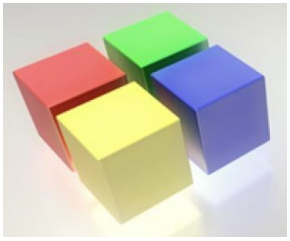


Code::Blocks v Čechách a na Slovensku

web: **http://www.itac.webnode.cz**
e-mail: **RobertHK@seznam.cz**

Úvodem:
Zajímáte se o IDE Code::Blocks? Zde je pro začátek překlad manuálu Code::Blocks 10.05; v1.1 v češtině.

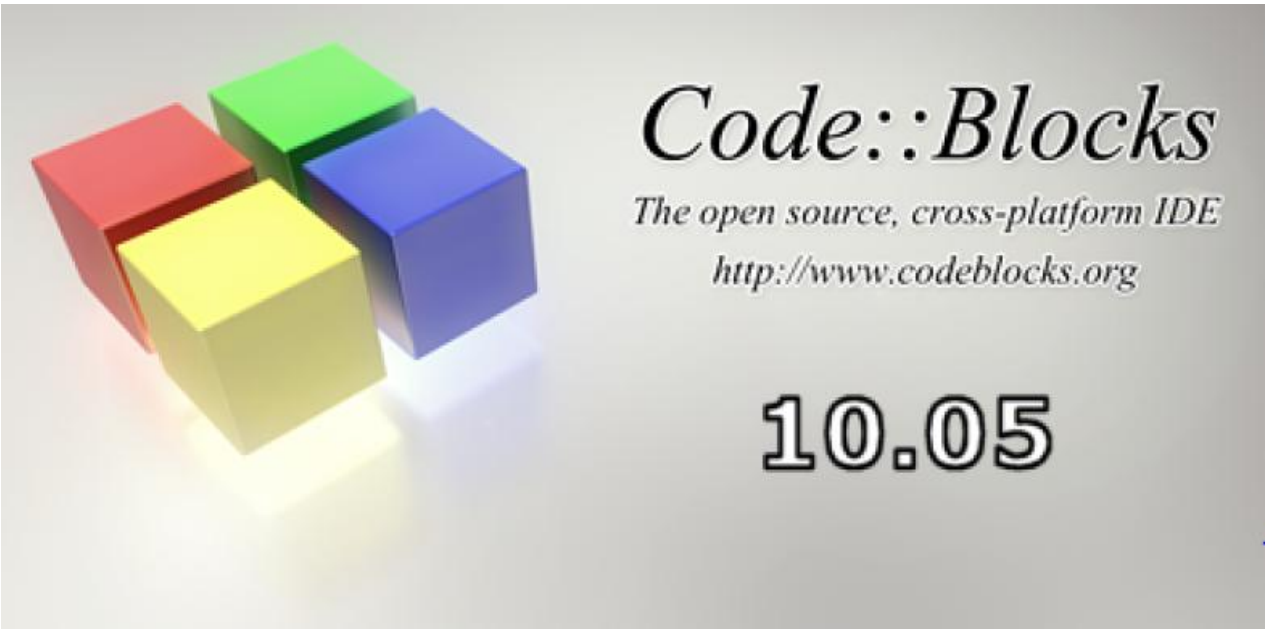
Historie:
Dovolil jsem si (primárně pro svou vlastní potřebu), přeložit originální uživatelský (en) manuál z oficiálních stránek <http://www.codeblocks.org/>. Překlad jsem se snažil (zejména co se týče prostorového uspořádání textů i obrázků) co nejvěrněji zachovat na pozicích vymezených stránkami originálu. A to z důvodu snadnějšího ověření pravdivosti přeloženého textu (abych se mohl kdykoliv vrátit k originálu za účelem případného upřesnění mého překladu). Jak se blížil závěr překládání, bylo mi líto nechat si vše jen pro sebe. A tak bych se rád o tento překlad podělil s těmi začátečníky v Code::Blocks, kteří se stejně jako já nechtějí zdržovat v době počátečního seznamování se s tímto skvělým IDE jazykovou stránkou problematiky. Byl bych rád, kdyby se ke mně někdo přidal, eventuelně provedl odbornou korekturu použitých termínů (nejsem ani rodilý inglišmen, ani ajťák od kolébky...).

Cíle:
Chtěl bych (po jistém "usazení se" překladu), strukturu originálu graficky upravit (zdánlivě nesmyslná prázdná místa mezi odstavci slouží pouze pro revizi přeloženého textu vůči originálu – viz výše). Dále by bylo vhodné k tomuto manuálu přiřadit různé uživatelské příručky které (a mnohé jsou skvělé) vznikají jaksi spontánně a jsou na netu různě roztříštěné na všech možných diskuzních fórech. Ať již se návody týkají instalace, vytváření konzolových aplikací, widgetů, Qt, GTK+ ..., všechny mají něco společného. A to je: nezbytná nutnost zvládnout IDE Code::Blocks jako takový.

Poděkování:
Chtěl bych poděkovat každému, kdo by se chtěl konstruktivně přidat k vývoji uceleného pohledu na IDE Code::Blocks v našich podmínkách (ČR + SR). Dále bych rád poděkoval každému, kdo by projevil (stejně jako já, tj. nezištně a bezplatně) touhu podělit se o své zkušenosti na všech možných platformách a implementacích tohoto IDE.

V neposlední řadě bych moc rád poděkoval autorovi těchto stránek, který mi umožnil celý tento (doufám) všeobecně prospěšný projekt „uvést do povědomí obce programátorské“....

S pozdravem Róbert Mojžiš



CodeBlocks

Manuál cz

Verze 1.1

Poděkování týmu:

Anders F. Björklund (afb), Biplab Kumar Modak (biplab), Bartomiej wiecki (byo), Paulo 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)

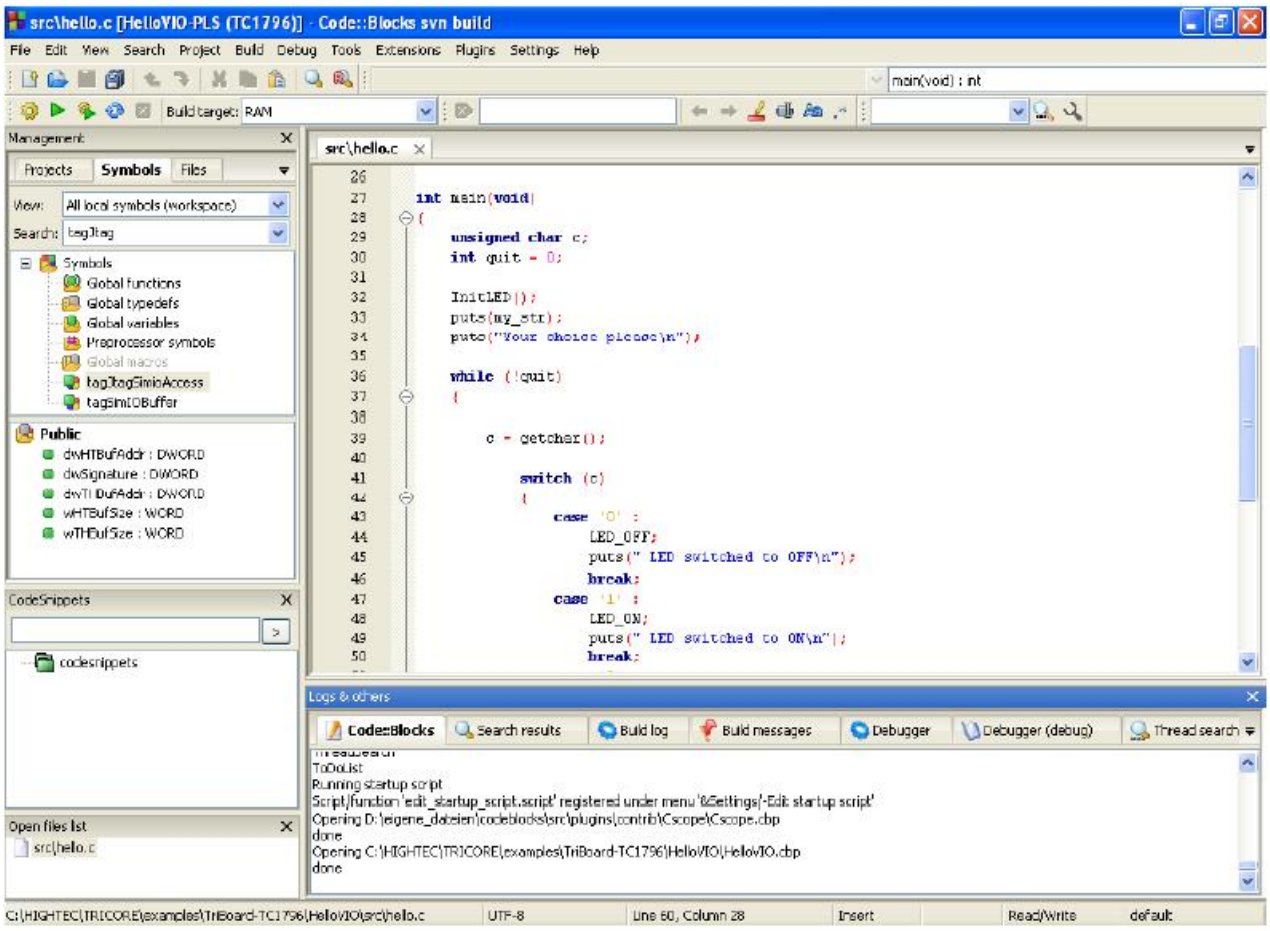
Z anglického originálu přeložil R. Mojžiš.

OBSAH

1 Řízení projektů CodeBlocks

Návody ke kapitole 3 na str. 53 a? na stránce? jsou oficiální dokumentací ze stránek CodeBlocks Wiki a jsou k dispozici pouze v angličtině.

Níže uvedený obrázek ukazuje design uživatelského rozhraní CodeBlocks.



Obrázek 1.1: IDE CodeBlocks

Management:	toto okno obsahuje rozhraní "Projekty", které se bude v následujícím textu nazývat jen pohled na projekt. Tento pohled zobrazuje všechny projekty otevřené v jistém čase. Záložka "symboly" okna Management zobrazuje symboly,proměnné atd..
Editor:	v editoru je zobrazen zdroj pojmenovaný hello.c otevřený se zvýrazněním syntaxe
Open files list:	zobrazuje seznam všech souborů otevřených v editoru, v tomto příkladu: hello.c.
Code Snippets:	fragmenty kódu lze zobrazit přes menu "Zobrazit" → "Code Snippets". Zde můžete Spravovat textovými moduly, odkazy na soubory a odkazy na URL.
Logs & Others:	toto okno se používá pro výstup výsledků vyhledávání, log zpráv kompilátor, atd.

Stavový řádek podává přehled o následujících nastaveních:

- Absolutní cesta souboru otevřeného v editoru.
- Editor používá výchozí kódování znaků vašeho hostitelského operačního systému. Toto nastavení se zobrazí jako výchozí.
- Aktuální pozice kurzoru v editoru (řádek, sloupec).
- Nastavení režimu klávesnice pro vkládání textu (Vložit nebo Přepsat).
- Aktuální stav souboru. Upravený soubor bude označen s úpravami, jinak je tato položka prázdná.
- Povolení souboru. Soubor nastavený pouze pro čtení se ve stavovém řádku zobrazí jako pouze pro čtení v okně "Seznam otevřených souborů". Tyto soubory budou zvýrazněny pomocí ikony překrytí zámkem.

> Poznámka:
> V aktivním editoru si může uživatel vybrat kontextovou nabídku Vlastnosti.
> V zobrazeném dialogovém okně lze vybrat na kartě "Obecné" možnost "Soubor pouze ke čtení". Tato možnost bude mít přístup pouze pro čtení odpovídajícího souboru v CodeBlocks, ale původní možnost číst a zapisovat atributy souboru se nemění.

- Pokud začnete v CodeBlocks volbou příkazového řádku - - osobnost = <profil> pak stavový řádek zobrazí aktuálně používaný profil, jinak bude zobrazen standardní (default). Nastavení CodeBlocks jsou uloženy v odpovídajícím konfiguračním souboru <personality>. conf.

CodeBlock nabízí velmi flexibilní a komplexní řízení projektů. Následující text se bude zabývat pouze některými aspekty řízení projektu.

1.1 Vzhled projektu

V CodeBlocks jsou zdroje a nastavení procesu sestavení uložené v projektu Soubor <název>. CBP. C/ C++ zdroje a odpovídající hlavičkové soubory jsou typické součásti projektu. Nejjednodušší způsob jak vytvořit nový projekt je provedení příkazu "Soubor" → "Projekt" a vybrat průvodce. Potom můžete přidat soubory do projektu prostřednictvím kontextového menu "Přidat soubory" v okně Správa (Management).

CodeBlocks řídí projekty v kategoriích shodně s jejich příponami. Toto jsou přednastavené kategorie:

Sources	Zdroje - obsahuje zdrojové soubory s příponami *. c, *. cpp,
ASM sources	ASM zdroje - obsahuje zdrojové soubory s příponami *. s; *. S, *. ss, *. asm.
Headers includes	Hlavičky vložené, mezi jinými, soubory s příponou *. h;.
Resources	Zdroje zahrnují soubory pro popis rozvržení oken wxWidgets okna s příponami *.res, *. xrc;. Soubory těchto typů jsou uvedeny na kartě "Resources" v okně Manangement.

Nastavení pro typy a kategorie souborů lze nastavit pomocí kontextového menu "strom projektu" → "Upravit typy souborů a skupin. Zde také můžete definovat vlastní, uživatelské kategorie pro tento soubor

s vlastními příponami. Například pokud chcete vyjmenovat seznam skriptů linkeru s příponou *.ld v kategorii nazvanou Linkerscript, tak musíte vytvořit novou kategorii.

Poznámka:
Pokud deaktivujete funkci "strom projektu" → "Kategorizace podle typů souborů" v kontextovém menu, bude vypnuto zobrazení kategorie a soubory budou uvedeny tak, jak jsou uloženy v systému souborů.

1.2 Poznámky k projektům

V CodeBlocks mohou být pro projekt uloženy tzv. poznámky. Tyto poznámky by měly obsahovat stručný popis a tipy pro odpovídající projekt. Tím, že se tyto informace zobrazují při otevření projektu, mají ostatní uživatelé rychlý přehled o projektu. Zobrazení poznámek lze zapnout nebo vypnout na záložce Notes ve vlastnostech projektu.

1.3 Šablony projektu

CodeBlocks je dodáván s celou řadou projektových šablon, které jsou zobrazovány při vytváření nového projektu. Je však také možné ukládat vlastní šablony pro sbírku vlastních specifikací pro spouštění kompilátoru, optimalizace použití, specifické spouštěcí nástroje atd. v šablonách. Tyto šablony budou uloženy do adresáře v Documents and Settings \ <User> \ Application Data \ CodeBlocks \ UserTemplates. Pokud budou šablony otevřené pro všechny uživatele, musí být zkopírovány do příslušného instalačního adresáře CodeBlocks. Tyto šablony se zobrazí při dalším spuštění CodeBlocks v "New" → "Project" → "User templates".

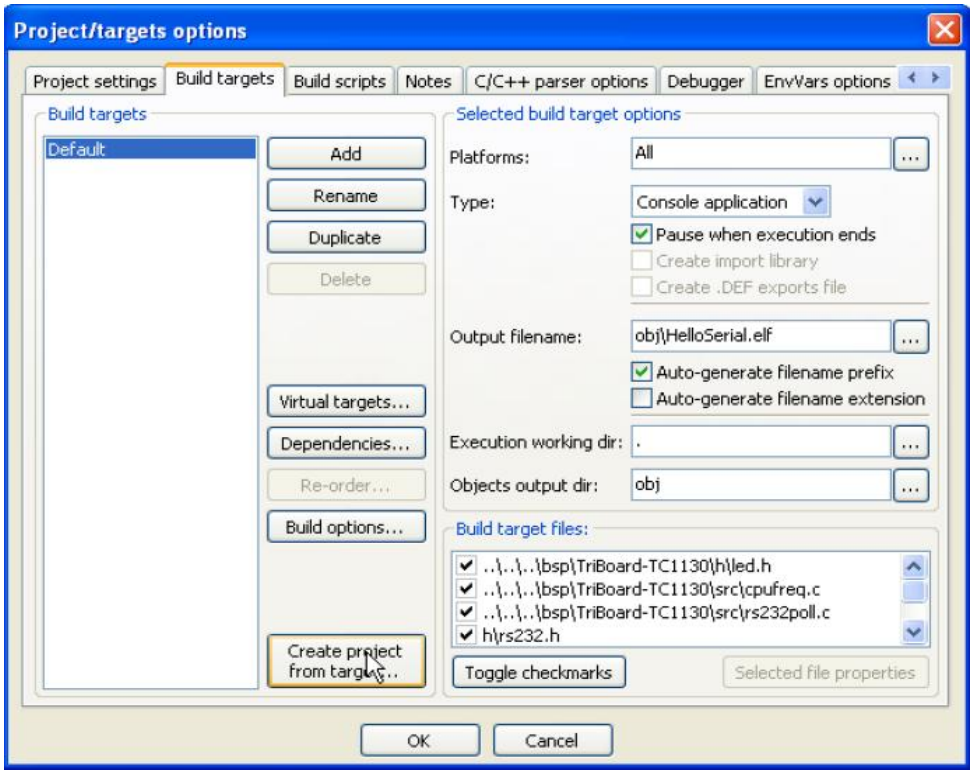
Poznámka:
Šablona dostupná v průvodci projektem může být editována výběrem pomocí kliknutí na pravé tlačítko.

1.4 Vytvoření projektu z cílů sestavení

V projektech je nutné mít dostupné různé varianty projektu. Varianty jsou vytvářeny jako tzv. cíle. Ty se liší s ohledem na jejich volby kompilátoru, informace o ladění a/nebo výběrem souborů. Vytvořený cíl lze také přesunout do samostatného projektu. Aby to mohlo být uděláno, klikněte na "Project" → "Properties", zvolte variantu na kartě 'Build Targets' a klikněte na tlačítko "Create project from targets" (viz obrázek 1.2 na straně 4).

1.5 Virtuální cíle

V CodeBlocks mohou být projekty dále členěny na tzv. Virtuální cíle. Často používané struktury projektu se skládají ze dvou sestavovacích cílů. Jeden "Ladící" cíl, který



Obrázek 1.2: Cíl sestavení

obsahuje informace o ladění a jeden cíl "Release", který je bez této informace. Přidáním virtuálních cílů přes "Project", "Properties" → "Build Targets " mohou být kombinovány jednotlivé cíle sestavení. Například virtuální cíl "All" vytvoří verze Target a Release současně. Virtuální Cíle jsou uvedeny v panelu symbol překladače pod Build Targets.

1.6 Kroky před a po sestavení (pre-build, post-build)

CodeBlocks umožňuje vykonávat dodatečné operace před, nebo po kompilaci projektu. Tyto operace jsou nazývané Pre-built nebo Post-built Steps. Typické kroky Post-build jsou:

• Vytvoření Intel Hexaformátu z konečného objektu

• Manipulace s objekty objcopy

• Generování výpisu obsahu souboru podle objdump

Například:

Rozklad (disassembly) objektu pod Windows. Prolomení souboru vyžaduje volání cmd s parametrem /c.

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

Archivace projektu může být dalším příkladem Post-built kroku. Za tímto účelem se vytvoří Build Target 'Archiv', který v Postbuilt kroku začlení následující instrukci:

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

Pomocí tohoto příkazu, bude aktivní projekt a jeho zdroje, hlavičky a objekty zabaleny do souboru zip.

Přitom se extrahují vestavěné proměnné $ (PROJECT_NAME) a $ (TODAY), název projektu a aktuální datum (viz oddíl 3.2 na straně 54). Po provedení "archiv" cíle bude komprimovaný soubor uložen v adresáři projektu.

V adresáři share / CodeBlocks / scripts najdete několik příkladů skriptů. Skript můžete přidat přes menu "Settings" → "Scripting" a registrací v menu. Pokud např. provedete z nabídky Skript make_dist, pak všechny soubory patřící k projektu budou komprimovány v archivu <project>. tar.gz.

1.7 Přidávání skriptů v cílech sestavení

CodeBlocks nabízí možnost využití nabídky činností ve skriptech. Skript představuje další míru svobody pro řízení výroby vašeho projektu.

> Poznámka:
> Skript může být také zahrnutý do Build Target.

1.8 Pracovní plocha a závislosti projektu

V CodeBlocks mohou být otevřené rozličné projekty. Uložením otevřených projektů přes ' File' → ' Save workspace' je můžete soustředit v jediné pracovní ploše pod <název>.workspace.
Jestli otevřete <název>.workspace během dalšího spuštění CodeBlocks, všechny projekty se opět zobrazí.

Komplexní softwarové systémy se skládají ze součástí, které jsou řízeny v různých projektech CodeBlocks. Nad to, s generací takových softwarových systémů jsou mezi těmito projekty často závislosti.

Příklad:

Projekt A obsahuje základní funkce, které jsou k dispozici i pro jiné projekty ve formě knihovny. Teď: jestli jsou zdroje tohoto projektu upravené, pak musí být přestavěna i knihovna Pro udržování konzistence mezi projektem B který funkce používá a projektem A který funkce realizuje , závisí projekt B na projektu A. Potřebné informace o závislostech projektů je uložen v příslušné pracovní ploše tak, že každý projekt je vytvořen samostatně. Použitím závislostí je také možné ovládat pořadí, ve kterém mohou být projekty vygenerovány. Závislostí na projekty lze nastavit pomocí výběru menu "Projekt" → "Vlastnosti" a poté kliknutím na tlačítko "Závislostí projektu".

