Installieren und Verwenden von wxWidgets

Klaus Kusche, Juni 2025

Ich habe ***wxWidgets 3.3.x noch nicht*** getestet,
daher bitte bis auf weiteres die ***Version 3.2.x verwenden*** (aktuell 3.2.8)!

1.) CodeBlocks Vers. 20.03 mit *integriertem* MinGW 8.1.0, offizieller WxWidgets-Download

1.) Wir brauchen ein ***vorkompiliertes*** **wxWidgets-*Dev*-Paket** (zum Entwickeln, mit *Headerfiles* und *Libraries*), *nicht* nur die Runtime und *nicht* nur die Sourcen! Der lokale C/C++-Compiler und das vorkompilierte wxWidgets-Dev-Paket müssen ***versionsmäßig exakt zusammenpassen***!!!

- In den Rechnerräumen der DHGE sollte irgendeine Version bereits fertig ausgepackt auf den Rechnern liegen:
  - Auf *neu installierten Rechnern* in **`C:\Library\wx...`**
  - Auf Systemen mit älterer Installation eventuell direkt unter **`C:\`** oder **`D:\`**
- Ich habe ein *alternatives wxWidgets-Paket* …**`Computerix`** gebaut, siehe *Kapitel 3!*
- Bitte *prüfen*, ob das vorhandene wxWidgets *zur CodeBlocks-Version passt*:
  …**`gcc810`**… ist für CodeBlocks **20.03**, für 25.03 braucht man ***mein*** …**`gcc1420`**…-Paket!
  Die Pakete funktionieren nur mit dem MinGW-**`gcc`**, den CodeBlocks *mitinstalliert*!
- Sonst für CodeBlocks Version 20.03, das intern als Compiler die *MinGW-Portierung* des **`gcc`** in der *Version 8.1 (64 bit)* verwendet, von https://github.com/wxWidgets/wxWidgets/releases/ folgende *zwei Files* herunterladen (dabei 3.2.8 durch die *aktuelle 3.2.x-Version* ersetzen, es sollte jede 3.2.x funktionieren, zur jeweiligen Version *hinunterscrollen* und dort den *Punkt „Assets“ aufklappen*, dort liegen die Download-Links):

  https://github.com/wxWidgets/wxWidgets/releases/download/v3.2.8/wxMSW-3.2.8_gcc810_x64_Dev.7z

  https://github.com/wxWidgets/wxWidgets/releases/download/v3.2.8/wxWidgets-3.2.8-headers.7z

  Eine **alternative Download-Quelle** ist
  https://sourceforge.net/projects/wxwidgets.mirror/files/v3.2.8/
- Ein *neues, sinnvoll benanntes Verzeichnis* anlegen (z.B. **`wxw-3.2.8-gcc810`**).
  Zur Minimierung des Klick-Aufwandes beim Einrichten in CodeBlocks empfehle ich:
  - *Entweder möglichst weit oben* in der Ordner-Hierarchie (direkt unter **`C:`**, **`D:`** oder **`Z:`**)
    (wenn man wxWidgets für *mehrere* CodeBlocks-Projekte nutzen will)
  - Oder *direkt im Projekt-Ordner* des CodeBlocks-Projektes
    (wenn man die wxWidgets *nur für ein Projekt* nutzen will)

- Die beiden Zip-Files dort hinein auspacken, sodass das Verzeichnis einen **`include`**-Ordner und einen **`lib`**-Ordner (mit Unter-Ordnern) enthält.

2.) CodeBlocks starten und ein ***neues Konsol-Projekt*** (***"Console application"***) mit *Sprache C++* anlegen (***kein*** wxWidgets-Projekt, der CodeBlocks wxWidgets-Wizard kommt mit der aktuellen wxWidgets-Verzeichnisstruktur nicht zurecht!).

Der *Projektname* (wird auch Name des **`.exe`**-Files!) und alle im Projekt verwendeten Datei und Ordner-Namen sollen *keine Umlaute oder Zwischenräume* enthalten!

