DXライブラリの導入と設定ガイド Windows 版

STUDIO-UNSLOW

2026/08/16 更新

ちょっと大変だけど
大事な作業です
頑張って設定しましょう

DXライブラリとは

ゲームやアプリを作るための強力なライブラリ

- **DXライブラリ**はC++言語用の便利で強力なゲームライブラリです
  画像や音楽の読み込み、描画、キー入力などの複雑な処理を
  簡単に扱うことができます
- ゲームやアプリの開発を効率的に行うための便利な機能が沢山用意されています
  ゲームプログラミングの学習やC++を使用したオリジナルのゲーム制作に最適です
- 詳しくは公式サイトを参照してください
  https://dxlib.xsrv.jp/

この資料は

DXライブラリを使うための導入と設定手順を解説します

- DXライブラリの利用にはVisual Studio 等の開発環境を設定する必要があります
  設定は初心者には少し難しい場合があります
- この資料では初心者でも迷わず設定できるように手順を1つずつ順番に解説します
  わからない用語があっても、いったん気にせず先に進んでください
- ただ、**手順を入れ替えると設定がうまくいかない場合があります**
  説明を読み飛ばさないようにだけ注意してください

この資料と公式ガイドの違いについて

Windows 11とVisual Studio 2026 環境を前提に進めます

- DXライブラリ公式ガイドはVisual Studio 2022ベースですが
  この資料は Windows 11 と Visual Studio 2026 の利用を前提にしています
  異なる環境での設定は、公式ガイドを参考にしてください
- 基本的な導入手順は同じです
  画面表示や初期設定が異なる場合は**この資料の手順を優先**してください
- 迷った場合や資料にない表示がされた場合は、その場で講師に確認してください

Visual Studio 2026 のインストール

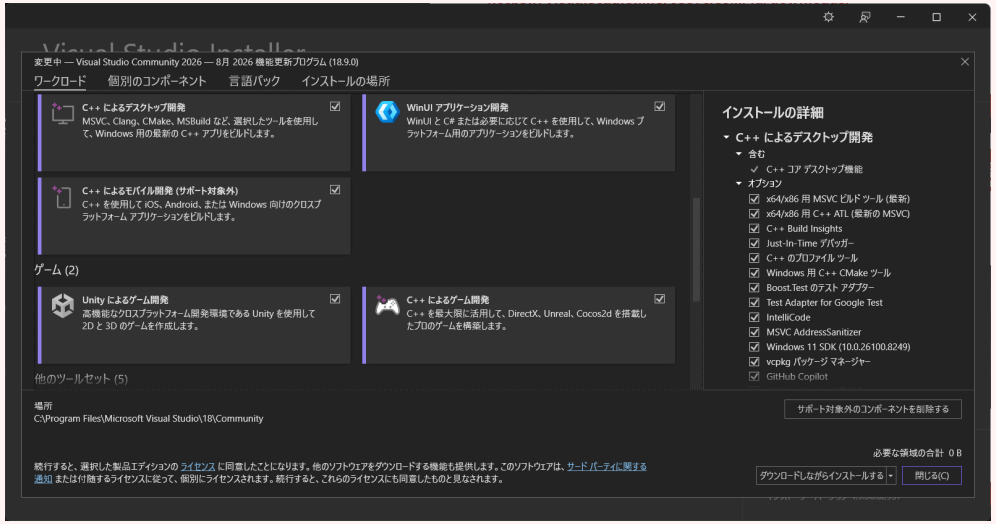
DXライブラリを使うための開発環境を準備しましょう

- Visual Studio はMicrosoft社が提供する統合開発環境です
  この資料では Visual Studio 2026 を使用して
  DXライブラリを利用するための設定を行います
- インストールされていない場合は
  公式サイトからダウンロード、インストールしてください
  https://visualstudio.microsoft.com/ja/downloads/
- Community 版は無料で利用できます
  利用規約を確認のうえ、インストールしてください

Visual Studio 2026 のインストール確認

C++によるデスクトップ開発 ワークロードが必要です

- まず Visual Studio 2026 が
  インストール済みかを確認してください
- 次に Visual Studio インストーラーを起動して
  C++によるデスクトップ開発 ワークロードが
  インストール済みかを確認してください
- もしインストールされていない場合は
  チェックを入れてインストールしてください



