

# 2回目 グラフ作成ライブラリmlibの使い方 ウィンドウ初期設定関連

グラフ作成ライブラリmlibの使い方、ウィンドウ初期設定関連

Visual-Cのインストール

Mlibのダウンロード

Visual-C のプロジェクトの作成

def.hの使い方

ウィンドウサイズの変え方

プッシュボタン、チェックボックス、ラジオボタン

メモウインドウ設定、Printf

# Visual C++ 2010 Express のインストール

<http://www.microsoft.com/japan/msdn/vstudio/express/>



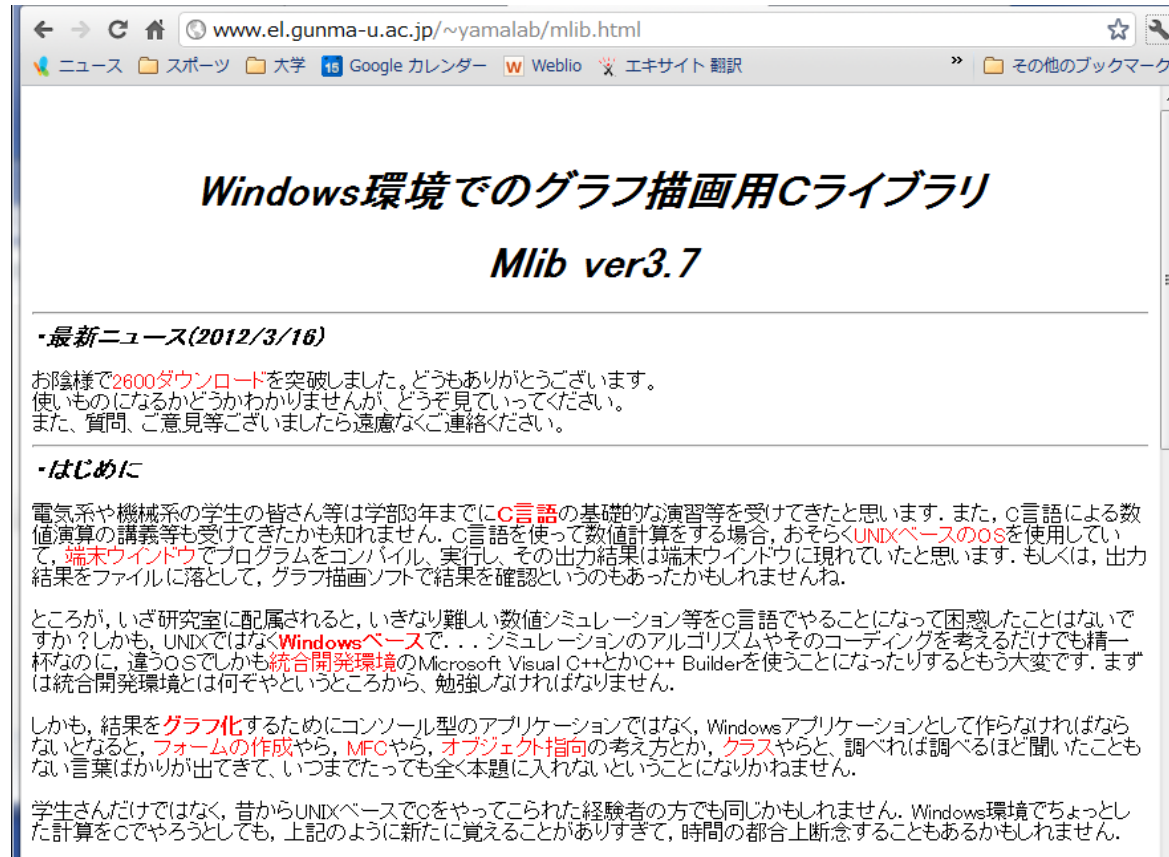
クリック後にダウンロードされた vc\_web.exeを実行

Platform SDK も含まれており、自動的にインストールされる

ダウンロード、インストールで10分程度

# mllib のダウンロード

<http://www.el.gunma-u.ac.jp/~yamalab/mllib.html>



← → C 家 www.el.gunma-u.ac.jp/~yamalab/mllib.html ☆ 鍵

ニュース スポーツ 大学 Google カレンダー Weblio エキサイト 翻訳 » その他のブックマーク

## Windows環境でのグラフ描画用Cライブラリ

### Mlib ver3.7

---

**・最新ニュース(2012/3/16)**

お陰様で2600ダウンロードを突破しました。どうもありがとうございます。  
使いものになるかどうかわかりませんが、どうぞ見ていってください。  
また、質問、ご意見等ございましたら遠慮なくご連絡ください。

---

**・はじめに**

電気系や機械系の学生の皆さん等は学部3年までに**C言語**の基礎的な演習等を受けてきたと思います。また、C言語による数値演算の講義等も受けてきたかも知れませんが、C言語を使って数値計算をする場合、おそらく**UNIXベースのOS**を使用している、**端末ウィンドウ**でプログラムをコンパイル、実行し、その出力結果は端末ウィンドウに現れていたと思います。もしくは、出力結果をファイルに落として、グラフ描画ソフトで結果を確認というのもあったかもしれませんね。

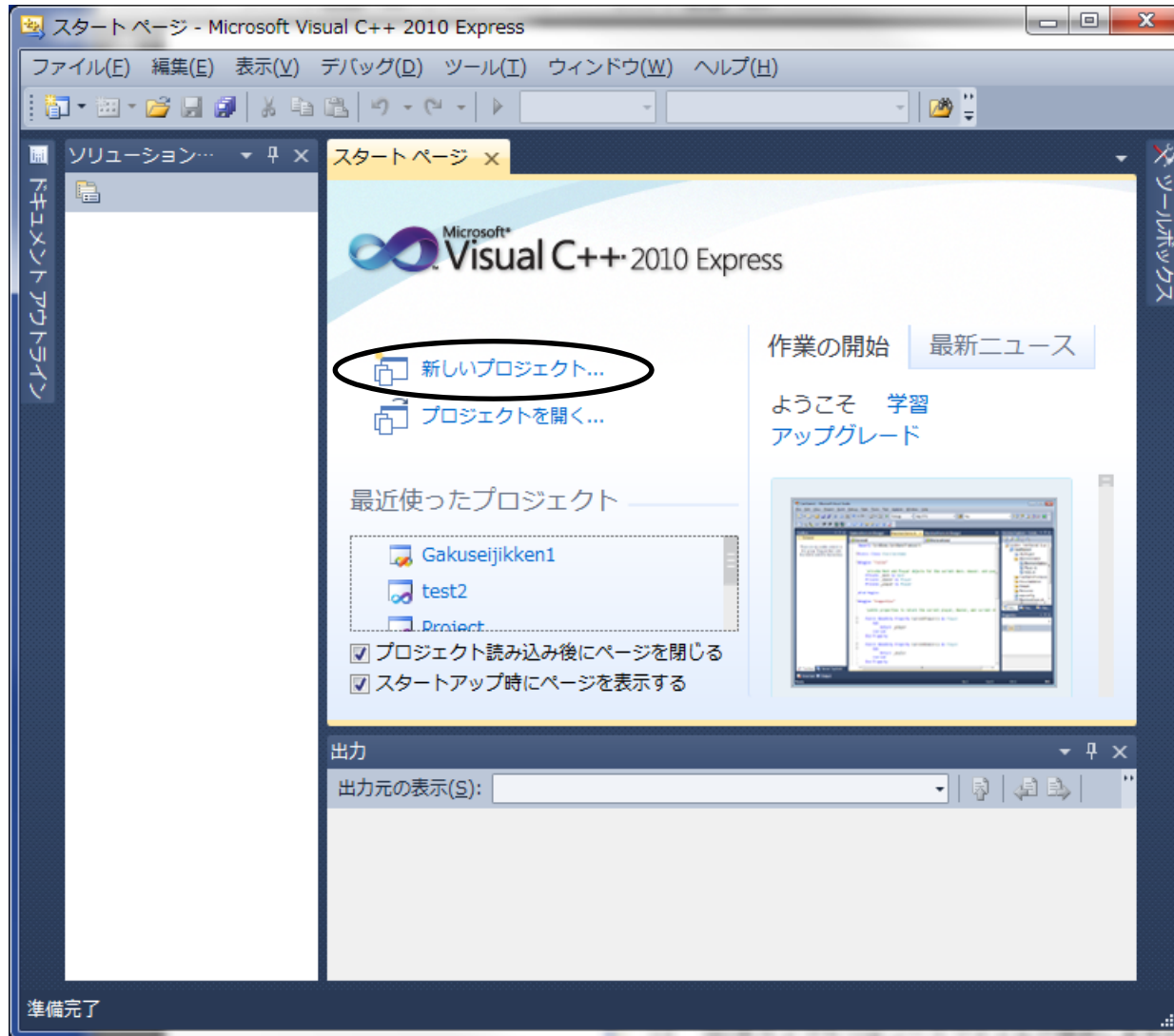
ところが、いざ研究室に配属されると、いきなり難しい数値シミュレーション等をC言語でやることになって困惑したことはないですか？しかも、UNIXではなく**Windowsベース**で...シミュレーションのアルゴリズムやそのコーディングを考えるだけでも精一杯なのに、違うOSでしかも**統合開発環境**のMicrosoft Visual C++とかC++ Builderを使うことになったりするともう大変です。まずは統合開発環境とは何ぞやというところから、勉強しなければなりません。