1.9 Vložení souboru Assembleru

V okně Správa projektu jsou v kategoriích zdrojů souborů uvedeny soubory ASM. Uživatel si může změnit seznam souborů v kategoriích (viz oddíl 1.1 na straně 2). Pravým kliknutím na jeden z uvedených souborů Assembleru se otevře kontextové menu. Zvolte "Vlastnosti" k otevření nového okna. Nyní vyberte záložku "Vytvořit" pro aktivaci dvou oblastí:

"Compile File" a "Link soubor '. Pak vyberte "Advanced" a proveďte následující kroky:

- Nastavte "kompilátor proměnné" na CC
- V kompilátoru vyberte "Pro tento překladač"
- Zvolte "Použít vlastní příkaz k vytvoření tohoto souboru"
- V okně zadejte:

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

Proměnné jsou označeny $ (viz kapitola 3.4 na straně 58). Ty jsou automaticky, takže stačí vyměnit možnost Assembler <asopt> podle vlastního nastavení.

1.10 Editor a Nástroje

1.10.1 Výchozí kód

Firemní kódovací pravidla požadují zdrojové soubory k tomu, aby měly sériová, standardní provedení. CodeBlocks umožňuje automaticky zahrnout předdeklarovaný obsah na začátku souboru při vytvoření nových C/C++ zdrojů a hlaviček. Tento předdeklarovaný obsah je nazývaný implicitním kódem. Toto nastavení může být vybraný pod ' Stettings ' → ' Editor' implicitní kód. Jestli vytvoříte nový soubor pak makro expanduje proměnné, to se provede například definováním přes menu ' Settings' → ' Global Variables'. A nový soubor může být vytvořený přes nabídkový 'File' → 'New' → 'File'.

Například:

