

Baixo nível, alto rendimento

Flexibilidade é a principal característica do Eclipse. O plugin CDT adapta o IDE às linguagens C e C++, resultando num ambiente de desenvolvimento ergonômico e com ótimas funções.

por Peter Kreussel

O Eclipse é uma IDE especializada em tudo e em nada — é assim que os desenvolvedores do Eclipse descrevem a meta de seu projeto. Portanto, para ser de fato útil, há que se lançar mão de um dos inúmeros plugins. O CDT^[1] (C/C++ development tooling) adapta o Eclipse para C e C++. Os recursos oferecidos pelo plugin vão desde um diálogo pop-up para auto-completar no editor, passando pelo realce de sintaxe até um monitor dos registradores do processador, um *disassembler* e a verificação de regiões de memória.

Editores adequados à linguagem empregada facilitam a programação: o realce de sintaxe já indica erros de digitação quando o cursor estiver no local correto. Além de C e C++, o realce também entende *makefiles*.

No navegador de arquivos, os erros de sintaxe são evidenciados com facilidade: arquivos errados, bem como projetos e diretórios que recebem esses arquivos, recebem um X vermelho. Apesar do tamanho reduzido dos ícones e das subjanelas deixar bastante espaço para o editor, eles conseguem mostrar o tipo de objeto, o status do teste de sintaxe e, quando se gerencia versões, também o status da sincronização.

A função de autocompletar também herda o visual ergonômico do Eclipse: o plugin CDT marca as sugestões para completar através de ícones facilmente reconhecíveis, como classes, variáveis ou palavras-chaves da linguagem de programação (figura 1). Contudo, para o assistente de código reconhecer as classes e métodos, é necessário que ele saiba o diretório

de inclusão Eclipse. Frequentemente, mas nem sempre, os caminhos de inclusão são suficientes. Quando o usuário seleciona uma classe ou função a partir da lista pop-up, o editor mostra sua assinatura.

Na procura

A função de procura funciona através de uma indexação que ocorre em segundo plano. O indexador propriamente dito foi recentemente aprimorado, e ficou mais rápido desde a versão 3.1.1: o código-fonte inteiro do Firefox, composto por 10 mil arquivos C e C++, é lido em um computador atual em menos de 3 minutos. A função de procura faz a identificação de acordo com elementos de sintaxe, como classes, métodos, funções, variáveis e campos. Há formas de limitar a procura a um subconjunto de arquivos de projeto.

A função normal de procura e substituição de código do Eclipse está disponível no editor CDT. Porém, raramente é utilizada, pois há uma outra ferramenta de refatoração (ou *refactoring*) capaz de levar em consideração a sintaxe do arquivo. Essa alternativa, portanto, consegue renomear apenas variáveis, mesmo quando há funções com o mesmo nome.

Cris DeRaud - www.sxc.hu

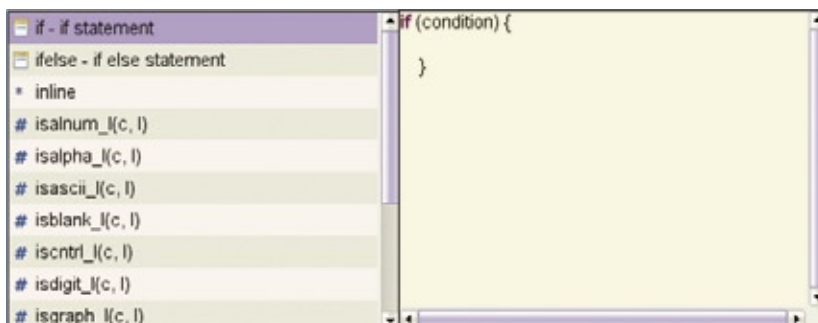


Figura 1 O recurso de autocompletar é muito eficiente, oferecendo uma lista de possibilidades e a sintaxe básica do comando.

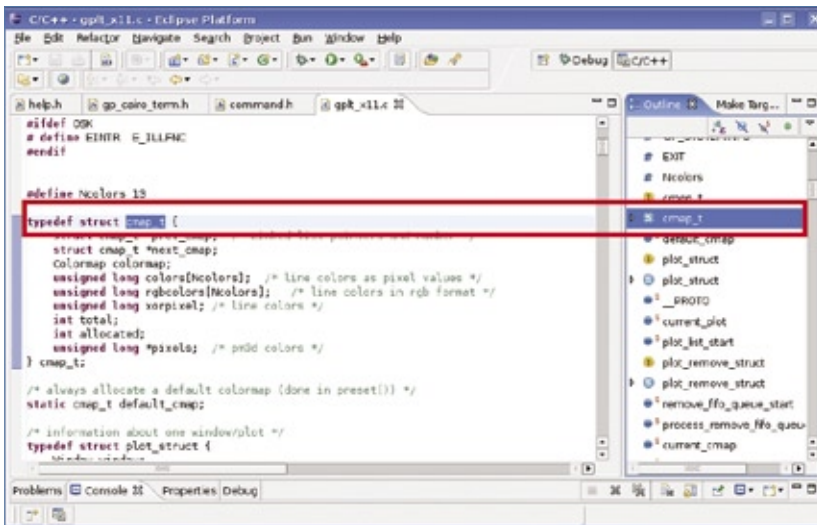


Figura 2 A aba *Outline* mostra a estrutura do código, listando classes, funções e declarações de variáveis.