しかも、結果を**グラフ化**するためにコンソール型のアプリケーションではなく、Windowsアプリケーションとして作らなければならぬとなると、**フォームの作成**やら、**MFC**やら、**オブジェクト指向**の考え方や、**クラス**やらと、調べれば調べるほど聞いたこともない言葉ばかりが出てきて、いつまでたっても全く本題に入れないということになりかねません。

学生さんだけではなく、昔からUNIXベースでCをやっていた経験者の方でも同じかもしれません。Windows環境でちょっとした計算をCでやろうとしても、上記のように新たに覚えることがありすぎて、時間の都合上断念することもあるかもしれません。

このページの下の方から **mllib.h** と **def.h** をダウンロードして、  
分かりやすいところにおいておく

# mllibのインストール



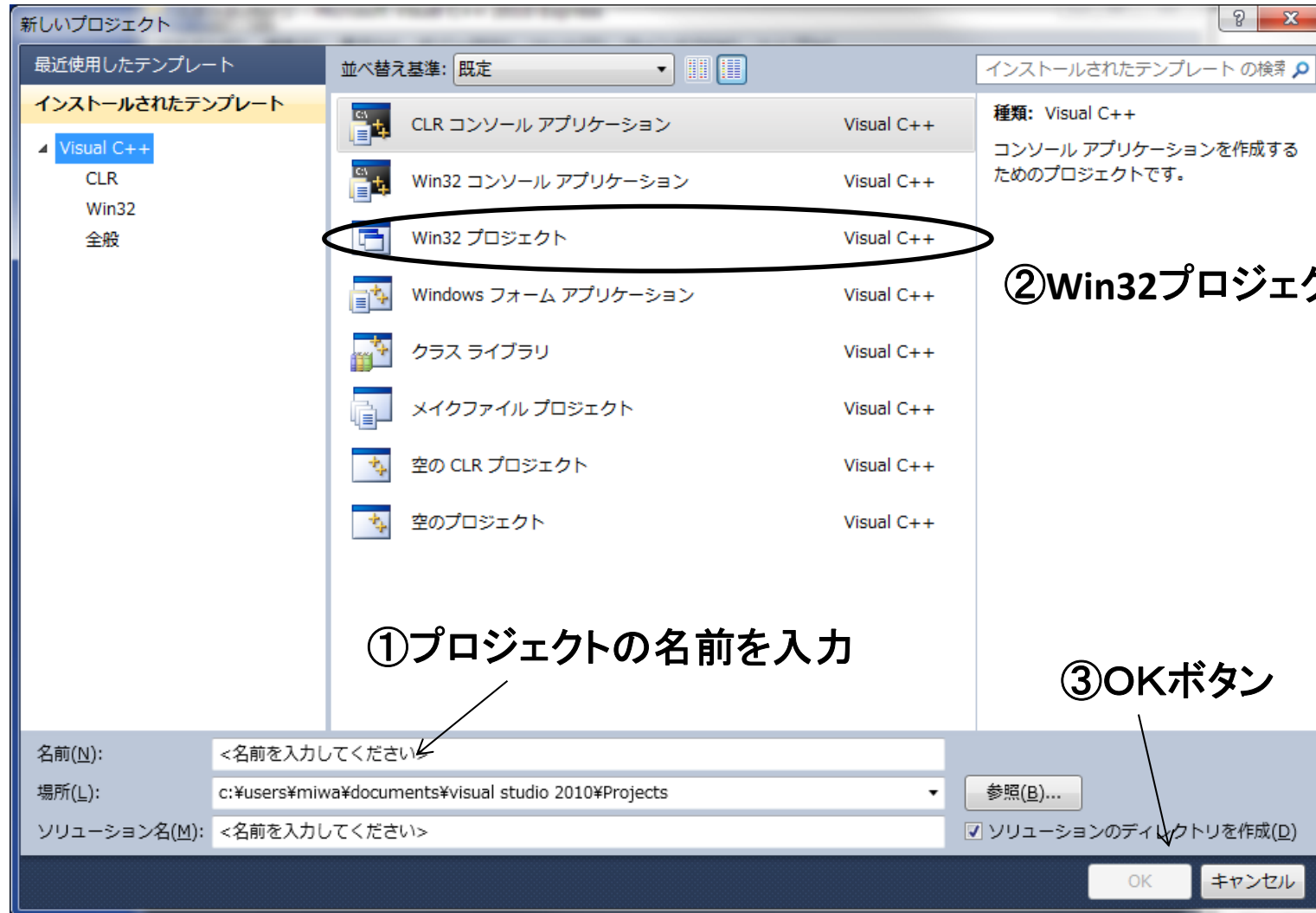
Visual C++ 2010E を起動する

新しいプロジェクトを作成

visualC ではプロジェクト単位でプログラムやコンパイルに必要なインクルードファイル、実行可能ファイル等を登録し、管理する

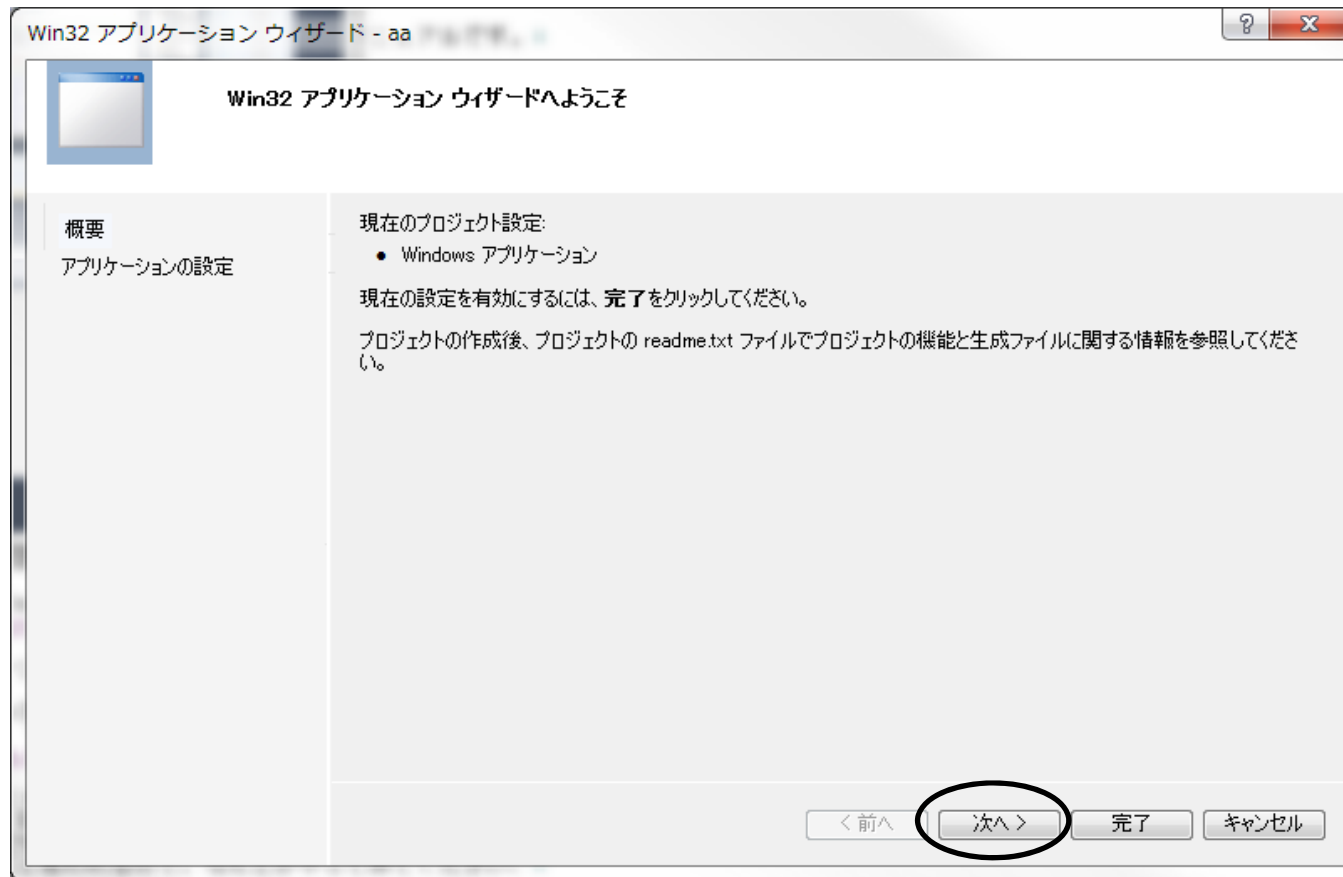
# プロジェクトの新規作成

以下に指示するどおりに作らないと、コンパイルできないので注意



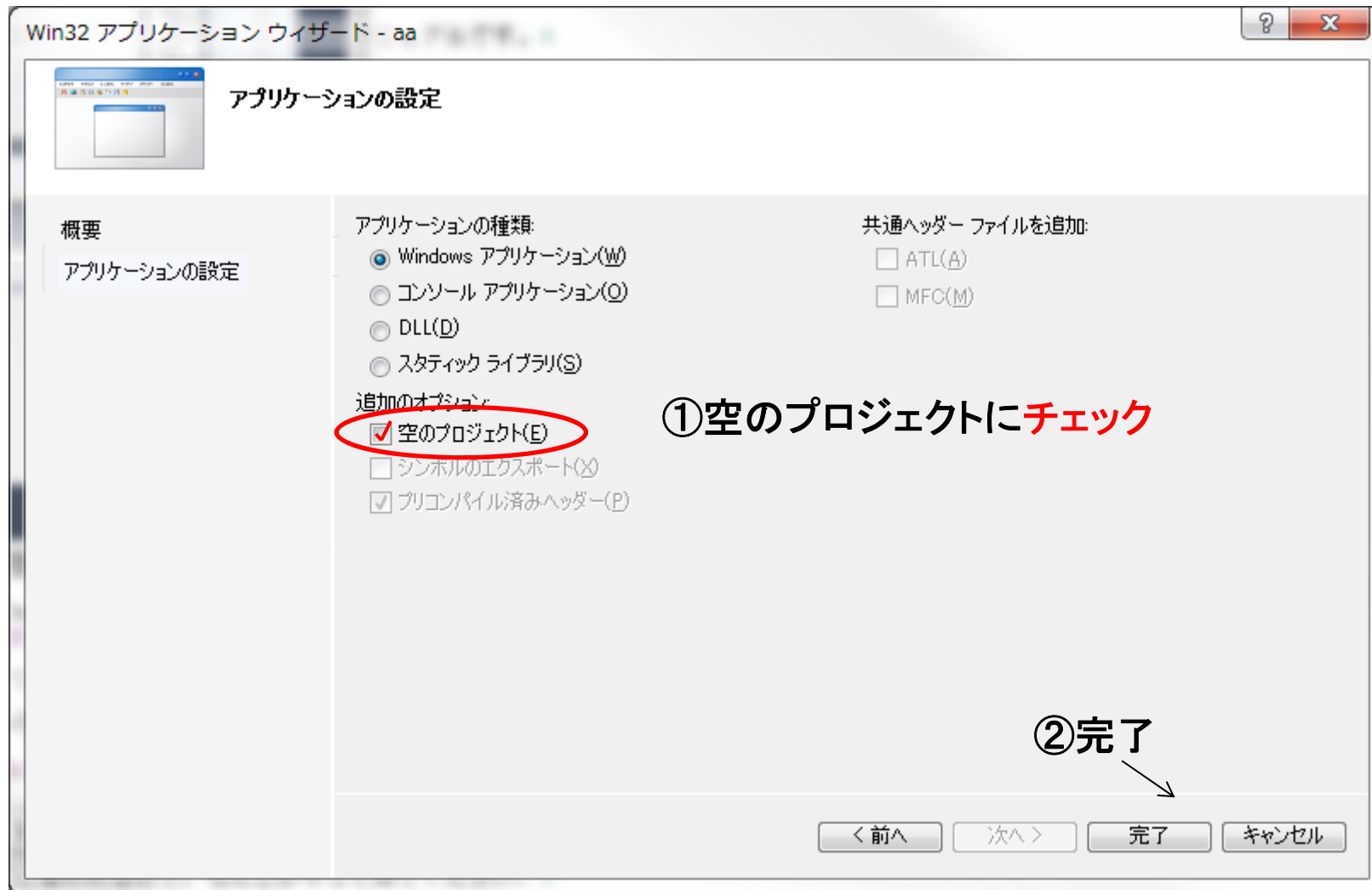
# Win32アプリケーションウィザード

アプリケーションウィザードが開く