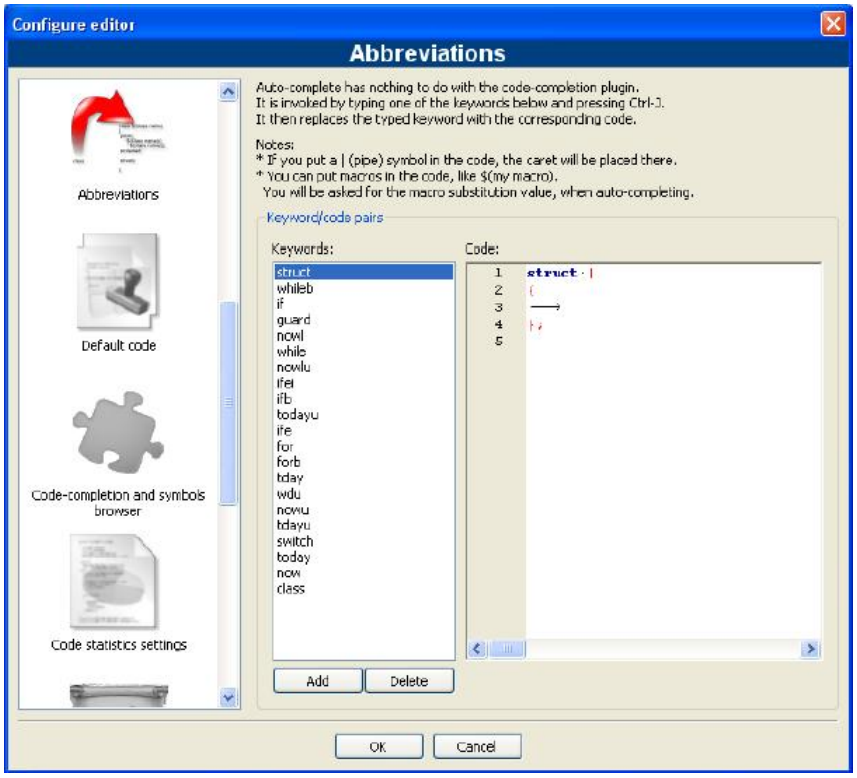
```
/*************************************************************************
* Project: $(project)
* Function:
*************************************************************************
* $Author: mario $
* $Name: $
*************************************************************************
*
* Copyright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

1.10.2 Zkratky

Mnoho typizací lze vložit do CodeBlocks definováním zkratek. To se provádí volbě "Nastavení" → "Editor" a definováním zkratek pod názvem <name>, které pak můžou být volány klávesovou zkratkou Ctrl-J (viz obr. 1.3 na straně 7).

Parametrizace je rovněž umožněna vložením proměnných zkratkou $(NAME).

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



Obrázek 1.3: Definice zkratek

Při provádění zkratky `<name>` ve zdrojovém textu a provedení Ctrl-`J`, je včleněn obsah požadované proměnné.

1.10.3 Osobnosti - personálie

Nastavení CodeBlocks jsou uložena jako aplikační data v souboru nazvaném `<user>.conf` v adresáři CodeBlocks. Tento konfigurační soubor obsahuje informaci o posledních otevřených projektech, nastavení editoru, zobrazení tabulky symbolů atd. Ve výchozím nastavení je 'standardní' osobnost stanovena tak, že konfigurace je uložena v souboru default.conf. Jestli je CodeBlocks vyvolán z příkazové řádky s parametrem
`--personality=myuser`, nastavovací hodnoty budou uloženy v souboru myuser.conf. Jestli profil neexistuje, tak bude automaticky vytvořen. Tato procedura umožňuje vytvořit odpovídající profily pro různé pracovní kroky. Jestli začnete CodeBlocks z příkazové řádky s dodatečným parametrem `--personality=ask`, tak pro všechny dostupné profily bude zobrazeno výběrové pole.

> Poznámka:
> Název aktuálního profilu / osobnosti se zobrazí v pravém rohu stavového řádku.

1.10.4 Konfigurační soubory

Nastavení CodeBlocks jsou uložena v default.conf profilu CodeBlocks v adresáři Data aplikací. Při použití osobností (viz pododdíl 1.10.3 na str. 7),

konfigurační údaje budou uloženy v `<personality>`. conf souboru.

Nástroj cb_share_conf, který se nachází v instalačním adresáři CodeBlocks, se používá pro správu a ukládání nastavení.

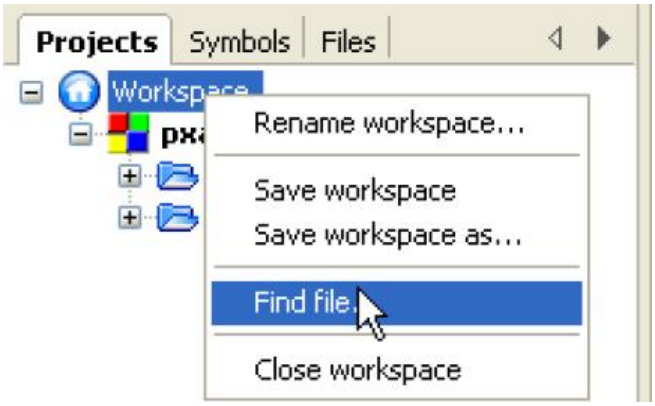
Pokud si přejete definovat standardní nastavení pro několik uživatelů počítače, konfigurační soubor default.conf musí být uložený v adresáři `\Documents and Settings\Default User\Application Data \CodeBlocks.` Při prvním spuštění zkopíruje CodeBlocks nastavení ze 't User' do aplikačních dat aktuálních uživatelů.

Chcete-li vytvořit přenosnou verzi CodeBlocks na USB flash disk, postupujte následovně. Zkopírujte CodeBlocks instalace na USB flash disk a uložte konfigurační soubor v default.conf tohoto adresáře. Konfigurace bude použita jako globální nastavení. Věnujte prosím pozor, aby byl soubor uložen jako zapisovatelný, jinak změny nastavení nelze uložit.

1.10.5 Navigace a vyhledávání

V CodeBlocks existují různé způsoby rychlé navigace mezi soubory a funkcemi. Typickým postupem je nastavování záložek. Pomocí zkratky Ctrl-B je ve zdrojovém souboru záložka nastavena nebo odstraněna. Přes Alt-PgUp můžete přejít na předchozí záložku a pomocí Alt-PgDn můžete přeskočit na další záložku.

Jestli vy vyberete pracovní pole nebo v pohledu projektu v pracovním poli, budete schopný pátrát po souboru v projektu. Jen vyberte ' najděte soubor ' z kontextového menu, pak napište název souboru a soubor bude vybrán. Jestli jste dali zpět, tento soubor bude otevřený v editoru (viz Obrázek 1.4 na stránce 8).



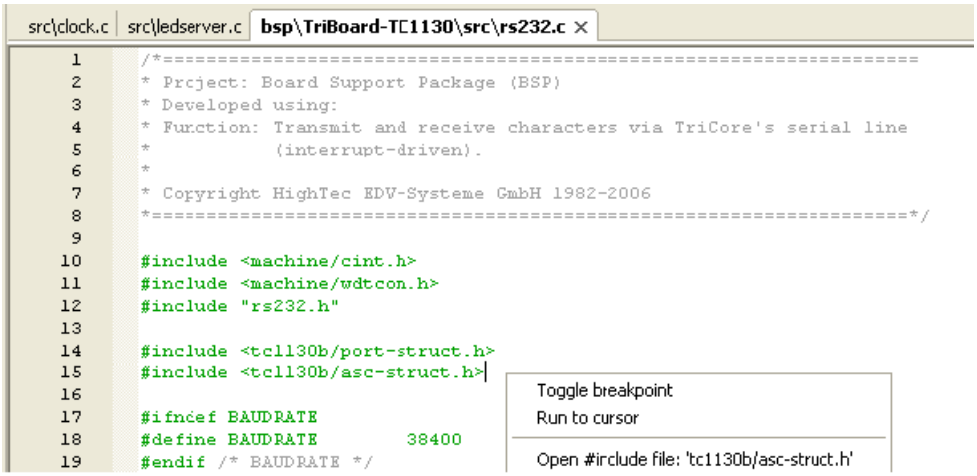
Obrázek 1.4: Vyhledávání souborů

V CodeBlocks můžete snadno manévrovat mezi hlavičkou / zdrojovými soubory jako:

1. Nastavte kurzor na místo, kde je včleněn hlavičkový soubor a tento soubor otevřete pomocí Kontextového menu "Open include file" (viz Obrázek 1.5 na straně 9).

2. Přepínat mezi hlavičkou a zdrojem pomocí kontextového menu "Swap header / source".

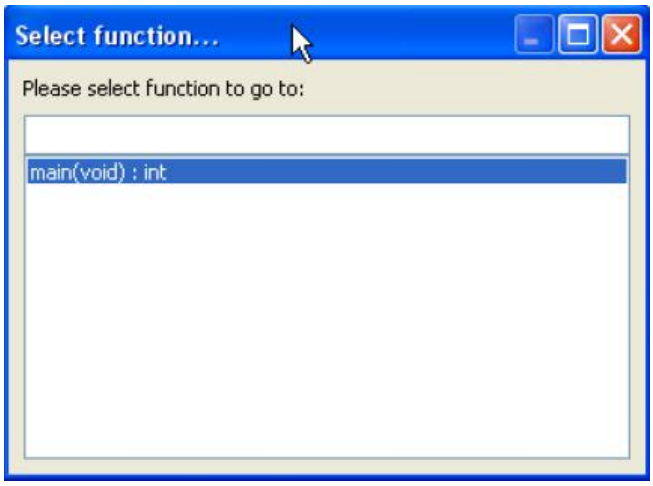
3. V editoru vyberte z kontextu např. definici a zvolte "Find declaration", čímž se otevře soubor včetně jeho deklarací.



Obrázek 1.5: Otevření hlavičkového souboru

CodeBlocks nabízí několik způsobů vyhledávání v rámci souboru nebo adresáře. Dialogové okno pro vyhledávání je otevřeno přes "Search" → "Find" (Ctrl-F), nebo "Find in Files" (Ctrl-Shift-F).

Alt-G a Ctrl-Alt-G jsou další užitečné funkce. V dialogu, který se otevře při použití této klávesové zkratky můžete vybrat soubory / funkce a pak se přejde na realizaci vybrané funkce (viz obr. 1.6 na str. 9) nebo otevře vybraný soubor v editoru. Můžete použít zástupné znaky jako * nebo ? atd. pro inkrementální vyhledávání v dialogovém okně.



Obrázek 1.6: Hledání funkcí

> Poznámka:
> Pomocí zkratky Ctrl-PgUp můžete přeskočit na předchozí funkce, a pomocí Ctrl-PgDn, můžete přeskočit na další funkci.

V editoru můžete otevřít nový dialog Otevřít soubor pomocí Ctrl - Tab a můžete přepínat mezi uvedenými položkami. Pokud je Ctrl-klávesa stisknuta, pak lze soubor vybrat různými způsoby:

1. Pokud levým tlačítkem myši vyberete položku, bude vybraný soubor otevřen.
2. Pokud stisknete klávesu Tab, můžete přepínat mezi uvedenými položkami. Uvolněním klávesy Ctrl se vybraný soubor otevře.

3. Pokud se budete pohybovat myší přes záznamy uvedených položek, bude aktuální výběr zvýrazněn. Uvolněním klávesy Ctrl se vybraný soubor otevře.

4. Je-li ukazatel myši mimo označený výběr, pak můžete využít kolečka myši k přepínání mezi vstupy. Uvolněním klávesy Ctrl se vybraný soubor otevře.

Běžným postupem při vyvíjení software představuje zápolení se sadou funkcí které jsou implementovány v rozmanitých souborech. Doplněk (plug-in) Browse Tracer (sledovací prohlížeč) vám pomůže tento problém vyřešit tím, že vám ukáže pořadí, v jakém byly soubory vybrány. Potom můžete pohodlně procházet voláním funkcí procedury (podívejte se na oddíl 2.8 na straně 38).

Zobrazení čísla řádků v CodeBlocks může být aktivováno pomocí "Settings" → "General Settings" v poli "Show line number". Zkratka Ctrl-G nebo příkaz "Search" → "Goto line" vám usnadní skok na požadovaný řádek.

Pokud budete držet klávesu CTRL a vyberte text v CodeBlocks Editor umožňuje provádět např. Vyhledávání Google pomocí kontextového menu.

Poznámka:
Pokud budete držet klávesu CTRL a vyberete text v editoru, budete moct provádět např. vyhledávání v Google pomocí kontextového menu.

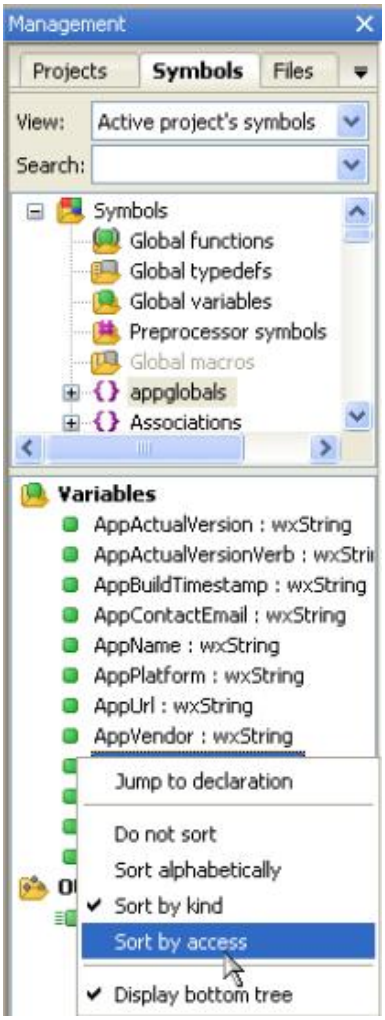
1.10.6 Zobrazení symbolů

Okno Management CodeBlocks nabízí stromové zobrazení symbolů C / C + + zdroje pro navigaci přes funkce nebo proměnné. Jako rozsah tohoto pohledu je možné nastavit aktuální soubor nebo projekt, nebo celý pracovní prostor.

Poznámka:
Vstupní doba nebo název vyhledávaného symbolu ve vstupní masce 'Search' prohlížeče symbolů má za následek filtrovaný pohled na symboly jestli se nějaké vyskytly.

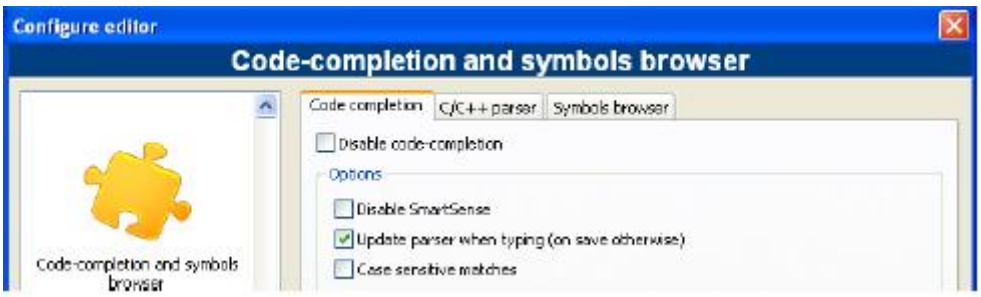
Existují symboly následujících kategorií:

Globální funkce	Seznam implementovaných globálních funkcí.
Globální typedefs	Seznam implementovaných definicí typedef.
Globální proměnné	Zobrazuje symboly globálních proměnných.
Symboly preprocesoru	Seznam instrukcí pre-procesoru vytvořených #define.
Globální makra	Seznam instrukcí maker pre-procesoru



Obrázek 1.7: Zobrazení symbolů

Struktury a třídy jsou zobrazeny ve "spodním stromu" a posloupnost pořadí lze změnit pomocí kontextového menu. Pokud je kliknutím myši vybrána kategorie kde se symbol nachází, symbol se zobrazí v dolní části okna (viz 1.7 na str. 11). Poklepáním na symbol se otevře soubor, ve kterém je symbol definován nebo ve kterém jsou implementovány funkce a přejde na příslušný řádek. Auto-refresh prohlížeče symbolu bez uložení souboru lze aktivovat přes menu "Settings" → "Editor" → "Code Completion" (viz obr. 1.8 na straně 11). Pro projekty s mnoha symboly bude ovlivněn výkon.



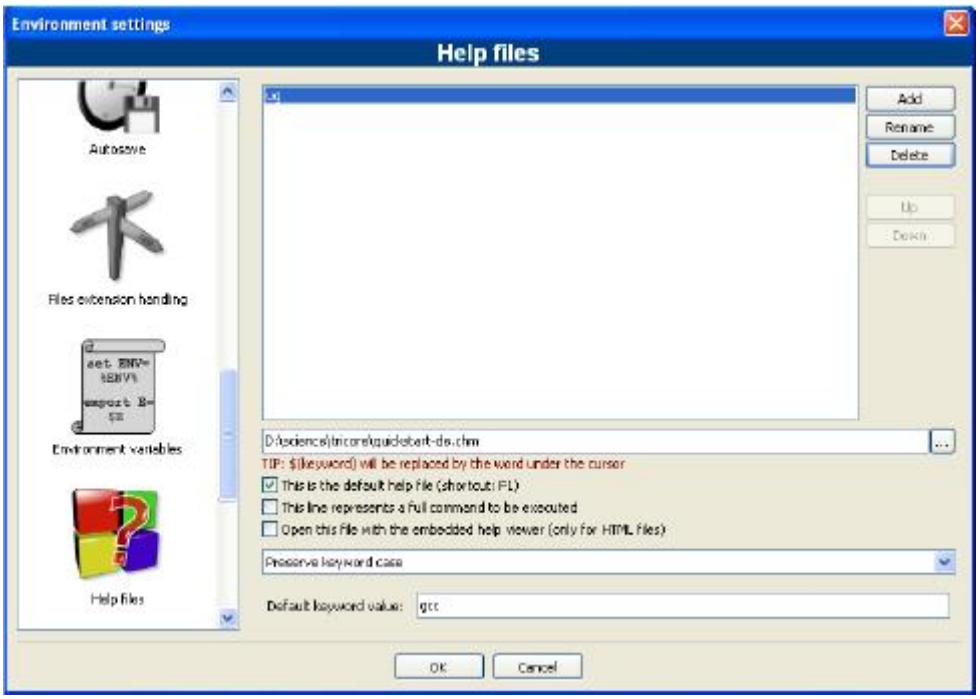
Obrázek 1.8: Umožní rozbor v reálném čase

Poznámka:
V editoru se seznam tříd zobrazí pomocí kontextového menu "metoda vložení deklaraci třídy implementace" nebo "Všechny metody třídy bez implementace".

1.10.7 Vložení externích souborů nápovědy

Vývojové prostředí CodeBlocks podporuje zahrnutí externích souborů nápovědy pomocí menu "Settings" → "Environment". Zahrnutí příručky dle vaší volby v chm formátu v ' souborech nápovědy' vyberou ' toto je výchozí soubor nápovědy' (viz Obrázek 1.9 na stránce 12). Vstup $ (klíčové slovo) je zástupný symbol pro vybranou položku v editoru. Teď můžete v CodeBlocks kliknutím myši vybrat funct v otevřeném zdrojovém souboru a odpovídající dokumentace se objeví při stlačení F1.

Jestli jste si začlenili víc souborů nápovědy, můžete si v editoru vybrat podmínky v kontextovém menu 'Locate in' pro vyhledání v CodeBlocks.

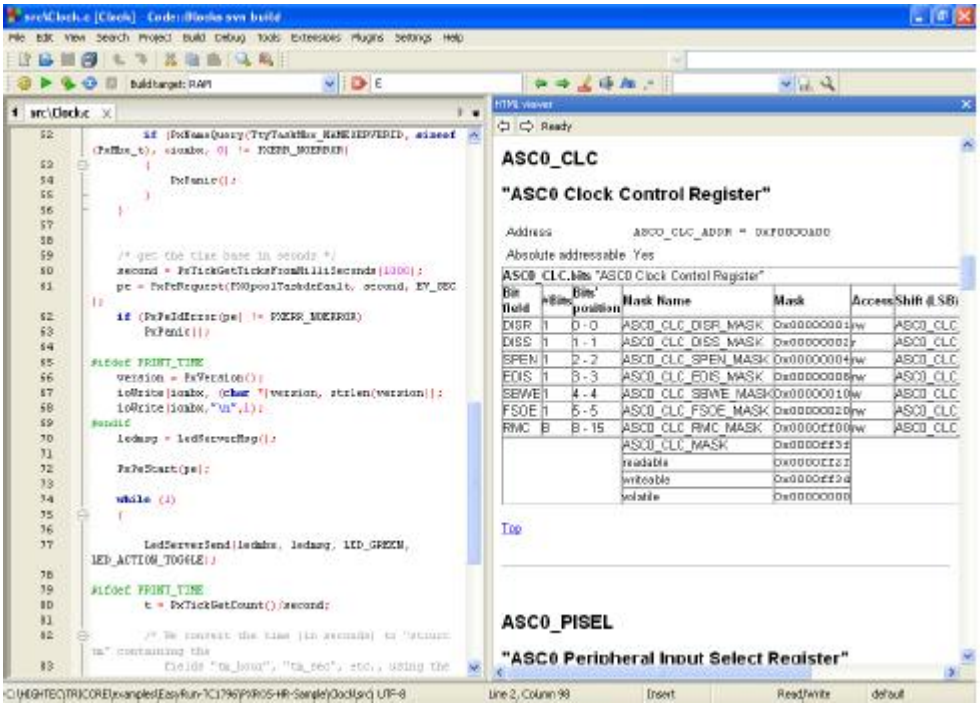


Obrázek 1.9: Nastavení souboru nápovědy

V CodeBlocks ještě můžete přidat podporu pro stránky manuálu. Stačí přidat položku "man" a zadat cestu:

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

CodeBlocks poskytuje "Vložený HTML Prohlížeč", který může být použit pro zobrazení jednoduchého html souboru a hledání klíčových slov v tomto souboru. Stačí nastavit cestu k HTML souboru, které by měly být analyzovány a umožnit volbu "Open this file with embedded help viewer" z menu "Settingsí" → "Environment" → "Help Files".



Obrázek 1.10: Vestavěný prohlížeč HTML

Poznámka:
Vyberete-li soubor ve formátu HTML poklepáním v průzkumníku souborů (viz kapitola 2.7 na straně 34), spustí se vložený prohlížeč HTML stejně dlouhý dokud žádné spojení pro html soubory v manažeru souborů rozšíření.

1.10.8 Vložení externích nástrojů

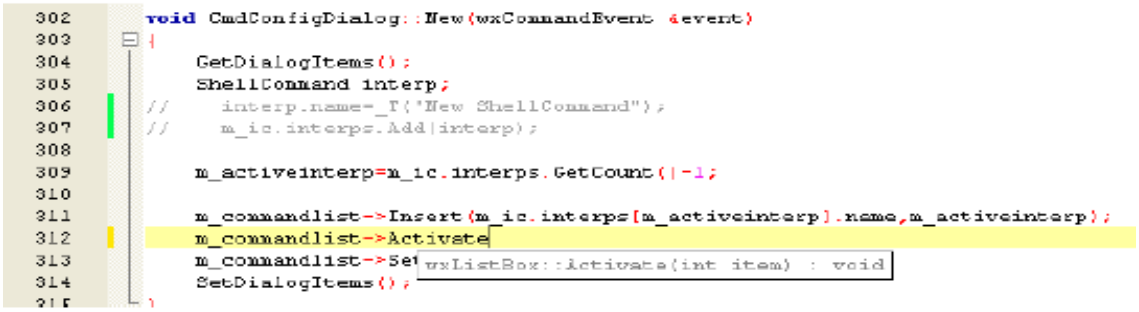
Vložení externích nástrojů je možné v CodeBlocks přes "Nástroje" → "Konfigurovat Nástroje" → "Přidat". Vestavěné proměnné (viz oddíl 3.2 na straně 54) mohou být přístupné také pro nástroj parametrů. Kromě toho existuje několik druhů zahájení voleb pro spouštění externích aplikací. V závislosti na možnosti, jsou externí aplikace po ukončení CodeBlocks zastaveny. Pokud je aplikace mají zůstat otevřený po ukončení CodeBlocks, musí být možnost "Spustit nástroj viditelný oddělený".

1.11 Tipy pro práci s CodeBlocks

V této kapitole budeme prezentovat několik užitečných nastavení pro CodeBlocks.

1.11.1 Sledování změn

CodeBlocks obsahuje funkci pro sledování změn ve zdrojovém souboru a změny ukazuje v proužku pro změny. Změny jsou označeny žlutou a úpravy, které jsou již uloženy budou používat zelenou barvu (viz obr. 1.11 na str. 14). Mezi změnami se můžete pohybovat přes menu "Search" → "Goto next changed line" nebo " Search " → " Goto previous changed line ". Stejné funkce je přístupná i přes klávesové zkratky Ctrl-F3 a Ctrl-Shift-F3.



Obrázek 1.11: Sledování změn

Tuto funkci lze povolit nebo zakázat v poli "Use Changebar" v menu" Settings " →" Margins and caret".

> Poznámka:
> Pokud je upravený soubor uzavřen, pak se změny v historii jako undo / redo a change bars ztratí. Přes menu "Edit" → " Clear changes history " nebo odpovídající kontextové menu můžete vymazat historii změn, i když je soubor stále otevřený.

1.11.2 Výměna dat s jinými aplikacemi

Data mohou být vyměňována mezi CodeBlocks a dalšími aplikacemi. Pro tuto komunikaci se používá DDE (Dynamic Data Exchange) pro Windows a pro různé operační systémy. Ten je založen na komunikaci TCP.

Pomocí tohoto rozhraní lze různým instancím CodeBlocks zasílat příkazy s následující syntaxí:

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

Pomocí tohoto rozhraní lze různým instancím CodeBlocks zasílat příkazy s následující syntaxí:

V současné době jsou k dispozici tyto příkazy:

`Open`	Příkaz `[Open("d:\temp\test.txt")]` používá parametr, v našem případě se jedná o soubor určený s absolutní cestu a otevře se ve stávající instanci CodeBlocksu, nebo se první instance spustí v případě potřeby.
`OpenLine`	Tento příkaz otevře soubor na daném čísle řádku instance CodeBlocks. Číslo řádku je uvedeno jako: line. `[OpenLine("d:\temp\test.txt:10")]`
`Raise`	Soustředí pozornost na instanci CodeBlocks. Parametr nesmí být zastaralý.

1.11.3 Konfigurace environmentálních proměnných

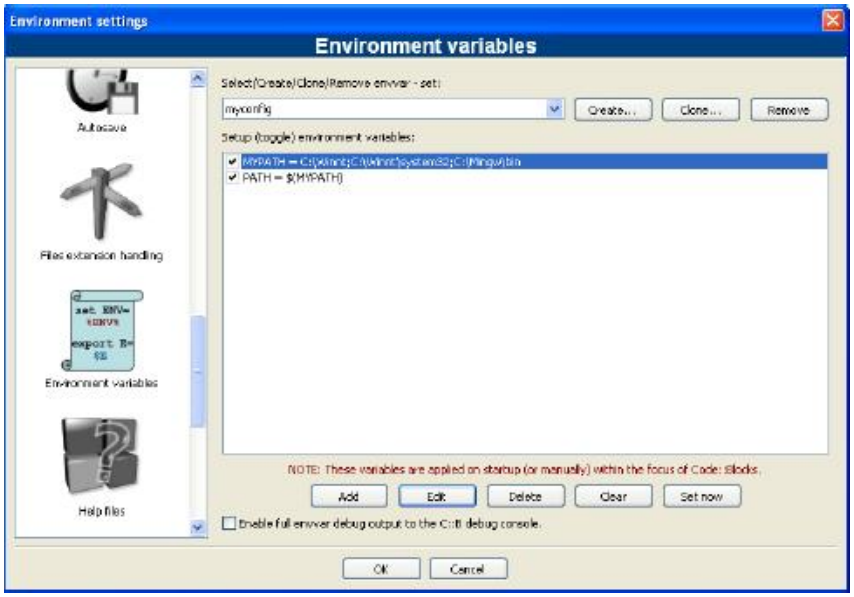
Konfigurace pro operační systém je určen tzv. environmentální proměnné. Například environmentální proměnná PATH obsahuje cestu k nainstalovanému kompilátoru.
Operační systém bude zpracovávat tuto environmentální proměnnou od začátku až do konce. Položky na konci budou tedy prohledávány poslední. Pokud jsou instalovány různé verze překladače nebo jiné

aplikace, mohou nastat následující situace:

• Je volána nesprávná verze software

• Instalované softwarové balíky se volají navzájem

Tak to by mohlo být v případě, že různé verze kompilátorů a další nástroje jsou povinné pro různé projekty. V takovém případě je jednou z možností pro změnu environmentální proměnné v systému řízení pro každý projekt. Nicméně, tento postup je náchylný k chybám a není flexibilní. Pro tento požadavek CodeBlocks nabízí elegantní řešení. Je možné vytvořit různé konfigurace environmentálních proměnných, které se používají pouze interně v CodeBlocks. Navíc mezi těmito konfiguracemi můžete přepínat. Obrázek 1.12 na straně 15 ukazuje dialog, který můžete otevřít pomocí "Environment Varibales" v "Settingsí" → "Environment". Konfigurace je vytvořena pomocí tlačítka "Create".



Obrázek 1.12: Environmentální proměnné

Přístup a rozsah environmentální proměnné zde vytvořené se omezuje na CodeBlocks. Tyto environmentální proměnné můžete rozšířit právě tak jako jiné proměnné CodeBlocks pomocí $(NAME).

> Poznámka:
> Konfiguraci environmentální proměnné pro každý projekt lze vybrat v kontextovém menu "Properties" na kartě "EnvVars options".

Například

Můžete napsat použitý environment do post-build kroku (viz oddíl 1.6 na str. 4), Soubor <project>. env a archivovat ve vašem projektu.

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

nebo pro Linux

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

1.11.4 Přepínání mezi pohledy

V závislosti na požadovaném úkolu může být užitečné mít různé konfigurace, nebo pohledy CodeBlocks a uložit tyto konfigurace / pohledy. Ve výchozím nastavení jsou nastavení (např. zobrazit / skrýt panely symbolů, rozvržení atd.) uložené v default.conf konfiguračním souboru. Pomocí příkazového řádku --personality=ask se v průběhu spuštění CodeBlocks mohou zvolit různá nastavení. Nehledě na toto globální nastavení může nastat situace, kdy chcete za chodu přepínat mezi různými pohledovými okny a palet symbolů. Editace souborů a ladění projektů jsou dva typické příklady takových situací. CodeBlocks nabízí mechanismus pro ukládání a výběr různých perspektiv proto, aby uživatel nemusel často ručně otevírat a zavírat okna. Chcete-li uložit perspektivu, vyberte v nabídce "View" → "Perspectives" → "Save current", zadejte název do <name>. Příkaz "Settings" → "Editor" → "Keyboard Shortcuts" → "View" → "Perspectives" → "<name>" umožňuje pro tento proces definovat klávesové zkratky. Tento mechanismus umožňuje přepínat mezi různými pohledy jednoduše pomocí horkých kláves.

> Poznámka:
> Dalším příkladem je editace souboru v režimu celé obrazovky bez panelu symbolů. Pro tento účel si můžete si vytvořit pohled jako "Full" a přiřadit mu klávesovou zkratku.

1.11.5 Přepínání mezi projekty

Je-li současně otevřeno několik projektů nebo souborů, musí uživatel určit způsob jak rychle přepínat mezi projekty nebo soubory. Pro takové situace má CodeBlocks několik zkratek.

Alt-F5 Z pohledu projektu aktivuje předchozí projekt.

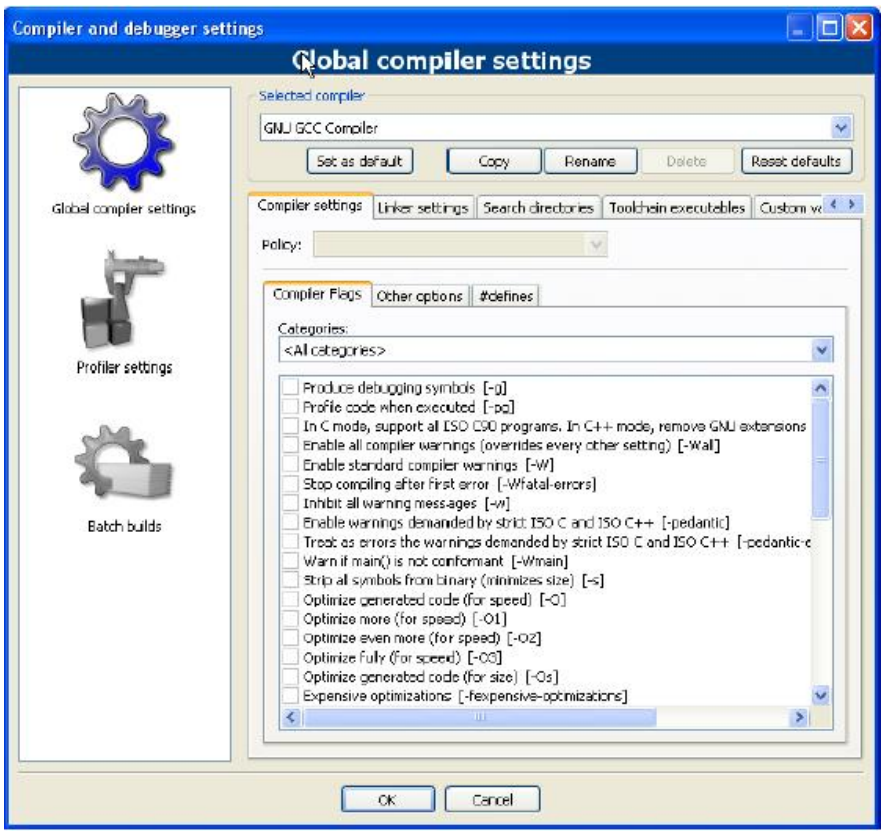
Alt-F6 Z pohledu projektu aktivuje další projekt.

F11 Přepínače editoru mezi zdrojovým souborem <name> . cpp a odpovídajícím hlavičkovým souborem `<name>.h`.

1.11.6 Rozšířené nastavení kompilátoru

Během průběhu sestavování projektu jsou na kartě Build Log v okně zpráv zobrazené zprávy kompilátoru. Pokud si přejete získat detailní informace, zobrazení lze prodloužit. Za tímto účelem klikněte v rozevíracím poli na tlačítko "Settings" → "Compiler and Debugger" a zvolte "Other Settings".

Dejte pozor na správný výběr kompilátoru. "Full command line" Nastavení v oblasti překladačů Přihlášení výstupy kompletní informace v protokolu sestavení. Kromě toho,



Obrázek 1.13: Nastavení detailních informací

může být tento výstup nahrán do souboru HTML. Pro tento účel zvolte 'Save build log to HTML file when finished'. Kromě toho CodeBlocks nabízí průběhu tohoto procesu sestavení v okně Build Log, které lze aktivovat pomocí nastavení "Display build progress bar".

1.11.7 Zoomování v editoru

CodeBlocks poskytuje velmi efektivní editor. Tento editor umožňuje měnit velikost, ve kterém je otevřený text zobrazen. Pokud používáte myš s kolečkem, stačí stisknout klávesu Ctrl a vybraný text pomocí kolečka myši přibližovat a oddalovat.

> Poznámka:
> Pomocí zkratky Ctrl-Numepad-/ nebo menu "Edit", "Special commands" → "Zoom" → "Reset" je obnovena původní velikost písma aktivního souboru v editoru.

1.11.8 Režim zalamování

Při editaci textových souborů v CodeBlocks, například *. txt by mohlo být užitečné mít text zalomený, což znamená, že se dlouhé řádky zobrazí v několika řádcích na obrazovce tak, aby mohly být řádně upraveny. Funkci zalomení textu "Word wrap" lze aktivovat pomocí "Settings" → "Editor" → "Others options", nebo nastavením políčka "Word wrap". Pomocí kláves Home a End umístěte kurzor na začátek nebo na konec zalomených řádků, resp. při nastavení "Settings" → "Editor" → "Other options" a "Home key always move to caret to
First column", kurzor se umístí na začátek nebo na konec aktuálního řádku, respektive, pokud je stisknuto

HOME nebo END. Je-li kurzor umístěn na začátek prvního řádku aktuálního odstavce je vhodné použít kombinaci kláves "Alt-Home". Totéž platí obdobně pro "Alt-End" pro umístění kurzoru na konec posledního řádku aktuálního odstavce.

1.11.9 Režim výběru v editoru

CodeBlocks podporuje různé režimy pro výběr a vkládání řetězců.

1. Pomocí levého tlačítka myši lze v aktivním editoru vybrat text pak tlačítkem myši uvolnit. Kolečkem myši může uživatel přejít na pozici. Prostředním tlačítkem myši lze vkládat dříve vybraný text. Tato funkce je k dispozici pro soubor a může ukládat do schránky souboru.

2. Stisknutím klávesy "Alt" se aktivuje tzv. obdélníkový režim výběru. Výběr může být zvětšen pomocí levého tlačítka myši. Pokud je klávesa Alt je uvolněna, tento výběr lze kopírovat nebo vložit. Tato funkce je užitečná, pokud chcete vybrat nějaký sloupec, např. z pole a kopírovat nebo vkládat obsah.

3. V menu "Settings" → "Editor" → "Margins und Caret" mohou být aktivovány tzv. "virtuální mezery". Tato volba umožňuje, že výběr v blokovém výběrovém režimu může začít nebo skončit uvnitř prázdného řádku.

4. V menu "Settingsí" → "Editor" → "Margins und Cared" může být aktivován vícenásobný výběr "Multiple selection". Při držení klávesy Ctrl si uživatel může levým tlačítkem myši vybrat různé řádky. Výběr bude vložen do schránky (clipboardu) pomocí kláves Ctrl-C a Ctrl-X. Kombinace kláves Ctrl-V vloží obsah schránky na aktuální pozici kurzoru. Další možností je tzv. "Povolit psaní (a mazání)" (Enable typing and deleting). Toto může být aktivováno pro více výběrů. Tato funkce je užitečná, pokud chcete přidat pre-procesor, jako je instrukce # ifdef v různých zdrojových řádků, nebo chcete-li přepsat nebo nahradit text na několika místech.

> Poznámka:
> Většina Linuxových správců oken Alt-Left ClickDrag přesunout okno, takže si budete muset zakázat toto chování správci oken pro první vybraný pracovní blok.

1.11.10 Sbalení kódu

CodeBlocks podporuje tzv. sbalení kódu (folding). Díky této funkci je možné sbalit např. funkcí v rámci editoru CodeBlocks. Sklopený bod je označen symbolem mínus v levém okraji pohledu editoru. Na okraji jsou vidět začátek a konec sbaleného kódu jako svislá čára. Pokud levým tlačítkem myši kliknete na symbol minus, kód bude sbalený nebo rozbalený. Přes menu "Edit" → "Folding" si můžete vybrat sbalení. V editoru uvidíte sbalený kód jako kontinuální vodorovnou čáru.

Poznámka:
Skládací styl a omezení hloubky skládání lze nastavit přes menu "Settings" → "Editor" → "Folding".

CodeBlocks poskytuje funkci sbalení i pro příkazy pre-procesoru. Pro aktivaci této funkce zvolte " Fold preprocessor commands " z menu " Settings "→" Editor " ve vstupu balení.

Další možností je vytvořit uživatelsky definované skládací body. Zahajovací bod skládání je zapsán jako komentář s otevřením složenou závorkou a konec je komentář s pravou složenou závorku.

