

AmiDevCpp Manual

Heinz-Raphael Reinke

March 6, 2009



Contents

1	What is AmiDevCpp ?	3
2	Installation	3
2.1	Warning !	3
2.2	The Setup Program	3
3	getting started	6

1 What is AmiDevCpp ?

AmiDevCpp is a Cross Development Environment. You can use it on your Windows PC to develop Applications for all Amiga like Operating Systems.

It consists of three Parts:

- The wx-DevC++ IDE
- The Cygwin Enviroment for Windows
- The Cygwin based Cross Compilers

The Cross Compilers can produce executables for various platforms:

- Windows *x86*
- AmigaOS3 *68k*
- AmigaOS4 *ppc*
- MorphOS *ppc*
- AROS *x86*
- AROS *x86_64*
- AROS *ppc*

2 Installation

2.1 Warning !

If you already have a Cygwin Environment, and need it to work, dont install AmiDevCpp ! AmiDevCpp comes with a complete cygwin Environment. It's Setup Program will overwrite your settings.

2.2 The Setup Program

AmiDevCpp comes with a typical Windows install program, that will put all files in place and make all settings for you. When you start the setup, you are first asked for the language of your choice.



Figure 1: Choose the language for the Installation Dialog

Click OK to proceed with the Installation. In the next Dialog, you get a warning about the cygwin problem.

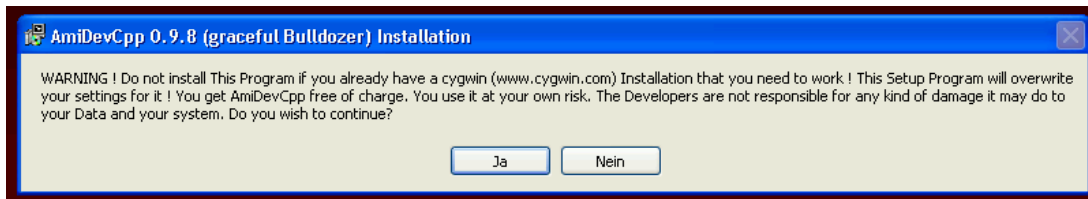


Figure 2: Read this carefully !

You now need to accept the License to proceed.



Figure 3: accept the license

In the next Dialog, you have to choose in which language AmiDevCpp should be installed.



Figure 4: choose you language

There are various languages available.



Figure 5: choose you language

Now you can choose, where AmiDevCpp shall be installed

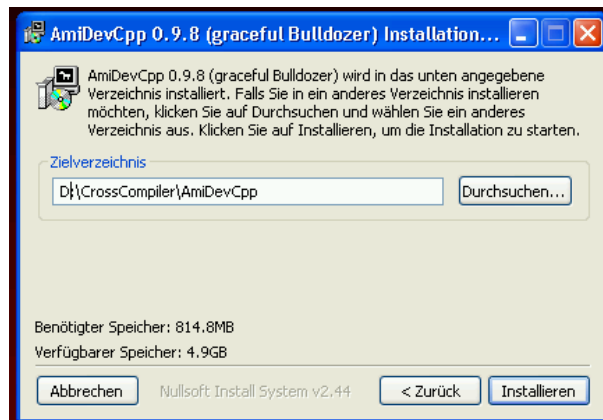


Figure 6: select location

You can name the AmiDevCpp directory as you prefer it, but the path **MUST NOT** contain empty spaces ! Something like "C:\Program Files\AmiDevCpp" will install, but you will get lots of errors when compiling something. Click Install.

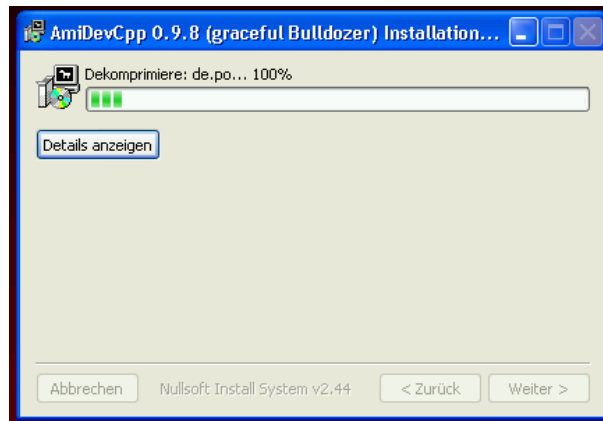


Figure 7: Please wait ...