面倒なWindows アプリケーションのひな形プロジェクトを簡単に作ってくれるが、、

# Win32アプリケーションウィザード

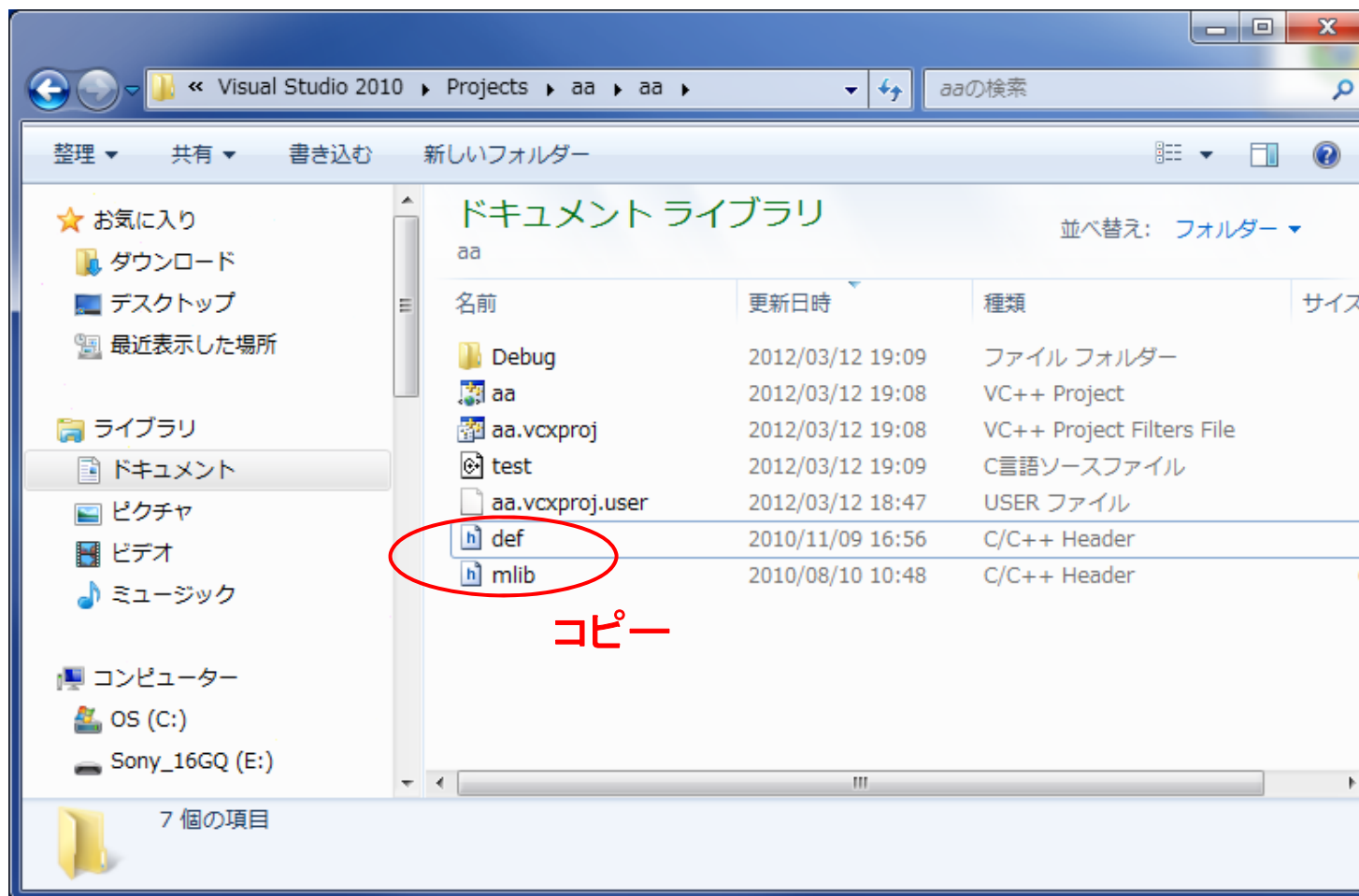


mllibがひな形の役割なので、空のプロジェクトのみ作っておく

# プロジェクトの作成確認

自分のホームディレクトリのVisual Studio 2010¥Projectsに作成したプロジェクトのディレクトリができていることを確認

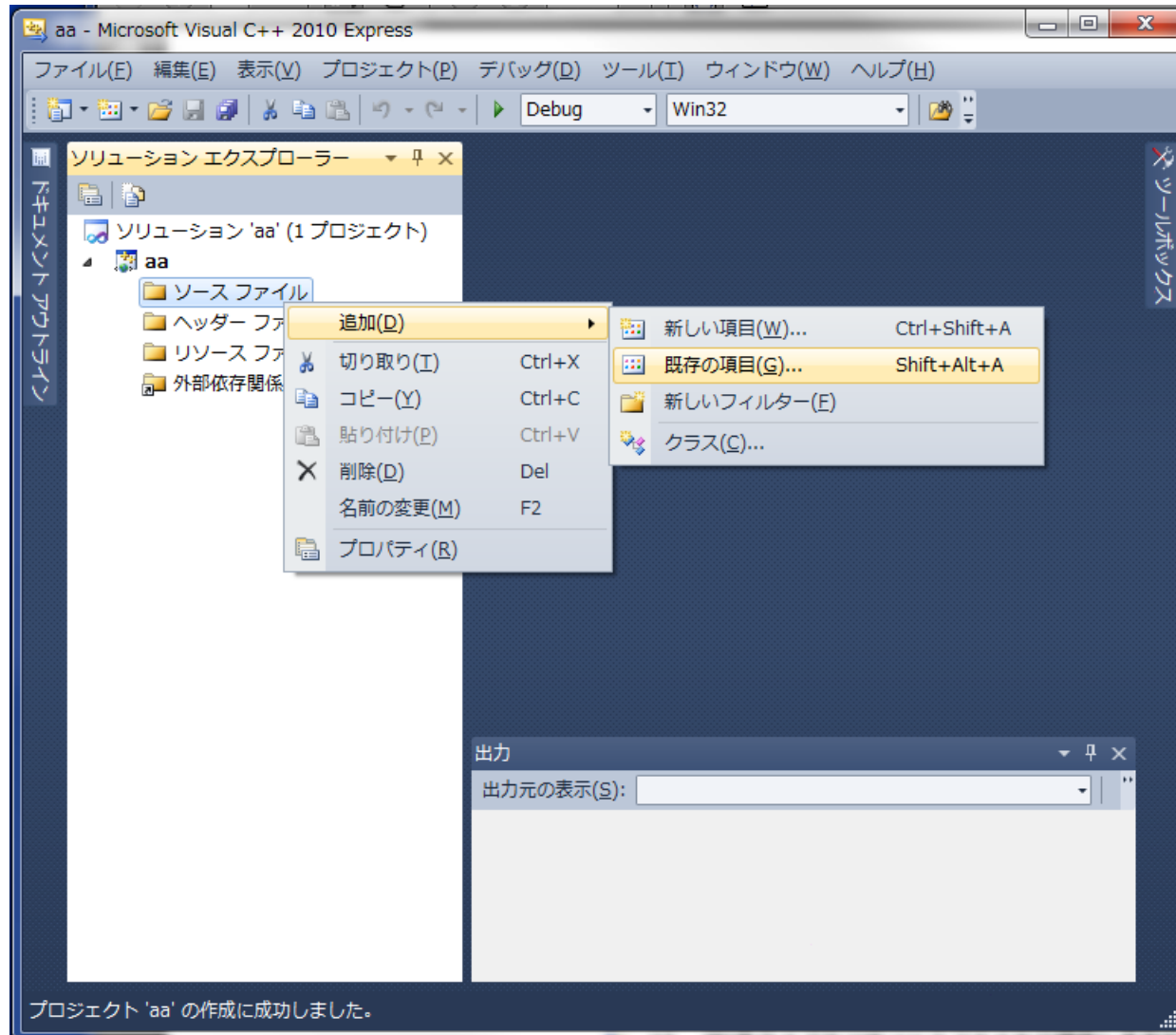
Visual Studio 2010¥Projects¥プロジェクト名¥プロジェクト名  
にmlib.h とdef.hをコピーしておく