```
//{
kód s uživatelsky definovanými sbalením
//}
```

1.11.11 Automatické dokončení

Když v CodeBlocks otevřete projekt, vyhledávací adresáře překladače analyzují zdroje a hlavičky vašeho projektu. Navíc je analyzována oprávněnost klíčových slov. Informace z analýzy jsou požity pro automatické dokončení CodeBlocks. Prosím kontrolujte si nastavení editoru, jestli je tato vlastnost povolena. Auto dokončení je přístupné pomocí zkratky Ctrl-mezerník. V menu " Settings " → " Editor " → " Syntax highlighting " můžete přidávat oprávnění uživatelsky definovaným klíčovým slovům.

1.11.12 Hledat zbytky souborů

Jestli je soubor z disku odstraněn, ale zůstal včleněn do souboru projektu <projekt>.cbp, pak bude tento "zbytkový soubor" v pohledu projektu zobrazen symbolem rozbití. Místo mazání (deleting) souborů bys měl používat nabídku "Remove file from project".

Ve velkých projektech s mnoha podadresáři může být hledání poškozených (rozbitých) souborů časově dost náročné. Pro jednoduché řešení tohoto problému nabízí CodeBlocks doplněk (plug-in) ThreadSearch (viz oddíl 2.6 na straně 30). Pokud vložíte hledaný výraz do ThreadSearch a vyberete možnost "Project files" nebo "Workspace files", pak bude ThreadSearch zpracovávat všechny soubory které jsou zahrnuty do projektu nebo pracovního prostoru. Pokud je poškozený soubor nalezen, ThreadSerch oznámí chybu v nenalezeném souboru.

1.11.13 Vkládání knihoven

V možnostech sestavení projektu můžete přidat používané knihovny přes tlačítko "Add" v položce "Link libraries " v "Linker Settings". Přitom je možné použít buď absolutní cestu do knihovny, nebo jen dát jméno bez přípony lib a přípony souboru.

Například:
Pro knihovnu nazvanou <path>nlibsnlib<name>.a, stačí napsat <name>. Linker pak odpovídajícím hledáním cest správně včlení knihovny.

> Poznámka:
> Dalším způsobem jak zahrnovat knihovny je v oddílu 2.10 na straně 39.

1.11.14 Pořadí spojování objektů

Při sestavování jsou objekty name.o vytvořeny ze zdrojů name.c / cpp. Linker pak spojuje jednotlivé objekty do aplikace name.exe nebo pro vestavěné systémy do name.elf. V některých případech může být vhodné předdefinovat pořadí, ve kterém jsou objekty propojeny. V CodeBlocks, toho lze dosáhnout přidělením priorit. V kontextovém menu "Properties", můžete definovat prioritu souboru na kartu "Vytvořit". Nižší priorita má za následek dřívější spojení souboru.

1.11.15 Automatické ukládání

CodeBlocks nabízí způsoby jak automaticky ukládat projekty a zdrojové soubory, nebo vytvoření záložních kopií. Tato funkce může být aktivována v menu "Settingsí" → "Environment" → "Autosave". Přitom, "Save to . save file" by měla být stanovena jako metoda pro vytvoření záložní kopie.

1.11.16 Nastavení přípon souborů

V CodeBlocks, můžete si vybrat mezi několika způsoby ošetření přípon souborů. Dialog pro nastavení lze otevřít pomocí "Settings" → "File extension handling." Můžete buď použít jednotlivé přípony souborů (otevřít s přidruženou aplikaci) přidělené pro aplikace Windows, nebo měnit nastavení pro každou příponu tak, že bude zahájeno při spuštění uživatelsky definovaného programu (při spuštění externího programu), nebo se soubor otevře v editoru CodeBlocks (otevře se v editoru CodeBloks).

> Poznámka:
> Pokud je uživatelsky definovanému programu přiřazena určitá přípona souboru, nastavení "Disable CodeBlocks za běhu externího programu " by mělo být deaktivováno, protože jinak se CodeBlocks ukončí při každém spuštění programu s touto příponou.

1.12 CodeBlocks v příkazové řádce

IDE CodeBlocks může být spuštěn z příkazové řádky bez grafického rozhraní. V takovém případě existuje několik parametrů dostupných pro řízení procesu sestavení projektu. Vzhledem k tomu, že CodeBlocks je zapisovatelný příkazy, může být vytváření spustitelných souborů integrováno do vašich vlastních pracovních procesů.

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

// CZ: CodeBlocks.exe /na /nd --žádná úvodní obrazovka --postavený <název>.cbp
- -cíl='vydání'//
```

`<filename>` Určuje název souboru projektu *. cbp nebo pracovní plochy *. workspace filename. Například <filename> je project.cbp. Umístěte tento argument na konec příkazového řádku, těsně před přesměrováním výstupu, pokud existuje.

--file=<filename>[:line]
Otevře soubor v Code:: Block a případně přejde na konkrétní řádek.

/h, --help Zobrazuje nápovědu, pokud jde o argumenty příkazového řádku.

/na, --no-check-associations
Neprovádějte žádné kontroly sdružení souborů (pouze pro Windows).

/nd, --no-dde Nespouštějte server DDE (pouze Windows).

/ni, --no-ipc Nespouštějte server IPC (pouze pro Linux a Mac).

/ns, --no-splash-screen
Při načítání aplikace skryjte úvodní obrazovku.

/d, --debug-log
Zobrazení protokolu ladění aplikace.

--prefix=<str>
Nastavuje adresář sdílených dat předpony.

/p, --personality=<str>, --profile=<str>
Nastaví osobností uživatelů. Můžete použít dotaz jako parametr seznamu všech dostupných osobností.

--rebuild Čistí a sestavuje projekt nebo pracovní prostor.

--build Sestavuje projekt nebo pracovní prostor.

--target=<str>
Nastaví cíl pro dávkové sestavování, například --target='Release'.

--no-batch-window-close
Držte záznamové okno dávky viditelné po dokončení sestavování dávky.

--batch-build-notify
Po dokončení sestavování dávky se zobrazí zpráva.

--safe-mode Při spuštění jsou zakázány všechny doplňky (plugins).

`> <build log file>`
Umístěno na úplně poslední pozici příkazového řádku může být použit k přesměrování Standardního výstupu do záznamového souboru. Toto není možností samotného CodeBlocks, ale pouze standardního DOS / * nix shellu přesměrování výstupu.

1.13 Klávesové zkratky

I když jsou IDE, jakým je i CodeBlocks ovládány primárně pomocí myši, klávesové zkratky zůstávají velmi užitečným způsobem urychlení a zjednodušení pracovních procesů. V níže uvedené tabulce autoři shromáždili některé z klávesových zkratek.

1.13.1 Editor

Funkce	Klávesová zkratka
Zpět poslední akce	Ctrl-Z
Opakovat poslední akci	Ctrl-Shift-Z
Prohodit hlavička / zdroj	F11
Komentovat zvýrazněný kód	Ctrl-Shift-C
Odkomentovat zvýrazněný kód	Ctrl-Shift-X
Automatické doplnění / Zkratky	Ctrl-Space/Ctrl-J
Přepnout záložku	Ctrl-B
Jdi na předchozí záložku	Alt-PgUp
Jdi na další záložku	Alt-PgDown

Toto je seznam klávesových zkratek poskytovaných komponenty editoru CodeBlocks. Tyto klávesové zkratky nelze provázat.

Funkce	Klávesová zkratka
Vytvoření nebo odstranění záložek	Ctrl-F2
Jdi na další záložku	F2
Vybrat další záložku	Alt-F2
Hledat výběr	Ctrl-F3
Hledat výběr zpětně	Ctrl-Shift-F3
Najít odpovídající vnořené přeskoky procesoru	Ctrl-K

1.13.2 Soubory

Funkce	Klávesová zkratka
Nový soubor nebo projekt	Ctrl-N
Otevřít existující soubor nebo projekt	Ctrl-O
Uložit aktuální soubor	Ctrl-S
Uložit všechny soubory	Ctrl-Shift-S
Zavřít aktuální soubor	Ctrl-F4/Ctrl-W
Zavřít všechny soubory	Ctrl-Shift-F4/Ctrl-Shift-W

1.13.3 Zobrazit

Funkce	Klávesová zkratka
Zobrazit / skrýt panel zpráv	F2
Zobrazit / skrýt panel řízení	Shift-F2
Aktivovat předchozí (v projektu strom)	Alt-F5
Aktivovat další (v projektu strom)	Alt-F6

1.13.4 Hledání

Funkce	Klávesová zkratka
Najít	Ctrl-F
Najít další	F3
Najít předchozí	Shift-F3
Najít v souboru	Crtl-Shift-F
Nahradit	Ctrl-R
Nahradit v souboru	Ctrl-Shift-R
Jdi na řádek	Ctrl-G
Jdi na další změněný řádek	Ctrl-F3
Jdi na předešlý změněný řádek	Ctrl-Shift-F3
Jdi na soubor	Alt-G
Jdi na funkci	Ctrl-Alt-G
Jdi na předešlou funkci	Ctrl-PgUp
Jdi na další funkci	Ctrl-PgDn
Jdi na deklaraci	Ctrl-Shift-.
Jdi na implementaci	Ctrl-.
Otevřít vložený soubor	Ctrl-Alt-.

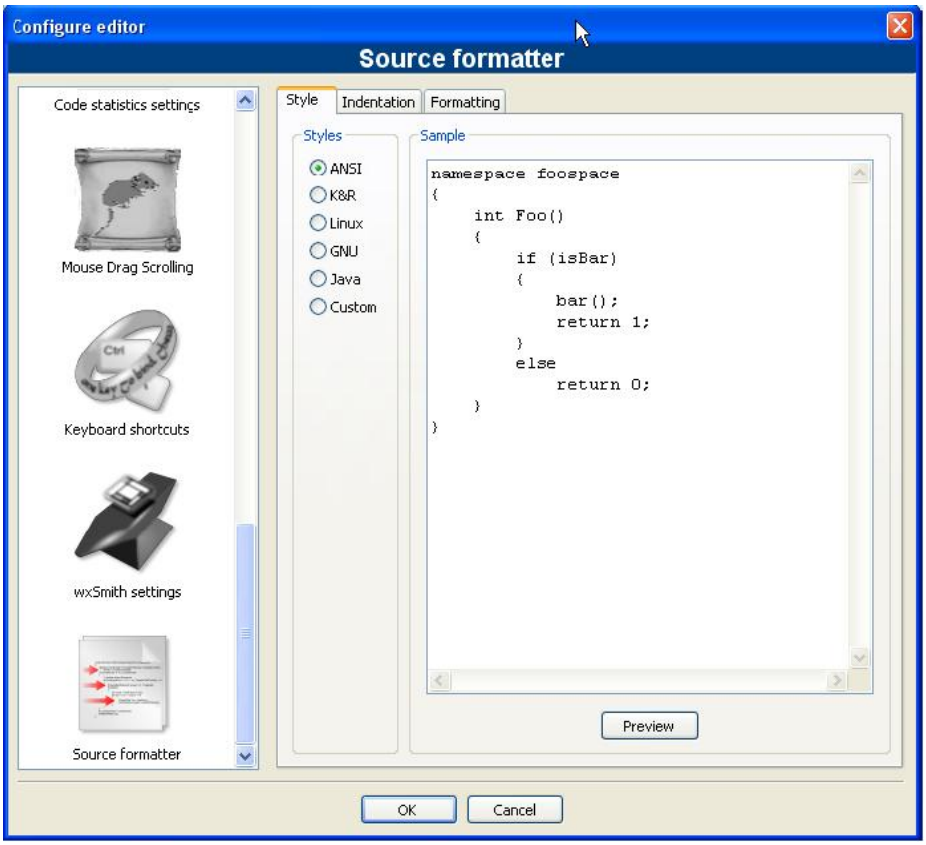
1.13.5 Sestavování

Funkce	Klávesová zkratka
Sestavit	Ctrl-F9
Kompilace aktuálního souboru	Ctrl-Shift-F9
Spustit	Ctrl-F10
Sestavit a spustit	F9
Přestavět	Ctrl-F11

2 Plugins - doplňky

2.1 Astyle

Umělecký styl je odsazovač zdrojového kódu, formátovač zdrojového kódu a zkrášlovač zdrojového kódu programovacích jazyků C, C + +, C #. Může být použit k výběru z různých stylů pravidel pro kódování v CodeBlocks.



Obrázek 2.1: Formátování zdrojového kódu

Při odsazování zdrojového kódu máme jako programátoři tendenci používat mezerníky a tabulátory. Některé editory navíc při stlačení klávesy 'tab' standardně vkládají mezery místo tabulátorů. Další editory automaticky „zkrášlují“ řádky nastavením bílého znaku před řádek kódu, nebo vkládají do kódu mezery tam, kde byl dosud tabulátor používán pouze pro odsazení.

Jedním ze standardních problémů, kterým programátoři čelí při přechodu z jednoho editoru do druhého je to, že při změnách zdrojového kódu mezi editory je počet zobrazených mezer každého znaku tabulátoru se kód který obsahuje mezery a tabulátory (a který byl doposud perfektně členěn) se při přechodu na jiný editor stává náhle nepřehledným. Dokonce, i když se jako programátor postaráte o použití pouze mezer nebo tabulátorů, pohled na zdrojové kódy jiných lidí může být stále problematický.

Pro řešení tohoto problému byl vytvořen umělecký styl / - filtr napsaný v C++ tak, že automaticky přeodrážkuje a přeformátuje zdrojové soubory C / C++ / C#.

Poznámka:
Při kopírování kódu například z internetu nebo manuálně, bude tento kód automaticky přizpůsoben pravidlům kódování v Code::Blocks

2.2 Útržky kódu

Zásuvný modul (plug-in) CodeSnippets umožňuje stromové zobrazení náhledu na uspořádání textových modulů a odkazů na soubory shodně s kategoriemi. Tyto moduly se používají pro ukládání často používaných souborů a pojmů v textových modulech a jejich řízení z centrálního místa. Představte si následující situaci: několik často používaných zdrojových souborů je uloženo v různých souborových adresářích systému. Zobrazení CodeSnippets nabízí možnost vytvářet kategorie a nižší kategorie odkazů na požadované soubory. Díky těmto funkcím můžete nezávisle řídit přístup k souborům bez ohledu na místo uložení v systému souborů a rychle se pohybovat mezi soubory, aniž by bylo nutné prohledávat celý systém.

Poznámka:
Můžete použít proměnné Code::Blocks, nebo proměnné prostředí, např. v souboru odkazů $ (VARNAME) / name.pdf na odkaz parametrizace v prohlížeči CodeSnippets.

Seznam textových modulů a vazby mohou být uloženy v okně CodeSnippets pomocí pravého tlačítka myši a výběrem "Uložit Index" z nabídky kontextového menu. Takto vytvořený soubor codesnippets.xml se pak nachází v podadresáři codeblocks, adresáře Documents and Settings \ Application Data. V Linuxu je tato informace uložena v souboru podadresáře .codeblocks, ve vašem adresáři HOME. Konfigurační soubory CodeBlocks budou načteny při příštím startu. Pokud chcete Obsah CodeSnippets uložit na jiném místě, zvolte vstup "Save Index as". Chcete-li načíst tento soubor, během příštího spouštění CodeBlocks zvolte "Load Index File" nebo začleňte adresář v menu "Settings" pod "Code Snippets". Tato nastavení jsou uložena v příslušné evidenci aplikačních dat codesnippets.ini.

Pro zařazení do kategorie použijte menu "Add SubCategory". Kategorie může obsahovat Snippets (textové moduly) nebo soubor odkazů (File Links). Textový modul je vytvořen pomocí příkazu "Add Snippet" v kontextové nabídce. Obsah je integrován do textu modulu jako "New snippet" výběrem pasáže textu z editoru CodeBlocks a přetažením na zobrazený modul dialogového okna vlastností. Poklepáním na nově zařazené položky nebo výběrem "Edit text" se otevře editor pro obsah.

Výstup textu modulu je v CodeBlocks ovládán pomocí příkazu kontextového menu "Apply", nebo přetažením do editoru. Ve Windows lze obsah úryvku přetáhnout rovněž do jiných aplikací. V prohlížeči CodeSnippets můžete zkopírovat vybranou položku a přetáhnout ji do jiné kategorie.

Kromě toho můžou být textové moduly s parametrem proměnné <name> , které lze zpřístupnit přes $ (name) (viz obr. 2.2 na straně 26). Hodnoty proměnných lze získat ve vstupním poli, pokud je textový modul pojmenovaný pomocí příkazu kontextového menu "Apply".

Obrázek 2.2: Úprava textového modulu

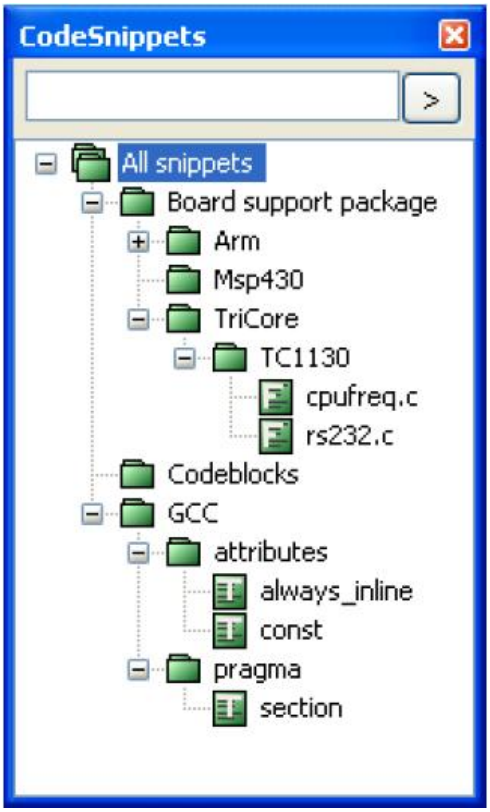
Kromě textových modulů může být rovněž vytvořeno spojení souborů. Pokud po vytvoření textového modulu klepnete na příkaz kontextové nabídky "Properies", tak si můžete zvolit cíl odkazu kliknutím na tlačítko "Link Target". Tímto způsobem se automaticky převede text do modulu odkazu na soubor. V CodeSnippets, budou veškeré textové moduly označeny symbolem T, propojení na soubor symbolem F a adresy URL symbolem U. Pokud si chcete otevřít vybraný soubor (link) v zobrazení codesnippets, vyberte z kontextového menu "Open File" nebo držte klávesu "Alt" a poklepejte na soubor.

> Poznámka:
> Do textových modulů můžete přidat i URL (např. http://www.codeblocks.org). URL lze otevřít pomocí kontextového menu "Open URL" nebo pomocí drag and drop ve svém oblíbeném webovém prohlížeči.

S tímto nastavením, je-li otevřen odkaz (link) na soubor ve formátu PDF, automaticky se spustí prohlížeč PDF Wiewer codesnippets. Tato metoda umožňuje uživateli přístup k souborům, které jsou rozšířeny po celé síti, jako jsou data CAD, uspořádání, dokumentace atd. běžných aplikací jednoduše pouze pomocí odkazu. Obsah codesnippets je uložen v souboru codesnippets.xml konfigurace je uložena v souboru codesnippets.ini v adresáři dat aplikace. Tento ini soubor bude například obsahovat cestu k souboru codesnippets.xml.

CodeBlocks podporuje používání různých profilů. Tyto profily se nazývají osobnosti. Start CodeBlocks s volbou příkazového řádku --personality=<profile> vytvoří nový, nebo použije existující profil. Nastavení pak nebudou uložena v místě adresáře dat aplikací v souboru default.conf, ale místo toho v <personality>. conf. Doplněk Codesnippets pak uloží vlastní nastavení v souboru <personality>.codesnippets.in. Nyní, když v Codesnippets nastavení vložíte nový obsah <name.xml> pomocí "Load Index File", bude tento obsah uložen v příslušném souboru ini. Výhoda této metody spočívá v tom, že v případě různých profilů, lze zdárně propojit různé konfigurace textových modulů.

Doplněk nabízí další funkce vyhledávání pro navigaci mezi kategoriemi a úryvky kódu. Rozsahy vyhledávání úryvků, kategorií, nebo úryvků a kategorií mohou být upraveny. Zadáním požadovaných hledaných výrazů je automaticky vybrán odpovídající záznam v zobrazení. Obrázek 2.3 na straně 27 ukazuje typické zobrazení v okně CodeSnippets.



Obrázek 2.3: Zobrazení CodeSnippets

Poznámka:
Při použití modulů v rozsáhlém dokumentu, by obsahy těchto modulů měly být uloženy v souborech pomocí "Convert to File Link", aby se redukovalo využití paměti v systému. Pokud odstraníte codesnippet nebo soubor link bude přesunut do kategorie koše (.trash). Pokud přidržíte klávesu Shift, položky budou smazány.

2.3 Přírůstkové hledání

Pro efektivní vyhledávání v otevřených souborech poskytuje CodeBlocks tzv. přírůstkové (inkrementální) vyhledávání. Tento způsob vyhledávání je pro otevření souboru zahájen přes menu "Search" → "Incremental Search" nebo klávesovou zkratkou Ctrl-I. Zaměření je pak automaticky nastaveno na vyhledávací masku příslušného panelu nástrojů. Jakmile zadáte hledaný výraz, bude pozadí vyhledávací masky upraveno v souladu s výskytem výrazu. V případě nálezu výrazu, který se nachází v aktivním editoru bude příslušná pozice v textu označena barvou. Ve výchozím nastavení bude aktuální nález zvýrazněn zeleně. Toto nastavení lze změnit pomocí "Settings" → "Editor" → " Incremental Search " (viz.? Na straně?). Po stisknutí klávesy Enter bude provedeno hledání dalšího výskytu hledaného řetězce v souboru. K výběru předchozího výskytu lze pomocí Shift-Return. Tato funkce je podporována Scintillou v případě, že inkrementální vyhledávání používá platné výrazy.

Poznámka z Wiki: Scintilla je opensource editační komponenta s pokročilými vlastnostmi jako například obarvování zdrojového kódu.

```
m_pToolbar->EnableTool(X
if (m_pControl != 0)
{
    m_SearchText=m_pText
    m_pToolbar->EnableTo
    m_pToolbar->EnableTo
    m_NewPos=m_pControl-
    m_OldPos=m_NewPos;
}
else
{
    m_pToolbar->EnableTo
    m_pToolbar->EnableTo
}
```

Pokud hledaný řetězec nelze nalézt v rámci aktivního souboru, je tato skutečnost zvýrazněna na pozadí vyhledávací masky červeným zobrazením.