Es reicht, die *Debug-Konfiguration* auszuwählen (kein Release nötig).

3.) Zum Testen mein http://www.computerix.info/examples/wxw-hello.cpp o.ä. herunterladen und den Code in das **`main.cpp`** des Projektes hineinkopieren oder den File anstelle von **`main.cpp`** in das Projekt aufnehmen.

4.) In den ***Compiler-Einstellungen*** des Projektes (links in der Projekt-Ansicht: Rechtsklick auf das Projekt selbst, *"Build options..."*):

a) Zuerst auf das *dritte* Karteiblatt *"Search directories"* gehen und dort bei *"Compiler"* mit [Add] *zwei Verzeichnisse* der wxWidgets-Installation eintragen (in *dieser Reihenfolge!*):

wxWidgets-Installationspfad`\lib\gcc810_x64_dll\mswu`
wxWidgets-Installationspfad`\include`

Wenn das *wxWidgets-Installationspaket* irgendwo *zentral* auf **`C:`** oder **`D:`** liegt: ***"nein"*** ankreuzen, wenn CodeBlocks fragt, ob es die *Pfade relativ speichern* soll. Wenn die wxWidgets-Installation direkt *im Projektordner* liegt, oder wenn sie gemeinsam mit dem Projekt auf demselben *Stick oder Netzlaufwerk* liegt: ***"ja"*** ankreuzen!

b) Genauso bei *"Search directories"* / *"Linker"* das *Lib-Verzeichnis* der wxWidgets-Installation eintragen:

wxWidgets-Installationspfad`\lib\gcc810_x64_dll`

c) Dann auf das vorige Karteiblatt "*Linker Settings*" gehen und dort im rechten Feld *"Other linker options:"* Folgendes eintragen:

```
-lwxmsw32u_core -lwxbase32u
```

Für komplexe wx-Projekte braucht man *eventuell weitere Libraries*, aber für unsere einfachen Programme sollten diese beiden ausreichen.

d) Bei den *„Compiler Settings"* (*erstes* Karteiblatt) als *Sprachstandard* entweder **C++11** oder **C++14** *ankreuzen.*

5.) Mit diesen Einstellungen sollte sich eine Musterlösung bzw. dein Programm fehlerfrei compilieren, linken und ***im CodeBlocks*** (Run-Button) starten lassen.

2.) Verwendung in einem cmd-Fenster

Wenn man den **.exe** ***außerhalb*** *von CodeBlocks* in einem **cmd**-Fenster
bzw. *ohne die wxWidgets- und MinGW-Installation* starten will
(oder z.B. *auf einen Stick bzw. anderen Rechner* kopieren will),
scheitert das Ausführen mit zahlreichen **.dll**-Fehlern.

wxWidgets liefert leider nur **.dll**'s mit, *keine* Libraries zum statisch Linken.
In *„Build options"* / *„Compiler Settings"* / *„Compiler Flags"* die drei Optionen
„Static …" ankreuzen hilft daher *nicht*.

Es gibt drei Möglichkeiten, das Problem zu lösen:

1. **Mit *meinem* wxWidgets-Download:**

   Ich habe ein wxWidgets-Paket für CodeBlocks 20.03 gebaut, das sich nur ***statisch*** *linken* lässt. Es ist genauso wie mein wxWidgets-Paket für CodeBlocks Version 25.03 (siehe Kapitel 3 !!) herunterzuladen, zu installieren und in CodeBlocks zu verwenden.

   Die beim Kompilieren entstehenden **.exe**-Files sind zwar *riesig*,
   aber sollten *standalone ausführbar* sein (*ohne* wxWidgets und *ohne* MinGW).

