

Лекция 3. Система построения проектов CMake

Разработка многоплатформенного ПО

19 октября 2016 г.

Схема работы

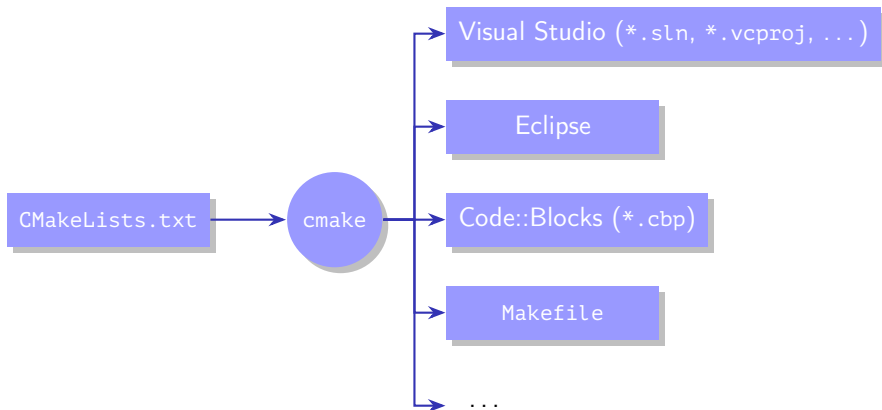


Рис. 1: генерация проектов при помощи утилиты CMake

Структура проекта

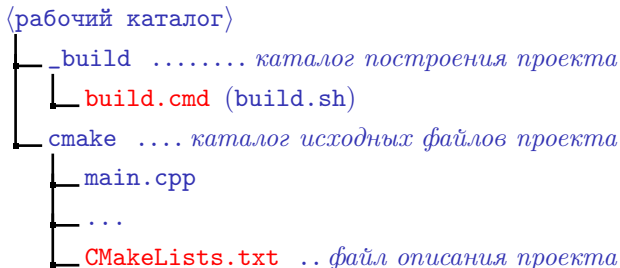


Рис. 2: структура каталога простого проекта

Пример

Пример (build.cmd, использование переменной окружения PATH)

```
@set PATH=C:\Program Files (x86)\CodeBlocks\MinGW\bin;%PATH%

cmake -G "CodeBlocks - MinGW Makefiles" ../test_cmake
```

Пример (build.cmd, использование переменной CMake CMAKE_PREFIX_PATH)

```
@set PATH=C:\Qt\Qt5.7.0\Tools\mingw530_32\bin;%PATH%

cmake^
  -G "MinGW Makefiles"^
  -D CMAKE_PREFIX_PATH="C:\Qt\Qt5.7.0\5.7\mingw53_32"^

..\qt-examples-2
```

Пример

Пример (CMakeLists.txt)

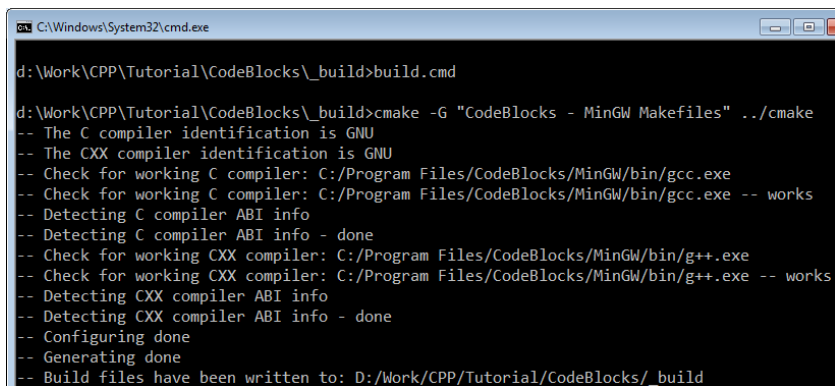
CMakeLists.txt для тестового проекта

```
project(test_cmake)
```

```
add_executable(test_cmake main.cpp)
```

Конец файла

Пример (продолжение)



```
C:\Windows\System32\cmd.exe

d:\Work\CPP\Tutorial\CodeBlocks\_build>build.cmd

d:\Work\CPP\Tutorial\CodeBlocks\_build>cmake -G "CodeBlocks - MinGW Makefiles" ../cmake
-- The C compiler identification is GNU
-- The CXX compiler identification is GNU
-- Check for working C compiler: C:/Program Files/CodeBlocks/MinGW/bin/gcc.exe
-- Check for working C compiler: C:/Program Files/CodeBlocks/MinGW/bin/gcc.exe -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working CXX compiler: C:/Program Files/CodeBlocks/MinGW/bin/g++.exe
-- Check for working CXX compiler: C:/Program Files/CodeBlocks/MinGW/bin/g++.exe -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Configuring done
-- Generating done
-- Build files have been written to: D:/Work/ CPP/ Tutorial/ CodeBlocks/ _build
```