Visual Studio インストーラーのワークロード選択画面

DXライブラリをダウンロードして配置する

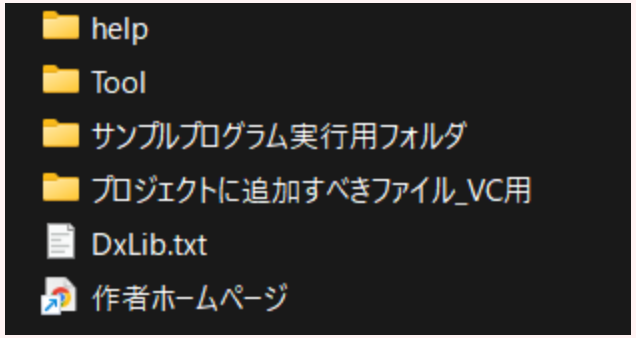
公式サイトから入手してください

- 公式サイトのダウンロードページ
  https://dxlib.xsrv.jp/dxdload.html にアクセスして
  「DXライブラリ Windows版 VisualStudio（C++）用」から
  ZIPファイルをダウンロード、解凍してください
- 解凍されたフォルダはお使いPCストレージの**わかりやすい場所**に置いてください
  例: `C:\DxLib_VC\`
- 必要に応じて公式の導入ガイドも確認してください
  https://dxlib.xsrv.jp/use/dxuse_vscom2026.html

DXライブラリのフォルダ構成を確認する

フォルダとファイルに不足がないかを確認してください

- 解凍したフォルダの中に
  プロジェクトに追加すべきファイル_VC用が
  あることを確認してください
- さらに、そのフォルダの中に `DxLib.h` があれば
  準備完了です
- 不足があったり文字化けが起きている場合は
  講師に相談してください



DXライブラリのフォルダ構成
「プロジェクトに追加すべきファイル_VC用」と
その中のDxLib.hを確認すること

環境変数を設定する

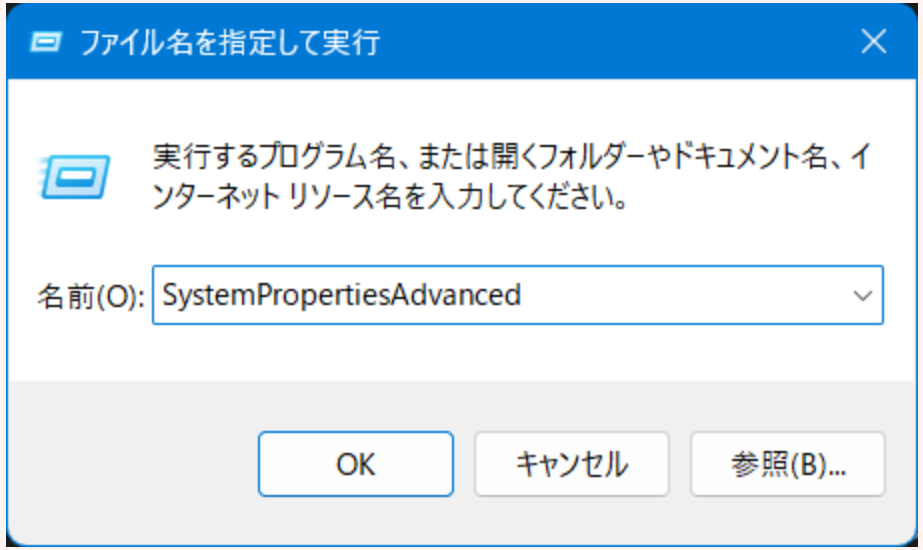
DXライブラリのインストール場所を登録する

- DXライブラリを置いた場所を**環境変数**に登録します
- **環境変数**はOSが管理する変数です
  プログラム間で設定値を共有するために利用されます
- ここでは `DXLIB_ROOT` という名前の環境変数で
  DXライブラリの配置場所を登録します
  次ページからの手順で登録を完了してください

環境変数 DXLIB_ROOT を登録する(1/4)

「ファイル名を指定して実行」からシステムのプロパティを開く

1. キーボードの **Windows キー + R** を押して「ファイル名を指定して実行」を開きます
2. 「**名前(O)**」に `SystemPropertiesAdvanced` と入力します
3. **OK** を押すかEnter キーを押してシステムのプロパティを開きます



「ファイル名を指定して実行」の名前(O)にSystemPropertiesAdvancedを入力

環境変数 DXLIB_ROOT を登録する(2/4)

