Compile Bforartists at Windows 7 with Cmake and Visual Studio 2013

Overview

Introduction

This is a step by step tutorial packed with tips and tricks to help you compile your first Bforaritsts build – opening doors to develop your artistic ideas with the sourcecode.

Outline

To compile, we need to gather the correct tools, materials, blueprints, and then we can build the “house” - which is **Bforartists**. The following tutorial will explain each process step by step with tips and tricks to make sure you can troubleshoot and build without any errors.

Once you have no errors in the building process, you will be free to work in a Github branch and commit and push new improvements to Bforartists – with set tasks that can be worked on collaboratively and approved to be included in future official releases.

So let’s begin by getting the right tools, then the correct materials and blueprints – then we can use the tools to build our house.

Bforartists - This File is Public Domain

Enjoy!

Tool Requirements

Required Tools

In order to build Bforartists, you need building “tools” like **compilers,** and **sourcing tools** for the “materials” and “blueprints” like the **repository** and **libraries,** and other third party “resources” that are classified as **dependencies.**

Building Tools

Microsoft Visual Studio 2013 Community. https://my.visualstudio.com/Downloads?q=visual%20studio%202013&wt.mc_id=o~msft~vscom~older-downloads

Quick Tip: *While Microsoft Visual Studio 2015 Community is supported, official builds are still done using 2013. And 2015 made big trouble here. So you better keep your hands away from the 2015 version for now*

Cmake: http://www.cmake.org/download/

Sourcing Tools

Git: https://git-scm.com/
Tortoisegit: https://tortoisegit.org/

Quick Tip: *You can of course also do everything from the Git command line. But Tortoisegit makes things much easier with a GUI.*

TortoiseSVN for the dependencies. http://tortoisesvn.net/index.de.html

Other Tools

Python 2.7 : https://www.python.org/downloads/
Python 3.5 : https://www.python.org/downloads/release/python-353/
CUDA 9.1.85 Developer Tools: https://developer.nvidia.com/cuda-91-download-archive

Quick Tip: *Hitting “Next” “Next” “Next” to install the above is generally ok and risk free.*

Optional Tools

We need an installed version of Blender in case you want to modify the icons. You need the Windows Installer version, the one with the *.msi ending. Make sure that the path to the Blender.exe is added to the Windows system variables http://download.blender.org/release/

Quick tip: *Check out Page 9 to learn how to add System Variables*

7Z or a similar zip software that can handle tar.gz archives. http://www.7-zip.org/

Nice to have but not necessary is **Grepwin** to search the source code:
http://stefanstools.sourceforge.net/grepWin.html

Same counts for software like **Meld**: http://meldmerge.org/ or **Atom:** https://atom.io/ to have an overview, syntax highlight and document comparison.

Downloading the Repository

So we have downloaded and installed the tools – one after the other. Now it's time to download the repository (the blueprints) and get the libs (the materials).

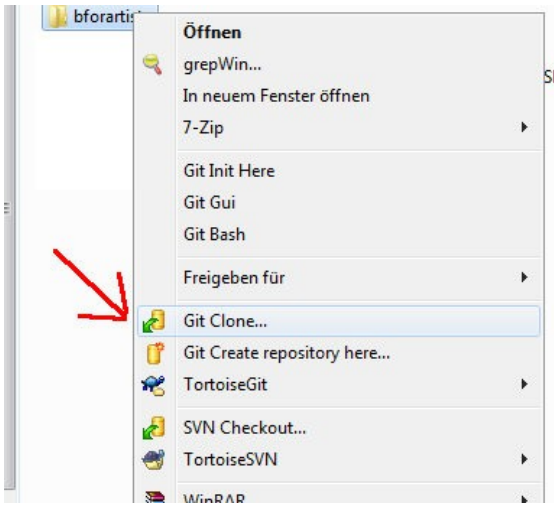
1. Choose a location.

Quick Tip: *When possible a folder directly at a drive.*

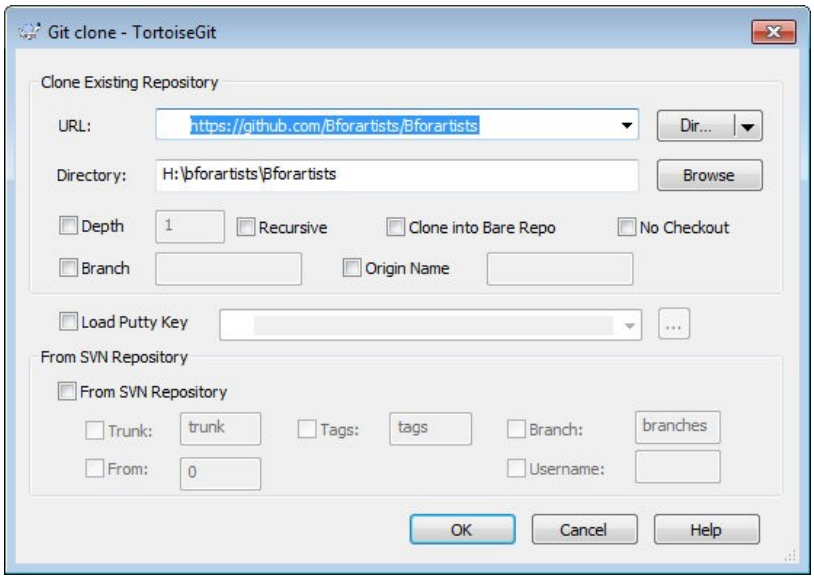
I have chosen drive **H:**, and have created the folder **bforartists**, which is our target folder.

2. Open your Browser and navigate to https://github.com/Bforartists/Bforartists . This makes the repository URL recognized by Git.

3. Then right click at our target folder and choose git clone in the rmb menu.



4. We have navigated to the Bforartists Github repository before in step 2. , so Git should have copied the right link into the URL field already. If not, then type in the URL manually. https://github.com/Bforartists/Bforartists



5. Click okay. Git will download the Bforartists repository now.

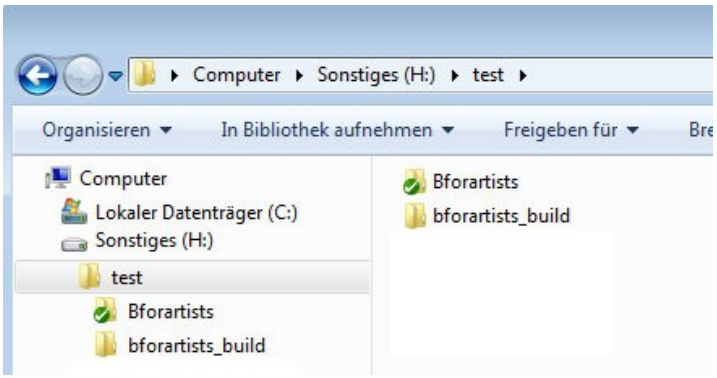
6. When everything goes well then we have the following result (notice the green tickbox):

Creating the Binary Folder

On with some more preparations.