2. **Mit dem offiziellen wxWidgets-Download aus Kapitel 1 und kopierten .dll's:**

   Man muss:

   - Folgende *zwei* **.dll** aus *wxWidgets-Installationspfad***lib\gcc810_x64_dll**
     in dasselbe Verzeichnis kopieren, in dem auch der **.exe** liegt:

     wxbase32u_gcc810_x64.dll und **wxmsw32u_core_gcc810_x64.dll**

   - Folgende *drei* **.dll**'s aus **C:\Programme\CodeBlocks\MinGW\bin**
     ebenfalls zum **.exe**-File *kopieren*:

     libgcc_s_seh-1.dll , **libstdc++-6.dll** und **libwinpthread-1.dll**

   Falls das nicht reicht und sich Windows bei Starten des Programms
   immer noch über fehlende **.dll**-Files beschwert:
   Die in der Fehlermeldung angezeigten **.dll**'s in den beiden Verzeichnissen oben
   oder in …**\MinGW\x86_64-w64-mingw32\lib** suchen
   und ebenfalls in das **.exe**-Verzeichnis kopieren.

   Auch mit dem Tool "*Dependencies*" (https://github.com/lucasg/Dependencies)
   kann man anzeigen, welche **.dll**-Files ein **.exe** wirklich braucht.

3. **Mit dem offiziellen wxWidgets-Download aus Kapitel 1 und erweitertem PATH:**

   Stattdessen kann man im **cmd**-Fenster den **PATH** so *erweitern*, dass er *vorne*
   das wxWidgets-**dll**-Verzeichnis und das MinGW-**bin**-Verzeichnis enthält.

   Mit *Administratorrechten* kann man auch den *systemweiten* **PATH** erweitern.
   Das **cmd**-Fenster muss *danach neu geöffnet* werden, damit die Änderung wirkt.

3.) Meine WxWidgets-Pakete für CodeBlocks Vers. 25.03 mit *integriertem* MinGW 14.2.0 und CodeBlocks Vers. 20.03 mit *integriertem* MinGW 8.1.0

Für CodeBlocks Version 25.03, das intern als Compiler
die *WinLibs-Portierung* des **gcc** in der *Version 14.2 (64 bit)* verwendet,
gab es zum Zeitpunkt dieser Anleitung (Juni 2025) noch ***keinen fertigen Download***:
Das Paket **`wxMSW-3.2.8_gcc1420_x64_Dev.7z`** auf der Download-Seite ist
trotz gleicher **gcc**-Versionsnummer ***nicht** kompatibel* zum MinGW in CodeBlocks 25.03,
es nutzt eine andere **gcc**-Portierung (es soll aber ein kompatibles Paket kommen?).

Ich habe für CodeBlocks Version 25.03 ein eigenes Paket gebastelt:
https://www.computerix.info/wx/wxMSW-3.2.8_gcc1420_Computerix.7z

Und ich habe auch ein gleichartiges Paket für CodeBlocks 20.03 mit gcc 8.1.0 gebaut:
https://www.computerix.info/wx/wxMSW-3.2.8_gcc810_Computerix.7z

Beides sind *Pakete zum **statischen** Linken* (siehe Kapitel 2),
deshalb sind in CodeBlocks etwas ***andere Projekt-Einstellungen*** nötig.

Das Paket einfach *analog zu Kapitel 1* irgendwohin (z.B. ganz oben oder ins Projekt)
entpacken und *in das Projekt eintragen*:

- In meinen Paketen heißt das *Unterverzeichnis* von **`lib`** einfach **`lib\gcc_lib`**
  statt **`lib\gcc810_x64_dll`** o.ä.. Die Version habe ich weggelassen, und es handelt
  sich ja *nicht* um einen Build mit **`dll`**'s, sondern um einen Build mit statischen **`lib`**'s.

  Davon abgesehen ist die Verzeichnis-Struktur ident, und damit auch die *Suchpfade*,
  die man im CodeBlocks für den *Compiler und Linker* eintragen muss.