システムのプロパティから環境変数の設定画面を開く

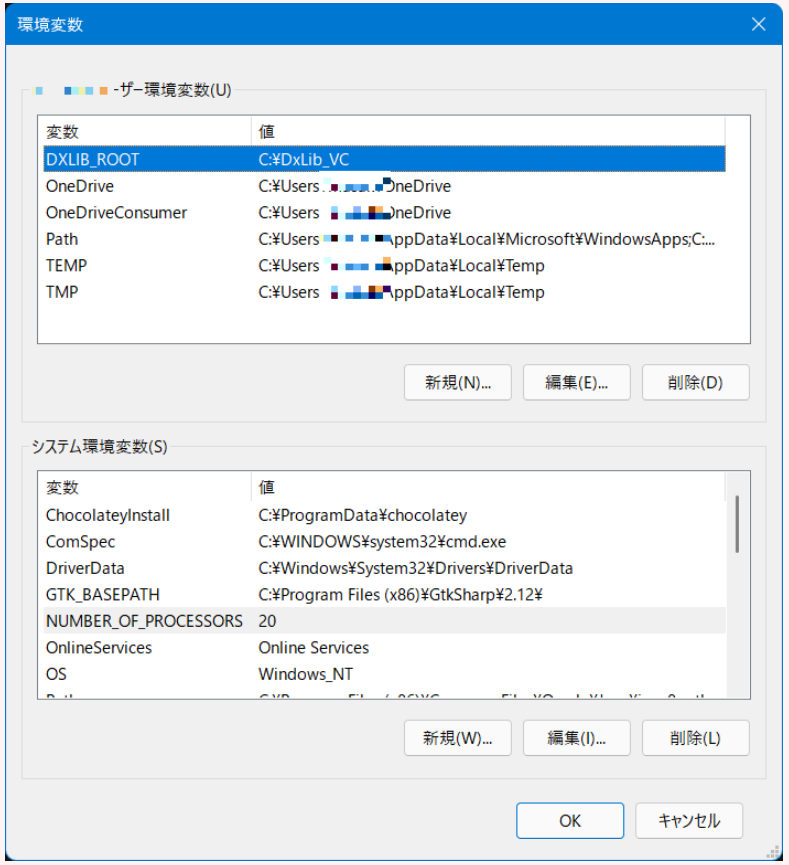
1. システムのプロパティが開いたら
   上の「**詳細設定**」タブが選択されていることを
   確認してください
2. 右下の「**環境変数(N)...**」を押します

システムのプロパティの詳細設定タブ

環境変数 DXLIB_ROOT を登録する(3/4)

ユーザー環境変数の一覧を確認する

1. ユーザー環境変数の一覧を確認してください
2. 既存の `DXLIB_ROOT` がない場合は
   新規(N)... で環境変数を追加してください
3. 既存の `DXLIB_ROOT` がある場合は
   編集(E)... で値を確認してください



ユーザー環境変数の一覧

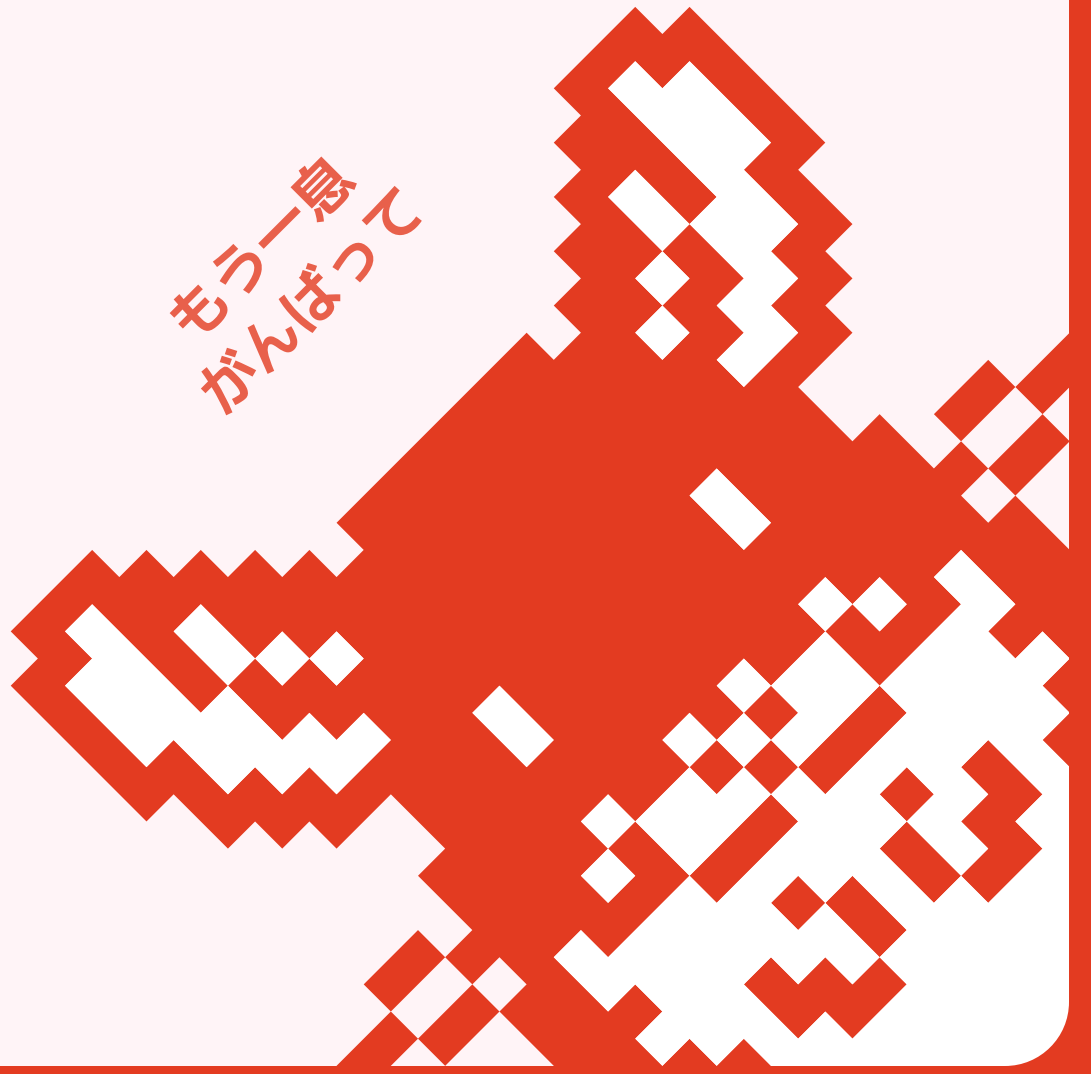
環境変数 DXLIB_ROOT を登録する(4/4)