ESC	Opustit přírůstkový způsob vyhledávání.
ALT-DELETE	Vymazat vstup přírůstkového vyhledávacího pole

Ikony v panelu přírůstkového vyhledávání mají následující význam:

Odstranění textu z vyhledávací masky dílčího Search Toolbar.

Navigace mezi výskyty hledaného řetězce.

Kliknutím na toto tlačítko jsou výsledky výskytu vyhledávaného řetězce uvnitř editoru barevně zvýrazněny (místo pouze počátečního výskytu).

Aktivace této možnosti omezuje hledání na pasáž textu označenou v editoru.

Tato možnost znamená, že bude probíhat hledání velkých a malých písmen.

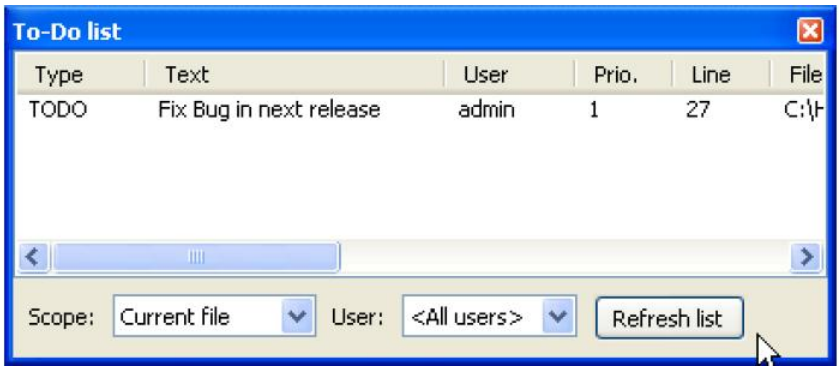
Platný výraz může být použit do vstupního pole pro přírůstkové vyhledávání.

> Poznámka:
> Standardní nastavení tohoto panelu lze nastavit v "Settings"
> → "Editor" → "Incremental Search".*

2.4 Seznam ToDo

U složitých softwarových projektů, kterých se účastní různí uživatelé je často kladen požadavek na různé úkoly pro různé uživatele. Za tímto účelem CodeBlocks

nabízí Todo List. Tento seznam lze otevřít pomocí "Search" → "To-Do list". Ten obsahuje úkoly, které mají být provedeny, spolu s jejich prioritami, typy a odpovědnými uživateli. V seznamu můžete filtrovat úkoly pro uživatele a / nebo zdrojové soubory. Řazení do sloupců lze dosáhnout kliknutím na titulek příslušného sloupce.



Obrázek 2.4: Zobrazení seznamu ToDo.

> Poznámka:
> To-Do list může být zakotven v konzoli správce. Vyberte možnost
> " Include the To-Do list in the message pane" z menu "Settings" → "Environment".

Pokud jsou v CodeBlosks otevřené zdroje, můžou být Todo přidány do seznamu pomocí příkazu kontextového menu " Add To-Do item". Do vybraného řádku zdrojového kódu bude přidán komentář.

```
// TODO (user#1#): add new dialog for next release
```

Při přidávání úkolů bude zobrazeno dialogové okno, kde je možné provést následující nastavení (viz obr. 2.5 na straně 30).

Uživatel — Jméno uživatele <user> v operačním systému. Úkoly pro ostatní uživatele zde mohou být vytvořeny taky. Přitom: odpovídající uživatelské jméno musí být vytvořeno jako Add new. Přiřazení Todo se pak provádí pomocí výběru z údajů uvedených pro uživatele.

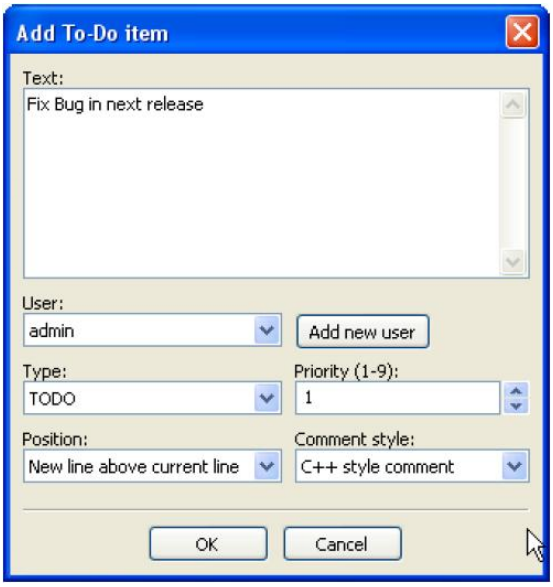
> Poznámka:
> Všimněte si, že uživatelé nemají nic společného s osobnostmi (personalies), používaných CodeBlocks.

Type — Ve výchozím nastavení je nastaven typ úkolu.

Priority — Především důležité úkoly v CodeBlocks mohou být vyjádřena prioritami (1 - 9).

Position — Toto nastavení určuje, zda mají být komentáře zahrnuty před, po, anebo na přesnou pozici kurzoru.

Comment Style — Výběr stylu formátů pro komentáře (např. doxygen).



Obrázek 2.5: Dialog pro přidání do ToDo.

2.5 Exportér zdrojového kódu

Často nastává nutnost převodu zdrojových kódů do jiných aplikací nebo do e-mailů. Pokud bude text pouze zkopírovaný, formátování bude ztraceno a text velmi nejasný. Exportní funkce CodeBlocks slouží jako pomocný prostředek v takových situacích. Požadovaný formát pro export souborů lze zvolit ve "File" → "Export". Program pak přijme název souboru a cílový adresář otevřeného zdrojového souboru a navrhne jej pro uložení exportovaného souboru. Odpovídající příponou souboru bude v každém případě určen formát exportu. K dispozici jsou následující formáty exportu:

html *A text-based* formát, který lze zobrazit ve webovém prohlížeči nebo v aplikacích pro zpracování textu.

rtf *Ritch Text Format* je textový formát, který lze otevřít v aplikacích pro zpracování textu jako je Word nebo OpenOffice.

odt *Open Document Text* formát je standardní formát, který byl specifikován společností Sun and O'Reilly.
Tento formát lze zpracovat ve Wordu, OpenOffice a jiných aplikacích pro zpracování textu.

pdf *Portable Document Format* je možné otevřít pomocí programů, jako je Acrobat Reader.

2.6 Hledání vláken (thread)

Poznámka Wiki: Vlákno (též vlákno řízení, anglicky thread) označuje v informatice odlehčený proces, pomocí něhož se snižuje režie operačního systému při změně kontextu, které je nutné pro zajištění multitaskingu (zdánlivého běhu více úloh zároveň, který je zajištěn jejich rychlým střídáním na procesoru) nebo při masivních paralelních výpočtech. Zatímco běžné procesy jsou navzájem striktně odděleny, sdílí vlákna nejen společný paměťový prostor, ale i další struktury.

Pomocí nabídky "Search" → "Search Thread“ je možné skrýt příslušné doplňky, nebo je zobrazit jako záložky Message Consoles. V CodeBlocks lze zobrazit náhled na zobrazený výskyt řetězce znaků v souboru, adresáře nebo pracovního prostoru. Přitom se seznam výsledků vyhledávání zobrazí po pravé straně konzole ThreadSearch.

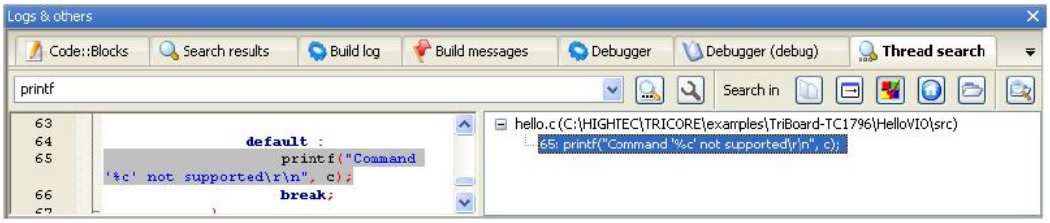
Kliknutím na položku v seznamu se zobrazí náhled na levé straně. Dvojitým kliknutím v seznamu, je vybraný soubor otevřen v editoru CodeBlocks.

> Poznámka:
> Rozsah přípon souborů vložených do hledání je stanovený předem a možná jej budete muset upravit.

2.6.1 Vlastnosti

Doplněk Hledat vlákno (Thread Search) nabízí následující funkce:

- Více-vláknové "Hledání v souborech".
- Náhled na výsledky pouze pro čtení uvnitř editoru.
- Soubory otevřené v poznámkovém bloku
- Kontextové menu "Hledat výskyty 'pro vyhledávání v souborech se slovem pod kurzorem.



Obrázek 2.6: Panel Thread Search

2.6.2 Používání

1. Nastavte si vlastní preference vyhledávání (viz obr. 2.7 na straně 32).
   Jakmile je doplněk nainstalován, existují 4 způsoby jak spustit hledání:

   a) Type / Vyberte slovo do vyhledávacího pole rozbalovacího panelu a stiskněte Enter nebo klikněte na tlačítko Vyhledat na panelu téma hledání zpráv poznámkového bloku.

   b) Typ / Vyberte slovo v rozbalovacím panelu vyhledávacího pole a stiskněte Enter nebo klikněte na tlačítko Vyhledat.

   c) Klikněte pravým tlačítkem myši na libovolné "slovo" v aktivním editoru a klikněte na "Hledat události".

   d) Klikněte na tlačítko Vyhledat / vlákno hledání (Search/Thread) pro nalezení aktuálního slova v aktivním editoru.

   > Poznámka:
   > V závislosti na aktuální konfiguraci nemusí být body 1, 2 a 3 k dispozici.

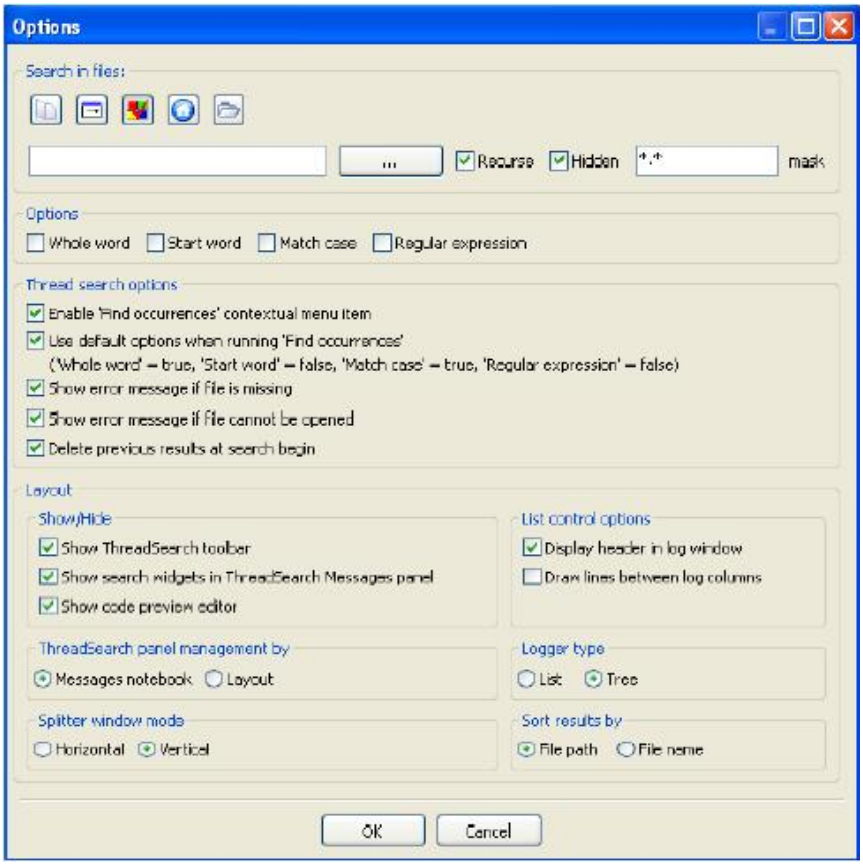
2. Pro zrušení stávajícího vyhledávání klikněte znovu na tlačítko Hledat.

3. Po jednom kliknutí na položku se v náhledu editoru zobrazí výsledek na správném místě.

4. Dvojklikem na výsledné položce otevírá nebo nastavuje editor zápisníku editoru na správném místě.

2.6.3 Konfigurace

Pro přístup k panelu nástroje Hledání Vláken klikněte na (viz obr. 2.7 na straně 32):



Obrázek 2.7: Konfigurace Hledání vláken

1. Ve vyhledávacím panelu zapněte Volby (Options).

2. Tlačítko Možnosti na panelu nástrojů téma hledání.

3. V menu Settings/Environment a potom v levém sloupci na položku nastavení Thread.

> Poznámka:
> V závislosti na aktuální konfiguraci nemusí být body 1, 2 a 3 k dispozici.

Hledání částečně definuje nastavení analyzovaných souborů.

- Políčka Projekt a Pracovní prostor se navzájem vylučují.

- Cestu k adresáři lze editovat nebo nastavit pomocí tlačítka Vybrat.

- Maska je nastavení specifikace souboru oddělena tečkou".". Například: *. cpp, *. c, *. h.

2.6.4 Volby

Whole word - označení kontroluje, jestli výraz vyhledaný v řádku odpovídá nalezenému celému výrazu bez znamének + a - před vyhledávaným výrazem.

Start word - je-li označeno, řádek odpovídá hledanému výrazu - jestli je hledaný výraz nalezen na začátku slova bez znamének + a - před vyhledávaným výrazem.

Match case - kontroluje citlivost vyhledávání podle malých a velkých písmen (case sensitive).

Regular expression – je hledaný výraz platným výrazem.

> Poznámka:
> Pokud chcete vyhledávat platné výrazy jako je N, budete muset nastavit volbu "Rozšířené vyhledávání RegEx" z menu "Nastavení" → "Editor" → "General Settings".

2.6.5 Volby vyhledávání vláken

Enable 'Find occurrences contextual menu item'. Pokud je zatrženo, nalezený výskyt "Zaměřeného slova" je vložen do kontextového menu editoru.

Use default options when running 'Find occurrences'. Je-li zaškrtnuto, je výchozí nastavení souboru aplikováno na menu zahájení "Najít výskyty ". Je povolena výchozí možnost "Pouze celá slova" a "malá a velká písmena".

Delete previous results at search begin. Pokud ThreadSearch je konfigurován s 'Tree View "pak výsledky vyhledávání budou vypsány hierarchicky,

- první uzel obsahující hledaný výraz

- jsou uvedeny nad soubory, které obsahují hledaný výraz

- v tomto seznamu se zobrazí číslo řádku a odpovídající obsah výskytu

Pokud hledáte podle různých podmínek, seznam se stane matoucí. Proto mohou být předchozí výsledky hledání používáním této volby vyčištěny pro nové hledání.

> Poznámka:
> V seznamu výskytů mohou být jednotlivé, nebo všechny položky odstraněny pomocí kontextového menu 'Odstranit položku' nebo 'Odstranit všechny položky. "

2.6.6 Dispozice

Display header in log window – když je zaškrtnuto, zobrazí v záhlaví okna seznamy výsledků ovládání.

> Poznámka:
> Jestli bude odznačeno, sloupec již neumožní změnu velikosti ale prostor bude zálohovaný.

Draw lines between columns - v režimu seznamu kreslí čáry mezi sloupci.

Show ThreadSearch toolbar - zobrazí nástrojovou lištu pluginu hledání vláken (threadsearch).

Show search widgets in ThreadSearch Messages panel - Pokud je zaškrtnuto, editor zobrazí pouze výsledky seznamu a náhled. Všechny ostatní hledání widgety jsou skryty (náhradní prostor).

Show code preview editor - náhledy na kód mohou být skryty buď touto volbou, nebo dvojitým kliknutím na střední hranici okna splitter. To je místo, kde lze ukázat znovu.

2.6.7 Ovládání panelů

Můžete si vybrat různé režimy řízení okna ThreadSearch okno. S nastavením 'Message Notebook' bude dokové okno v panelu zprávy. Pokud zvolíte nastavení 'layout' budete moci odpojit okna z panelu zpráv a dát ji někam jinam.

2.6.8 Typ zápisu

Pohled na výsledky vyhledávání lze zobrazit různými způsoby. Nastavení 'List' (seznam) zobrazí všechny výskyty jako seznam. Požadovaný režim 'Tree' (strom) shromažďuje všechny výskyty v souboru jako uzel.

2.6.9 Separátní režim zobrazení

Uživatel může měnit horizontální nebo vertikální rozdělení okna náhledu a výstupní okno výsledků hledání.

2.6.10 Třídění výsledků hledání

Pohledy na výsledky vyhledávání mohou být řazeny podle cesty nebo názvu souboru.

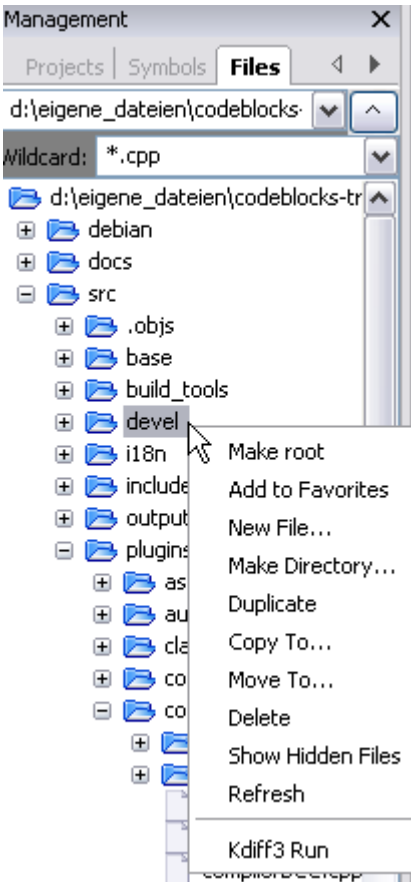
2.7 Správce souborů a PowerShell

Prohlížeč souborů (File Explorer, obr. 2.8 na straně 35) je součástí pluginu správce souborů. Můžete jej najít na záložce "Files". Skladba File Exploreru je zobrazena na obrázku 2.8 na straně 35.

Na vrcholu najdete pole pro zadání cesty. Kliknutím na tlačítko na konci tohoto pole, bude rozbalovací seznam polí historii posledních záznamů, ve kterém lze navigovat pomocí posuvníku. Šipkou na pravé straně pole se pohybujete nahoru adresářovou strukturou jednoho adresáře.

V zobrazeném poli 'Wildcard - zástupných' můžete zadat filtr termínu pro zobrazení souboru. Odchod z pole prázdné nebo zadáním * Výsledky ve všech lesa zobrazení. Zadáním například * c;. *. h,

má za následek, že se zobrazí pouze zdrojové a hlavičkové soubory typu C. Otevření rozbalovacího pole znovu vypíše seznam historie posledních záznamů.



Obrázek 2.8: Manažer soborů

Stisknutím klávesy Shift a kliknutím vyberete skupinu souborů nebo adresářů, stisknutím klávesy Ctrl a kliknutím vyberete více samostatných souborů nebo adresářů.

Pokud je ve File Explorer vybrán pouze jeden, nebo více adresářů, lze pomocí kontextového menu lze spustit následující operace:

Make Root - definuje aktuální adresář jako kořenový adresář.

Add to Favorites - nastaví ukazatel na adresář a uloží jej jako oblíbený. Tato funkce umožňuje rychle procházet
mezi často používanými adresáři, a to i na různých síťových discích.

New File - ve zvoleném adresáři vytvoří nový soubor.

New Directory - ve zvoleném adresáři vytvoří nový podadresář.

Pokud ve File Explorer vyberete jeden nebo více souborů nebo adresářů, lze pomocí kontextového menu spustit následující operace:

Duplicate - zkopírujte soubor / adresář a přejmenujte jej.

Copy To - otevře dialog pro zadání cílového adresáře, ve kterém má být zkopírovaný soubor / adresář uložen.

Move To - výběr přesune do cílového umístění.

Delete - odstraní vybrané soubory / adresáře.

Show Hidden Files - aktivuje / deaktivuje zobrazení skrytých systémových souborů. Pokud je aktivní, je položka menu zaškrtnutá.

Refresh - aktualizuje zobrazení adresářového stromu.

Pokud ve File Explorer vyberete jeden nebo více souborů, lze pomocí kontextového menu spustit následující operace:

Open in CB Editor - otevře vybraný soubor v editoru CodeBlocks.

Rename - přejmenuje vybraný soubor .

Add to active project - do aktivního projektu přidá soubor (y).

> Poznámka:
>
> Soubory / adresáře vybrané v Průzkumníku souborů jsou k dispozici pomocí pluginu PowerShell, proměnné mpaths.

Uživatelem definované funkce lze nastavit přes menu příkazem "Settings" → "Environment" → "PowerShell. V masce PowerShell je nová funkce, která může být pojmenována náhodně - vytvořena pomocí tlačítka "New".

V poli "ShellCommand Executable" je uveden spustitelný program a do pole v dolní části okna je možné programu předávat další parametry. Kliknutím na funkce v kontextovém menu, nebo v nabídce PowerShell se funkce spustí a zahájí zpracování vybraných souborů / adresářů. Výstup je přesměrován do samostatného okna shellu.