# プロジェクトへのヘッダーファイルの追加

作ったプロジェクトにファイルを追加しないと、コンパイルに反映されない



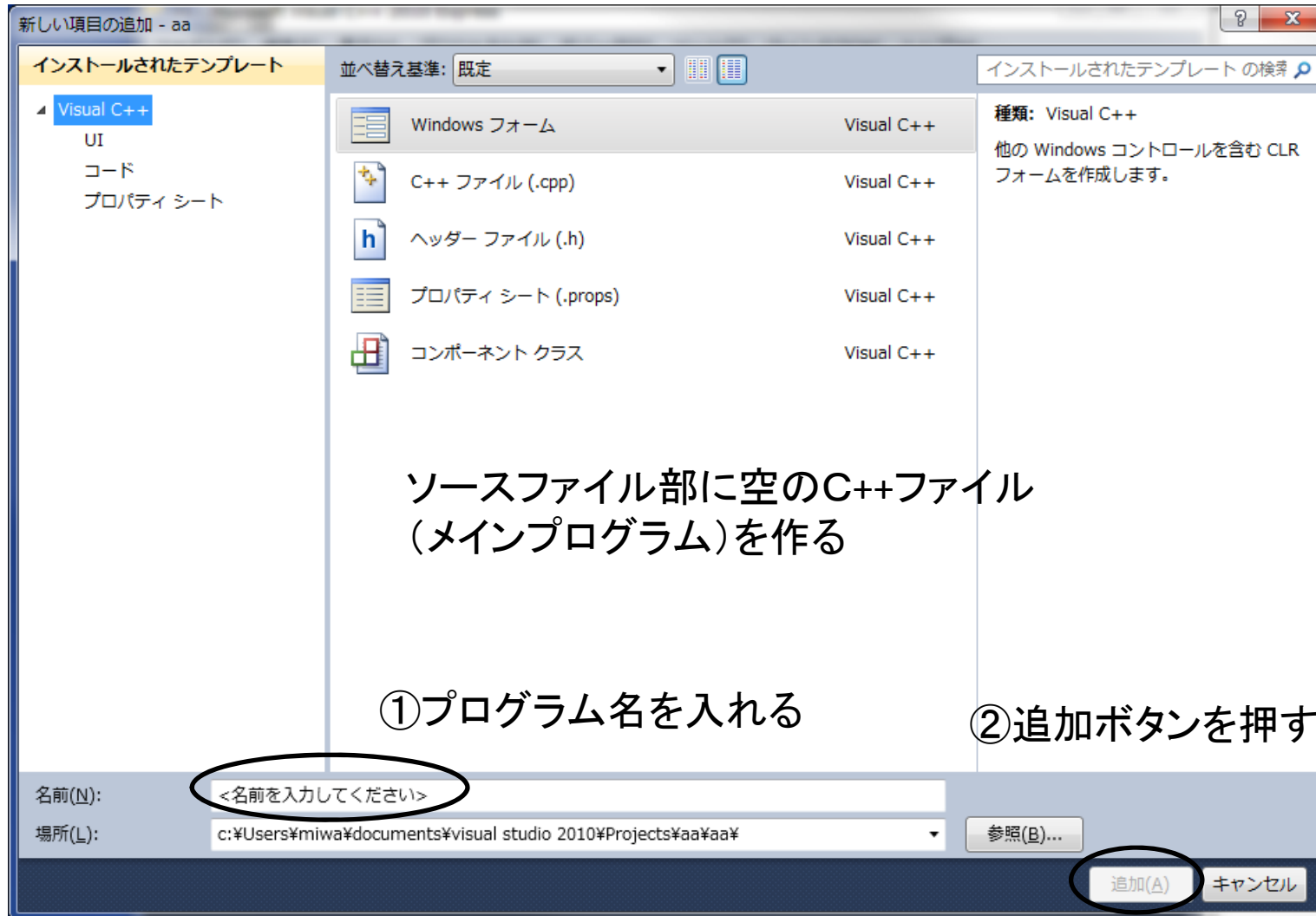
プロジェクトは作成したヘッダーファイルやプログラムが入ったコンテナ

ソリューションエクスプローラーのソースファイルやヘッダーファイルのアイコンを右クリックでファイルを追加できる。

作成したプロジェクトがあるディレクトリと同じディレクトリにコピーした `mllib.h`, `def.h` をヘッダーファイル部に追加する。

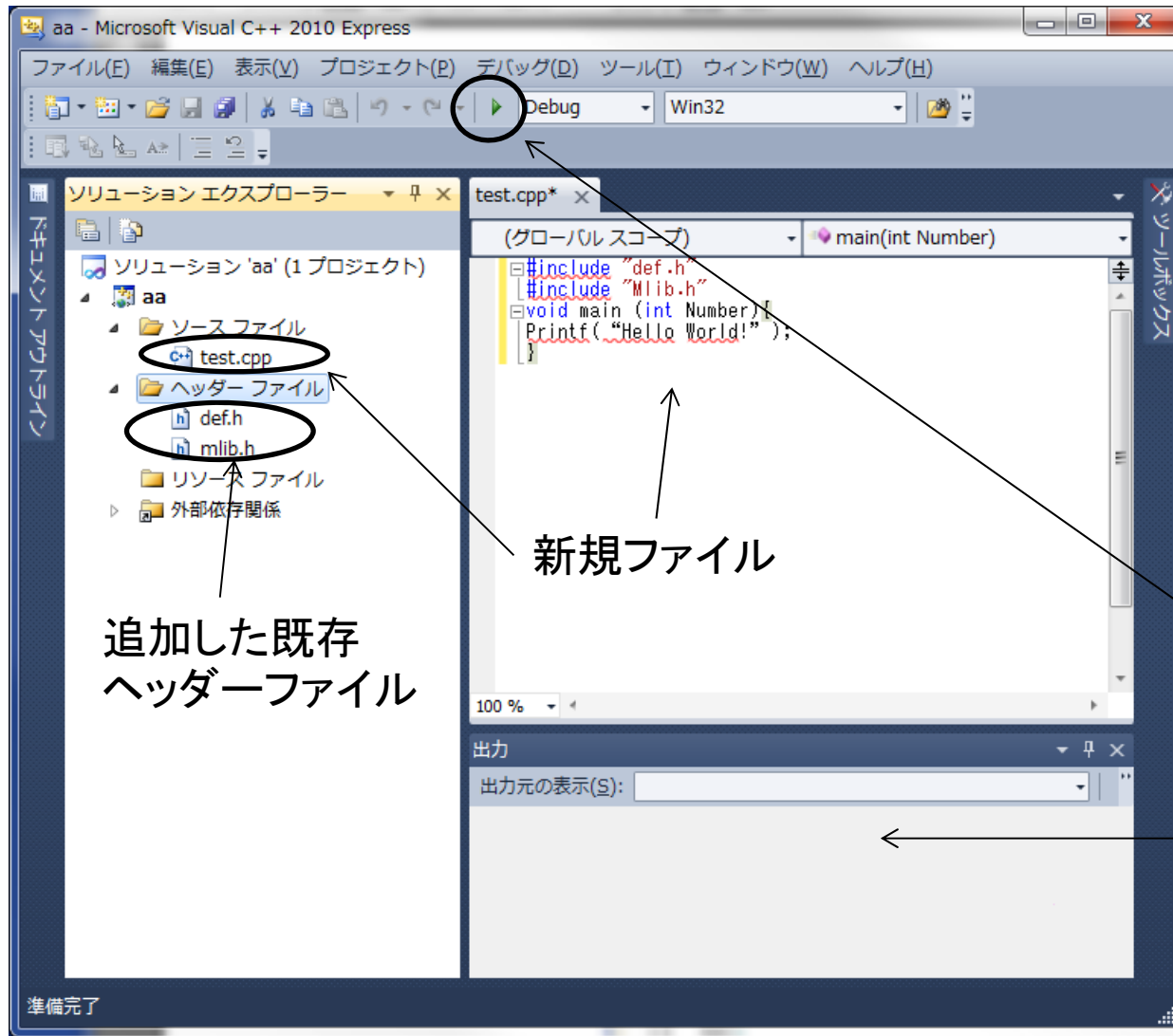
# プロジェクトへの新規ファイルの作成

ソースファイルのアイコンを右クリックでファイルを新たに作成追加できる。



Visual Studio 2010\Projects\プロジェクト名\プロジェクト名 の下に作成される

# サンプルプログラムの入力



①作成した新規Cファイル  
に以下のプログラムを入力

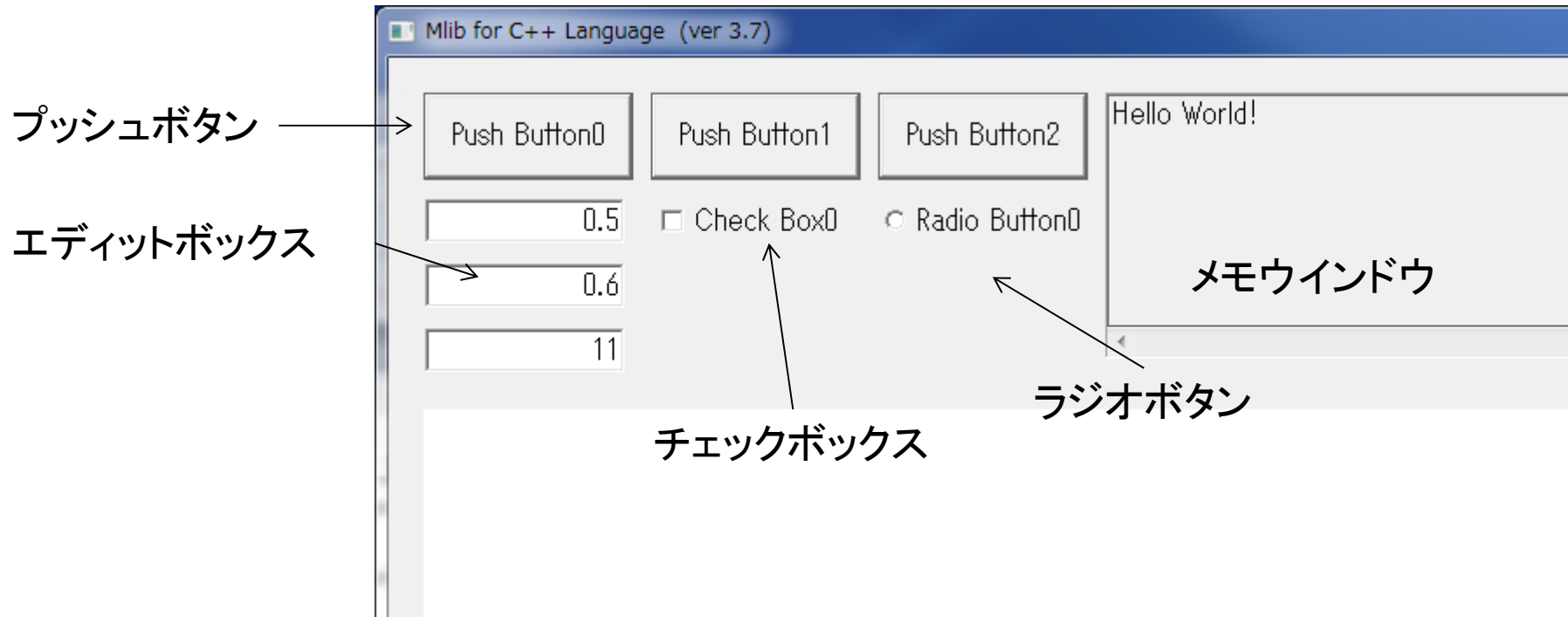
```
#include "def.h"
#include "mlib.h"
void main(int Number){
    Printf("Hello World!");
}
```

②コンパイル実行ボタンを押す

③コンパイルエラーがあればここにメッセージが出る、  
エラーがでなくなるまでプログラムを修正する

# サンプルプログラムの実行結果

プッシュボタンを押すと、メモウインドウに「Hello World!」と表示される



def.h には、ウインドウやボタンのサイズ、配置の情報が入っている。

mllib.h には様々な関数やウインドウ処理のOSとのやり取りをする関数が入っている。

def.h と mllib.hのインクルードの順番は必ずdef.hを先にインクルードする

プッシュボタンが押されたとき、main関数を実行する。(main関数は必ず必要)  
main関数は整数型引数Numberを受け取る。Numberには押されたプッシュボタンの番号が入る。

