

NAME

cmake-generators - CMake Generators Reference

INTRODUCTION

A *CMake Generator* is responsible for writing the input files for a native build system. Exactly one of the *CMake Generators* must be selected for a build tree to determine what native build system is to be used. Optionally one of the *Extra Generators* may be selected as a variant of some of the *Command-Line Build Tool Generators* to produce project files for an auxiliary IDE.

CMake Generators are platform-specific so each may be available only on certain platforms. The **cmake(1)** command-line tool **--help** output lists available generators on the current platform. Use its **-G** option to specify the generator for a new build tree. The **cmake-gui(1)** offers interactive selection of a generator when creating a new build tree.

CMAKE GENERATORS

Command-Line Build Tool Generators

These generators support command-line build tools. In order to use them, one must launch CMake from a command-line prompt whose environment is already configured for the chosen compiler and build tool.

Borland Makefiles

Generates Borland makefiles.

MSYS Makefiles

Generates MSYS makefiles.

The makefiles use `/bin/sh` as the shell. They require `msys` to be installed on the machine.

MinGW Makefiles

Generates a make file for use with `mingw32-make`.

The makefiles generated use `cmd.exe` as the shell. They do not require `msys` or a unix shell.

NMake Makefiles

Generates NMake makefiles.

NMake Makefiles JOM

Generates JOM makefiles.

Ninja

Generates `build.ninja` files (experimental).

A `build.ninja` file is generated into the build tree. Recent versions of the `ninja` program can build the project through the `all` target. An `install` target is also provided.

Unix Makefiles

Generates standard UNIX makefiles.

A hierarchy of UNIX makefiles is generated into the build tree. Any standard UNIX-style make program can build the project through the default make target. A `make install` target is also provided.

Watcom WMake

Generates Watcom WMake makefiles.

IDE Build Tool Generators

These generators support Integrated Development Environment (IDE) project files. Since the IDEs configure their own environment one may launch CMake from any environment.

Visual Studio 6

Generates Visual Studio 6 project files.

Visual Studio 7

Generates Visual Studio .NET 2002 project files.

Visual Studio 7 .NET 2003

Generates Visual Studio .NET 2003 project files.

Visual Studio 8 2005

Generates Visual Studio 8 2005 project files.

It is possible to append a space followed by the platform name to create project files for a specific target platform. E.g. Visual Studio 8 2005 Win64 will create project files for the x64 processor.

Visual Studio 9 2008

Generates Visual Studio 9 2008 project files.

It is possible to append a space followed by the platform name to create project files for a specific target platform. E.g. Visual Studio 9 2008 Win64 will create project files for the x64 processor; Visual Studio 9 2008 IA64 for Itanium.

Visual Studio 10 2010

Generates Visual Studio 10 (VS 2010) project files.

It is possible to append a space followed by the platform name to create project files for a specific target platform. E.g. Visual Studio 10 2010 Win64 will create project files for the x64 processor; Visual Studio 10 2010 IA64 for Itanium.

For compatibility with CMake versions prior to 3.0, one may specify this generator using the name Visual Studio 10 without the year component.

Visual Studio 11 2012

Generates Visual Studio 11 (VS 2012) project files.

It is possible to append a space followed by the platform name to create project files for a specific target platform. E.g. Visual Studio 11 2012 Win64 will create project files for the x64 processor; Visual Studio 11 2012 ARM for ARM.

For compatibility with CMake versions prior to 3.0, one may specify this generator using the name Visual Studio 11 without the year component.

Visual Studio 12 2013

Generates Visual Studio 12 (VS 2013) project files.

It is possible to append a space followed by the platform name to create project files for a specific target platform. E.g. Visual Studio 12 2013 Win64 will create project files for the x64 processor; Visual Studio 12 2013 ARM for ARM.

For compatibility with CMake versions prior to 3.0, one may specify this generator using the name Visual Studio 12 without the year component.

Xcode

Generate Xcode project files.

EXTRA GENERATORS

Some of the *CMake Generators* listed in the [cmake\(1\)](#) command-line tool `--help` output may have variants that specify an extra generator for an auxiliary IDE tool. Such generator names have the form `<extra-generator> - <main-generator>`. The following extra generators are known to CMake.