We need a folder where **Cmake** builds the Binaries and where **Visual Studio** compiles the result. In order to do so, create another folder besides the repository.

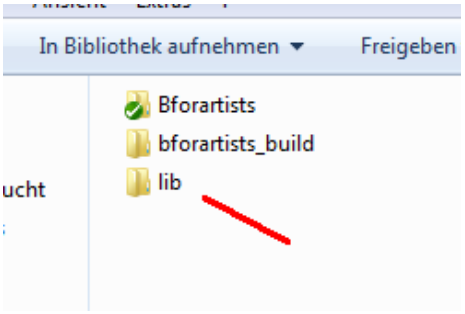
You can name this folder however you like. I will call it **bforartists_build**.



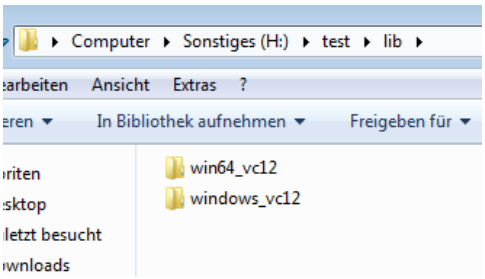
Installing the Dependencies

What is missing now are the dependencies, or “materials”. Bforartists is a fork of Blender, and uses the same dependencies. It doesn't host and update this dependencies though. We use the official dependencies from Blender.

1. Create a folder called **lib** beside the repository. This name is important, it has to be **lib**!



2. Inside the lib folder create a folder called **windows_vc12** when you want to build a 32 Bit version of Bforartists. Or a folder called **win64_vc12** when you want to build a 64 Bit version. Or both. These Names are important!



Quick Tip: *the 32 Bit version of Bforartists is quickly becoming outdated and not supported officially. It is recommended you build in the 64 Bit version.*

3. Now, right click on the **win64_vc12** folder, and choose **SVN Checkout**.

We could also do this by command line.

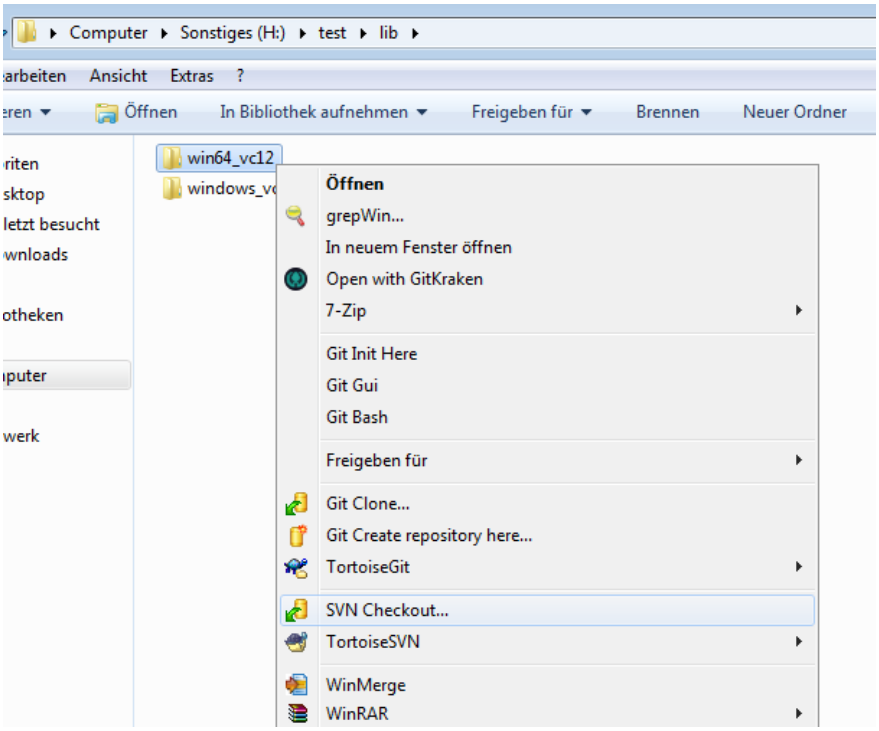
For 64 Bit :

```
svn checkout https://svn.blender.org/svnroot/bf-blender/trunk/lib/win64_vc12
lib/win64_vc12
```

For 32 Bit:

```
svn checkout https://svn.blender.org/svnroot/bf-blender/trunk/lib/windows_vc12
lib/windows_vc12
```

But we have SVN installed, so we use SVN Checkout via the right click GUI.