# def.h のウィンドウ初期設定部

```
def.h
(グローバルスコープ)
#include<windows.h>
#include<tchar.h>
#include<stdarg.h> //可変引数処理
#include<math.h>
#include<stdio.h>

/*****
/*   ウィンドウ情報の変更箇所   */
*****/

#define PROGRAM_TITLE "Mlib for C++ Language (ver 3.7)" //タイトルバーのキャプション
#define MAINWIN_W 1200 //メインウィンドウの幅
#define MAINWIN_H 1000 //メインウィンドウの高さ

#define PB_NUM 3 //使用するプッシュボタンの数
#define CK_NUM 1 //使用するチェックボックスの数
#define RD_NUM 1 //使用するラジオボタンの数
#define ED_NUM 3 //使用するエディットボックスの数
#define ST_NUM 0 //使用するテキストの数

#define FIGMAX 20 //フィギュアウィンドウに描ける図の最大数
#define PALLET_NUM (8*32) //2D画像のカラーマップの諧調 (8~256までで8の倍数を指定)

#define BUFSIZE_W 81 //メモウィンドウの水平文字数
#define BUFSIZE_H 1024 //メモウィンドウの記憶行数
#define DISPSIZE_H 7 //メモウィンドウの表示行数
#define MEMOWIN_H (19*(DISPSIZE_H+1)) //メモウィンドウの高さ
#define PNMAX 20 //フィギュアウィンドウに記憶できる線種の最大数
#define BUFNMAX 25 //軸キャプション, 数値ラベル等の最大文字サイズ

/*****
/*   ここまで   */
*****/
```

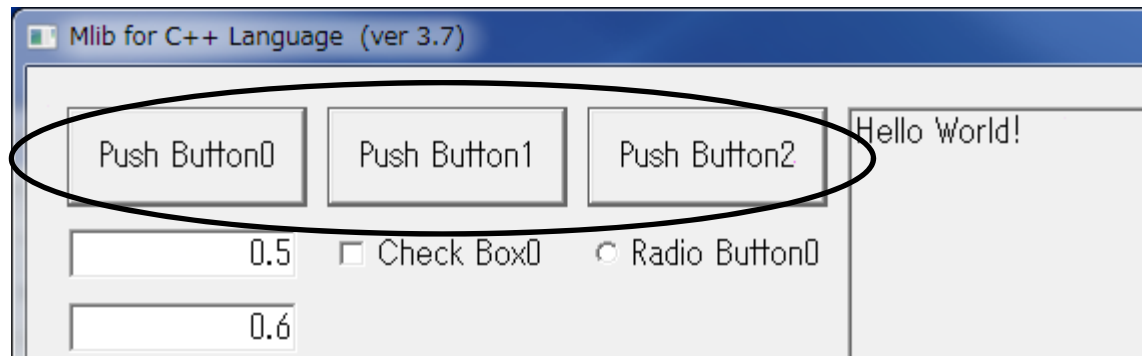
変更するとグラフィックウィンドウ、メモウィンドウのサイズが適宜調整して表示される。