Now the files are being copied. This will take some time and in the end, AmiDevCpp will consume more than 800MB of your precious Disk Space.

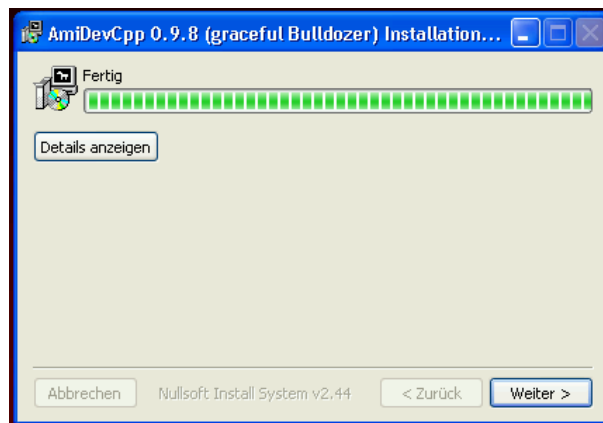


Figure 8: It's done !

Now everything is in place and you can start developing Amiga Apps.

3 getting started

When you have started AmiDevCpp, you will see this interface.

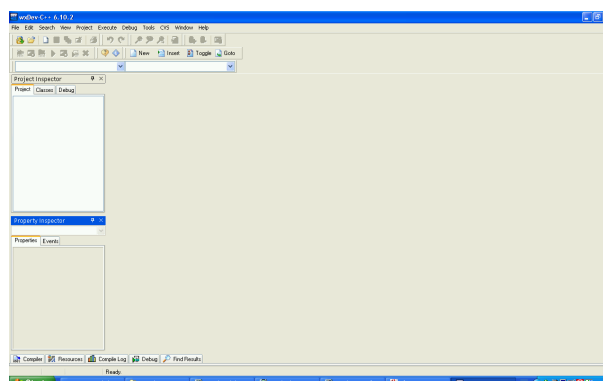


Figure 9: The wxDevC++ IDE

The easiest way to start a new project, is using one of the many templates, that ship with AmiDevCpp.

To do that, just open the menu "File new Project"

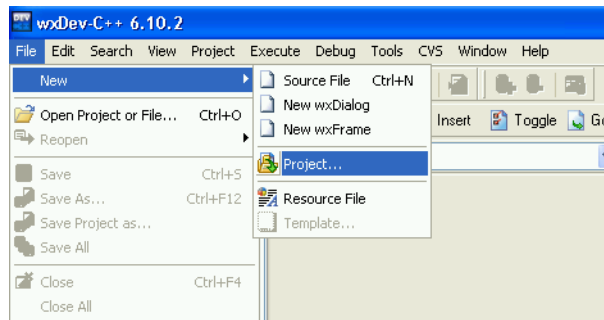


Figure 10: Start a new project

You may now think that AmiDevCpp is crashed because nothing happens and the GUI does not respond anymore, but it is not. I can take more than a minute before the new projects dialog appears. Fortunately you only have to do this once when you start a new project.

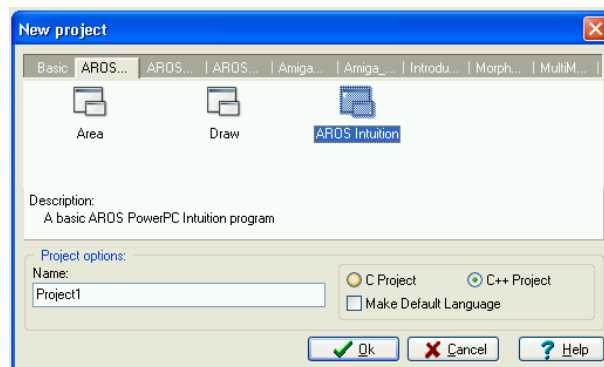


Figure 11: Choose a Template

You can choose between lots of templates for various platforms. We are going to choose the "AROS Intuition" example. You can also select if the programming language for the new project shall be C or C++.

