Compiling Orx with MinGW-w64/gmake on Windows

There are 5 steps required to build Orx using gmake

1. Getting Orx from source
2. Getting MinGW-w64 (not mingw32)
3. Updating the PATH environment variable
4. Creating a build project for gmake
5. Compiling

Getting Orx from source

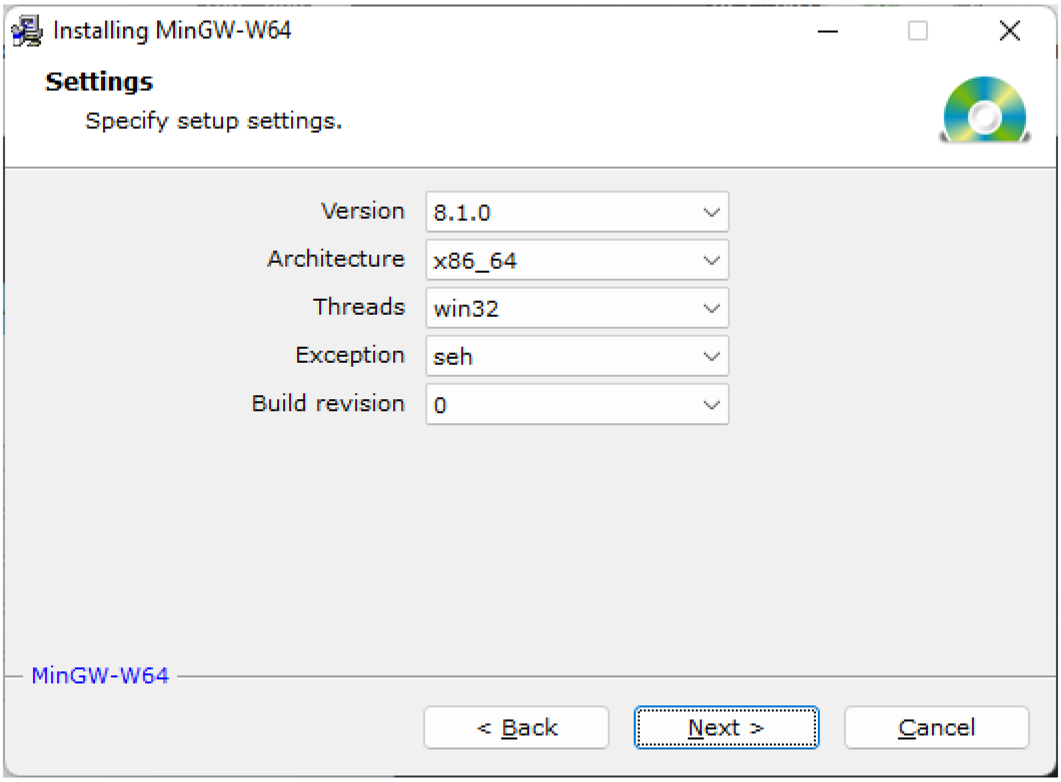
This step is already covered under cloning orx. Please follow those instructions first.

Getting MinGW-w64

Orx works with MinGW-w64. Do not download the Mingw32 project, Orx does not support this compiler anymore. You can download the installer from: https://sourceforge.net/projects/mingw-w64/files/Toolchains%20targetting%20Win32/Personal%20Builds/mingw-builds/installer/mingw-w64-install.exe/download

> The current minimum version of MinGW supported by Orx is 8.1.0.

