Intellisense - **クイックガイド**

1. "" をダウンロード
2. ダウンロードしたファイルを Visual Studio プロジェクトに追加
3. プロジェクトを閉じ、プロジェクトの *.ncb ファイルを削除。
4. プロジェクトを開く。
5. 特に .NET 2002 は、そのファイルを解析している間にクラッシュする傾向がある。その場合には、開きなおす。
6. 新しい wxWedgets クラスを使うときは、それがまだ定義されていない場合には、_wx_intellisense.h に "wx/something.h" と "wx/msw/something.h" を追加する。

この方法は、完全ではない。幾つかのクラスは、幾つか、または全ての関数が表示できない。しかし、多くのクラスは適度にうまくいく。

Click "here", if you are interested in the details behind _wx_intellisense.h.

Integrated (F1) Onlinehelp and wxWidgets documentation

I will no longer be making available the 'setup.msi' file here to integrate online help into visual studio. Instead I have created an installer called wxVisualSetup, which enables Intellisense, Online Help and a Project Wizard for wxWidgets.

Intellisense **と** wxWidgets

Visual Studio .NET を使用しているが、この発見は VC++ 6.0 と恐らく .NET 2003 にも応用できる。

plugin::pdf::PDFParser=HASH(0x53e96c)Visual Studio .NET を使用しているのは、オリジナルの筆者のことです

問題点

MS Intellisense の単純なヘッダファイルの解析方法では、#if, #define にうまく対処できない。

■ **問題点** (1)

以下を例としてあげる。：

wxprec.h

```
#include "wx/platform.h"        // __WXMSW__ などを定義
#include "wx/wx.h"              // __WXMSW__ が定義されているときに、"wx/msw/wx.h" をインクルード
```

platform.h は、マクロ _ _WXMSW_ _ を定義する。Intellisense が wxprec.h を解釈するとき、#include に従って platform.h を読み込み、そして、#define _ _WXMSW_ _ を見つける。しかし、実際にプリプロセスする時と異なり、マクロ _ _WXMSW_ _ は、Intellisense が platform.h をクローズするときに定義がキャンセルされる。これにより、Intellisense が wx.h を解析するとき、_ _WXMWS_ _ は未定義となり、msw/wx.h は Intellisense データベースにインクルードされない。

結果、ほぼ全ての wxWidgets シンボルが Intellisense データベースから除かれる。

■ 問題点 (2)

Intellisense はマクロ定義を解決することができない。下記の例では、wxWindowMSW クラスは Intellisense に認識されるが、wxWindow は認識されない。：

```
#define wxWindowMSW wxWindow
class wxWindowMSW {
};
```

加えて、マクロ WXDLLEXPORT は正しく解析されず、以下の宣言も予想通り、うまくいかない。

```
class WXDLLEXPORT wxClass {};
```

解決方法

両方の問題点を解決する、非常に簡単な解決方法が存在し、非常にうまくいく。

1. 新規ファイル wx_intellisense.h を作成し、プロジェクトに追加する。このファイルが他のソースファイルからはインクルードされてはならない。必要な作業は、それをプロジェクトファイルツリーに追加することだけである。

2. 必要な wxWidgets の各シンボルに対し、wx_intellisense.h に関連するヘッダファイルをインクルードする。include/wx の一般的な .h ファイルと、include/wx/msw ディレクトリにある実装用の .h ファイルの両方のヘッダファイルを追加する必要がある。

例：wxFrame を Intellisense に登録したい場合、wx_include.h に下記を追加する。

```
#include "wx/frame.h"
#include "wx/msw/frame.h"
```

plugin::pdf::PDFParser=HASH(0x53e96c)wx_include.h ではなく、wx_intellisense.h の間違い？

3. wx_intellisense.h を保存する。Intellisense はそのファイルを解析し、新しい #include を見つけて、解析可能な wxFrame クラスをインクルードする。もし Intellisense が解析できない場合、Visual Studio を終了し、*.ncb を削除してから、プロジェクトを開きなおし、wx_intellisense.h を保存しなおす。

これで、問題点 (1) は解決できるはずである。単に wxWidgets/include/wx と wxWidgets/include/wx/mws の全てのヘッダファイルを wx_intellisense.h に追加するだけである。これは、既に試してあるが、Intellisense が多くのファイルを解析するときに少し不安定になる。Intellisense データベース（*.ncb ファイル）をリビルドしようとした時に、Visual Studio が何回か無反応になった。そのため、自分のプロジェクトで実際に必要になるヘッダファイルだけをインクルードし、その度にそのファイルをビルドした。少し実験してみたほうが良いかもしれない。

問題点 (2) は、同様に単純な解決方法がある。単に以下を wx_intellisense.h の最後の #include の後に、追加するだけである。

```
class wxWindow:public wxWindowMSW {
};
```

これは、wxWindowMSW の全てのメンバで wxWindow クラスを宣言するものであり、まさに wxWindow そのものである。ここでの定義は、wx_intellisense.h が実際にはコンパイルされないため、完全に偽りのものである。

そして、WXDLLEXPORT の問題のため、wx_intellisense.h の冒頭で WXDLLEXPORT を定義する必要がある。

```
#define WXDLLEXPORT /* */
```

wxWidgets を DLL として使用する場合でも、このファイルはコンパイルされないため、この記述をそのままにしておいて良い。

一般的に、もしも未定義なクラスが見つかり Ctrl-Space を押してもうまくいかない場合には、インクルードファイルを wx_intellisense.h に追加して、それを保存すればよい。

wx_intellisense.h の例

もしシンボルの表示に失敗している場合には、正しいヘッダファイルを追加しなければならない。例として、もし wxBitmap を使用する場合、#include "wx/msw/bitmap.h" の直前に #include "wx/bitmap.h" を追加しなければならない。そうでなければ、幾つかの wxBitmap シンボルは未定義になる。このファイルを出発点として使用すること。

```
#define WXDLLEXPORT /* */
[... 中略 ...]
#include "wx/msw/missing.h"
#include "wx/msw/setup.h"
#include "wx/window.h"
#include "wx/msw/window.h"
#include "wx/msw/winundef.h"
#include "wx/sizer.h"

#undef wxWindow
class wxWindow:public wxWindowMSW
{
};
```