使用するコントロールに合わせて数を入力

メモウィンドウ関連  
変えなくてもよい

def.h で定めた定数はすべて大文字  
各種コントロールのサイズ、位置は下の方で設定

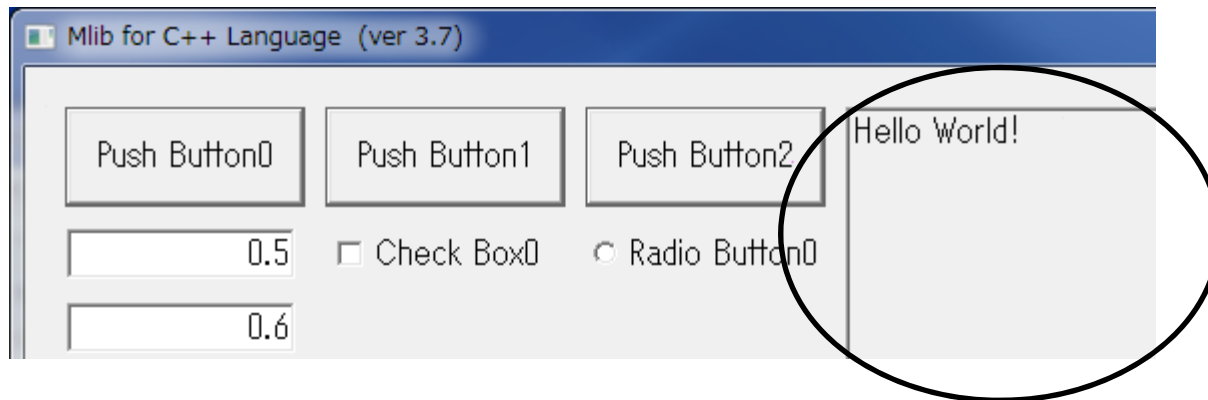
# プッシュボタン



- ・def.h内の定数PB\_NUMで指定した個数のプッシュボタンをウインドウ内に配置できる
  - ・0から昇順にボタン番号が割り振られる。
  - ・ボタンを押すことにより、main()関数で指定したプログラムが実行される。
  - ・押されたボタン番号が仮引数Numberに入っているため、プログラム内でswitch文やif文等で押されたボタンの種類による処理の場合分けができる。
- ・ボタンの位置、サイズ、表示される文字はdef.h の component()関数内にあるグローバル変数のsPB[ボタン番号]構造体を変更すればよい。

sPB[i].x	ボタン左上のx座標
sPB[i].y	ボタン左上のy座標
sPB[i].w	ボタンの幅
sPB[i].h	ボタンの高さ
sPB[i].name	ボタンのキャプション(TCHAR型)

# メモウインドウ

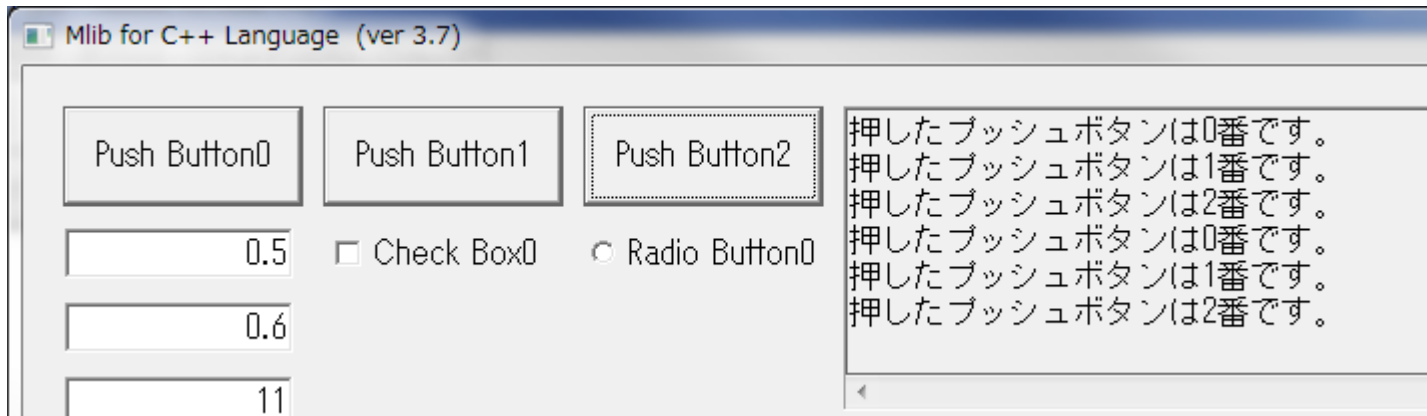


- ・変数やコメント等をプログラム中で表示させることが可能であり、情報処理センターのPCでいう(表示のみの)端末ウインドウに対応。
- ・メモウインドウ内はスクロールさせることができるため、多数の表示が必要なときに使用する。
- ・1行の文字数や、記憶できる行数、表示行数はdef.hの定数`BUFSIZE_W`、`BUFSIZE_H`、`DISPSIZE_H`でそれぞれ指定できる。
- ・表示には関数`Printf("表示文字列", 変数等...)`を使用、`printf`関数と同じ使い方。
- ・メモウインドウは一つだけ配置できる
- ・位置、サイズ、キャプションは`sME`構造体を変更すればよい。

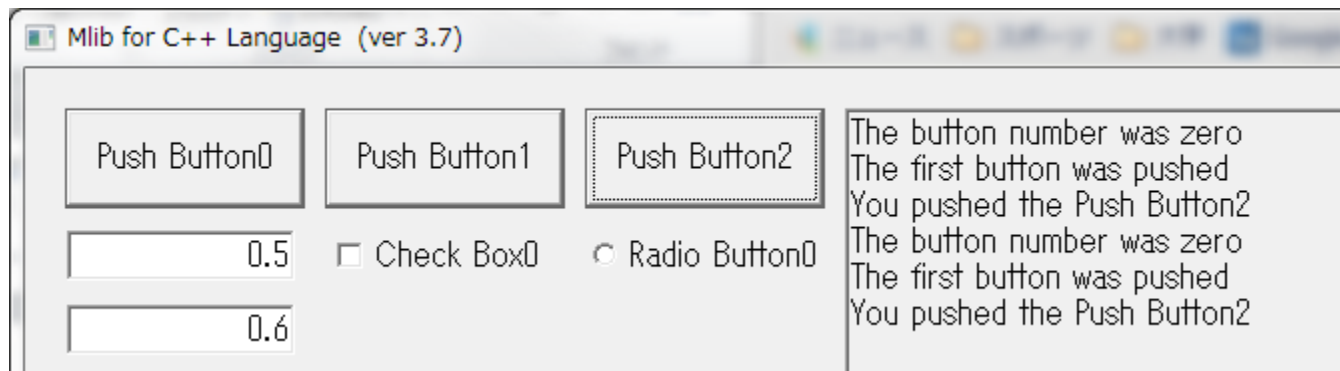
<code>sME.x</code>	エディットボックス左上のx座標
<code>sME.y</code>	エディットボックス左上のy座標
<code>sME.w</code>	エディットボックスの幅
<code>sME.h</code>	エディットボックスの高さ( <code>MEMOWIN_H</code> で決まる)