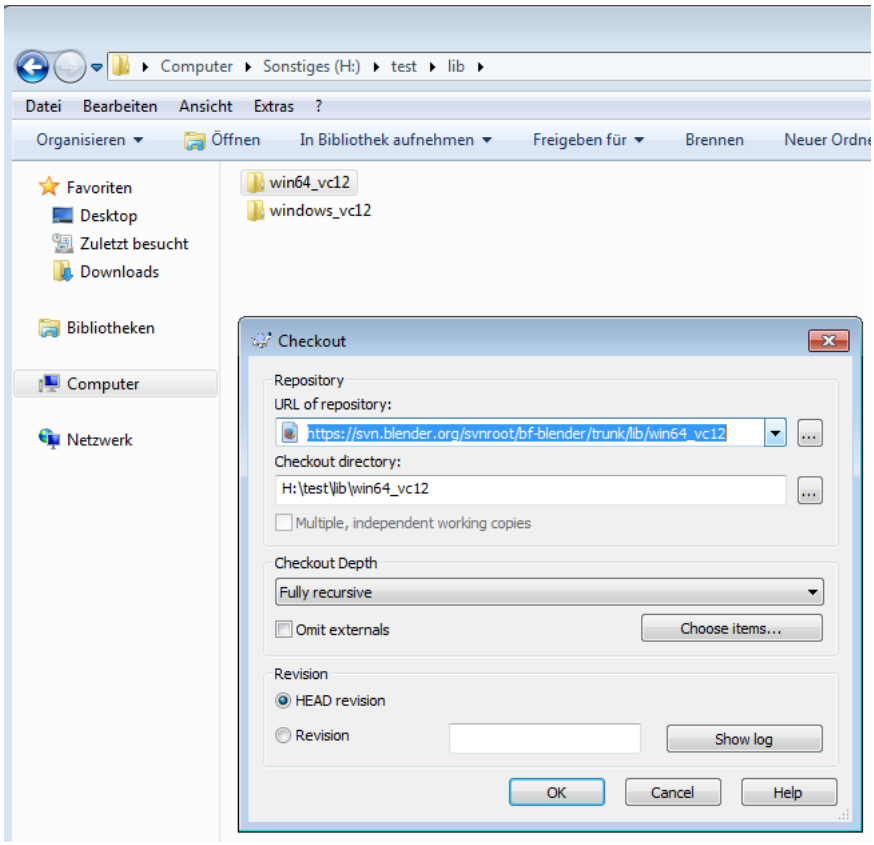


4. Once the check out is open, add the following link to get the correct lib

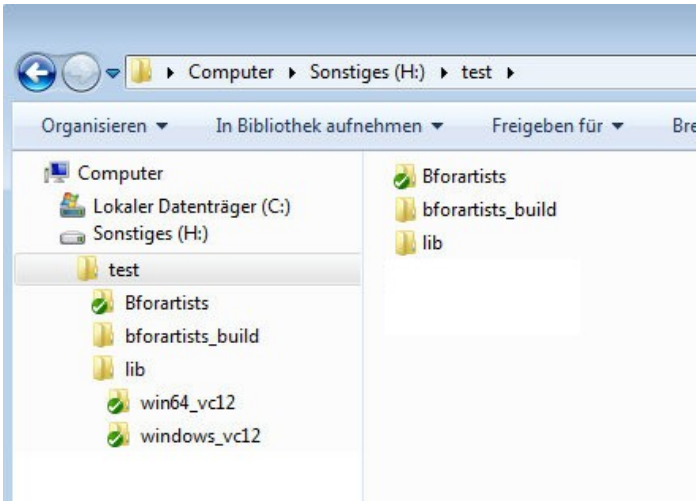
For the 32 Bit version we need the url:
https://svn.blender.org/svnroot/bf-blender/trunk/lib/windows_vc12

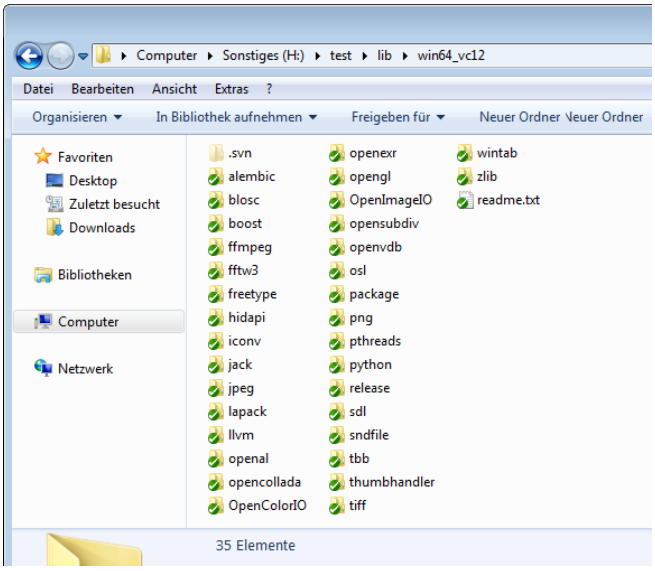
For the 64 Bit version we need the url:
https://svn.blender.org/svnroot/bf-blender/trunk/lib/win64_vc12

5. Click okay, and the SVN repository with the dependencies should download.



6. After these steps, the downloaded results in the target folder should look like this now:





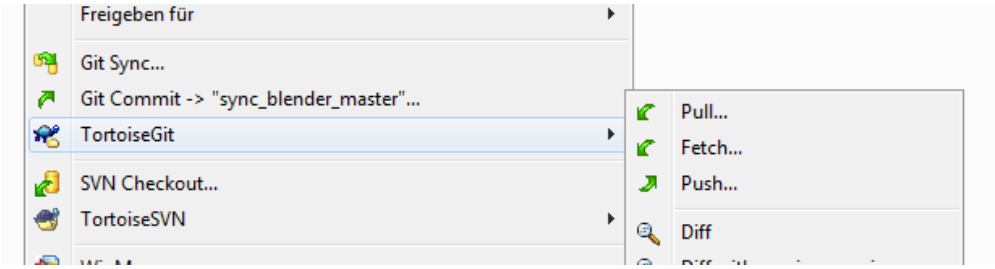
I have installed everything in a folder called **test** here. Note that the creation of the folder structure for the libs is required so that Cmake finds the libraries automatically. The folder called **windows_vc12** contains the 32 bit libraries. The folder called **win64_vc12** contains the 64 bit libraries.

Updating the Repository

Before you compile you might want to update the repository (the blueprints) to download the latest changes.

1. First “fetch”. This updates the repository index with branches, changes etc. But does not update the repository. To update the repository you do a “Pull”.

2. So first **fetch**, then **pull**. And then your repository should be up to date again.



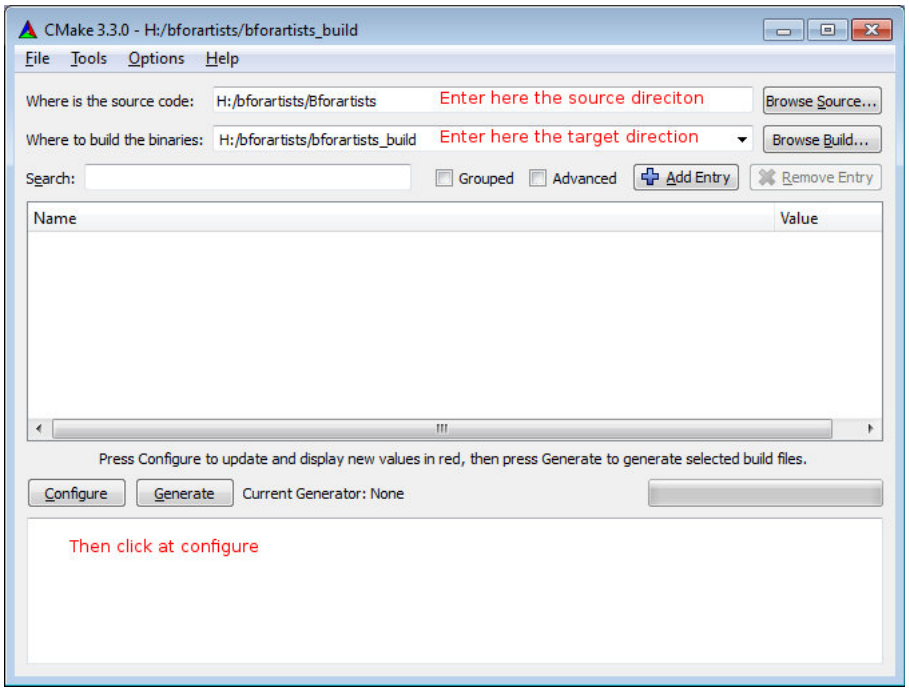
Running CMake

The Materials (Repository and Libraries) and Tools (Compilers) preparation is done using the Sourcing tools (Tools). Next chapter is the compilation itself. Get ready!