Například: Položka v menu "PowerShell "→ " SVN " v kontextovém menu je vytvořena pro 'SVN'. $ file v tomto kontextu znamená soubor vybraný v prohlížeči File Explorer, $ mpath vybrané soubory nebo adresáře (viz oddíl 3.2 na straně 54).

```
Add;$interpreter add $mpaths;;;
```

Tento a každý další příkaz otevře submenu, v tomto případě tzv. "Extensions (Rozšíření)" → "SVN" → "Add ". Kontextové menu bude rozšířeno. Kliknutím na příkaz v kontextové nabídce bude SVN příkazu Přidat proces vybrané soubory / adresáře.

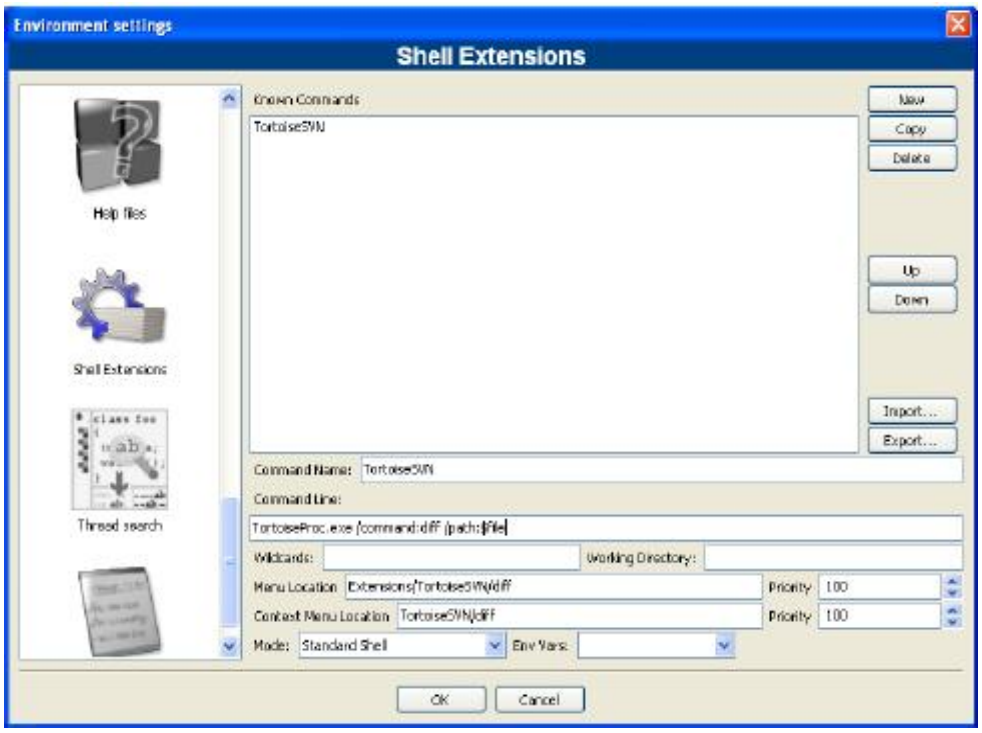
Tortoise SVN je rozšířený SVN program integrovaný do průzkumníka. Program TortoiseProc.exe z TortoiseSVN je možné spustit z příkazové řádky a zobrazený dialog sbírá vstup od uživatele. Takže můžete provést příkazy, které jsou k dispozici v kontextovém menu v Průzkumníku i v příkazovém řádku. Proto je možné integrovat také rozšíření prostředí v CodeBlocks. Například příkaz

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

bude rozdíl vybraného souboru v CodeBlocks File Explorer se základním SVN. Viz obr. 2.9 na straně 37, jak začlenit tento příkaz.

Poznámka:

Pro soubory které jsou pod kontrolou SVN a jsou-li aktivní, File Explorer zobrazuje překryvné ikony přes menu "Zobrazit" → "SVN dekoratéry".



Například:

Můžete použít Průzkumníka porovnání souborů nebo adresářů. Postupujte podle následujících kroků:

1. Přidejte název přes menu "Settings" → "Environment" → "PowerShell. To je vidět jako položku v menu překladače z kontextového menu.

2. Vyberte absolutní cestu spustitelného souboru Diff (např. kdiff3). Program je přístupný pomocí proměnné $ interpreter.

3. Přidejte parametry překladače.

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

Tento příkaz se provede pomocí vybraných souborů nebo adresáře jako parametr.Výběr je dostupný přes proměnnou $ mpaths. Jedná se o jednoduchý způsob, jak diferencovat soubory nebo adresáře.

Poznámka:

Plug-in podporuje použití přípon shell v proměnných CodeBlocks.

`$interpreter`	Volání spustitelného souboru
`$fname`	Název souboru bez přípony
`$fext`	Přípona vybraného souboru

`$file`	Název souboru.
`$relfile`	Název souboru bez info o cestě.
`$dir`	Název vybraného adresáře.
`$reldir`	Název adresáře bez info o cestě.
`$path`	Úplná cesta.
`$relpath`	Relativní cesta k souboru nebo adresáři.
`$mpaths`	Seznam aktuálních vybraných souborů nebo adresářů.
`$inputstr{<msg>}`	Řetězec, který je zapsán v okně zprávy.
`$parentdir`	Nadřazený adresář (../).

> Poznámka:
> Položky rozšíření shellu jsou k dispozici také jako lokální nabídky editoru v CodeBlocks.

2.8 Procházení sledováním

Browse Tracker je plug-in, který v CodeBlocks pomáhá při navigaci mezi naposledy otevřenými soubory. Seznam posledních souborů je uložen v historii. Pomocí menu "View" → "Browse Tracker" → "Clear all" vymaže historii.

V okně "Browsed Tabs" se můžete pohybovat mezi položkami naposled otevřených souborů pomocí nabídky z menu "View" → "Browse Tracker" → "Backward Ed (zpět) / Forward Ed (vpřed)" nebo pomocí klávesových zkratek Alt-Left/Alt-Right. Nabídka Browser Tracker je dostupná rovněž jako kontextové menu. Markery (značky) jsou uloženy v souboru rozložení <projectName>. bmarks.

Společný postup při vývoji software je boj se sadou funkcí které jsou realizovány v různých souborech. Plug-in BrowseTracks vám pomůže tento problém vyřešit tím, že ukáže vám pořadí, ve kterém byly soubory vybrány. Pak můžete pohodlně procházet volané funkce.

Plug-in umožňuje i prohlížení značek (markerů, ukazovatelů) v rámci každého souboru v editoru CodeBlocks. Na pozici kurzoru v každém souboru si pro zapamatování můžete nastavit značky v nabídce "View" → "Browse Tracker" → "Set BrowseMarks", nebo výběrem řádky pomocí levého tlačítka myši. Marker ... je zobrazen na levém okraji. Pomocí menu "View" → "Browse Tracker" → "Prev Mark / Next Mark , nebo klávesovou zkratkou Alt-up/Alt-down se v souboru můžete pohybovat pomocí značek. Pokud se v souboru chcete nechat navigovat mezi značkami seřazenými podle čísla řádku pak si vyberte menu "View" → "Browse Tracker" → "Sort BrowseMark".

Při "Clear BrowseMark" je na zvoleném řádku marker (značka, ukazovatel) odstraněn. Je-li marker nastaven na řádku: přidržení levého tlačítka myši na 1 / 4 sekundy při současně stisknuté klávese Ctrl smaže ukazatel pro tento řádek. Přes menu "Clear All BrowseMarks" nebo Ctrl + levým tlačítkem myši na jakémkoliv neznačeném řádku se v rámci souboru markery obnoví.

Konfiguraci Plug-inu lze provést přes menu "Settings" → "Editor" → "Browse Tracker".

Mark style - značky prohlížení jsou zobrazeny standardne jako . . . v rámci rozpětí. Při nastavení "Book Marks" budou zobrazeny jako záložky s modrými šipkami na okrajích. Při skrytí je potlačeno zobrazování značek Procházet.

Toggle Browse Mark key - značky lze nastavit nebo odstranit buď kliknutím levým tlačítkem myši nebo kliknutím při stisknuté klávesy CTRL.

Toggle Delay - doba (pro prodlevu) držení levého tlačítka myši pro vstup do režimu Browse Marker.

Clear All BrowseMarks - vymazat všechny značky - držte klávesu Ctrl buď jednoduchým, nebo dvojklikem na levé tlačítko myši.

Konfigurace pluginu je uložena v adresáři dat aplikace v souboru default.conf. Pokud v CodeBlocks použijete funkci osobnosti, konfigurace se načte ze souboru <personality>. conf.

2.9 Podpora SVN

Podpora systému pro správu verzí SVN je začleněn jako součást CodeBlocks - plugin TortoiseSVN. Přes menu → "TortoiseSVN" → "Plugin settings" můžete v záložce "Integration" nastavit dostupné příkazy SVN.

Menu integration - přidá vstup TortoiseSVN - s různými nabídkami nastavení.

Project manger - aktivuje příkazy TortoiseSVN v kontextovém menu manažera řízení projektu.

Editor - aktivuje příkazy TortoiseSVN v kontextové nabídce v editoru.

V konfiguraci pluginu můžete nastavit, které příkazy svn jsou dostupné prostřednictvím menu nebo z kontextového menu. Záložka integrace nabízí položky "Edit main menu (Upravit hlavní menu)" a "Edit pop-up menu (Upravit rozbalovací menu) " pro konfiguraci těchto příkazů.

> Poznámka:
> File Explorer v CodeBlocks používá různé ikony překrývání pro označení stavu svn. Tyto příkazy jsou zahrnuty v kontextové nabídce TortoiseSVN.

2.10 Vyhledávač knihoven

Pokud chcete v aplikaci použít některé knihovny, budete muset svůj projekt k jejich používání nastavit. Proces takové konfigurace může být obtížný a nepříjemný, protože každá knihovna může využívat vlastní možnosti systému. Dalším problémem je, že nastavení se liší v závislosti na platformách, což má za následek neslučitelnost mezi projekty Unix a Windows.

LibFinder nabízí dvě hlavní funkce:

- Vyhledávání knihoven nainstalovaných v systému.

- Vkládání knihovny do projektu jen s několika kliknutími myší pro vypracování projektu nezávislého na platformě.

2.10.1 Vyhledávání knihoven

Vyhledávání knihoven je k dispozici v rámci menu "Plugins" → menu "Library Finder". Jeho smyslem je odhalit knihovny nainstalované ve vašem systému a ukládání výsledků do databáze LibFinder je (Všimněte si, že tyto výsledky nejsou zapsány do souborů projektu CodeBlocks). Hledání začíná dialogem, kde můžete zadat nastavení adresářů s nainstalovanými knihovnami. LibFinder bude skenovat rekurzivně (zpětně), takže pokud si nejste jisti, můžete si vybrat některý z obecných adresářů. Můžete dokonce zadat celé disky - v takovém případě bude proces vyhledávání trvat delší dobu, ale zato může odhalit více knihoven (viz obr. 2.10 na str. 40).

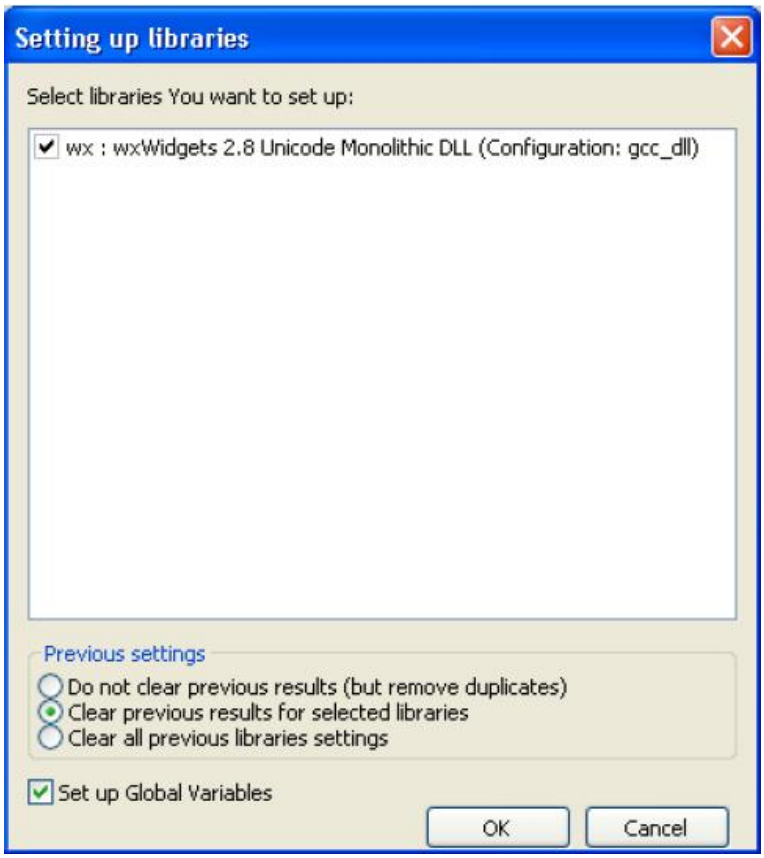
Obrázek 2.10: Seznam adresářů

Když LibFinder skenuje knihovny, používá zvláštní pravidla pro detekci přítomnosti knihovny. Každá sada pravidel se nachází v souboru XML. V současné době lze pomocí LibFinder vyhledávat knihovny wxWidgets 2.6/2.8, CodeBlocks SDK a GLFW - seznam bude v budoucnu rozšířen.

> Poznámka:
> Chcete-li získat více informací o tom, jak přidat podporu knihovny do LibFinder, přečtěte si src/plugins/contrib/lib_finder/lib_finder/readme.txt ve zdrojích CodeBlocks.

Po dokončení skenování ukazuje LibFinder výsledky (viz obr. 2.11 na str. 41).

V seznamu byste měli kontrolovat knihovny, které by měly být uloženy LibFinderem do databáze. Všimněte si, že každá knihovna může mít více než jedno platné konfigurační nastavení a přidání dodatečným nastavením je pravděpodobnější, že budou použity při sestavení projektů.



Obrázek 2.11: Výsledky hledání

Pod seznamem můžete vybrat, co udělat s výsledky předchozích testů:

Do not clear previous results - tato možnost funguje jako aktualizace dosavadních výsledků (přidává nové aktualizace a ty, které již existují). Tato možnost se nedoporučuje.

Second option (Clear previous results for selected libraries) - vymaže všechny výsledky pro knihovny, které jsou vybrány před přidáním nových výsledků. Toto je doporučená volba.

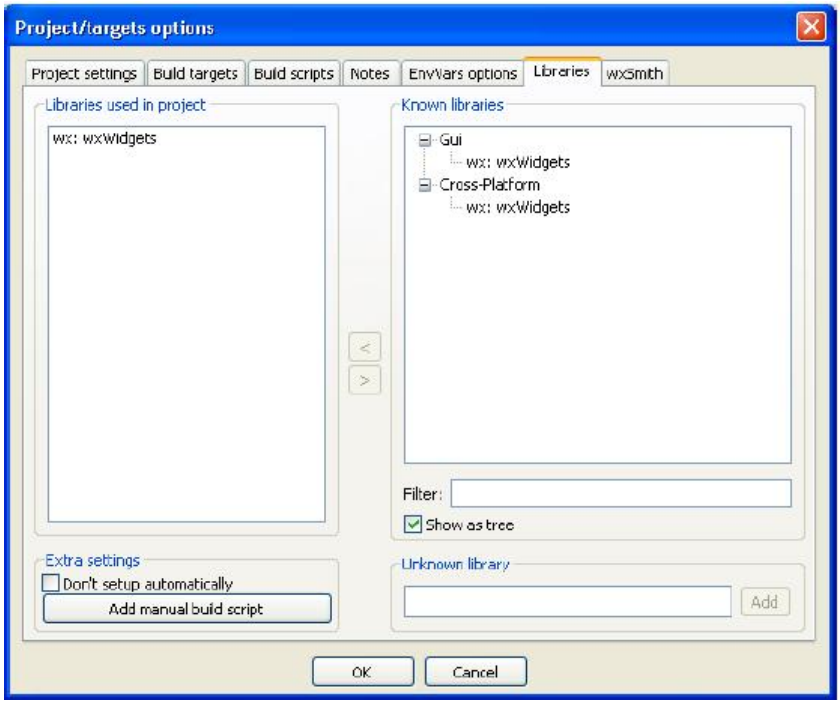
Clear all previous library settings - když zvolíte tuto možnost, bude před přidáním nového výsledku databáze LibFinder vymazána. To je užitečné v případě, že chcete vyčistit některé neplatné databáze LibFinderu.

Další možností v tomto dialogu je "Set up Global Variable (Nastavení globálních proměnných) ". Po zaškrtnutí této volby se LibFinder pokusí automaticky konfigurovat globální proměnné, které jsou také použity k jednání s knihovnami.

Máte-li v systému nainstalován pkg-config (automaticky je nainstalován na většině verzí linux), tak LibFinder poskytne knihovny také z tohoto nástroje. Není pro ně potřeba provádět žádné zvláštní skenování, automaticky jsou načteny při startu CodeBlocks.

2.10.2 Vkládání knihoven do projektů

LibFinder přidává do Project Properties (Vlastností projektu) další záložku "Libraries". Tato záložka ukazuje knihovny používané v projektu a libs známých LibFinder. Chcete-li přidat knihovnu do svého projektu, vyberte ji v pravém podokně a klepněte na tlačítko < . Chcete-li odstranit knihovnu z projektu, vyberte ji v levé části okna a klikněte na tlačítko > (viz obrázek 2.12 na straně 42).



Obrázek 2.12: Konfigurace projektu

Dostupné knihovny můžete filtrovat pomocí checkboxu vyhledávacího filtru "Show as Tree (Zobrazit jako strom)", který umožňuje přepínat mezi kategorizovaným a nekategorizovaným pohledem.

Pokud chcete přidat knihovnu, která v databázi LibFinder není k dispozici, můžete použít pole "Unknown library (neznámá knihovna)". Všimněte si, že byste měli zadat zkratku kódu knihovny (shortcode - což obvykle odpovídá názvu globální proměnné) nebo název knihovny v pkg-config. Seznam navrhovaných zkratek kódu (shortcodes) naleznete na globální proměnné. Využití této volby se doporučuje pouze při přípravě projektu, který bude postaven na jiných strojích, kde taková knihovna existuje a zda je správně detekována LibFinder. Máte přístup do globální proměnné CodeBlocks jako:

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

Volba zaškrtávacího políčka "Don't setup automatically (nepoužívejte instalační program automaticky)" bude informovat LibFinder, že při vypracování tohoto projektu by neměl přidávat knihovny automaticky. V takovém případě může uplatnit LibFinder ze skriptu vytvořit. Příkladem takového scénáře je generování a přidání do projektu stiskem tlačítka "Add manual build script (přidat ručně k sestavení skriptu)".

2.10.3 Použití LibFinderu a projekty vytvořené průvodci

Průvodci (Wizards) vytváří projekty, které nepoužívají LibFinder. Pro integraci s tímto pluginem budete muset ručně aktualizovat možnosti sestavení projektu. To lze snadno dosáhnout

odstraněním všech specifických nastavení knihovny a přidáním knihovny prostřednictvím záložky "Libraries" ve vlastnostech projektu.

Takový projekt se stane multiplatformní. Dokud používá knihovny, které jsou definovány v databázi LibFinder, bude projekt dle možnosti sestavení (project build's options) automaticky aktualizován tak, aby odpovídal specifikaci nastavení knihovny pro danou platformu.

2.11 AutoVersioning

Plug-in zvyšuje verzi a číslo sestavení aplikace pokaždé, když došlo ke změně a ukládá je do `version.h`, se snadno použitelnou deklarací proměnných. Má rovněž funkci pro provedení změn ve smyslu stylu SVN, editor verze programu, změny generátoru log a další...

2.11.1 Úvod

Myšlenka doplňku pro AutoVersioning vznikla během vývoje pre-alfa software, který vyžaduje informace o verzi a stavu programu. Bylo obsazeno kódování, aniž by se zaznamenával čas a číslo verze, a tak se rozhodl vytvořit plugin, který by mohl dělat svou práci s co nejmenšími zásahy.

2.11.2 Vlastnosti

Zde je seznam vlastností poskytovaných pluginem:

- Podpora C a C++.
- Automaticky generuje verze proměnných přírůstkem.
- Editor stavu software
- Integrovaný plán editor pro změnu chování automatické inkrementace verze hodnot.
- Datové deklarace jako den, měsíc a rok.
- Verze stylu Ubuntu.
- kontrola revize Svn.
- Změna generátoru Log
- Pracuje ve Windows a Linux.

2.11.3 Používání

Stačí přejít do menu "Project" → "Autoversioning". Zobrazí se okno Pop-up jako to následující:

Obrázek 2.13: Konfigurace Autoversioningu projektu

Pokud ve zprávě o konfiguraci projektu zmáčknete Ano, otevře se dialogové okno, které vám umožní nastavit informace o verzi projektu.

Po konfiguraci vašeho projektu pro automatizaci verzí bude nastavení, které jste zadali v konfiguračním dialogu uloženo v souboru projektu a bude vytvořen soubor version.h. Nyní pokaždé, když narazí na "Project" → "Autoversioning v menu nastavení popup dialog, nechte upravit verzi svého projektu a verzí souvisejících nastavení, pokud nechcete uložit nové změny provedené pluginem v souboru projektu.