- Run the `mingw-w64-install.exe` file.
- If you want to install the compiler for a 64-bit operating system, select the following options:

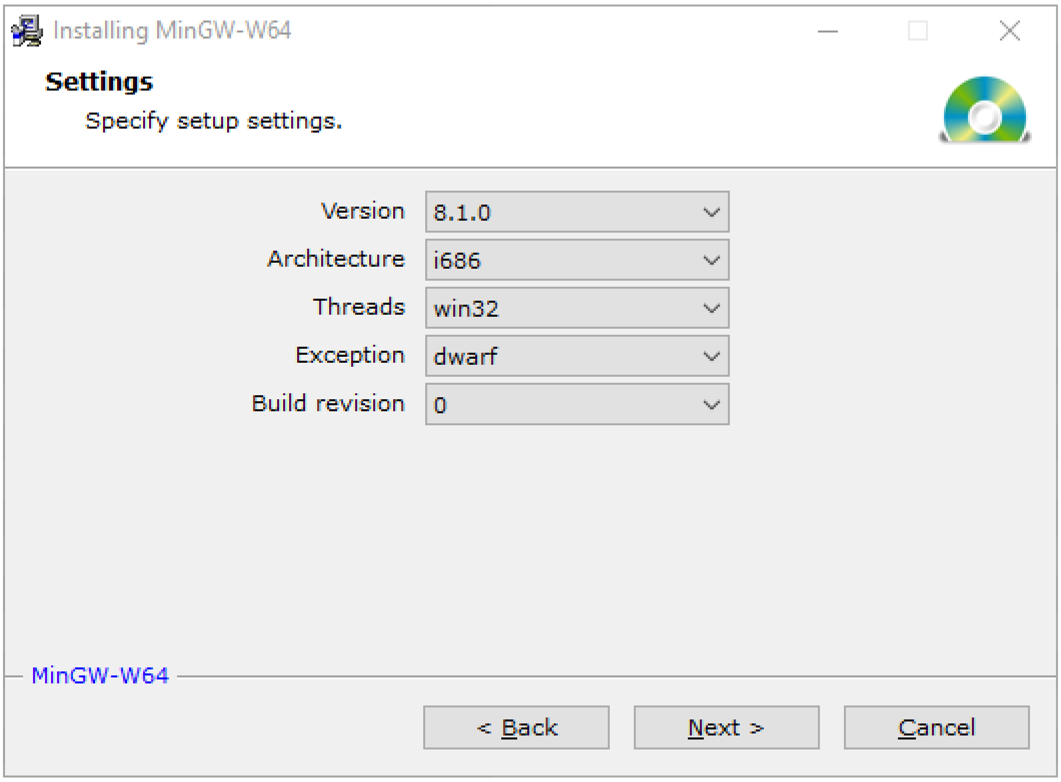


Settings

Specify setup settings.

Version: 8.1.0
Architecture: x86_64
Threads: win32
Exception: seh
Build revision: 0

- Choose an installation directory. To avoid dramas, choose a folder without spaces in the path. Something like `C:\MinGW64\x86_64-8.1.0-win32-seh` is best.
- Continue with the rest of the installation.
- If you want to install the compiler for a 32-bit operating system, select the following options:



Settings

Specify setup settings.

Version: 8.1.0
Architecture: i686
Threads: win32
Exception: dwarf
Build revision: 0

- Choose an installation directory. To avoid dramas, choose a folder without spaces in the path. Something like `C:\MinGW64\i686-8.1.0-win32-dwarf` is best.

- Continue with the rest of the installation.

See here for troubleshooting tips: https://stackoverflow.com/questions/46455927/mingw-w64-installer-the-file-has-been-downloaded-incorrectly

> The version of the compiler you chose above is regarding the toolchain itself, ie. on which host it's going to run (32bit Windows vs 64bit Windows). Please note all the versions of MinGW-w64 can build both 32bit and 64bit applications.

Updating the PATH environment variable

1. Edit the System Environment Variables
2. Add your MinGW bin folder locations into the PATH list, ie: `C:\MinGW64\x86_64-8.1.0-win32-seh\bin` and `C:\MinGW64\i686-8.1.0-win32-dwarf\bin` (depending where you installed it)

Creating a build project for gmake

When first cloning Orx, this project should already be created for you at: `C:\[somewhere]\orx\code\build\windows\gmake`.

If not, run the `setup.bat` script in the root of the Orx repo folder.