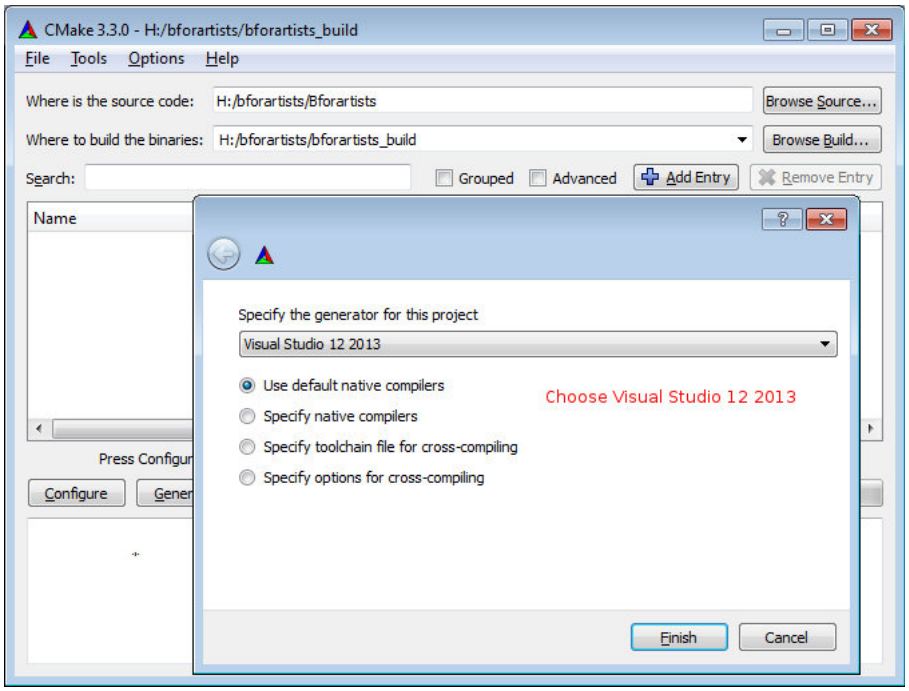
We have downloaded everything, we have installed everything. The repository is up to date. Time to fire up **Cmake** for the first time and create the solution files.

Cmake Preparation

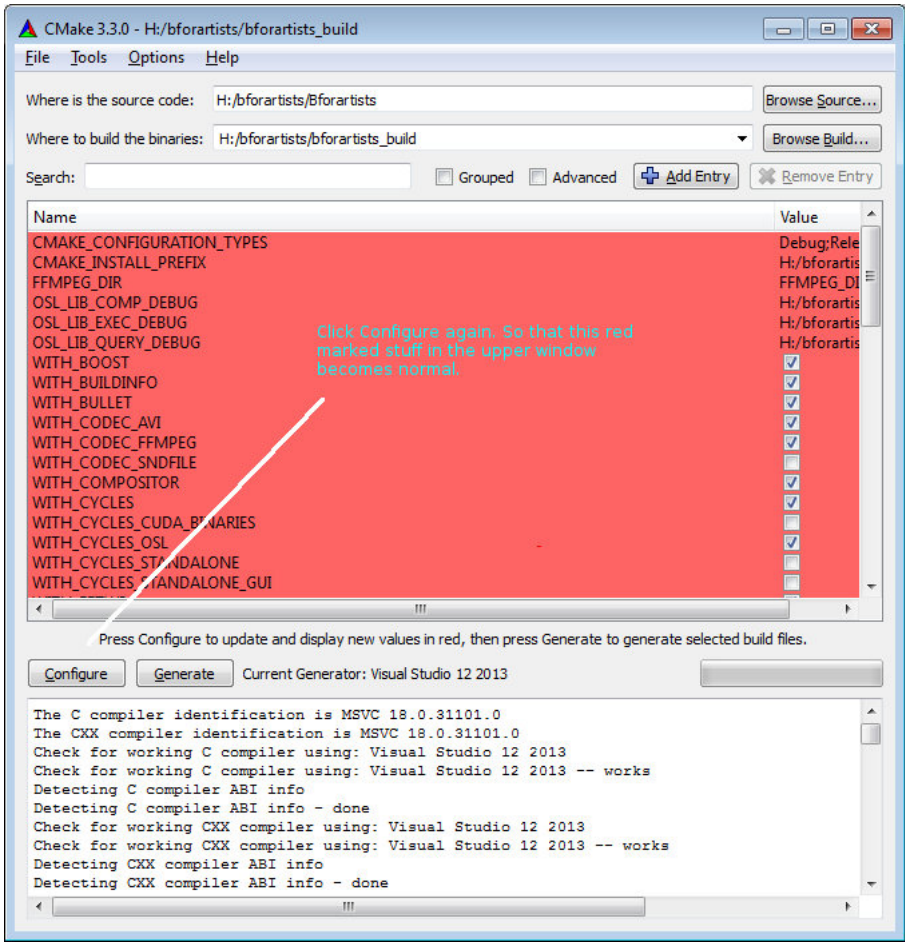
1. Enter the paths to the repository source code, enter the path to the build folder then click on **Configure**.



2. A window will pop up asking you what Visual Studio version to use. Choose **Visual Studio 12 2013** in the “Specify the Generator for this project” dropdown box to compile the 64 bit version.



3. Let Cmake run through once. Then click **Configure** again. The second run will turn the upper window back to normal – where the option toggles are displayed.



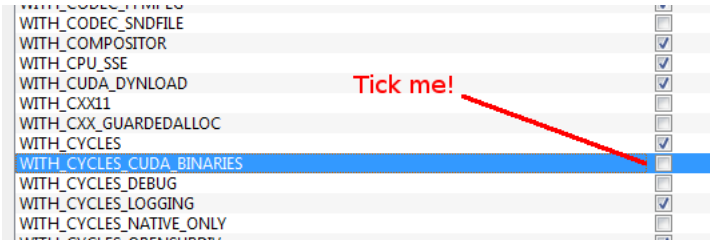
4. Sometimes, if there are incomplete libraries, missing or corrupt dependencies, and/or previous damaged builds, you might get errors in the bottom window. They need to be resolved, let’s move onto the next chapter.

