Tutorial de Instalação da SDL no Windows

Introdução

Este é um pequeno tutorial sobre como instalar a biblioteca SDL 2.0 em um ambiente Windows, e algumas recomendações de ferramentas de desenvolvimento.

1. TDM-GCC

Começamos obtendo um compilador. O TDM-GCC oferece uma versão atualizada dos compiladores de C e C++ da GNU Compiler Collection, permitindo que você use as features de revisões mais novas da linguagem C++. Ele também oferece versões para arquiteturas 32 e 64-bits. Usaremos a versão de 32.

http://tdm-gcc.tdragon.net/download

Execute o arquivo baixado. Recomenda-se fortemente usar o diretório padrão para a instalação (C:\TDM-GCC-32). Se quiser mudar, **não use diretórios com espaços no caminho.**

OPCIONAL: Você pode agora adicionar o caminho C:\TDM-GCC-32\bin à variável PATH do seu sistema. Isso permitirá que o gcc e o g++ sejam invocados pela linha de comando, e que seus programas encontrem DLLs colocadas nesta pasta automaticamente.

Para isso, se você estiver no Windows 7 ou Vista, acesse *Painel de Controle* > *Sistema e Segurança* > *Sistema*. No menu lateral esquerdo, escolha *Configurações Avançadas do Sistema* > *Variáveis de Ambiente*. Procure por *Path* no menu *Variáveis do Sistema*, e acrescente, separado do último caminho contido por ponto e vírgula, o caminho C:\TDM-GCC-32\bin. **Tome cuidado com essa variável; remover caminhos contidos nela pode desconfigurar programas instalados no seu computador.** Pode ser necessário reiniciar o computador para que as mudanças tenham efeito.

2. SDL

Após a instalação do MinGW, é necessário baixar a biblioteca SDL (Simple Directmedia Layer). Para isso, acesse http://libsdl.org/download-2.0.php e baixe o arquivo *SDL2-devel-2.x.x-mingw.tar.gz* na seção Development Libraries (usuários de Visual Studio devem usar exclusivamente os arquivos -*VC.zip*).

Extraia o conteúdo desse arquivo, que será uma pasta chamada SDL-2.x.x. Você quer as pastas *bin*, *include* e *lib* dentro de i686-w64-mingw32. Moveremos essas pastas para C:\SDL-2.x.x, você pode usar o caminho que quiser, mas, novamente, evite diretórios com espaço no caminho. Agora, é necessário baixar as bibliotecas complementares. São elas SDL2_image, SDL2_ttf e SDL2_mixer.

Essas bibliotecas podem ser baixadas nas suas respectivas páginas:

http://www.libsdl.org/projects/SDL_image/
http://www.libsdl.org/projects/SDL_mixer/
http://www.libsdl.org/projects/SDL_ttf/

Na página dessas bibliotecas, baixe o arquivo com o nome SDL2_[...]-devel-2.x.x-mingw.zip. Ao descompactar esses arquivos, você verá a mesma pasta com *bin*, *include* e *lib* dentro. Arraste essas pastas para C:\SDL-2.x.x, mesclando com as pastas de mesmo nome já existentes.

Daqui, a instalação da biblioteca já está mais ou menos pronta, bastando indicar as pastas para o TDM-GCC nos arquivos de projeto. No entanto, ainda é necessário reunir as DLLs necessárias para o nosso executável rodar.

Entre na pasta *lib* resultante da mescla anterior e copie todas as DLLs alí contidas para C:\SDL-2.x.x\bin. Lembre-se desta pasta: Você vai precisar copiar este conjunto de DLLs para a pasta de cada executável* que você compilar para que ele funcione.

* Note que, caso esteja usando uma IDE, ela pode exigir que as DLLs estejam no diretório setado como Working Directory, em vez de no do executável.

3. IDEs

Você pode usar a IDE que quiser, de fato, é recomendado usar uma que você tenha experiência em configurar. Se você não souber fazê-lo, seguem algumas recomendações de IDEs em que conseguimos configurar o build com SDL2.

O código no link a seguir pode ser usado para testar se a configuração está correta. Se ele não compilar, há um problema na sua instalação.

http://pastebin.com/Jgk50FB7

3a. Eclipse

Vá á ao menu *File* > *New* > *C++ Project*. Dê um nome para o projeto, por exemplo, *sdltest*. Use a toolchain MinGW GCC.

Criado o projeto, vá ao menu *Project* > *Properties*. Selecione *C/C++ Build* > *Settings*.

Em *GCC C++ Compiler* > *Preprocessor*, adicione o Defined Symbol "main=SDL_main".

Em *GCC C++ Compiler* > *Includes* adicione o Include Path C:\SDL-2.x.x\include\SDL2.

Em *GCC C++ Compiler* > *Dialect* > *Other Dialect Flags* escreva *-std=c++11*

Em *MinGW C++ Linker* > *Libraries* adicione o library search path "C:\SDL-2.x.x\lib".

Também em *MinGW C++ Linker* > *Libraries* adicione as seguintes libraries:

- MinGW32
- SDL2main
- SDL2_image
- SDL2_ttf
- SDL2_mixer
- SDL2

OBS.: A ordem dessas libraries PODE alterar/dar problemas na compilação, é aconselhado adicionar as libraries na ordem acima. Se o seu projeto não usar alguma das bibliotecas auxiliares, no entanto (image, ttf, mixer), não é necessário incluí-las na lista.

Só isso deve ser suficiente para o código de teste dado acima. Acontece que o indexador do Eclipse tem uma peculiaridade que reconhece algumas construções de C++ (como o std::unordered_map) como erro, e isso pode ser um incômodo. Para resolver isso: *Project* > *Properties* > *C/C++ General* > *Paths and Symbols* > *Symbols* > *GNU C++* > *Add*.