DXライブラリの場所を登録

- ユーザー環境変数の編集ウインドウで
  変数名に `DXLIB_ROOT` を
  変数値にDXライブラリを置いたフォルダ
  （例: `C:\DxLib_VC`）を入力してください
- Visual Studio を起動中の場合は再起動して
  環境変数を読み込ませてください

DXLIB_ROOT の編集

続いてVisual Studio
プロジェクトを作成して
DXライブラリを使えるよう
設定していきます

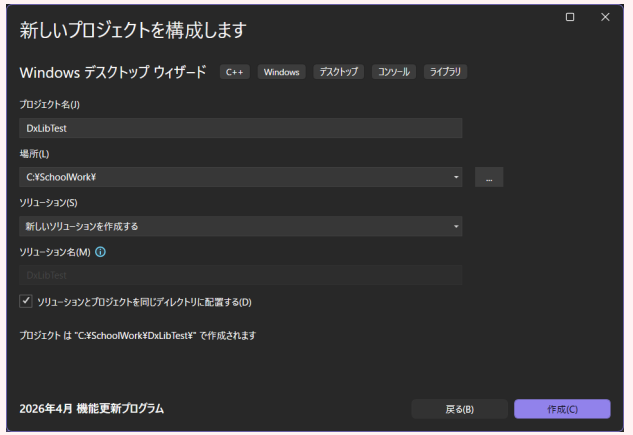


Visual Studioプロジェクトを新規作成する

新しいプロジェクトの作成を選ぶ

1. Visual Studio 2026を起動し、
   テンプレート一覧から
   Windows デスクトップ ウィザード を選ぶ
2. プロジェクト名にわかりやすい名前をつける
   （例: `DxLibTest`）
3. プロジェクトの場所を指定する
   （例: `C:\Dev\DxLib\`）
4. **ソリューションとプロジェクトを同じディレクトリに配置する(D)** にチェックを入れて作成する