# 演習1

- 1-1. メモウインドウに押したプッシュボタンの番号を表示するプログラムを作れ  
Printf関数はprintf関数と使い方は同じ

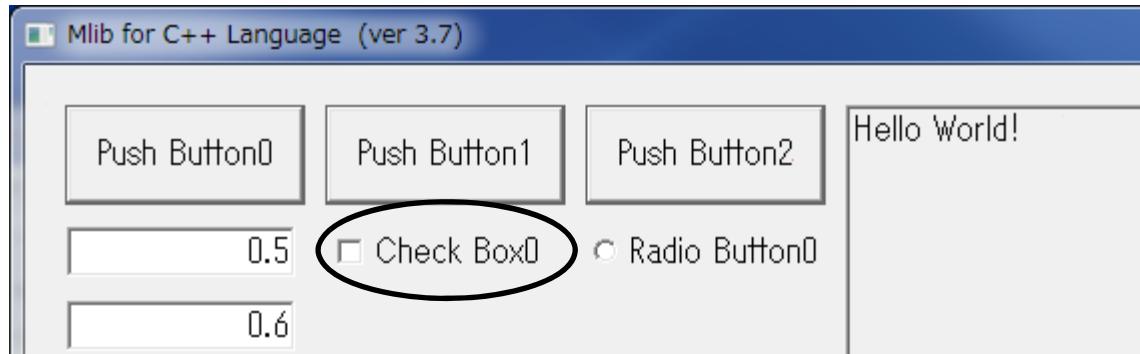


- 1-2. メモウインドウに押したプッシュボタンの番号を英語で表示するプログラムを作れ. if文、switch文を使う





# チェックボックス



- ・状態のON、OFFを変更できるボックスであり、クリックごとにON,OFFを切り替える。
- ・def.h内の定数CK\_NUMで指定した個数のチェックボックスを配置できる
- ・0から昇順に番号が割り振られる。
- ・位置、サイズ、キャプションはdef.hのcomponent()関数内にあるグローバル変数のsCK[チェックボックス番号]構造体を変更すればよい。

sCK [i].x	チェックボックス左上のx座標
sCK [i].y	チェックボックス左上のy座標
sCK [i].w	チェックボックスの幅
sCK [i].h	チェックボックスの高さ
sCK [i].name	チェックボックスのキャプション(TCHAR型)

- ・ON,OFFの状態はsCK[チェックボックス番号].chkに反映されONのとき1、OFFのとき0になっている。

# 演習2

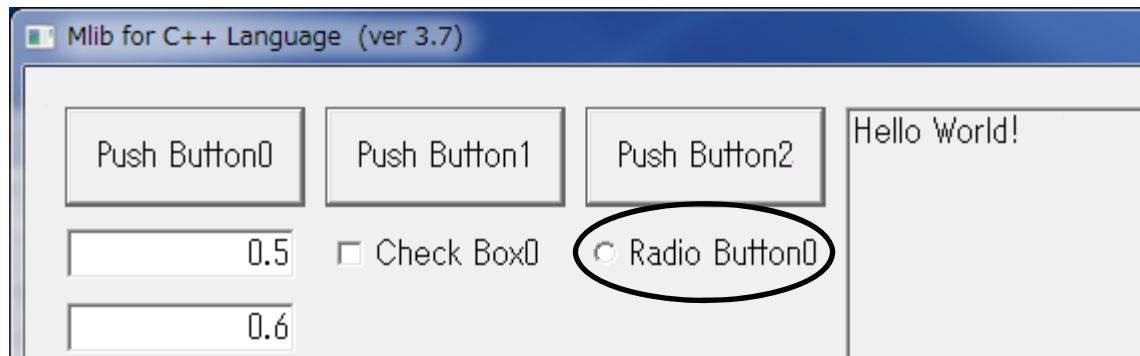
2-1. チェックボックスを二つ作成し、二つのチェックボックスのON, OFF状態をメモウインドウに表示するプログラムを作れ

Push Button0	Push Button1	Push Button2	チェックボックス0はオン    チェックボックス1はオフ
<input type="text" value="0.5"/>	<input checked="" type="checkbox"/> Check Box0	<input type="radio"/> Radio Button0	
<input type="text" value="0.6"/>	<input type="checkbox"/> Check Box1		

Push Button0	Push Button1	Push Button2	チェックボックス0はオン    チェックボックス1はオフ チェックボックス0はオフ    チェックボックス1はオン
<input type="text" value="0.5"/>	<input type="checkbox"/> Check Box0	<input type="radio"/> Radio Button0	
<input type="text" value="0.6"/>	<input checked="" type="checkbox"/> Check Box1		

Push Button0	Push Button1	Push Button2	チェックボックス0はオン    チェックボックス1はオフ チェックボックス0はオフ    チェックボックス1はオン チェックボックス0はオン    チェックボックス1はオン
<input type="text" value="0.5"/>	<input checked="" type="checkbox"/> Check Box0	<input type="radio"/> Radio Button0	
<input type="text" value="0.6"/>	<input checked="" type="checkbox"/> Check Box1		

# ラジオボタン



- ・グループ化されたラジオボタンの中から一つを選択するボタンであり、クリックごとにグループ内でON,OFFを切り替える。
- ・def.h内の定数RD\_NUMで指定した個数のラジオボタンを配置できる
- ・位置、サイズ、キャプションはsRD[ラジオボタン番号]構造体を変更すればよい。

sRD [i].x	ラジオボタン左上のx座標
sRD [i].y	ラジオボタン左上のy座標
sRD [i].w	ラジオボタンの幅
sRD [i].h	ラジオボタンの高さ
sRD [i].name	ラジオボタンのキャプション(TCHAR型)

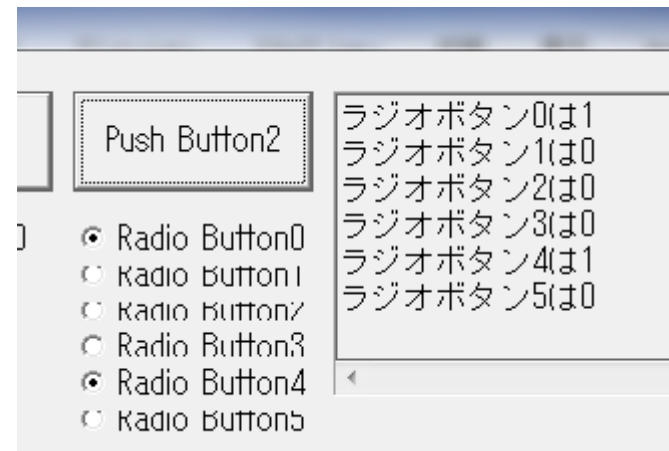
- ・選択の状態はsRD[ラジオボタン番号].chkに反映され**選択→1、それ以外→0**
- ・ラジオボタンのグループ化はdef.h内のsRD[].chkの初期値で設定する。以下の例ではラジオボタンの数が5個のときID=0,1 と2,3,4がそれぞれグループになる。  
sRD[0].chk=1;           sRD[2].chk=1;

# 演習3

3-1. ラジオボタングループを二つ作成し、1つのグループには3つつラジオボタンがあるとき、ボタンの状態をメモウインドウに表示するプログラムを作れ。



最初は何も選択されない



1, 4を選択したとき



5をクリックしたとき、  
3, 4, 5がグループなので、4がOFFになり、  
5がオンになる  
1, 2, 3、は別グループなので0がONのまま