Com a ajuda do veloz indexador, a ferramenta de refatoração também é eficaz em projetos grandes, como, por exemplo, no código-fonte do Firefox: renomear uma classe referida aproximadamente 350 vezes em cerca de 10 mil arquivos leva cinco minutos numa máquina Athlon 64 3700+ com 2 GB de memória.

Bem estruturado

Além da função de procura, a visão estruturada (**figura 2**) ajuda a compreensão do código-fonte: ela oferece uma visão geral sobre estruturas como `include`, declaração de variáveis e classes. Os tipos de objetos são evidenciados por símbolos. O navegador de arquivos mostra os registros da visão estruturada como subelementos dos arquivos. Uma margem colorida auxilia a reproduzir as modificações desde a última gravação.

O item *Navigate | Last Edit Location* leva o cursor à última posição visitada. Porém, essa função armazena apenas uma posição, diferentemente de um navegador web. Por último, mas não menos importante, o editor suporta a ocultação de subestruturas, como loops ou definições de funções, e ainda comenta linhas ou blocos pressionando uma tecla.

Compilação

Na instalação padrão, o Eclipse integra também o GNU Make. A plataforma de compilação do CDT emprega o princípio arquitetônico que rege todo o Eclipse: a comunicação com a ferramenta de compilação GNU acontece através de um plugin extensível. Com isso, o suporte a outros sistemas de compilação, como o *Cmake*, é facilmente implementável, e seu uso pode ser feito de dentro do próprio programa, como mostra a **figura 3**. Em caso de falhas nesse processo, o software reconhece

em qual arquivo de código-fonte ela ocorreu, assinalando-o no navegador de arquivos com uma marca de erro. Após a compilação, o Eclipse busca os arquivos binários no diretório de projeto. Assim como em arquivos de código-fonte, o navegador de arquivos mostra todos os arquivos de fonte que originaram o executável ou a biblioteca.

Em projetos C e C++ do tipo *managed* (gerenciados), o próprio software cuida da criação dos makefiles e outros arquivos necessários para as ferramentas de compilação GNU. Esses tipos de projetos são úteis nos casos de projetos importados que já possuam um Makefile. Posteriormente, eles também podem ser convertidos em projetos através do gerenciador de makefiles. Para fazer adaptações, há um editor próprio disponível.

Caça aos erros

Em C e C++, o fechamento de um programa sem informações de localização do erro pode ser muito enigmático. Nesses casos, o depurador é uma importante fonte de ajuda. Por padrão, o Eclipse se integra ao *GNU Debugger*, ou *GDB*, mas novamente é possível estender o suporte a outras alternativas. ▶

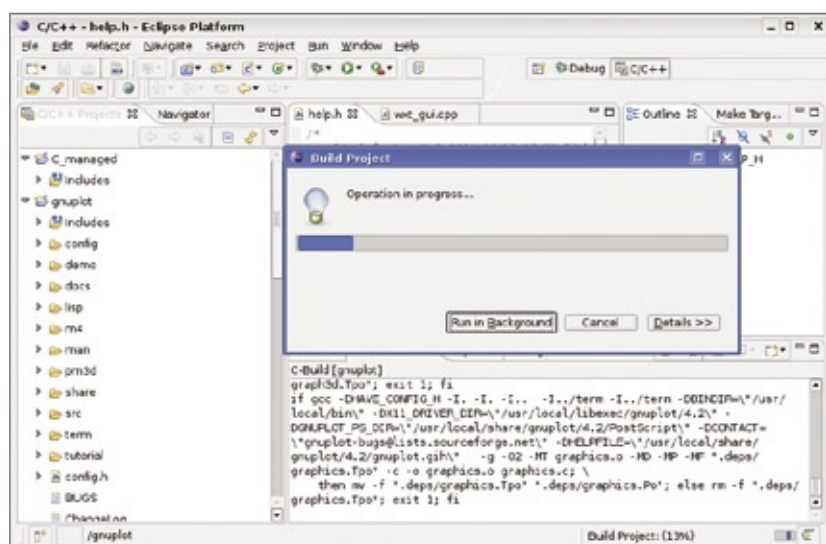


Figura 3 O CDT compila projetos de software com *make* e *GCC*. Graças à arquitetura do *Eclipse*, é fácil incluir o suporte a outras ferramentas de compilação.