2.11.4 Dialog notebook tabs

2.11.4.1 Hodnota Verze

Zde stačí zadat hodnotu odpovídající verze, nebo nechat vaše přírůstky verzí pluginem (viz obr. 2.14 na str. 45).

Major - přírůstky o 1, když dílčí verze dosáhne svého maxima.

Minor - přírůstky o 1, pokud číslo sestavení projde nastavenou časovou hranici, hodnota je nastavena na 0, když dosáhne své maximální hodnoty.

Build number (i jeho Release) - se zvýší o 1 pokaždé, když se zvýší číslo verze.

Revision - když se náhodou upraví a sestaví verze.

2.11.4.2 Status

Některá pole sledují stav software se seznamem předdefinovaných hodnot pro pohodlí (viz obrázek 2.15 na straně 45).

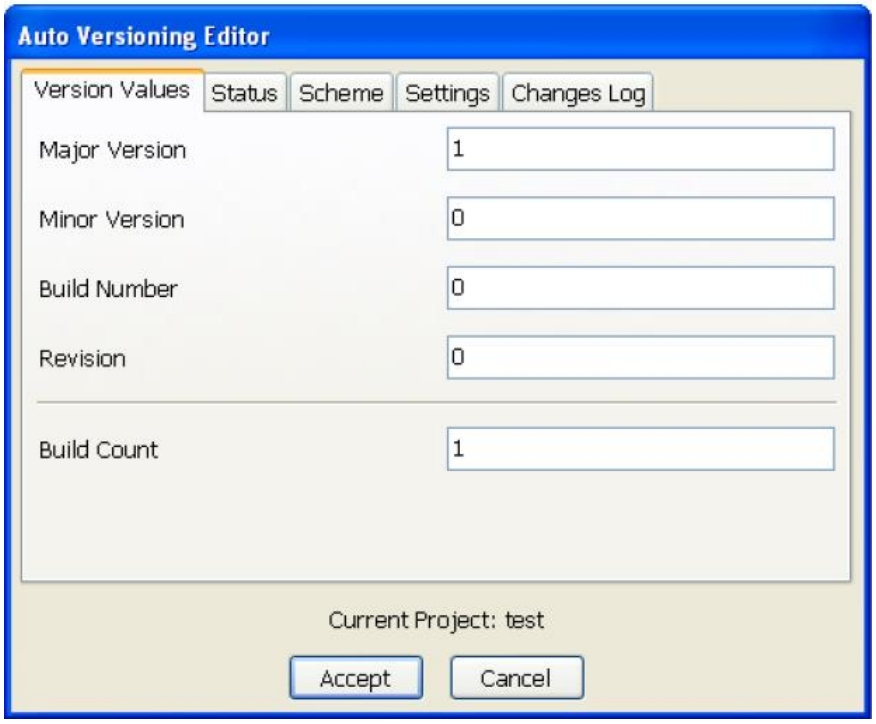
Software Status - typickým příkladem je třeba v1.0 Alpha

Abbreviation (Zkratka) - stejné jako stav software, ale takto: v1.0a

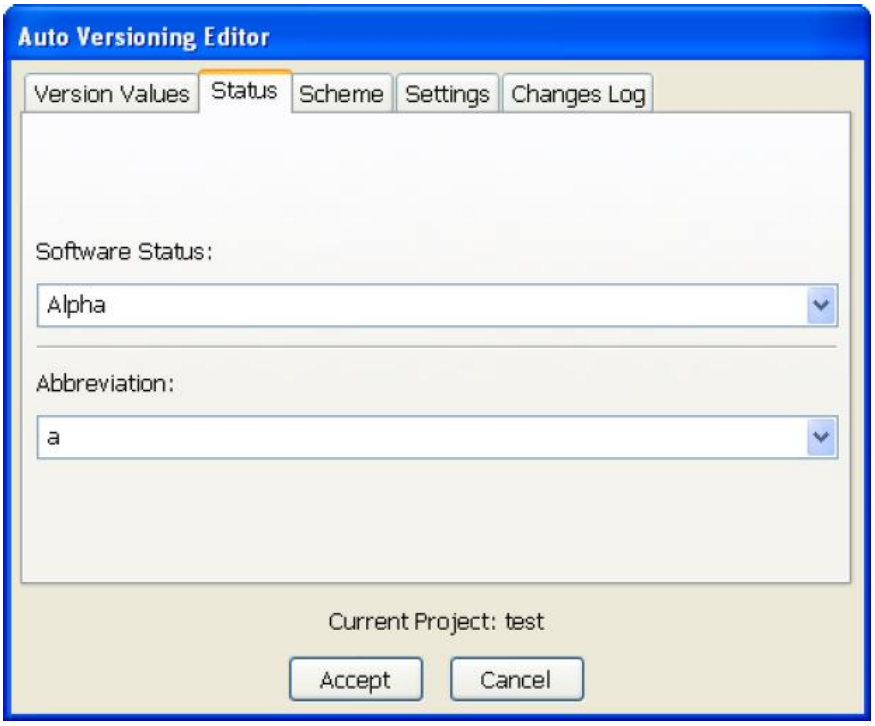
2.11.4.3 Scheme - systém

Umožňuje upravit, jak se bude zvyšovat o plugin verze hodnoty (viz obr. 2.16 na str. 46).

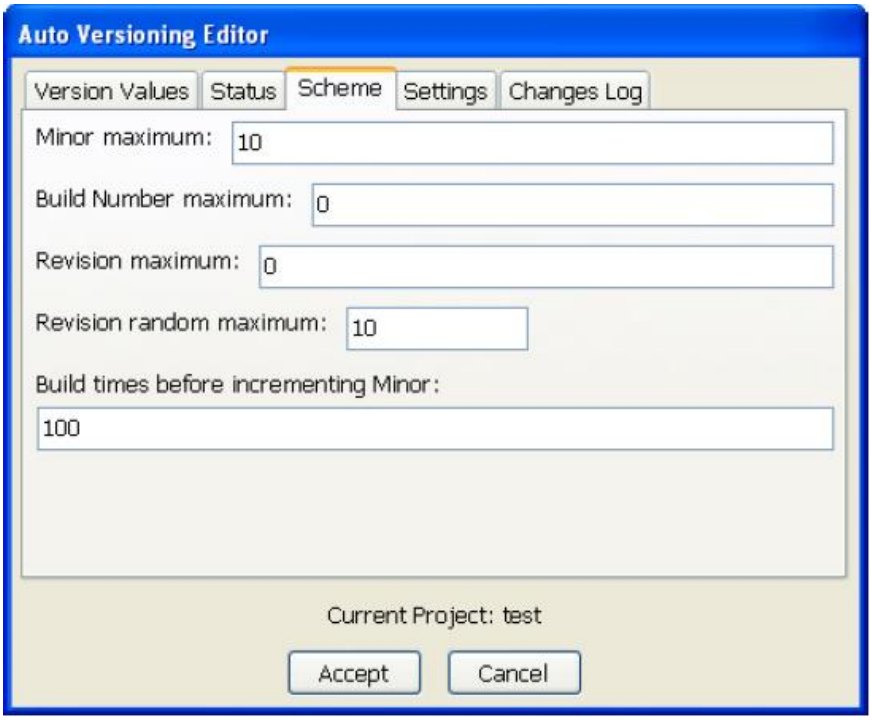
Minor maximum - maximální počet, který Menší hodnota může dosáhnout poté, co je této hodnoty dosaženo Major se zvýší o 1 a dalším projektem sestavený Minor je nastaven na 0.



Obrázek 2.14: Nastavení hodnoty verze



Obrázek 2.15: Nastavení stavu Auto Versioning



2.16: Schéma (systém) auto Versioning.

Build Number maximum - jakmile je dosaženo hodnoty, je sestavení dalším projektem nastaveno na 0. Dejte 0 pro neomezené.

Revision maximum - stejně jako maximální číslo sestavení. Dejte 0 pro neomezené.

Revision random maximum - k revizi přírůstků podle náhodných čísel, pro které se rozhodnete. Když zde zadáte 1, revize zřejmě bude zvyšovat o 1.

Build times befor incrementing Minor - po úspěšných změnách kódu a zpracování historie sestavení se bude zvyšovat, a když dosáhne této hodnoty Minor se bude zvyšovat.

2.11.4.4 Nastavení

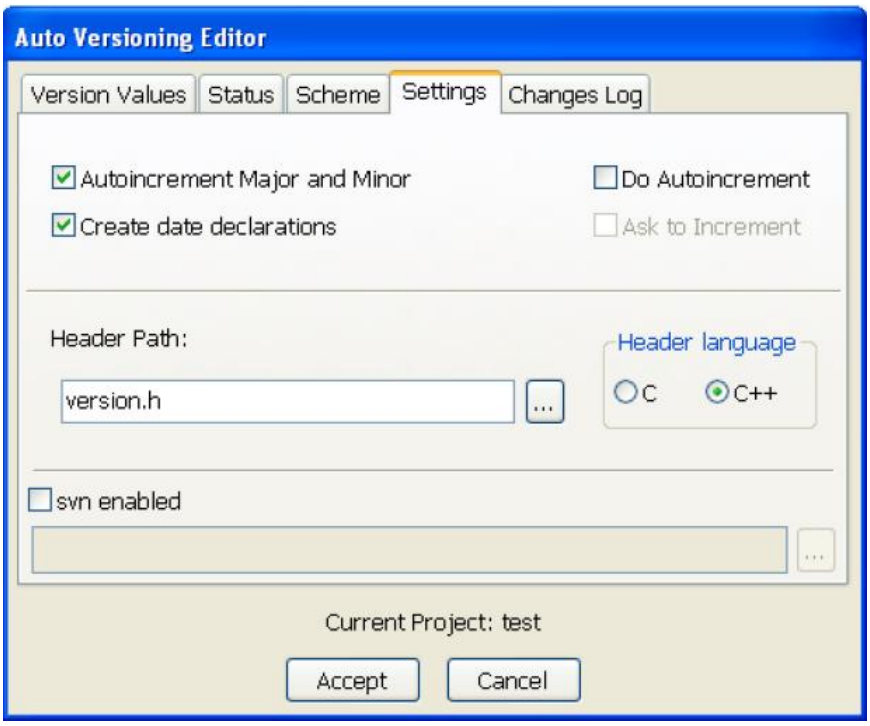
Zde si můžete nastavit některá nastavení chování autoverzí (viz obr. 2.17 na str. 47).

Autoincrement Major and Minor - umožňuje pluginu používat tento režim po krocích. Pokud není označen, bude zvyšovat pouze číslo sestavení a přírůstek revize.

Create date declarations - vytvoření datové položky v souboru version.h a verzi stylu ubuntu.

Do Auto Increment - při změnách automaticky mění přírůstek pluginu, k inkrementaci dojde před kompilací.

Header Language - výběr verze výstupního jazyka version.h



Obrázek 2.17: Nastavení Auto Versioning.

Ask to increment - Je-li označeno, zeptá se před kompilací (pokud byly provedeny), zda zvýšit číslo verze.

svn enabled - hledání revize a data svn v aktuální složce a vytváří správné položky ve version.h.

2.11.4.5 Změna Log

Umožňuje zadávat všechny změny provedené v rámci projektu pro vytvoření souboru ChangesLog.txt (viz obrázek 2.18 na straně 48).

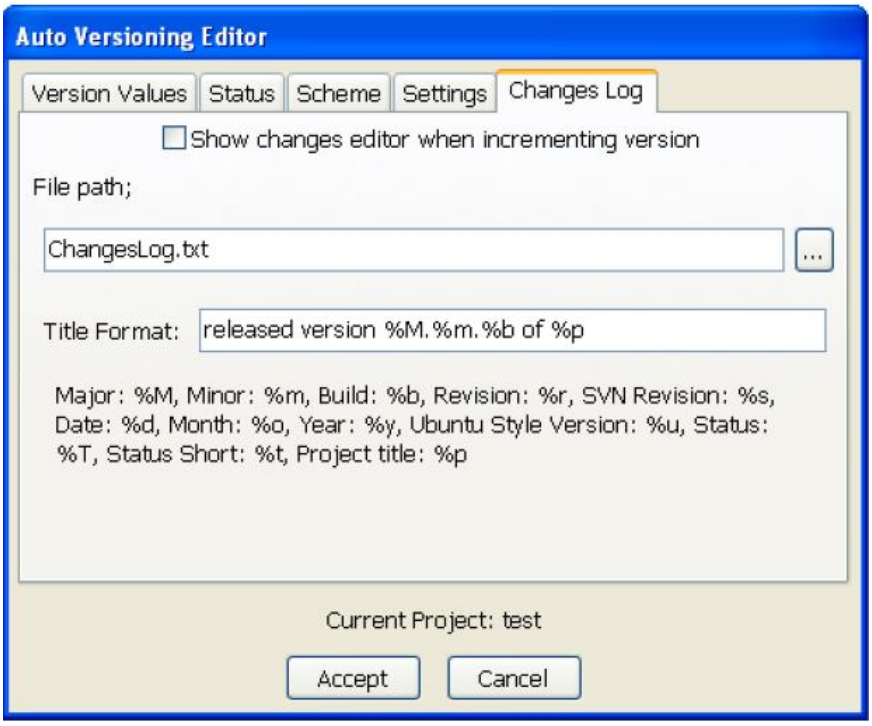
Show changes editor when incrementing version - ukáže na změny protokolu editor při povyšování verze.

Title Format - formát titulní tabulky se seznamem předdefinovaných hodnot.

2.11.5 Vložení kódu

Chcete-li používat proměnné generované pluginem, postačí # include <version.h>. Příklad kódu bude vypadat následovně:

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


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



Obrázek 2.18: Changelog of Autoversioning

2.11.5.1 Výstup version.h

Vygeneruje hlavičkový soubor. Zde je příklad obsahu souboru v režimu 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
```

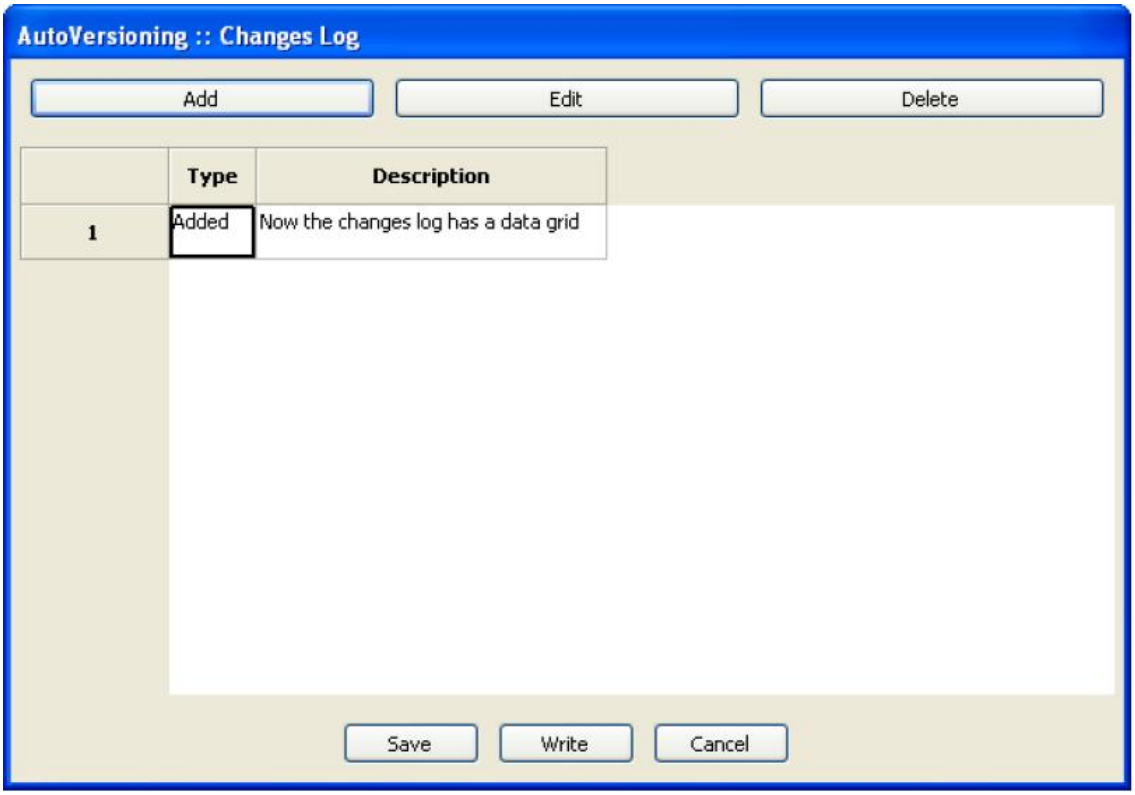
```
      #define RC_FILEVERSION_STRING "0, 10, 1086, 6349\0"
      static const char FULLVERSION_STRING[] = "0.10.1086.6349";

}
#endif //VERSION_h
```

V režimu C je stejný jako C + +, ale bez názvů:

```
#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 Změna zdroje log

Tento dialog je přístupný z menu "Project" → "Changes log". Také v případě kontrolování Show changes editor (Zobrazit změny editoru) při zvyšování verze na změny v nastavení log, otevře se okno, abyste mohli vstoupit do seznamu změn po změně zdrojů projektu nebo inkrementace událostí (viz Obrázek 2.19 na straně 50).



Obrázek 2.19: Změny v projektu

2.11.6.1 Přehled tlačítek:

Add - přidá řádek do datové tabulky (mřížky).

Edit - umožňuje úpravu vybraných buněk.

Delete - odstraní aktuální řádek v mřížce dat

Save - Ukládá aktuální data do dočasného souboru (changes.tmp) pro pozdější zpracování změn do souboru log.

Write - zpracovávat změny v mřížce dat souboru protokolu.

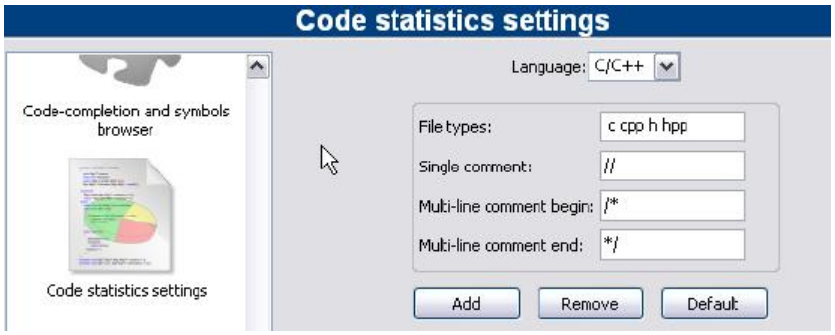
Cancel - zavře pouze okno bez přijetí jakéhokoliv opatření.