新しいプロジェクトのテンプレート選択

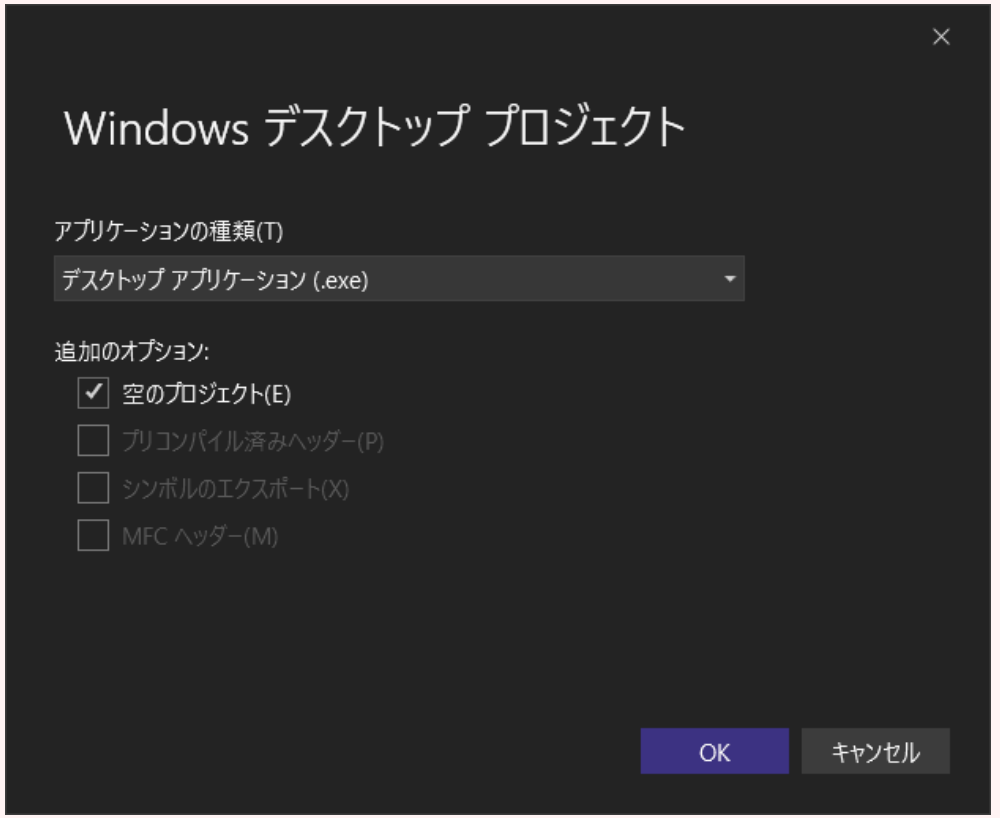


新しいプロジェクトの場所と名前の設定

Visual Studioプロジェクト作成時の設定

テンプレートの選択と設定

1. **Windows デスクトップ プロジェクト**の画面が表示されていることを確認する
2. **アプリケーションの種類(T)** を **「デスクトップ アプリケーション (.exe)」** に変更する
3. **「空のプロジェクト(E)」** にチェックを入れる
4. 右下の **OK** を押してプロジェクトを作成する



