C++Buildre旧バージョンのプロジェクトを利用

私は、C++Builder 5 を持っている。そのプロジェクトを C++Builder Community Edition に移行したので、その留意点などについて残す。

□最新のC++Buidlerで旧バージョンのプロジェクトを利用する方法」記載の手順で、自動コンバートは出来ないので新規プロジェクトを作成し、ソースをコピーして*.cppをプロジェクトに追加していく。一部、ソース修正が必要。

1. C++Builder Community EditionでC++プロジェクトを新規に作成し、プロジェクトを保存する。 2. プロジェクトにデフォルトで作成されるユニットを削除する 3. 旧プロジェクトのユニットを新しいプロジェクトに追加する（*.cpp,*.h,*.dfm） 4. プロジェクトのメインモジュール（プロジェクト名.cppやプロジェクト名PCH.h□に旧プロジェクトのコードをコピーする

ソースの修正が必要だった点

Windows環境ではファイル名の大文字小文字は区別されないが、#includeファイル名と実際のファイル名が合致していないと警告が出るので、ファイル名の大文字小文字を合わす。あと、ソースに漢字を含んでいる場合は、全て「UTF-8□文字コードBOM付に変更□□Wzエディターで変換し上書き保存）

ヘッダファイルが見つからない

コンパイルするとヘッダファイルが見つからないエラーが発生する場合がある。エラーとなるヘッダファイルの#include行を削除する。必要なヘッダは自動的に追加される。尚、<vcl.h>と<tchar.h>は共通プリヘッダファイル「<プロジェクト名>PCH.h」に記述があるので、各ユニットファイルへの記述は無くても大丈夫。

文字列の扱いの違い

RAD Studio 2009 バージョン以降で、C++BuilderはUnicode対応となり内部の文字列の扱いが変更になっている。従来の文字列はバイト単位のchar型だったが、漢字などのマルチバイト文字を扱うワイド文字型に変更になっている。その関係で以下のようなエラーが発生したりする。

```
[bcc32c エラ□] DSnd.cpp(46): no matching function for call to 'MessageBoxW'
  winuser.h(8941): candidate function not viable: no known conversion from
'const char [29]' to 'LPCWSTR' (aka 'const wchar_t *') for 2nd argument
```

この対応は、下記のように文字列の前に L 指定を追加する。

```
MessageBox(NULL, "バッファーのアンロックに失敗", "Error", MB_OK);
      ↓
MessageBox(NULL, L"バッファーのアンロックに失敗", L"Error", MB_OK);
```

従来のBCC32でコンパイル

デフォルトでは□clangベースのコンパイラを使うようになってるが、一部互換のない記述を使っている場合は、とりあえず従来のBCC32でコンパイルするようプロジェクトを設定変更する。

【参考】

- C++Builder とトライグラフ

レジストリやINIファイルを扱うヘッダの変更

レジストリ関連を操作するヘッダファイルは「`#include <Registry.hpp>`」に記述変更。
INIファイル関連を操作するヘッダファイルは「`#include <IniFiles.hpp>`」に記述変更。

【参考】

- Registry で □orm の位置を記録
- Form の位置を IniFile に記録
- TRegistryIniFile Class

リンクエラ□□System::UnicodeString, Vcl::Controls::TWinControl□

下記のようなリンクエラーが発生。

```
[ilink32 エラ□] Error: 未解決の外部シンボル '__fastcall
Vcl::Filectrl::SelectDirectory(System::UnicodeString,System::WideString,
System::UnicodeString&, System::Set, Vcl::Controls::TWinControl *)'
```

プロジェクトファイルの`<AllPackageLibs>`に`vclx.lib`を追加するとリンクが成功。

【参考】

- SelectDirectoryでリンカエラー

max,min関数でエラー

以前は、`<stdlib.h>`をインクルードすれば□max,min関数が使えていたが使えなくなっている模様。
下記のマクロを該当ソースに定義し修正して対応。

```
#define __max(a,b) (((a) > (b)) ? (a) : (b))
#define __min(a,b) (((a) < (b)) ? (a) : (b))
```

参考記事

---

1. bcc32 と msc(cl) のコマンドラインコンパイラの UNICODE 対応化オプション 2. 定義済みマクロ 3. 静的ランタイムライブラリ 4. 旧バージョンからの移行

From:
https://yokoweb.net/dokuwiki/ - **新石器Wiki**

Permanent link:
https://yokoweb.net/dokuwiki/devtool/ide/cpp_builder-rad_studio/cpp_builder-use-old-version?rev=1580367627

Last update: **2020/01/30 16:00**