If you need to run it, the expected output is:

```
Building configurations...
Running action 'gmake'...
Generating windows/gmake/Makefile...
Generating windows/gmake/orx.make...
Generating windows/gmake/orxLIB.make...
Generating windows/gmake/Bounce.make...
Done.
```

Compiling

1. cd into the gmake folder
2. mingw32-make (to compile the default configuration - debug/64-bit)

Expected output

```
"==== Building orxLIB (debug) ===="
Creating ../../../lib/dynamic
Creating obj/Debug/orxLIB
orxPlugin_EmbeddedList.cpp
orxAnim.c
orxAnimPointer.c
orxAnimSet.c
orxModule.c
orxType.c
orxClock.c
orxCommand.c
orxConfig.c
orxConsole.c
orxEvent.c
orxLocale.c
orxResource.c
orxSystem.c
orxThread.c
orxDebug.c
orxFPS.c
orxProfiler.c
orxDisplay.c
orxFont.c
orxGraphic.c
orxScreenshot.c
orxText.c
orxTexture.c
orxFile.c
orxInput.c
orxJoystick.c
orxKeyboard.c
orxMouse.c
orxParam.c
orxMath.c
orxVector.c
orxBank.c
orxMemory.c
orxFrame.c
orxFX.c
orxFXPointer.c
orxObject.c
orxSpawner.c
orxStructure.c
orxTimeLine.c
orxBody.c
orxPhysics.c
orxPlugin.c
orxCamera.c
orxRender.c
orxShader.c
```

```
orxShaderPointer.c
orxViewport.c
orxSound.c
orxSoundPointer.c
orxSoundSystem.c
orxHashTable.c
orxLinkList.c
orxString.c
orxTree.c
Linking orxLIB
Running post-build commands
cmd /c copy /Y ..\..\..\lib\dynamic\orx*.dll ..\..\..\bin
..\..\..\lib\dynamic\orxd.dll
        1 file(s) copied.
"==== Building orx (debug) ===="
Creating obj/Debug/orx
orxMain.c
Linking orx
"==== Building Bounce (debug) ===="
Creating ../../../bin/plugins/demo
Creating obj/Debug/Bounce
orxBounce.c
Linking Bounce
```

Your Orx Library

If you check in the C:\[somewhere]\orx\code\lib\dynamic folder, you'll see an liborxd.a and orxd.dll

Other configurations and 32-bit / 64-bit

If you wish to build the other configurations, this is the complete list:

- mingw32-make config=debug64
- mingw32-make config=profile64
- mingw32-make config=release64
- mingw32-make config=debug32
- mingw32-make config=profile32
- mingw32-make config=release32

Troubleshooting

Q. You get no errors, but no information either, just output like:

```
"==== Building orxLIB (debug) ===="
"==== Building orx (debug) ===="
```

```
"==== Building Bounce (debug) ===="
```

A. Try compiling again with debug messages again to get all possible information:

`mingw32-make -d config=debug64` or `mingw32-make -d config=debug32`

Q. If you cannot compile, and you see something in the messages or logs regarding: C:\Program Files (x86)\Git\bin\sh.exe

A. This is because you have installed Git Bash and it has added C:\Program Files (x86)\Git\bin to your windows path. Therefore git binaries are available to your windows console and are clashing with mingw. Remove the path from your Environment Variables:

`C:\Program Files (x86)\Git\bin`

Reopen your Windows Console and try compiling again.

Q. You don't get any specific errors in the debug output.

A. Ensure you passed the correct config value. It must be either `debug32`, `profile32`, `release32`, `debug64`, `profile64`, `release64` or not specified. If you typed any other value, you will get empty output like:

```
"==== Building orxLIB (wrong) ===="
"==== Building orx (wrong) ===="
"==== Building Bounce (wrong) ===="
```

From:
https://wiki.orx-project.org/ - **Orx Learning**

Permanent link:
https://wiki.orx-project.org/en/tutorials/orx/windows/compiling_orx_with_mingw32_gmake

Last update: **2022/06/28 23:08 (21 months ago)**