Windows デスクトップ プロジェクトの設定画面

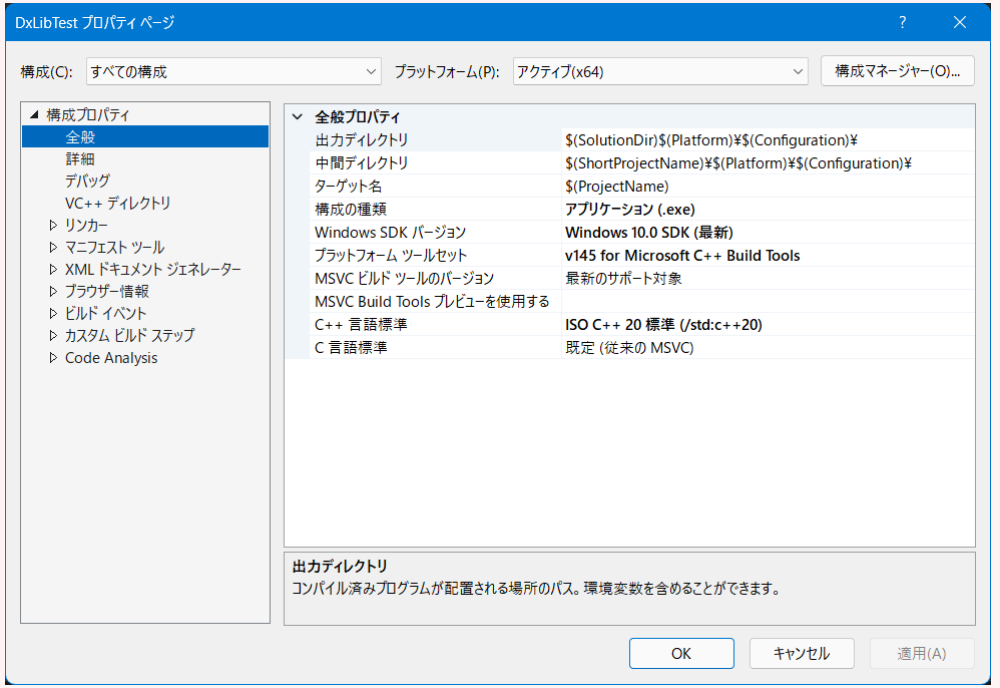
構成とプラットフォームを確認する

プロジェクトのプロパティを開く

1. ソリューションエクスプローラーで
   プロジェクト名を右クリック
   → **プロパティ**を選択する

2. 画面上部の **構成** を「**すべての構成**」に
   プラットフォーム を「**x64**」に変更する

- ※ この資料では64bit版の
  DXライブラリを使用します
  他のプラットフォームを選ぶとリンクエラーが
  発生する場合があります



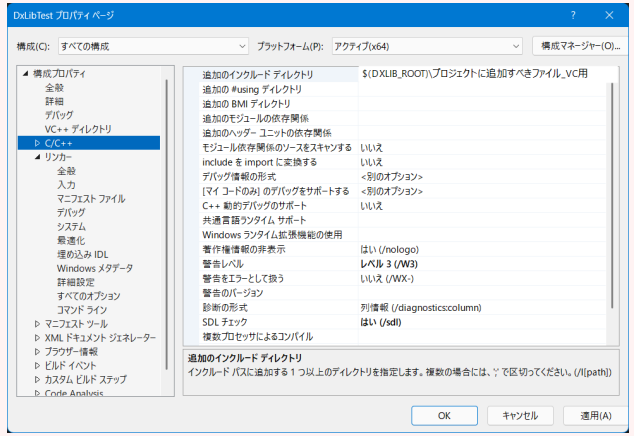
プロジェクトのプロパティ画面

DXライブラリへの参照を登録する

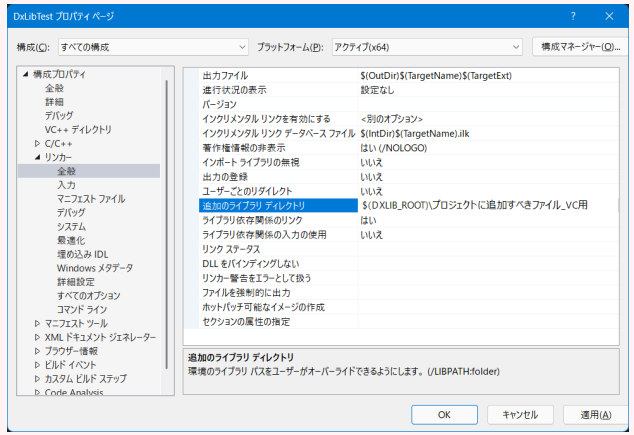
ヘッダファイルとライブラリファイルのディレクトリを設定

1. **C/C++ > 全般 > 追加のインクルードディレクトリ** に `$(DXLIB_ROOT)\プロジェクトに追加すべきファイル_VC用` を追加する
2. **リンカー > 全般 > 追加のライブラリディレクトリ** に `$(DXLIB_ROOT)\プロジェクトに追加すべきファイル_VC用` を追加する

- ※ `DXLIB_ROOT` は各自の環境変数です
  DXライブラリを置いた場所を設定してください



インクルードディレクトリの設定



ライブラリディレクトリの設定

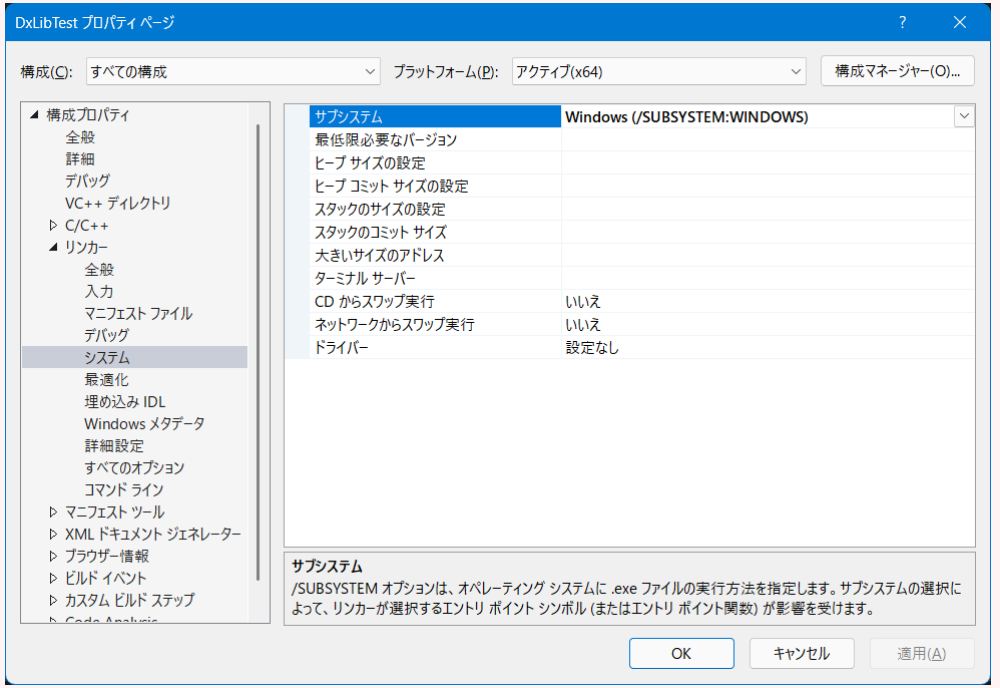
サブシステムを設定する

Windows アプリとして起動する設定

1. **リンカー > システム > サブシステム**

   `Windows (/SUBSYSTEM:WINDOWS)` に変更する

- ※ 初期設定済みの場合がほとんどですが
  Windows 以外に設定されていると
  想定と異なるウインドウ構成で起動したり
  リンクエラーになることがあります