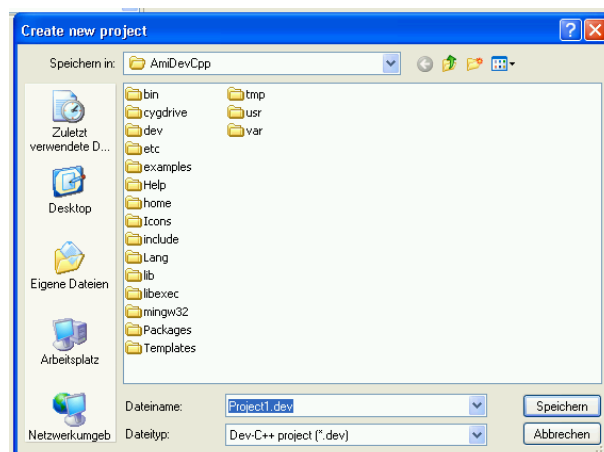


Figure 12: Choose directory

When you clicked OK, you are asked where to save the project file. The best is to create a directory inside the AmiDevCpp directory.

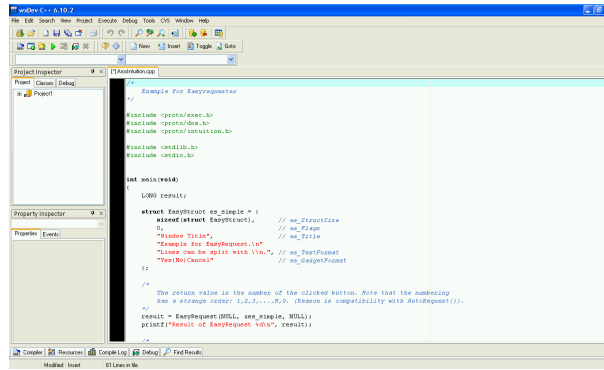


Figure 13: The source code

After saving the project file, the source code of the program is loaded into the editor.

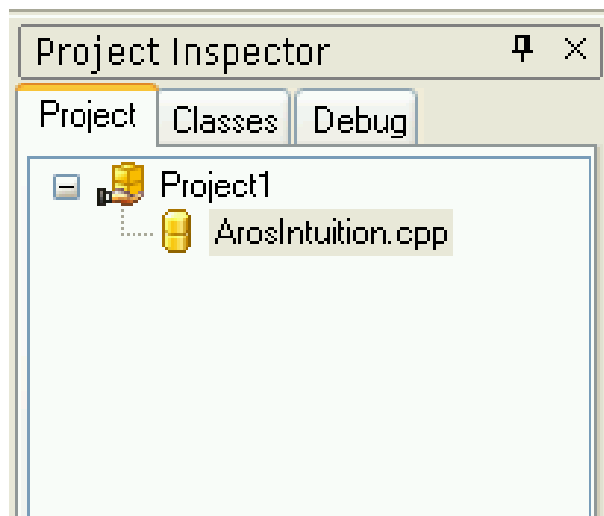


Figure 14: The Project inspector

The project inspector on the left, display all source files of the project. It can also display classes member functions etc.

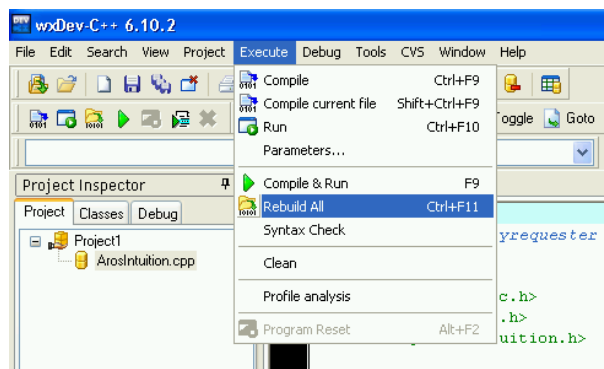


Figure 15: compile

To create a executable program, just open the menu "Execute" and click on "rebuild all".

