# Indication de compilation

Cette répertoire est un jeu de Tetris programmé sur platforme Windows. Pour le compiler, nous vous proposons deux solutions.

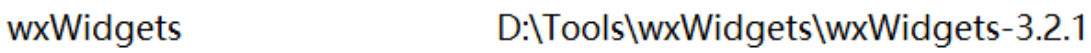
## Compilation par Visual Studio

FORTEMENT RECOMMENDE car on a développé notre projet avec Visual Studio. Les codes sont dans la répertoire `build_vs\Tetris`.

### Outils nécessaires

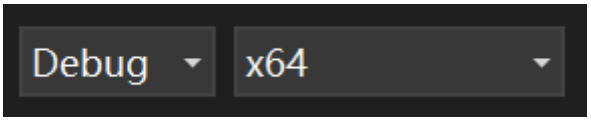
- Visual Studio 2022 avec outils de compilation v143 ou plus nouveaux (normalement Visual Studio 2022 satisfait toutes les demandes). Il faut installer C++ Desktop Development.

- wxWidgets pour Visual C++ (version 3.2.1 préférée). Pour l'installer, vous pouvez :
  - accéder au liens suivant : https://www.wxwidgets.org/downloads/
  - télécharger les source codes sous forme .zip
  - Dans la répertoire décompressée, trouver `wxWidgets-3.2.1\build\msw\wx_vc17.sln` et le lancer avec Visual Studio
  - Choisir le menu Build et clicker sur Batch Build
  - Choisir tous les termes et appliquer
  - Ajouter la répertoire de wxWidgets aux Paths avec le nom dédié (nom du Path wxWidgets) comme l'exemple suivant (sinon la compilation ne marche pas) :



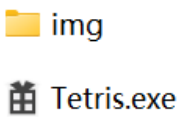

### Compilation

Entrer dans la répertoire `build_vs` et lancer `Tetris.sln` avec Visual Studio. En haut de la fenêtre, choisir la configuration de compilation comme l'exemple suivant :




Mode Debug est préférée et vous pouvez choisir x64 ou win32 par rapport à votre machine. Si vous voulez compiler sous mode Release, quelque lignes dans le fichier `MyButton.cpp` doivent être supprimées. Dans ce cas, vous pouvez référer aux commentaires dans le fichier.
Ensuite vous trouvez Build dans le menu et appliquez.
Après compilation, vous trouverez la répertoire `build_vs\x64\Debug` ou `build_vs\Debug` selon la configuration et l'éxecutable est dedans.

La dernière étape avant de lancer est de copier-coller la répertoire `build_vs\Tetris\img` vers la même répertoire que l'éxecutable :




Les images sont nécessaire pour l'interface utilisateur.

# Compilation par MinGW

Cette méthode de compilation n'est qu'un choix alternatif et n'est PAS recommendée. La librairie pour MinGW n'est pas la même que pour VC++ et le programme compilé perd toute sa disposition de l'interface utilisateur. Les codes sont dans la répertoire `build_mingw\src`. ATTENTION, les codes sont un peu différents que les codes pour VS et ne peuvent pas être compiler avec VS sans aucun changement !

## Outils nécessaires

- MinGW de version stable plus nouvelle. N'oubliez pas de mettre sa répertoire au Path pour utiliser `mingw32-make.exe`.
- wxWidgets pour gcc (version 3.2.1 préférée). Pour l'installer, les étapes de téléchargement et configuration du Path sont les même, mais pour compiler la librairie :
  - Dans la répertoire décompressée, entrer `wxWidgets-3.2.1\build\msw` et lancer :
    `mingw32-make.exe -f .\makefile.gcc`

  et patienter pour la compilation qui prend très long temps.

## Compilation

Entrer dans `build_mingw` et lancer :

```
mingw32-make.exe all
```

et vous trouverez l'éxecutable sous `build_mingw`.

# Version déjà compilée

Au cas où aucune méthode ne marche, vous pouvez trouver 4 version compilées avec Visual Studio dans la répertoire `executables`. Choisissez-vous x64 ou x86 selon votre machine et les versions Debug sont plus complètes.

# Commentaires

Dans les compilations vous pouvez voir les warnings qui indiquent que certaines fonctions sont déprises. Ce problème empêche la compilation de la version Release. Cependant, si on utilise une autre structure pour cette partie qui n'utilise pas les fonctions déprises, la version Debug qui offre une jolie interface utilisateur ne marche plus. Pour avoir un bon programme compilé, on a décidé de garder la réalisation avec ces fonctions. Si vous rencontrez les mêmes warnings, négligez-les.