Zde je příklad výstupu generovaného pluginem pro ChangesLog.txt souboru:

```
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 Statistika kódu



Obrázek 2.20: Konfigurace statistiky kódu

Na základě záznamů v konfiguraci masky, tento jednoduchý plugin detekuje proporce komentářů kódu a prázdných řádků v projektu. Hodnocení se volá přes příkazové menu "Plugins" → "Code Statistics".

2.13 Vyhledání dostupného zdrojového kódu

Tento plugin umožňuje zvolit název v editoru a hledat tento termín pomocí kontextového menu "Search at Koders v databázi [→ Koders]. Tento dialog nabízí další možnosti pro filtrování programových jazyků a licencí.

Tato databáze hledání vám pomůže najít zdrojový kód pocházející od jiných světových projektů konsorcií, vysokých škol a organizací jako je Apache, Mozilla, Novell Forge, SourceForge a mnoho dalších, které mohou být znovu použity, aniž byste museli pokaždé "znovu vynalézat kolo". Dbejte prosím na licenci zdrojového kódu v každém individuálním případě.

2.14 Profilovač kódu

Jednoduché grafické rozhraní pro GNU GProf Profiler.

2.15 Tabulka symbolů

Tento plugin umožňuje vyhledávat objekty a symboly v knihovnách. Možnosti a cesty pro příkazový řádek nm programu jsou definovány na záložce Možnosti.

Obrázek 2.21: Konfigurace tabulky symbolů

Kliknutím na "Search (hledat)" statistiky hledání, jsou výsledky NM programu zobrazeny ve zvláštním, samostatném okně nazvaném " SymTabs Result (výsledek)". Názvy objektů nebo knihoven, které obsahují symbol jsou uvedeny pod názvem "NMs Output“.

3 Variable Expansion - rozšíření proměnných

CodeBlocks rozlišuje mezi několika typy proměnných. Tyto typy slouží ke konfiguraci prostředí pro vytváření programu, a zároveň zlepšují udržovatelnost a přenositelnost. Přístup k proměnným CodeBlocks je dosaženo pomocí $ <name>.

Envrionment Variable (Proměnné prostředí) - jsou stanoveny při spuštění CodeBlocks. Mohou měnit systémové proměnné prostředí, jako PATH. To může být užitečné v případech, kdy je pro tvorbu projektů nezbytné nedefinovat prostředí. Nastavení pro proměnné prostředí v CodeBlocks jsou: 'Settings' → 'Environment' → 'Environment Variables'

Builtin Variables (Vestavěné Proměnné) jsou předdefinovány v CodeBlocks, a jsou přístupné přes jejich názvy (podrobnosti viz bod 3.2 na straně 54).

Command Macros (Příkaz Makra) - tento typ proměnných se používá pro řízení procesu sestavení. Pro další informace viz oddíl 3.4 na straně 58.

Custom Variables (Vlastní proměnné) - jsou uživatelem definované proměnné, které mohou být uvedeny v možnostech sestavení projektu. Zde si můžete například definovat derivát jako variabilní MCU a přiřadit mu odpovídající hodnotu. Pak nastavte volba kompilátoru-mcpu = $ (MCU), a CodeBlocks automaticky nahradí obsah. Touto metodou je možné nastavení dále parametrizovaných projektů.

Global Variables (Globální proměnné) - se používají především pro vytvoření CodeBlocks ze zdroje nebo pro vývoj aplikací wxWidgets. Tyto proměnné mají velmi zvláštní význam. Na rozdíl od všech ostatních, pokud je nastavíte jako proměnné a sdílíte soubor projektu s ostatními, kteří nemají *not* nastavení tohoto CodeBlocks GV vyzve uživatele k nastavení proměnné. Jedná se o velmi jednoduchý způsob, jak zajistit pro "další developer" ví, jak snadno nastavit. CodeBlocks bude ptát na všechny obvykle nutné cesty.

3.1 Syntaxe

CodeBlocks vnitřně jedná s následujícími funkčně totožnými sekvencemi znaků v sekvencích pre-build a post-build sestavení nebo při sestavování jako s proměnnými:

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

Názvy proměnných se musí skládat z alfanumerických znaků a nejsou case-sensitive (citlivé na malá a velká písmena). Proměnné začínající jedním znakem hash (#) jsou interpretovány jako globální uživatelské proměnné (viz viz oddíl 3.7 na straně 58).

Názvy uvedené níže jsou interpretovány jako vestavěné typy.

U proměnných, které nejsou globální proměnné, uživatelské, ani vestavěné typy, bude nahrazena hodnotou poskytovanou v souboru projektu, nebo proměnnou prostředí, pokud tento selže.

> Poznámka:
> Před-projektové definice mají přednost před po-projektovými definicemi.

3.2 Seznam dostupných vestavěných

Proměnné zde uvedené jsou vestavěné proměnné CodeBlocks. Nemohou být použity v rámci zdrojových souborů.

3.2.1 Pracovní plocha CodeBlocks

$(WORKSPACE_FILENAME), $(WORKSPACE_FILE_NAME), $(WORKSPACEFILE), $(WORKSPACEFILENAME)
: Název souboru z aktuálního pracovního prostoru projektu (.workspace).

$(WORKSPACENAME), $(WORKSPACE_NAME)
: Název pracovního prostoru, který je zobrazen v záložce panelu 'Řízení projektu'.

$(WORKSPACE_DIR), $(WORKSPACE_DIRECTORY), $(WORKSPACEDIR), $(WORKSPACEDIRECTORY)
: Umístění adresáře pracovní plochy.

3.2.2 Soubory a adresáře

$(PROJECT_FILENAME), $(PROJECT_FILE_NAME), $(PROJECT_FILE), $(PROJECTFILE)
: Název souboru aktuálně sestaveného projektu.

$(PROJECT_NAME)
: Název aktuálně kompilovaného projektu.

$(PROJECT_DIR), $(PROJECTDIR), $(PROJECT_DIRECTORY)
: Společný nadřazený adresář aktuálně kompilovaného projektu.

$(ACTIVE_EDITOR_FILENAME)
: Jméno souboru otevřeného v aktuálně aktivním editoru.

$(ACTIVE_EDITOR_LINE)
: Zpět na aktuální řádek v aktivním editoru.

$(ACTIVE_EDITOR_COLUMN
: Návrat na sloupec aktuálního řádku v aktivním editoru.

$(ACTIVE_EDITOR_DIRNAME)
: Adresář, který obsahuje právě aktivní soubor (ve vztahu k společné cestě na nejvyšší úrovni).

$(ACTIVE_EDITOR_STEM)
: Základní jméno (bez přípony) právě aktivního souboru.

$(ACTIVE_EDITOR_EXT)
Přípona aktivního souboru.

$(ALL_PROJECT_FILES)
Řetězec obsahující názvy všech souborů v aktuálním projektu.

$(MAKEFILE) Název souboru makefile.

$(CODEBLOCKS), $(APP_PATH), $(APPPATH), $(APP-PATH)
Cesta k aktuálně běžící instanci CodeBlocks.

$(DATAPATH), $(DATA_PATH), $(DATA-PATH)
"Sdílený" adresář aktuálně spuštěné instance CodeBlocks.

$(PLUGINS) Pluginy aktuálně běžící instanci CodeBlocks.

$(TARGET_COMPILER_DIR)
Instalační adresář kompileru, zvaný mistrovská cesta.

3.2.3 Cíl sestavení

$(FOOBAR_OUTPUT_FILE)
Specifikace umístění výstupu do souboru.

$(FOOBAR_OUTPUT_DIR)
Specifikace umístění výstupu do adresáře.

$(FOOBAR_OUTPUT_BASENAME)
Umístění výstupního souboru se zákl. názvem (bez cesty a přípony).

$(TARGET_OUTPUT_DIR)
Aktuální umístění adresáře výstupu.

$(TARGET_OBJECT_DIR)
Aktuální umístění adresáře objektu.

$(TARGET_NAME)
Název aktuálního cíle (umístění).

$(TARGET_OUTPUT_FILE)
Výstupní soubor z aktuálního umístění.

$(TARGET_OUTPUT_BASENAME)
Výstupní soubor základního názvu (bez cesty a přípony) ze stávajícího umístění.

$(TARGET_CC), $(TARGET_CPP), $(TARGET_LD), $(TARGET_LIB)
Nástroj pro sestavení spustitelných souborů (kompilátor, linker, atd) z aktuálního umístění.

3.2.4 Jazyk a kódování

$(LANGUAGE) Jazyk systému srozumitelným způsobem.

$(ENCODING) Kódování v jednoduchém jazyce.

3.2.5 Datum a čas

$(TDAY) Aktuální datum ve formátu YYYYMMDD (například 20051228)

$(TODAY) Aktuální datum ve formátu YYYY-MM-DD (například 2005-12-28)

$(NOW) Časové razítko ve formátu YYYY-MM-DD-hh.mm (například 2005-12-28-07.15)

$(NOW_L) Časové razítko ve formátu YYYY-MM-DD-hh.mm.ss (například 2005-12-28-07.15.45)

$(WEEKDAY) Den v týdnu běžným jazykem (např. "středa")

$(TDAY_UTC), $(TODAY_UTC), $(NOW_UTC), $(NOW_L_UTC), $(WEEKDAY_UTC)
Tyto jsou stejné jako předchozí typy, ale jsou vyjádřeny porovnáním s UTC.

$(DAYCOUNT) Počet dní uplynulých od libovolně zvoleného dne nula (1. ledna 2009). Užitečné jako poslední součást verze / číslo sestavení.

3.2.6 Libovolné hodnoty

$(COIN) Tato proměnná hodí virtuální mincí (jednou za vyvolání) a vrátí buď 0 nebo 1.

$(RANDOM) 16-bitové kladné náhodné číslo (0-65535)

3.2.7 Příkazy operačního systému

Proměnná je nahrazena pomocí příkazu operačního systému.

$(CMD_CP) Příkaz Kopírovat soubory.

$(CMD_RM) Příkaz Odstranit soubory.

$(CMD_MV) Příkaz Přesunout soubory.

$(CMD_MKDIR) Příkaz Vytvořit adresář.

$(CMD_RMDIR) Příkaz Odstranit adresář.

3.2.8 Vyhodnocení podmínky

```
$if(condition){true clause}{false clause}
```

Podmínka je vyhodnocena jako pravdivá (true), jestli:

- podmínkou je neprázdný znak, jiný než 0 nebo false.
- podmínkou je neprázdná proměnná, která neobsahuje 0 nebo false.
- podmínkou je proměnná, která je vyhodnocena jako pravdivá (implicitně jako předešlá podmínka).

Podmínka je vyhodnocena jako nepravdivá (false), jestli:

- podmínka je prázdná

- podmínka má hodnotu 0 nebo nepravda
- podmínka je prázdná proměnná, nebo obsahuje hodnoty 0 nebo nepravda

> Poznámka:
> Prosím, berte na vědomí, že ani varianty syntaxe proměnné % if (...), ani $ (if) (...) jsou k sestavení podporovány.

Například:

Pokud používáte několik platforem a chcete nastavit různé parametry v závislosti na operačním systému. V následujícím kódu skriptu jsou příkazy [[ ]]vyhodnocovány a <command> bude vykonán. To by mohlo být užitečné v post-build kroku.

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

3.3 Zápis přípon

Pro maximální flexibilitu můžete vložit skripty pomocí operátorů [[ ]] jako zvláštní případ proměnné expanze. Vložené skripty mají přístup ke všem standardním funkcím které jsou k dispozici pro skripty (dočasné listy) a pracují velmi podobně jako bash backticks (s výjimkou přístupu k jmennému prostoru CodeBlocks). Jako takové nejsou skripty omezeny na výrobu textového výstupu, ale můžou také manipulovat stavem CodeBlocks (projekty, cíle, apod.).

> Poznámka:
> Manipulace se stavem CodeBlocks by měla být prováděna spíše v pre-build skriptu, než se scénářem (skriptem).

Příklad s těmito znaky:

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

Výraz v backticks vrátí seznam všech spustitelných souborů *. elf ve všech podadresářích. Výsledek tohoto výrazu je možné použít přímo objdump. Nakonec výstup je přesměrován do souboru s názvem name.dis. Tak jsou procesy automatizovány jednoduchým způsobem, aniž byste museli programovat nějaké smyčky.

Příklad použití zápisu

Text skriptu je nahrazen výstupem generovaného skriptu, nebo v případě syntaktické chyby odstraněn.

Vzhledem k tomu,že podmíněné hodnocení probíhá před rozšířením skriptů, může být hodnocení podminkou pro funkci preprocesoru. Vestavěné proměnné (a uživatelské proměnné) jsou rozšířené po skripty, takže je možné se odkazovat na proměnné ve výstupu skriptu.

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

vloží název aktivního projektu do příkazového řádku.

3.4 Příkazy Maker

`$compiler`	Přístup k názvu překladače spustitelného souboru.
`$linker`	Přístup k názvu linkeru spustitelného souboru.
`$options`	Indikátor překladače.
`$link_options`	Indikátor linkeru.
`$includes`	Vložení cesty překladače.
`$c`	Vložení cesty linkeru.
`$libs`	Knihovny linkeru.
`$file`	Zdrojový soubor (úplný název).
`$file_dir`	Adresář zdrojového souboru bez názvu a přípony souboru.
`$file_name`	Název zdrojového souboru bez cesty a přípony souboru.
`$exe_dir`	Adresář spustitelného souboru bez jména a přípony souboru.
`$exe_name`	Název spustitelného souboru bez cesty a přípony souboru.
`$exe_ext`	Přípona spustitelného souboru bez cesty a názvu souboru.
`$object`	Objektový soubor.
`$exe_output`	Výstupní spustitelný soubor.
`$objects_output_dir`	Adresář výstupního objektového souboru.

3.5 Kompilace jednoho souboru

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

3.6 Sestavení objektového souboru do spustitelného

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

3.7 Globální proměnné kompilátoru

3.8 Synopse - osnova

Pracujete jako vývojář na projektu, který se spoléhá na knihovny třetí strany. To zahrnuje spoustu zbytečně se opakujících úkolů, jako je například nastavení budování proměnných v závislosti na uspořádání místního systému souborů. V případě projektových souborů je třeba dbát na to, aby se zabránilo náhodnému zavinění místní úpravě kopie. Pokud si člověk nedává pozor, tak se to může stát docela snadno například po změně stavu příznaků při sestavení zprávy. Koncepce globálních proměnných kompilátoru je nové, unikátní řešení pro CodeBlocks, které řeší tento problém. Globální proměnné

překladače vám umožní vytvořit projekt jednou, s libovolným počtem vývojářů za pomoci libovolného množství různých rozložení systémových souborů, který je schopen sestavit a rozvíjet tento projekt. Žádné místní (lokální) uspořádání informací není nutné změnit více než jednou.

3.9 Názvy a členy

Globální proměnné v kompilátoru CodeBlocks jsou diskriminovány před proměnnými per-projectu počátečním znakem hash (#). Globální proměnné kompilátoru jsou strukturovány, každá proměnná se skládá z názvu a volitelného členu. Jména jsou volně definovatelné, zatímco někteří členové jsou zabudováni do IDE. I když si můžete vybrat nějaký název proměnné, v zásadě je vhodné vybrat známý identifikátor pro společný balíček. A taky aby se snížilo množství informací které uživatel potřebuje na minimum. Tým CodeBlocks obsahuje seznam doporučených proměnných známých balíčků.

Základní členy řeší o stejné hodnotě jakou používá název proměnné bez členu (alias).

Členy jsou i lib jsou ve výchozím nastavení aliasu pro základní / include a základ / lib, resp. Nicméně, může uživatel předefinovat je-li vhodné jiné nastavení.

Obecně se doporučuje použít syntaxi $ (# variable.include) místo $ (# variable) / include, protože poskytuje větší flexibilitu a jinak je funkčně naprosto totožná (podrobnosti viz pododdíl 3.12.1 na str. 62 a na obr. 3.1 str. 60).

Členové cflags a lflags (příznaky) jsou ve výchozím nastavení prázdné a poskytují možnost používat je pro plnění stejně konzistentního souboru kompilátor / linker flags pro všechny verze na jednom stroji (PC). CodeBlocks umožňuje definovat vlastní proměnné členy, kromě těch vestavěných.

3.10 Omezení

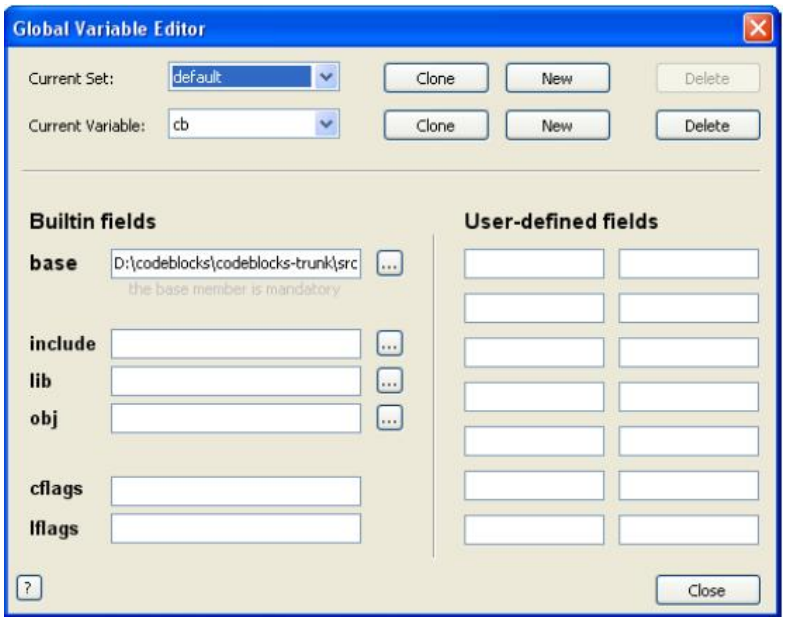
- Při nastavení nesmí být názvy globálních proměnných překladače prázdné, nesmí obsahovat mezery, musí začínat písmenem a musí sestávat z alfanumerických znaků. Azbuka nebo čínské znaky nejsou alfanumerickými znaky. Pokud jsou v CodeBlocks uvedeny neplatné sekvence názvů znaků, tak mohou být nahrazeny bez ptaní.

- Každá proměnná vyžaduje jeho základ má být definován. Všechno ostatní je volitelné, ale základ je naprosto povinný. Pokud nechcete definovat základ proměnné, nebude uložen (bez ohledu na to, jaké je další pole, které jste definoval).

- Nesmíte definovat vlastní člen, který má stejný název jako ty vestavěné. V současné době vlastní členy mohou přepsat ty vestavěné, ale obecně platí, že chování v tomto případě není definováno.

- Proměnné a členské hodnoty mohou obsahovat libovolné posloupnosti znaků, s výjimkou následujících třech omezení:

    - Nesmíte definovat hodnotu proměnné, která odkazuje na stejnou proměnnou nebo na některý z jeho členů.



Obrázek 3.1: Globální proměnné prostředí

- Člen nesmíte definovat hodnotou, která se odkazuje na stejný člen.

- Nesmíte definovat člen, nebo proměnnou hodnotou, která se odkazuje na stejné proměnné nebo členy přes cyklické závislosti.

CodeBlocks odhaluje ty nejviditelnější případy rekurzivní definice (ke které může dojít náhodou), ale neprovádí důkladnou analýzu všech možných zneužití. Zadáte-li nesmysl, nesmyslný bude i výsledek.

Například:

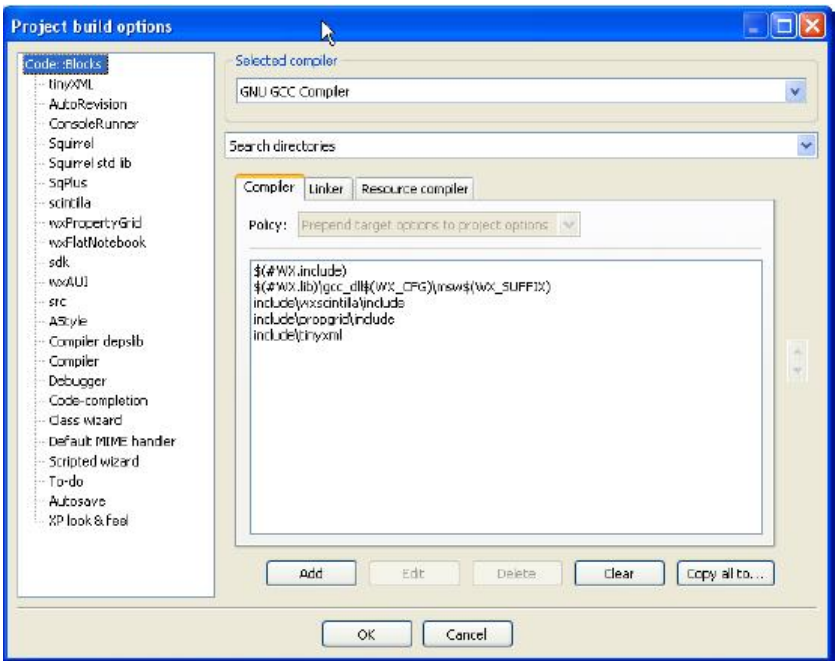
Definování wx.include jako $(#wx)/include je zbytečné, ale zcela legální. Definování wx.include $ (# wx.include)je proti pravidlům bude CodeBlocks detekováno. Definování wx.include $(#cb.lib), který je opět definován jako $ (# wx.include) vytvoří nekonečnou smyčku.

3.11 Používání globálních proměnných překladače

Vše, co musíte udělat pro použití globálních proměnných kompilátorem, je vložit je do svého projektu! Ano, je to tak snadné.

Když IDE zjistí přítomnost neznámých globálních proměnných, bude vyzváni k zadání jeho hodnoty. Tato hodnota bude uložena v nastavení, takže se nikdy nebudete muset zadávat informace dvakrát.

Pokud potřebujete změnit nebo odstranit proměnné později, můžete tak učinit v nastavení.



Obrázek 3.2: Globální proměnné

Příklad:

Snímek nahoře ukazuje globální proměnné pre-projektu. WX_SUFFIX je definován v projektu, ale WX jsou globální proměnné uživatele.

3.12 Nastavení proměnných

Někdy chcete použít různé verze stejné knihovny, nebo si vytvořit dvě větve téhož programu. I když je možné získat je spolu s globálními proměnnými kompilátoru, může to být únavné. CodeBlocks podporuje za tímto účelem sady proměnných. Soubor proměnných je nezávislý soubor proměnných, které jsou označeny názvem (jejich názvy mají stejná omezení jako názvy proměnných).

Pokud chcete přejít na jinou sadu proměnných, stačí si z nabídky vybrat jinou sadu. Různé soubory nemusí mít stejné proměnné, a stejné proměnné v různých souborech nemusí mít stejné hodnoty, nebo dokonce stejné vlastní členy.

Dalším kladem toho všeho je, že pro soubory, pokud máte tucet proměnných a chcete mít nový soubor s jednou z těchto proměnných směřující do jiného umístění, nemusíte znovu zadávat všechny údaje. Můžete jednoduše vytvořit klon aktuálního souboru, který pak bude opakovat všechny své proměnné.

Vymazání souboru také odstraní všechny proměnné v této sadě (ale ne v jiném souboru).Výchozí nastavení je vždy přítomno a nemůže být odstraněno.

3.12.1 Vlastní Členský Mini-tutorial

Jak bylo uvedeno výše, psaní $ (# var.include) a $ (# var) / include je ve výchozím nastavení přesně to samé. Tak proč byste měli chtít psát něco tak neintuitivního jako $ (# var.include)?

Podívejme se například na standardní instalaci Boost pod Windows. Obecně platí, že očekáváte funkční balíček ACME s vlastními hlavičkovými soubory pod ACME / include a jeho knihovny v rámci ACME / lib. Volitelně by mohl své hlavičky umístit do další podsložky , zvané vrchol. Takže po přidání správné cesty a volby kompilátoru a linkeru, byste očekávali # include <acme/acme.h> a link na libacme.a (nebo co to děje).

Katalog URL

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

[,!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/

[,!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/