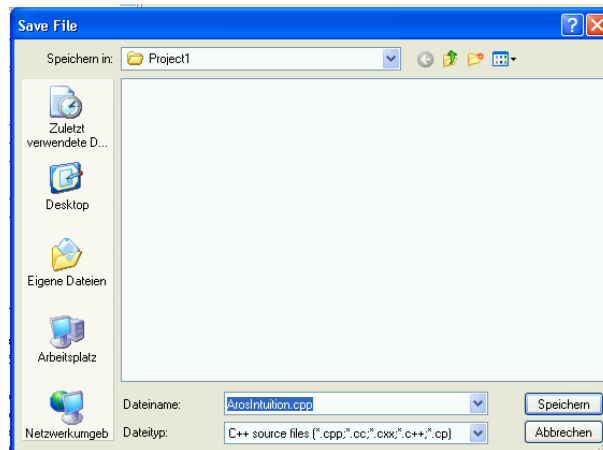


Figure 16: compiles...

The compilation starts immediately using the AROS i686 compiler.

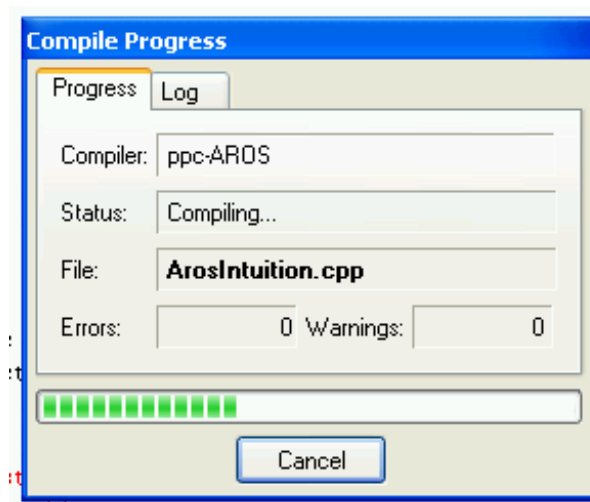


Figure 17: done

When the compilation process ended without errors, the compiler windows disappears.

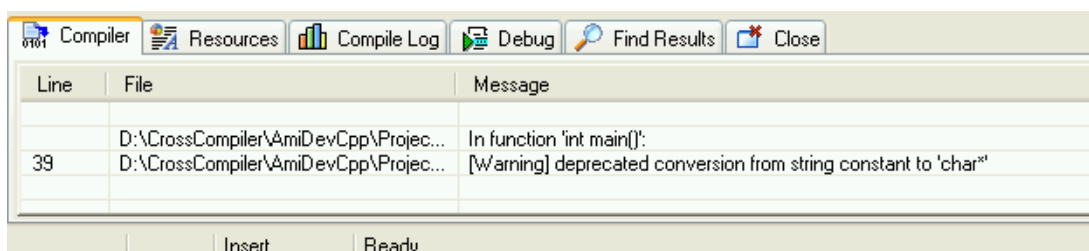


Figure 18: Warnings

If the compiler returned any errors or warnings, they will be display at the bottom of the IDE.

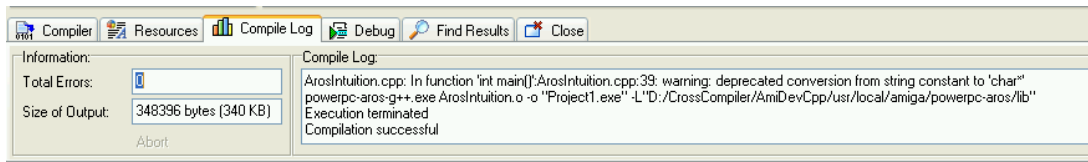


Figure 19: Compile log

Click on "compile log" to see all the commands and errors just like in a terminal.

