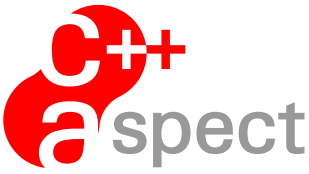

Documentation:

Ag++ Manual:

Georg Blaschke

Version 0.7

Contents

1 Introduction

The ag++ program provides a more intuitive frontend to the AspectC++ weaver (ac++) in a GNU environment. The only preliminaries are a working installation of GNU C++ compiler, which also can run within a cygwin environment. It basically wraps the functionality of the aspect weaver and the c++ compiler into one single program.

2 Invocation

The usage of ag++ is mainly influenced by the usage of the GNU g++ compiler and the synopsis is like:

```
ag++ [options] [files...].
```

Let's say, you want to **compile** a single file (here: main.cc) with g++, you have to run

```
g++ -c main.cc
```

in order to generate an object file.

To **weave** and **compile** a single file you simply invoke

```
ag++ -c main.cc
```

the same way like you did before with g++.

2.1 Concept

As ag++ is just a wrapper, it first generates the puma configuration file, then calls ac++ and afterwards g++. The intermediate files generated by ac++ are stored in the directory which is extracted from the -o option or in current directory. In some cases this may lead to a situation where the names of intermediate files interfere with each other.

2.2 Options

All available options are summed up in the options table (see table 1). The column labeled with *AC++* shows if the option is taken over from ac++ by ag++ ('X'), not supported by ac++ ('–') or modified by ag++ ('!'). All options which are taken over, are not described in this document. Consult the AC++ Compiler Manual instead. Options which are not listed in the option table are accounted as g++ options. Some g++ options can not be automatically handled correctly by the options parser of ag++. So all g++ options starting with -p, -a, -d and -r (e.g. -pipe, -ansi, -dletters ,-remap) have to be written between --Xcompiler (see 2.2.12) and --Xweaver (see 2.2.13). If such options passed to ag++ without using --Xcompiler they will be interpreted a ag++/ac++ options ; e.g. -pipe will be interpreted as -p "ipe".

Option	AC++	Description
--gen_config	–	Only generate Puma configuration file
--weave_only	–	Weave only
-c	!	Compile only
--keep_woven	–	Keep woven source code files
--c_compiler <arg>	–	Path to C++ compiler
--ac_compiler <arg>	–	Path to AspectC++ compiler
--config_command <arg>	–	Specify command which prints information about compiler
--Xcompiler	–	In case of doupt account following options as g++ options.
--Xweaver	–	In case of doupt account following options as ac++ options.
-p|--path <arg>	!	Defines a project directory
-d|--dest <arg>	X	Specifies a target directory for saving
-v|--verbose <arg>	!	Level of verbosity (0-9)
-o|--output <arg>	X	Name of the output file
--include_files	X	Generate manipulated header files (short version -i is not supported)
-a|--aspect_header <arg>	!	Name of aspect header file or 0
-r|--repository <arg>	X	Name of the project repository
--config <arg>	!	Parser configuration file
--no_line	X	Disable generation of #line directives
-k|--keywords	X	Allow AspectC++ keywords in normal project files
--real-instances	X	Let ac++ perform a full template analysis
--problem...	X	Enable back-end compiler problem workaround
--no_problem...	X	Disable back-end compiler problem workaround
--warn...	X	Show a specific ac++ warning that is suppressed by default
--no_warn...	X	Suppress a specific ac++ warning
-I <arg>	X	Include file search path
-D <name>[=<value>]	X	Macro definitions
-U <name>	X	Undefine a macro
--include <arg>	X	Forced include

Table 1: ag++ Compiler Option Summary

2.2.1 --gen_config

Just create a parser configuration and quit afterwards. The argument of the -o option specifies the name of the file. In any other case (no --gen_config and/or no -o option) a configuration file with the name 'puma.config' will be generated in the directory where ag++ was invoked.

2.2.2 --weave_only

Generate only woven source code files. With -o option and one file the generated output is named after the argument of the -o option.

2.2.3 --path

This options differs only slightly from the --path option of ac++. In ac++ it is mandatory to specify a project path, whereby ag++ the current working directory is used as project path by default. Especially for larger projects it is NOT wise to rely on the default project path, as weaving take a lot of time. See the AspectC++ Compiler Manual for a more detailed description of this option.

2.2.4 -c

Like the -c option of g++, this options effects the creation of object files of one or more source files.

2.2.5 -v|--verbose [<arg>]

Set the level of verbosity.

2.2.6 --aspect-header <arg>

This option differs from meaning in ac++ only if dependencies are generated (see 2.3).

2.2.7 --keep_woven

Don't remove intermediate woven files.

2.2.8 --c_compiler

Specify path to GNU C++ compiler˙ The default is g++.

2.2.9 --ac_compiler

Specify path to AspectC++ compiler˙ By default ag++ assumes, that the ac++ executable is located in the same directory like itself.

2.2.10 --config_command

Specify the command which prints information about the compiler. This information is necessary for generating the parser (puma) configuration file. The default value is "<compiler> <compiler options> -E -dM -v -x c++ <an empty file>".

2.2.11 --config <arg>

Path to a puma configuration file. If this option is available the configuration file will not be generated automatically.

2.2.12 --Xcompiler

ac++ and ag++ options that might interfere with g++ options are not recognized after using --Xcompiler in the argument list of an ag++ invocation.

2.2.13 --Xweaver

Enable the recognition of those ac++ and ag++ options which previously have been disabled by the usage of –Xcompiler.

2.3 Generating dependency information

To produce dependency files just pass the -M or -MM (consult the GNU C++ Compiler Manual) to ag++ . Dependency files generated by ag++ are slighty different from dependency files created by g++, as they contain dependencies to aspect header files. If the --aspect-header option is provided, only the header file specified as option argument is considered when building the dependency file; otherwise the dependency file will contain all aspect header files within the whole project path .

2.4 Examples

- ag++ --help
  Displays all options with a short description.

- `ag++ -o test Test.cc main.cc`
  Weave, compile and link the source files `Test.cc` and `main.cc`. The created executable will be named '`test`'.

- `ag++ --gen_config`
  Create a puma configuration file named `puma.config` within the current working directory.

- `ag++ --gen_config -o my.config`
  Create a puma configuration file named `my.config`.

- `ag++ --path src --include_files --dest gen/includes`
  Generate modified include files out of all include files found below `src` directory and store them under '`gen/includes`'.

- `ag++ -M -MFmain.dep main.cc`
  Generate dependency file `main.dep` from source file `main.cc` .

- `ag++ -p ../aspects -p .  --Xcompiler`
  This string could be used to substitute `g++` in a simple `make` environment.