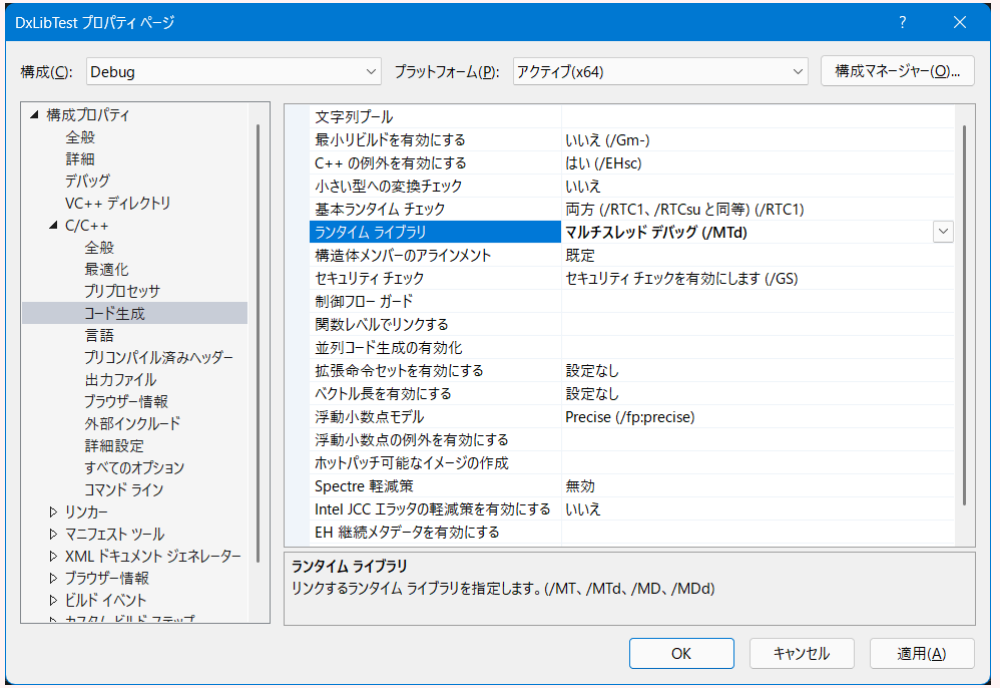


サブシステムの設定

ランタイムライブラリを設定する

Debug / Release ごとに設定

- **C/C++ > コード生成 > ランタイムライブラリ**を開いて、以下のように変更してください
- 構成を「**Debug**」にしてから:
  `マルチスレッド デバッグ (/MTd)` に変更する
- 構成を「**Release**」にしてから:
  `マルチスレッド (/MT)` に変更する
- `Debug` と `Release` は別々の設定です
  注意してください

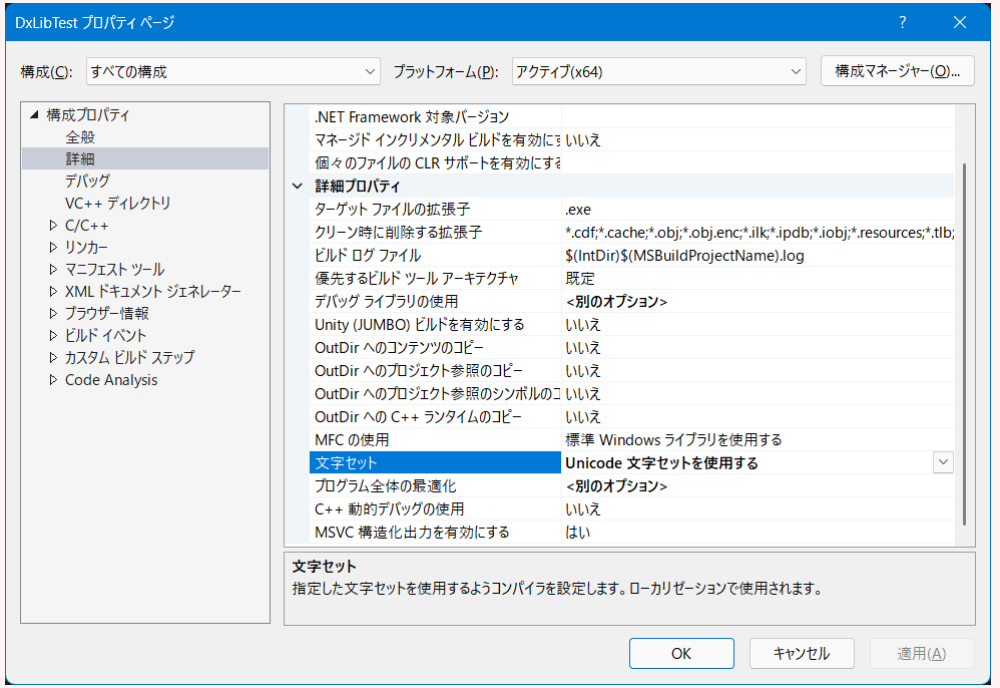


ランタイムライブラリの設定

Unicode 文字セットを設定する

文字列をUnicodeで扱う設定に変更

- **構成プロパティ > 詳細 > 文字セット** を開いて
  `Unicode 文字セットを使用する` に変更する
- この資料では `L"Hello DXLib!"` のような
  Unicode文字列を使用します
- 公式ガイドとは異なりますが
  文字化けを防ぐために必要な設定です
  必ず Unicode 文字セットを使用してください