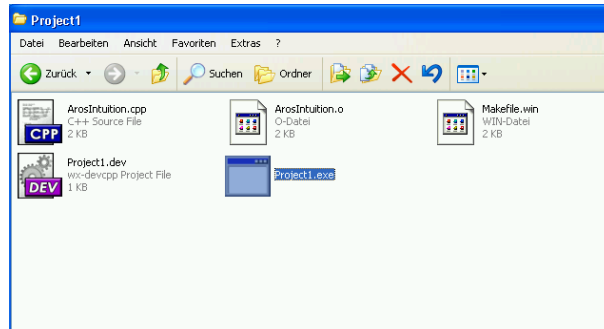


Figure 20: The executable

If you open the directory, where you saved the project file, you will find the just created executable. You can test it easily with WinAros (qemu) or the new mingw32-hosted AROS.

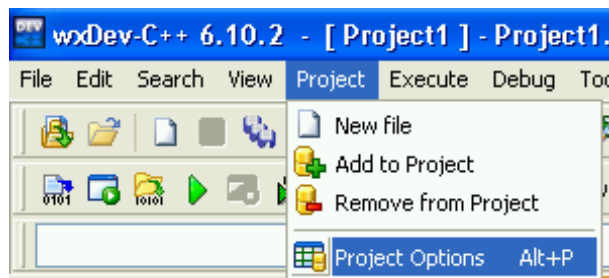


Figure 21: Project Menu

If you decide to compile the program for another platform, then open the Project Menu and click on project options.

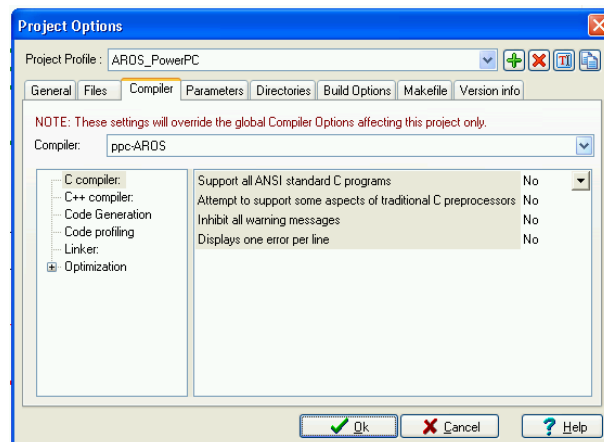


Figure 22: Project Options

Click on the Compiler Tab.

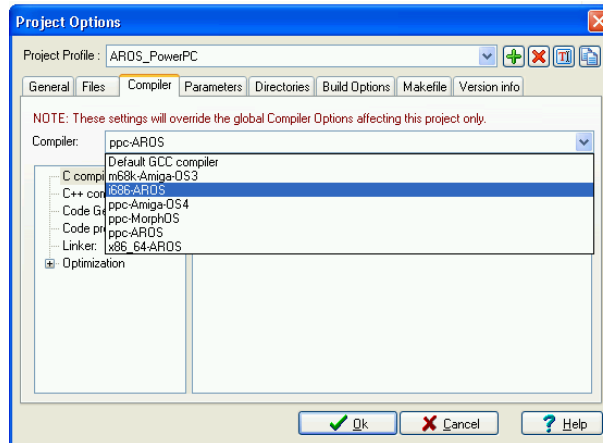


Figure 23: Project Options

Choose the Compiler that shall be used for the Project.

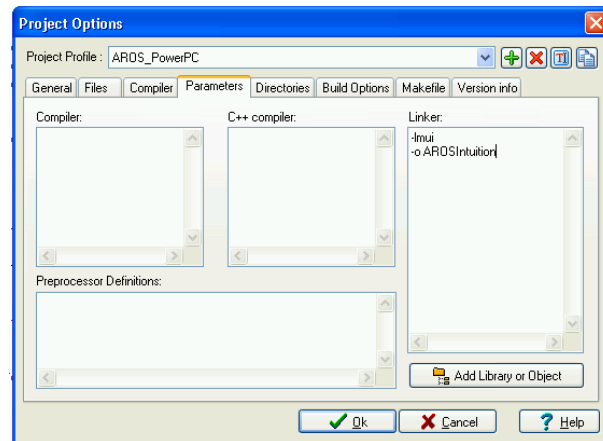


Figure 24: Project Options

In the Parameter Tab, you can add parameter that are passed to the compiler and linker. If you make a MUI Program you would add -lmui to the Linker field.

Another "rebuild all" will compile your project for the selected platform.