- Es müssen bei den *Linker Options* viel mehr Libraries zum Linken angegeben werden:
  `-static -static-libgcc -static-libstdc++ -lwxmsw32u -lwxpng -lwxzlib -ladvapi32 -lcomctl32 -lcomdlg32 -lgdi32 -lgdiplus -limm32 -lkernel32 -lmsimg32 -lole32 -loleaut32 -lopengl32 -lrpcrt4 -lshell32 -lshlwapi -luser32 -luuid -luxtheme -lversion -lwininet -lwinmm -lwinspool -lws2_32 -loleacc`

- Das Ankreuzen von **C++11** oder **C++14** als Sprachstandard bei den *„Compiler Settings“*
  (*erstes* Karteiblatt) ist bei *CodeBlocks 25.03 zwingend notwendig*,
  da CodeBlocks 25.03 per Default einen *höheren* C++-Standard verwendet,
  der beim Compilieren *Fehler* liefert!

4.) wxWidgets mit Visual Studio Code, CLion usw.

1.) Ein *wx-Paket* herunterladen und installieren, das *genau zum benutzten Compiler passt*:

- Unter
  https://github.com/wxWidgets/wxWidgets/releases/
  oder
  https://sourceforge.net/projects/wxwidgets.mirror/files/v3.2.8/
  liegen *vorkompilierte Pakete* für mehrere **gcc**-*Versionen*
  und *Visual-Studio-Versionen* (im **README** steht Genaueres zu den Versionen!).
- Oder den MinGW-**gcc** *aus CodeBlocks* zum Compilieren verwenden
  und eine *wx-Installation gemäß Kapitel 1 oder 3* nutzen.

  Achtung bei Verwendung des MinGW-**gcc** aus CodeBlocks, egal ob Cmake bzw. **make** oder **.bat**: Das **...\MinGW\bin**-Verzeichnis muss im **PATH** sein!

2.) Entweder am Internet nach entsprechenden Hinweisen für Cmake o.ä. suchen oder mit einem **.bat**-File compilieren.
Dazu an der **Befehlszeile im Kapitel 5** orientieren:

- Statt **$(wx-config --cflags)**
  werden die *Include-Pfade* mit **-I** angegeben, beispielsweise:
  -I*wxWidgets-Installationspfad***\lib\gcc810_x64_dll\mswu**
  -I*wxWidgets-Installationspfad***\include**
- Statt **$(wx-config --libs)**
  werden die *Library-Pfade* mit **-L** und *alle notwendigen Libraries* angegeben, zum Beispiel:
  -L*wxWidgets-Installationspfad***\lib\gcc810_x64_dll**
  -lwxmsw32u_core -lwxbase32u
- Bei den *anderen Compiler-Optionen* gehört u.a. **-std=c++11** dazu.

Also muss der Compiler-Aufruf in etwa so aussehen (alles in *einer* Zeile!):
g++ *andere Compiler-Optionen* **-std=c++11**
-I*wxWidgets-Installationspfad***\lib\gcc810_x64_dll\mswu**
-I*wxWidgets-Installationspfad***\include**
-o *exe-File-Name* *alle .cpp-Files nacheinander*
-L*wxWidgets-Installationspfad***\lib\gcc810_x64_dll**
-lwxmsw32u_core -lwxbase32u

5.) wxWidgets unter Linux und am Mac

- Man braucht das *Entwickler-Paket* der jeweiligen Linux-Distribution für wxWidgets, meist mit …**`devel`**… im Namen (nicht nur das Laufzeit-Paket).

  Auch auf dem Mac empfehle ich, ein *vorkompiliertes wxWidgets-Paket* zu suchen und zu installieren, zum Beispiel von:
  https://formulae.brew.sh/formula/wxwidgets (das haben die Studenten meist genutzt)
  https://anaconda.org/conda-forge/wxwidgets/files (ein halb-offizieller wx-Mac-Build)
  https://ports.macports.org/port/wxWidgets-3.2/ (hat auch schon einmal funktioniert)

  Zum selbst Kompilieren findet man zwar zahlreiche Anleitungen am Internet, und es funktioniert normalerweise auch gut, aber es dauert, eventuell stundenlang!

- Das Compilieren auf der *Befehlszeile* sollte wie folgt funktionieren (in *einer* Zeile!):

  `g++` *andere Compiler-Optionen* **`$(wx-config --cflags) -o`** *exe-File-Name*
  alle .cpp-Files nacheinander **`$(wx-config --libs)`**