5. If you got no errors, only warnings, then you can proceed to the next steps:

Activate all needed features.

Some features are off by default. **Cuda Binaries**, **Alembic, Open VDB** and **Open Subdiv** needs to be manually activated. We activate Cuda to have GPU acceleration in the software.

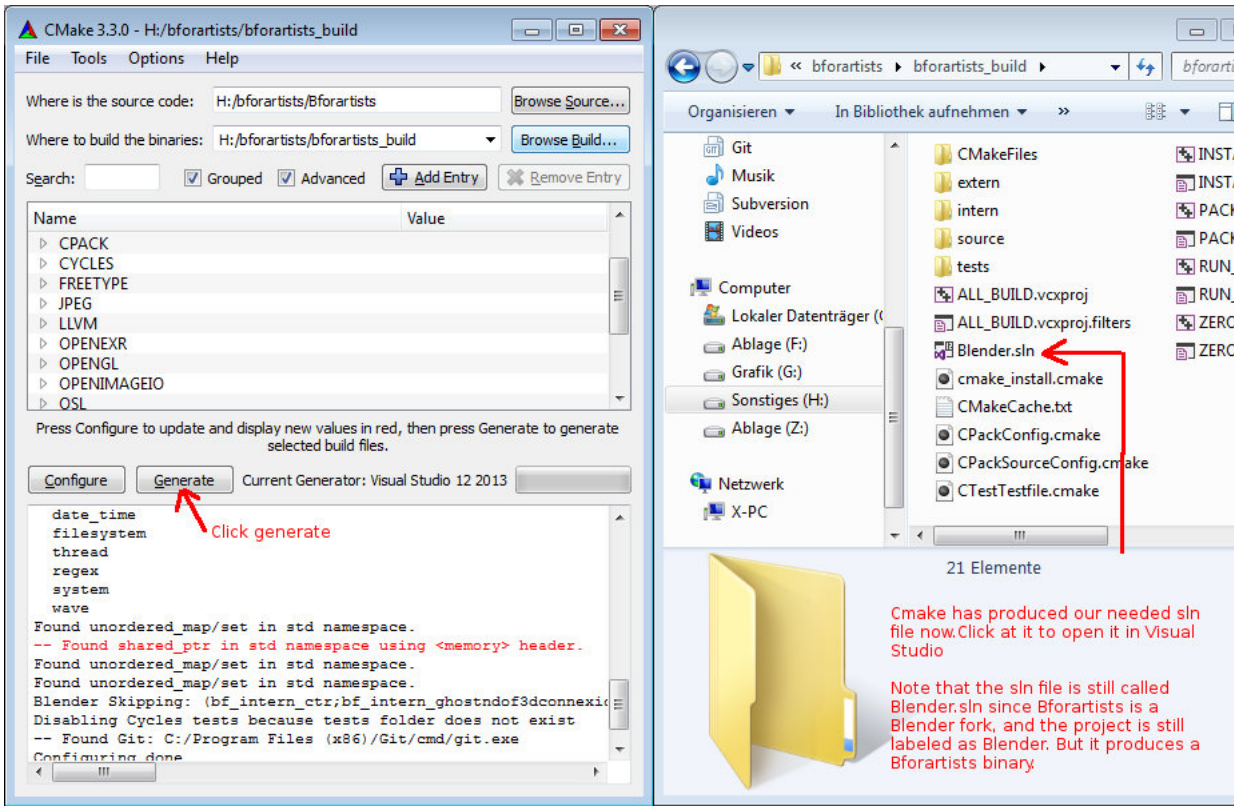
Quick tip: *the defaults are generally the fastest compiling times, but you can optionally disactivate Cuda Binaries and the former to speed things up – but beware the default is pretty good.*



Generate

When all errors are fixed Click on Generate. This will now create the solution files in the bforartists_build folder. And you can open the **Blender.sln** file in Visual studio.

Quick Tip: *Note that the sln file may be called called Blender.sln. Bforartists is a Blender fork. And so big parts of the file structure and the project name is still Blender.*

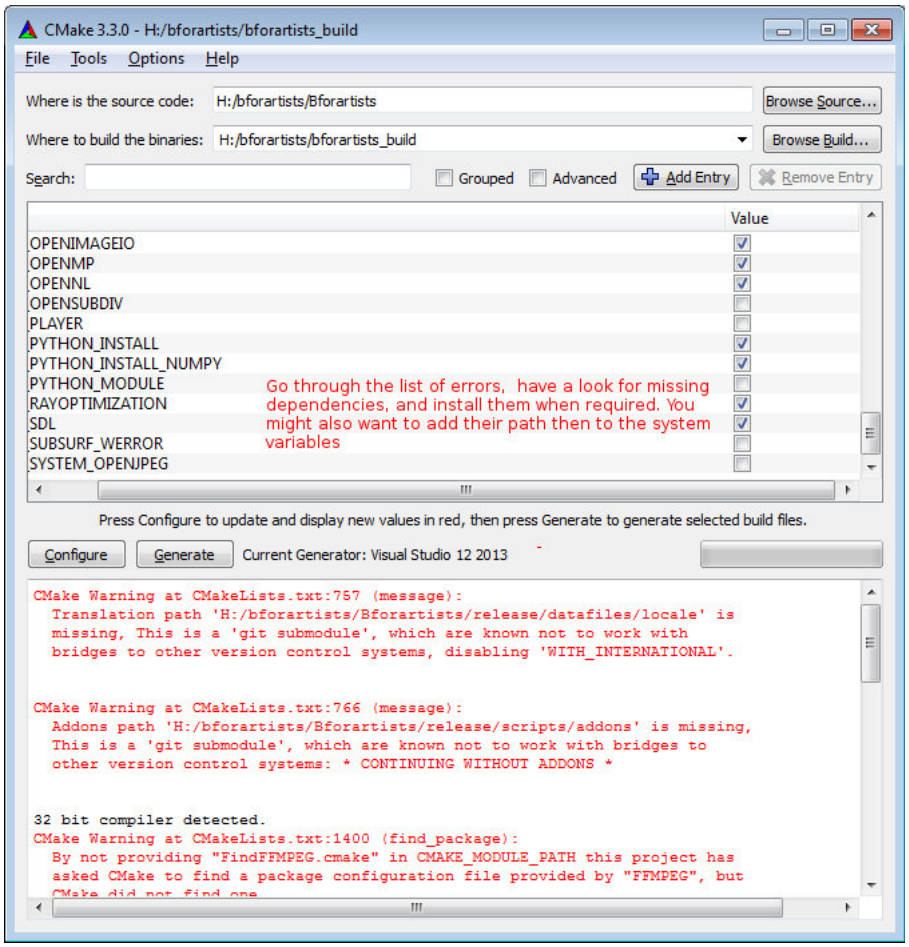


Troubleshooting Cmake Errors

You might get errors when you run Cmake for the first time: missing or corrupt dependencies for example. Make sure you have downloaded and installed all of them. You might also have missing path variables by which the installed software gets recognized by Cmake. These are just some of the errors. Feel free to drop a task in the Github tracker or Forum space for Bforartists asking for help if this step is over your head. We will do our best to help you out.