O CDT possui as funções de depuração padrão: um duplo clique no botão numérico do editor de código-fonte insere um *breakpoint*, que interrompe a execução do programa. Ao exibir variáveis em programas paralisados, o Eclipse mostra os valores dessas variáveis (figura 4). Um cursor de depuração assinala no código-fonte a posição na qual o programa foi paralisado. Os *watchpoints* também são grandes aliados em loops com muitas iterações: eles não paralisam a execução do programa em determinado ponto, mas sim quando os valores das variáveis correspondem a determinadas condições.

Monitoramento completo

O CDT também suporta um modo de monitoramento passo-a-passo. Nele, o usuário decide, individualmente, se o depurador também entrará nos códigos referidos por *include*, ou se apenas os comandos do arquivo principal serão moni-

torados. Todas as vezes em que, ao ser depurado, o programa fizer uma pausa, o usuário também poderá modificar manualmente os valores das variáveis.

Como C e C++ atualmente são linguagens de nível mais baixo, o CDT estende a plataforma de depuração do Eclipse para permitir o acompanhamento dos registradores do processador e também monitorar determinadas áreas da memória, incluindo também um *disassembler*.

Melhor que sua reputação

Antigamente, o Eclipse era lento e instável. O desenvolvedor do projeto CDT Doug Schaefer, em sua palestra na Eclipsecon 2006[2], recomendou o teste de importar o código-fonte do Firefox como projeto C++, compilá-lo e processá-lo. O resultado positivo confirmou que está superada a má fama desde a versão 3.2 dessa ótima plataforma de desenvolvimento. Daí em diante, o Eclipse tem bons tempos de resposta, e seu consumo de memória

(300 MB, já incluída a máquina Java) é eficiente para os recursos apresentados por esse software. A velocidade do indexador de procura e refatoração melhora ainda mais essa impressão. O travamento de programas ficou sensivelmente mais raro: nos testes realizados para este artigo, o IDE se despediu de nós uma única vez.

Junto ao Eclipse, o plugin CDT forma uma IDE muito boa para o desenvolvimento em C e C++. Seu ambiente flexível e claro, o editor, assim como as funções de procura e refatoração, atuam perfeitamente em conjunto, são intuitivos e aumentam o rendimento no trabalho.

Porém, é uma pena que o Eclipse e o CDT abandonem o usuário na criação de interfaces gráficas: programadores que necessitam de um bom editor de interfaces gráficas precisam lançar mão de um programa externo ou usar outros IDEs no lugar do Eclipse. O *Kdevelop*[3] contém um editor desses para *widgets Qt* e o *Anjuta*[4] possui um plugin experimental para a biblioteca GTK. ■

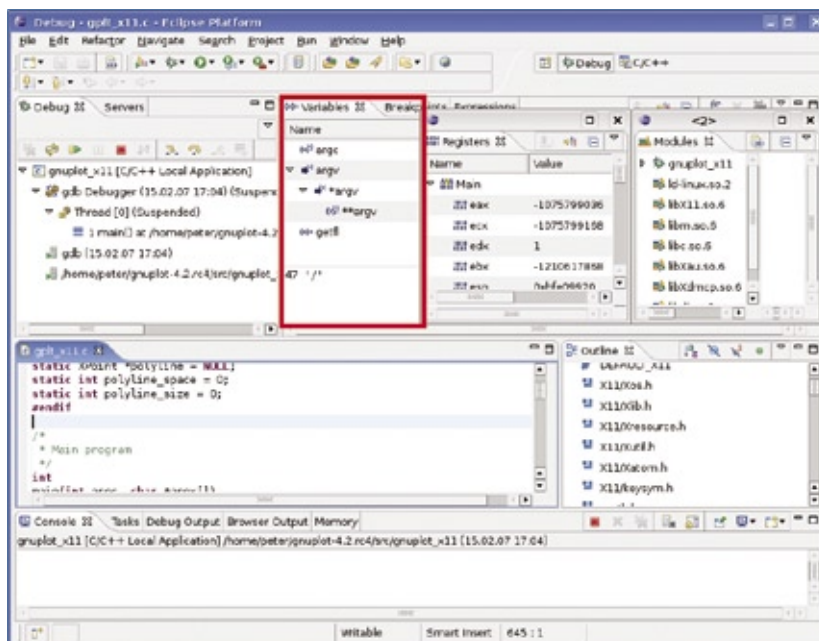


Figura 4 O depurador exibe os valores das variáveis, auxiliando a busca de falhas lógicas no programa.

Mais informações

- [1] CDT: <http://www.eclipse.org/cdt>
- [2] Palestra de Doug Schaefer sobre o CDT: http://cdt.eclipse.org/docs/CDT_DOM_Europe.ppt
- [3] KDevelop: <http://www.kdevelop.org>
- [4] Anjuta: <http://anjuta.sourceforge.net>

Sobre o autor

O Corel Draw foi o programa que capturou o interesse de **Peter Kreussel** em computadores. Peter fala com felicidade que o Inkscape e o Xara há muito tempo já ultrapassaram o Windows® e os caros programas da Adobe em sua máquina.