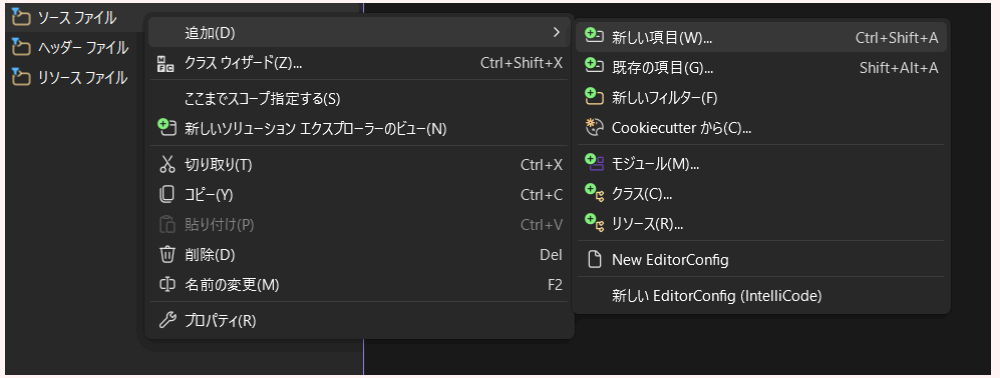
Unicode 文字セットの設定

プロジェクトに C++ ファイルを追加する

最初のソースファイルを作成

- Visual Studio ではソースコードを
  プロジェクトに追加する必要があります
- ここでは最初のソースファイルとして
  `main.cpp` を作成します

1. ソリューションエクスプローラーで
   ソース ファイル を右クリック
2. 「**名前**」に `main.cpp` と入力する
3. 「**追加**」を押してファイルを作成する



ソースファイルに新しい項目を追加するメニュー

main.cpp を追加するダイアログ

ビルドの準備

コードを入力してみましょう

- DXライブラリが正しく動作するかを最小限のコードで確認します
- 先ほど作成した `main.cpp` を開いて右のコードを入力してください

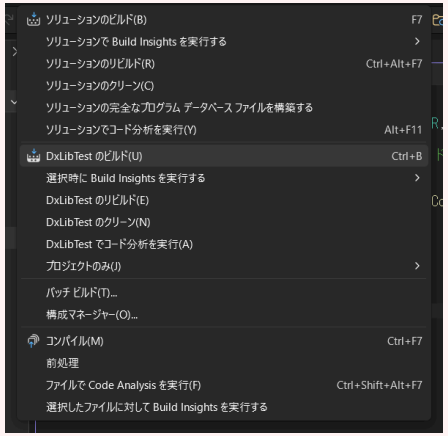
```cpp
#include "DxLib.h"

int WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)
{
    // ウィンドウモードで起動
    ChangeWindowMode(TRUE);

    // 初期化
    if (DxLib_Init() == -1)
        return -1;

    // 画面に文字を表示する
    DrawString(
        100, 100,
        L"Hello DXLib!",
        GetColor(255, 255, 255));

    // キー入力待ち
    WaitKey();

    // 終了処理
    DxLib_End();

    return 0;
}
```

最後の動作確認

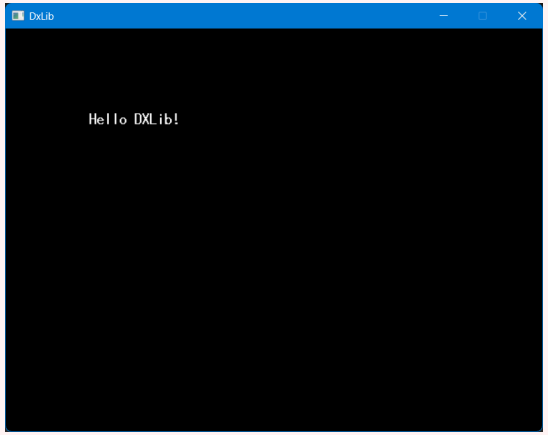
プログラムを実行してみましょう

1. メニューの**ビルド > ソリューションのビルド**で
   作成したプロジェクトをすべてビルドする
2. メニューの**デバッグ > デバッグなしで開始**で
   ビルドしたプログラムを実行する
3. ウィンドウと`Hello DXLib!`の文字が
   表示されれば成功です

- ※ うまく動かなかった場合は
  講師にエラーメッセージを見せてください



ソリューションをビルドするメニュー



Hello DXLib! が表示された実行結果

よくあるトラブルと注意点

以下を確認してください

- **ビルド時にDxLib.h が見つからない →**
  環境変数 `DXLIB_ROOT` と `$(DXLIB_ROOT)\プロジェクトに追加すべきファイル_VC用` が
  正確に入力されているかを確認
- **環境変数を設定したのにパスが見つからない →**
  `DXLIB_ROOT` の値を確認し、Visual Studio を再起動
- **未解決の外部シンボル XXXX と表示される →**
  プラットフォームが `x64` であることを確認
  Debugは `/MTd`、Releaseは `/MT` であることを確認
  追加のライブラリディレクトリが正しいことを確認

おつかれさまでした

これで DXライブラリを
使うための準備が
整いました！！