CodeBlocks

Generates CodeBlocks project files.

Project files for CodeBlocks will be created in the top directory and in every subdirectory which features a CMakeLists.txt file containing a PROJECT() call. Additionally a hierarchy of makefiles is generated into the build tree. The appropriate make program can build the project through the default make target. A make install target is also provided.

This extra generator may be specified as:

CodeBlocks - MinGW Makefiles

Generate with **MinGW Makefiles**.

CodeBlocks - NMake Makefiles

Generate with **NMake Makefiles**.

CodeBlocks - Ninja

Generate with **Ninja**.

CodeBlocks - Unix Makefiles

Generate with **Unix Makefiles**.

CodeLite

Generates CodeLite project files.

Project files for CodeLite will be created in the top directory and in every subdirectory which features a CMakeLists.txt file containing a PROJECT() call. The appropriate make program can build the project through the default make target. A make install target is also provided.

This extra generator may be specified as:

CodeLite - MinGW Makefiles

Generate with **MinGW Makefiles**.

CodeLite - NMake Makefiles

Generate with **NMake Makefiles**.

CodeLite - Ninja

Generate with **Ninja**.

CodeLite - Unix Makefiles

Generate with **Unix Makefiles**.

Eclipse CDT4

Generates Eclipse CDT 4.0 project files.

Project files for Eclipse will be created in the top directory. In out of source builds, a linked resource to the top level source directory will be created. Additionally a hierarchy of makefiles is generated into the build tree. The appropriate make program can build the project through the default make target. A make install target is also provided.

This extra generator may be specified as:

Eclipse CDT4 - MinGW Makefiles

Generate with **MinGW Makefiles**.

Eclipse CDT4 - NMake Makefiles

Generate with **NMake Makefiles**.

Eclipse CDT4 - Ninja

Generate with **Ninja**.

Eclipse CDT4 - Unix Makefiles

Generate with **Unix Makefiles**.

KDevelop3

Generates KDevelop 3 project files.

Project files for KDevelop 3 will be created in the top directory and in every subdirectory which features a CMakeLists.txt file containing a PROJECT() call. If you change the settings using KDevelop cmake will try its best to keep your changes when regenerating the project files. Additionally a hierarchy of UNIX makefiles is generated into the build tree. Any standard UNIX-style make program can build the project through the default make target. A make install target is also provided.

This extra generator may be specified as:

KDevelop3 - Unix Makefiles

Generate with **Unix Makefiles**.

KDevelop3

Generate with **Unix Makefiles**.

For historical reasons this extra generator may be specified directly as the main generator and it will be used as the extra generator with **Unix Makefiles** automatically.

Kate

Generates Kate project files.

A project file for Kate will be created in the top directory in the top level build directory. To use it in kate, the Project plugin must be enabled. The project file is loaded in kate simply by opening the ProjectName.kateproject file in the editor. If the kate Build-plugin is enabled, all targets generated by CMake are available for building.

This extra generator may be specified as:

Kate - MinGW Makefiles

Generate with **MinGW Makefiles**.

Kate - NMake Makefiles

Generate with **NMake Makefiles**.

Kate - Ninja

Generate with **Ninja**.

Kate - Unix Makefiles

Generate with **Unix Makefiles**.

Sublime Text 2

Generates Sublime Text 2 project files.

Project files for Sublime Text 2 will be created in the top directory and in every subdirectory which features a CMakeLists.txt file containing a PROJECT() call. Additionally Makefiles (or build.ninja files) are generated into the build tree. The appropriate make program can build the project through the default make target. A make install target is also provided.

This extra generator may be specified as:

Sublime Text 2 - MinGW Makefiles

Generate with **MinGW Makefiles**.

Sublime Text 2 - NMake Makefiles

Generate with **NMake Makefiles**.

Sublime Text 2 - Ninja

Generate with **Ninja**.

Sublime Text 2 - Unix Makefiles

Generate with **Unix Makefiles**.

COPYRIGHT

2000-2014 Kitware, Inc.