Cuda

You might get a **CUDA_SDK_ROOT_DIR-NOTFOUND** warning in the upper section. But Bforartists compiles nevertheless fine. In case you want to fix this, search for nvcc in the C:\Program Files\Nvidia Corporation folder.



Cmake Warning at intern/cycles/kernel/CmakeLists

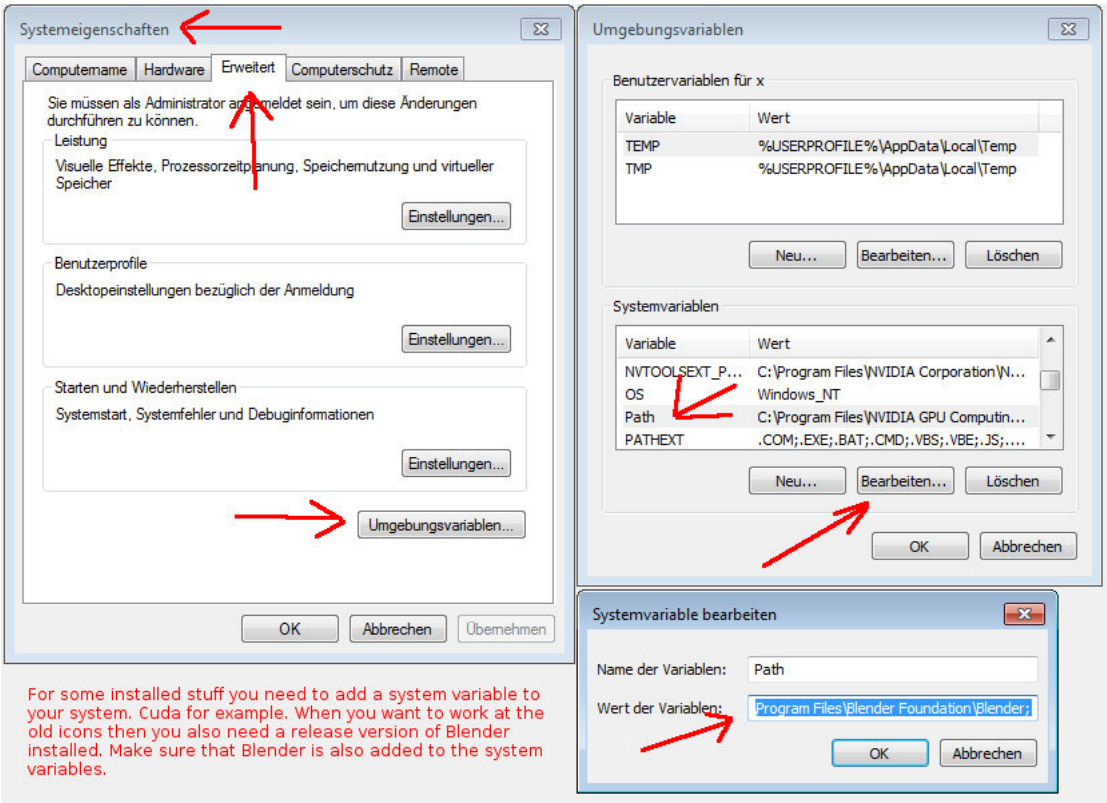
Sometimes you will get this warning (and thus cause compiling errors) if you have the incorrect CUDA Developer Tools installed. Uninstall and reinstall the correct version. 9.1.85 is the last tested version for Bforartists 1.0.0

```
64 bit compiler detected.
Visual Studio 2013 detected.
CMake Warning at build_files/cmake/platform/platform_win32.cmake:436 (message):
  LLVM debug libs not present on this system.  Using release libs for debug
  builds.
Call Stack (most recent call first):
  CMakeLists.txt:899 (include)


Found OpenMP_C: -openmp
Found OpenMP_CXX: -openmp
Found OpenMP: TRUE
Found CUDA: C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v9.2 (found version "9.2")
CUDA nvcc = C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v9.2/bin/nvcc.exe
nvcc not supported for this compiler version, using cycles_cubin_cc instead.
CMake Warning at intern/cycles/kernel/CMakeLists.txt:332 (message):
  CUDA version 9.2 detected, build may succeed but only CUDA 8.0 is
  officially supported


Blender Skipping: (bf_intern_ctr;bf_intern_opencl;extern_sdlew)
Disabling Cycles tests because tests folder does not exist
Configuring done
Generating done
```

Adding System variables

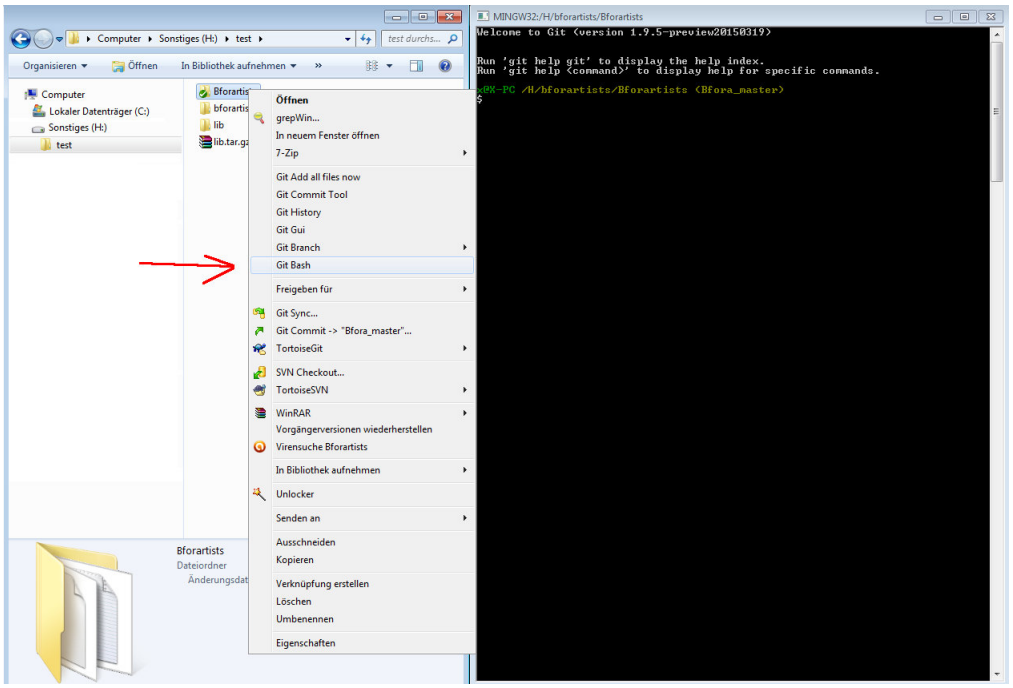
Some things requires to be added to the system variables.