Crie um símbolo com nome `__GXX_EXPERIMENTAL_CXX0X__` e sem valor. Ao confirmar, o Eclipse vai pedir pra reconstruir o índice, aceite. Dê Clean e em seguida Build no projeto. O editor deve parar de acusar erros.

3b. Qt Creator

O Qt Creator é uma IDE de C++ normalmente usada com o Qt SDK (bibliotecas para fazer programas com GUI), mas suporta projetos sem Qt e é uma boa IDE para a disciplina (pessoalmente, foi a que usei). Também tem a vantagem de ser um pouco mais leve que o Eclipse, por ser escrita em C++.

Você pode baixá-lo do site http://qt-project.org/downloads . Selecione a versão community. Durante a instalação, quando for escolher os componentes, você pode marcar apenas Qt > Tools > Qt Creator, mas se espaço em disco não for um problema, você pode querer instalar Qt > Qt 5.2.1 > MinGW 32-bit. Essas são as bibliotecas do Qt. Nos não às usaremos na disciplina, mas elas acompanham o utilitário QMake, que pode ser útil. Alternativamente, você pode usar o CMake, como descrito abaixo.

O que quer que você escolha, abra o Qt Creator. Crie um projeto novo do tipo Non-Qt. Há duas opções de projeto C++, uma com o qmake, e uma com o cmake. O qmake é um gerenciador de builds próprio do Qt Creator, é mais fácil de usar com a IDE, mas menos flexível. O cmake (que você pode baixar de http://www.cmake.org/), é um gerenciador de builds muito mais poderoso e útil fora do Qt Creator. Recomendo usá-lo, se possível.

Criado o projeto, copie o código de teste para o arquivo `main.cpp`.

Agora vá no arquivo do projeto, que vai depender de qual gerenciador de builds você escolheu.

Para o qmake (arquivo .pro):

- Adicione a linha `CONFIG += c++11` no início do arquivo
- Adicione a linha `LIBPATH += C:\SDL-2.x.x\lib` no final do arquivo
- Adicione a linha `LIBS += -lMinGW32 -lSDL2main -lSDL2 -lSDL2_image -lSDL2_ttf -lSDL2_mixer` no final do arquivo
- Adicione a linha `INCLUDEPATH += C:\SDL-2.x.x\include\SDL2` no final do arquivo

Para o cmake (cmakelists.txt):

- Antes de add_executable, adicione as seguintes linhas:

```
list(APPEND CMAKE_CXX_FLAGS "-std=c++11 ${CMAKE_CXX_FLAGS}")
include_directories(C:/SDL-2.x.x/include/SDL2)
link_directories(C:/SDL-2.x.x/lib)
```

```
link_libraries(Mingw32 SDL2main SDL2_image SDL2_ttf
SDL2_mixer SDL2)
```

➢ Ao executar o CMake (na criação do projeto ou via Build > Run CMake), você pode especificar build debug ou release passando como argumento `-DCMAKE_BUILD_TYPE=debug` ou `-DCMAKE_BUILD_TYPE=release`

Se tudo estiver certo, ao dar build no projeto agora, ele deve compilar sem problemas.

Mais uma coisa: Crie o hábito de setar o Working Directory para a pasta do projeto nos seus projetos da disciplina. Será útil para não ter que copiar os recursos do jogo para a pasta do executável sempre. Na aba lateral, clique em Projects, depois em Run > Working Directory.

3c. Visual Studio

(obs: As instruções abaixo são baseadas no VS 2012 e podem variar)

Admito que os alunos que usam VS na matéria constumam ter mais experiência com ele que eu, mas segue um pequeno tutorial de como incluir/linkar a SDL num projeto. Lembre-se que a instalação da biblioteca deve ser feita com os arquivos do VC, e não do MinGW, se você escolher por usar o VS.

Crie um projeto novo com o tipo *Visual C++* > *General* > *Empty Project.* Entre nas propriedades do projeto e altere os seguintes parâmetros:

- *C / C++* > *General* > *Additional Include Directories* para *C:\SDL-2.x.x\include*
- *C / C++* > *Code Generation* > *Runtime Library* para *Multi-Threaded DLL*
- *Linker* > *General* > *Additional Library Directories* para *C:\SDL-2.x.x\lib\x86*
- *Linker* > *System* > *Subsystem* para *Console* ou *Windows**
- *Linker* > *Input* > *Additional Dependencies* para *SDL2.lib;SDL2main.lib;SDL2_image.lib;SDL2_mixer.lib;SDL2_ttf.lib*

Observe que os Include e Library Directories podem variar em número e nome de acordo com suas escolhas na instalação da SDL. Separe-os por ';' se precisar de mais de um.

Ponto importante! Se você for usar o VS como IDE para a disciplina, é obrigatório setar também:

- *C++ > Language > Disable Language Extensions* para *Yes /Za*

Os trabalhos serão corrigidos no g++, o que quer dizer que, se você fizer uso das extensões da Microsoft, seu código não compilará.

* Setar para Windows elimina a janela de console. Isso pode ser desejável numa versão final, mas o console pode ser útil para debugar.

4. Possíveis Problemas

- *“undefined reference to 'WinMain@16”*
    - Refere-se à falta de uma função main no programa. Tanto a API do Windows quanto a SDL_main usam macros que alteram o significado de ‘main’ no programa, afetando a função que você escreveu. Em sistemas 64 bit, uma solução possível é abrir a SDL_main.h, procurar a linha

        ```
        #if defined(__WIN32__)
        ```

        Por

        ```
        #if defined(__WIN__)
        ```