Here's for example where you add the path to the Blender exe in case it is missing. Windows System control, Advanced system settings -> system properties.



Missing Language Paths in Blender

When you compile Blender, and not Bforartists, then you might have the above errors with the missing language paths. Bforartists does not have this problem anymore, we have unioned everything in one repository. But Blender has. And just for the sake, here's how to fix:



Type in the following commands into the bash, one after another. Wait until the single tasks are finished:

```
git submodule update --init --recursive
git submodule foreach --recursive git checkout master
git submodule foreach --recursive git pull --rebase origin master
```

This should download you the needed submodules. And Cmake shouldn't moan anymore

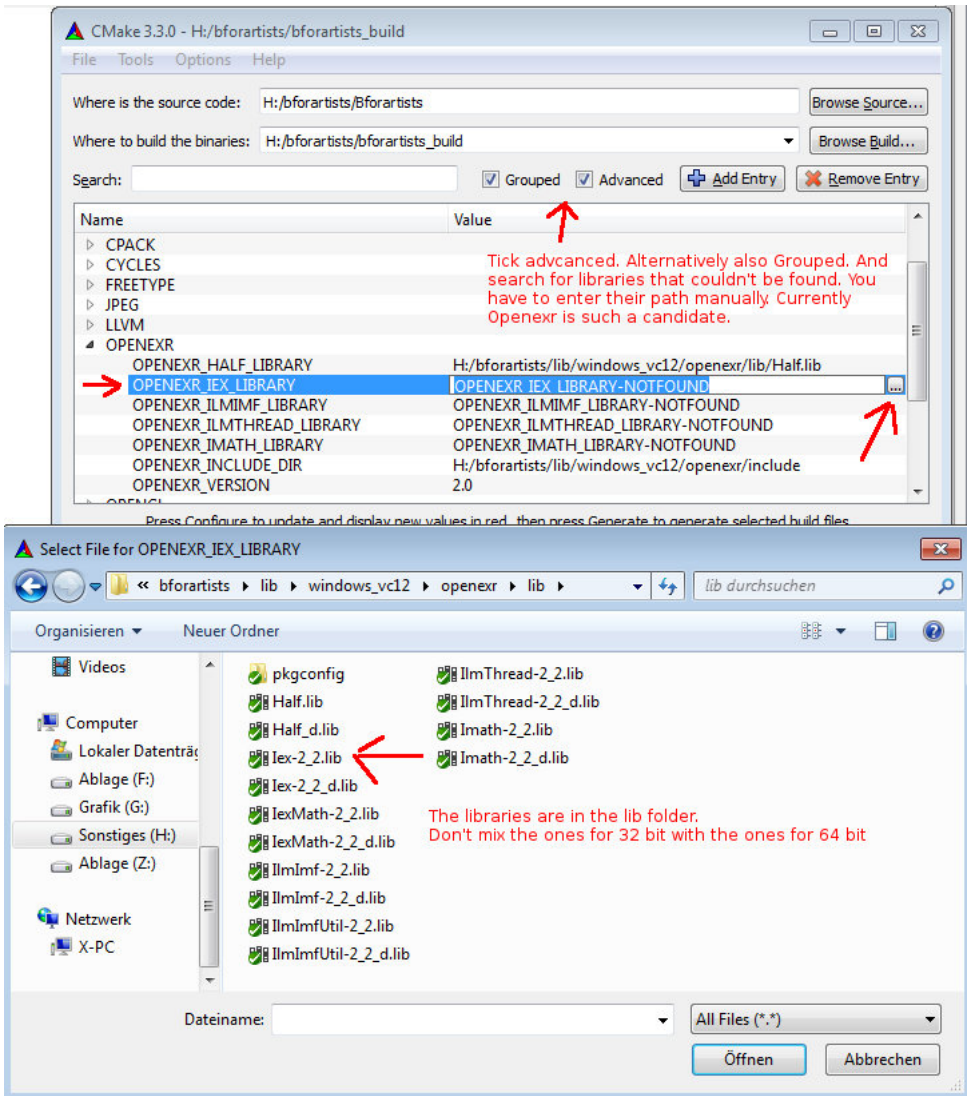
For later, when you want to update the submodules, type in

```
git pull --rebase
git submodule foreach --recursive git pull --rebase origin master
```

But this just as a sidenote. As told, Bforartists does not need this step.

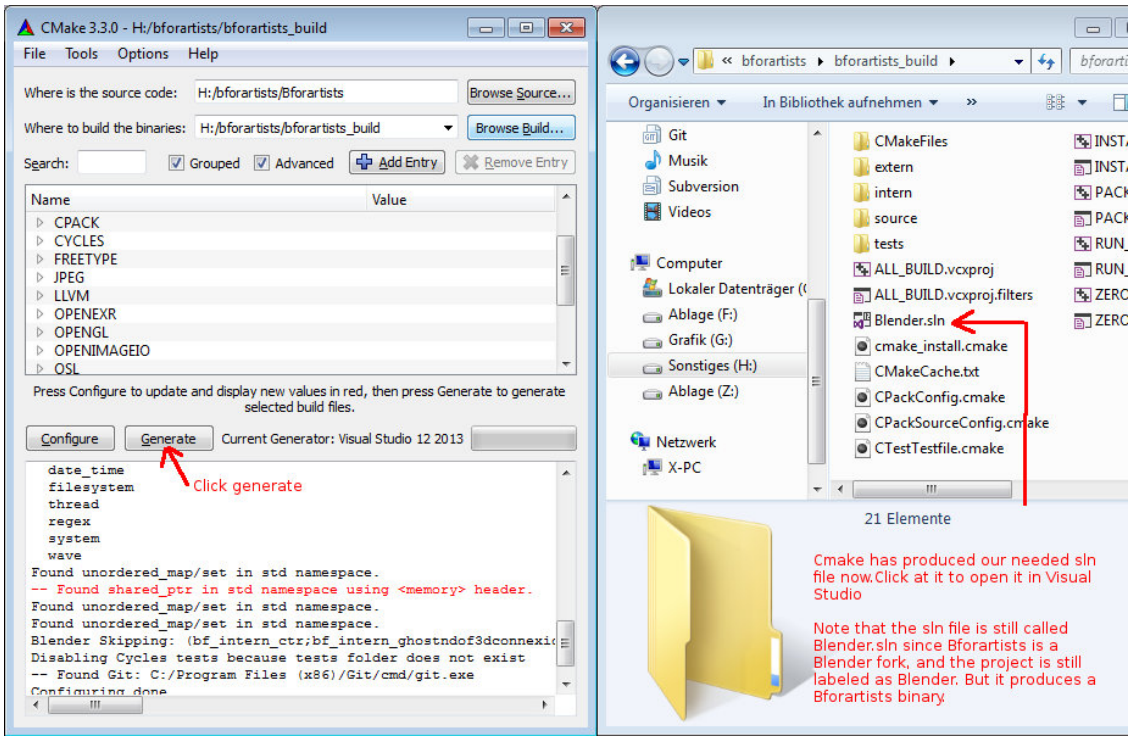
Open EXR

Most probably the Openexr libraries are not found on Windows, you can fix them by hand when you want. But it is not necessary. The libraries should nevertheless be found. You can happily ignore the warning. It's just a cosmetic problem in the Cmake GUI. In case you fix them by hand, then don't mix the libs for the 32 bit version with the libs for the 64 bit version.



Compiling

Finally we arrive to the compiling part, the best part. **Cmake** should have produced everything needed in the build folder.

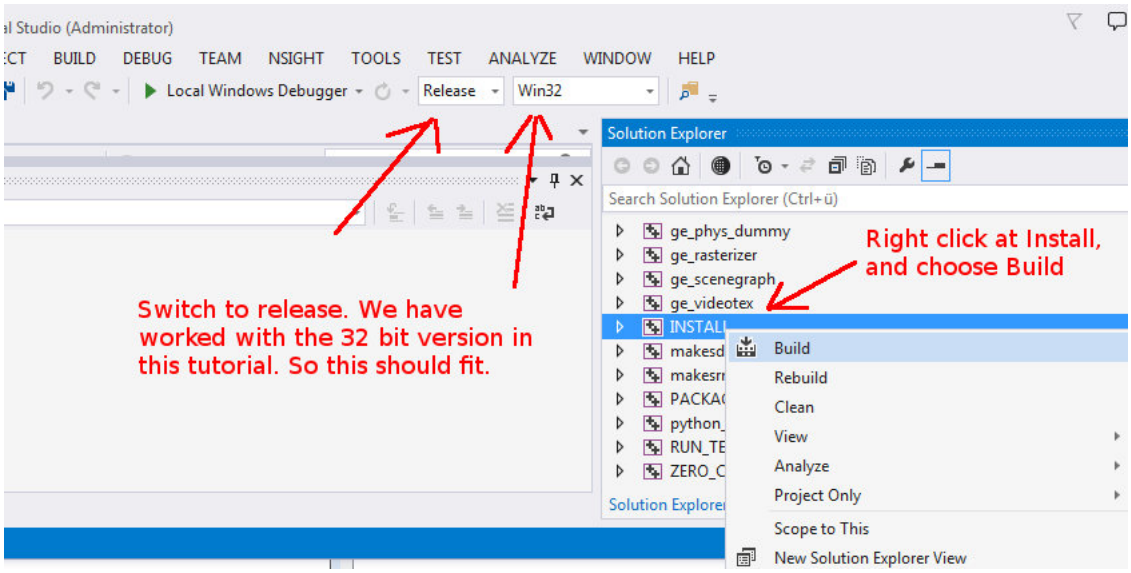


1. Click at the **Blender.sln** File to open Visual Studio.

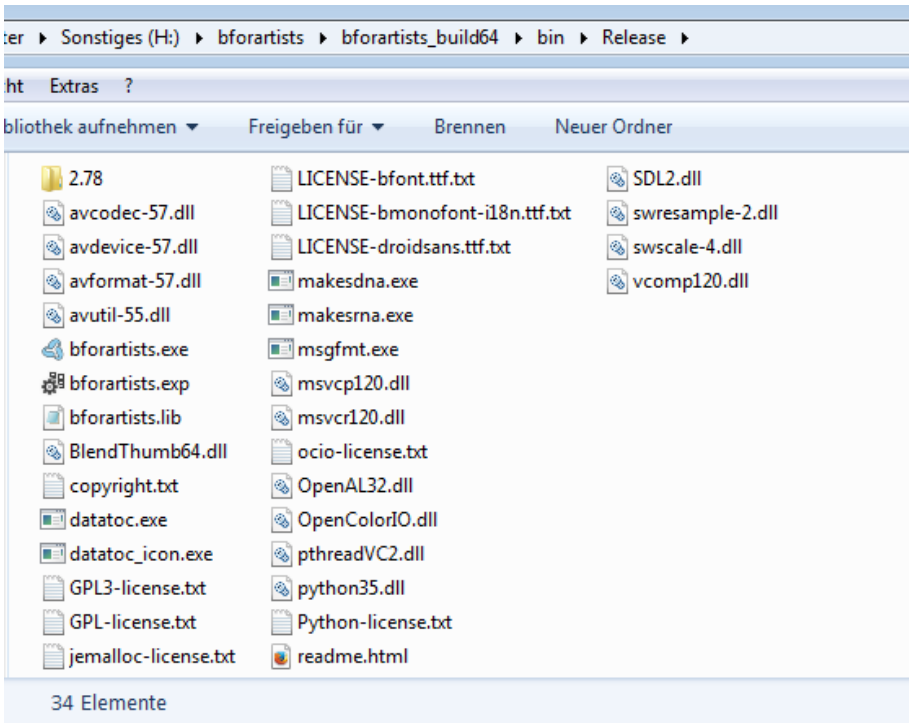
2. In VS, change the build type to **Release** (VS starts usually with **Debug**, that's of interest for development).

3. Select the **Install** item in the Solution Explorer, right click on it, and choose **Build**.

4. Visual Studio should now start the Build process.



5. When everything works as thought then you should have a result in the **bin/Release** folder.



Congratulations!

You have compiled Bforartists for the first time. You should be able to work with the source code now. Double click on the *.exe to run and see that everything is working fine. When it does, you are a winner.

Quick Tip: *Now that it's compiled, you can do Python code changes to the built data, and test it almost live. Don't forget to copy your python changes in the* ***bin/Release*** *folder back to the Repository or else when you rebuild, you will loose all changes!*

For C code changes, you will need to do the changes in the repository before building, then re-build to test. If you get errors on re-build, just delete the old build in the ***bin/Release*** *folder.*