Рис. 3: вывод программы CMake

Пример (продолжение)

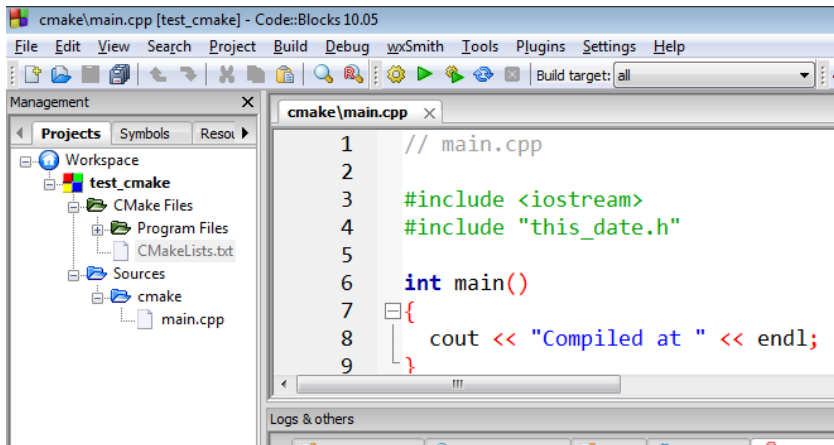


Рис. 4: среда Code::Blocks с открытым сгенерированным проектом

Структура проекта

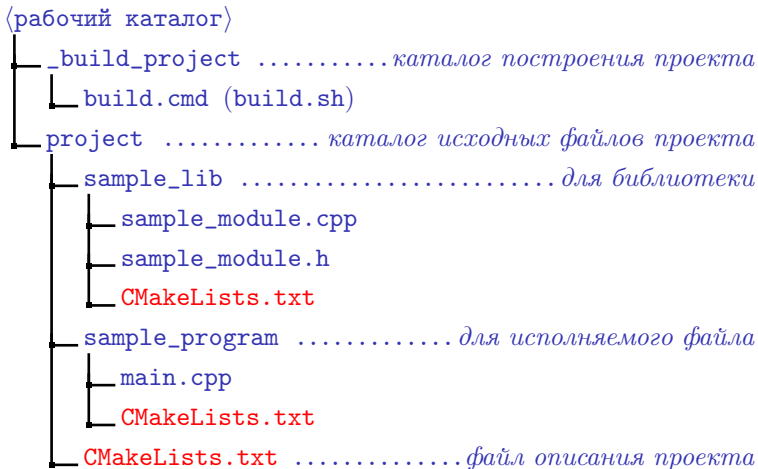


Рис. 5: структура каталога проекта с библиотекой

Пример

Пример (CMakeLists.txt)

```
cmake_minimum_required(VERSION 2.8)
```

```
add_subdirectory(sample_lib)
```

```
add_subdirectory(sample_program)
```

Пример

Пример (CMakeLists.txt)

```
cmake_minimum_required(VERSION 2.8)

add_subdirectory(sample_lib)
add_subdirectory(sample_program)
```

Пример (sample_lib/CMakeLists.txt)

```
project(sample_lib)

add_library(sample_lib sample_module.cpp sample_module.h)
```

Пример

Пример (sample_program/CMakeLists.txt)

```
project(sample_program)
```

```
add_executable(sample_program main.cpp)
```

```
include_directories(../sample_lib)
```

```
target_link_libraries(sample_program sample_lib)
```

Пример

Пример (build.cmd)

```
cmake -G "CodeBlocks - MinGW Makefiles" -DBUILD_SHARED_LIBS=0 ../project
```

Пример

Пример (build.cmd)

```
cmake -G "CodeBlocks - MinGW Makefiles" -DBUILD_SHARED_LIBS=0 ../project
```

Пример (sample_lib/CMakeLists.txt)

```
set(BUILD_SHARED_LIBS FALSE)
```

Пример

Пример (build.cmd)

```
cmake -G "CodeBlocks - MinGW Makefiles" -DBUILD_SHARED_LIBS=0 ../project
```

Пример (sample_lib/CMakeLists.txt)

```
set(BUILD_SHARED_LIBS FALSE)
```

Пример (sample_lib/CMakeLists.txt)

```
project(sample_lib)
```

```
add_library(sample_lib STATIC sample_module.cpp sample_module.h)
```

Стандартные переменные

CMAKE_INCLUDE_PATH	CMAKE_LIBRARY_PATH	CMAKE_PROGRAM_PATH
CMAKE_FRAMEWORK_PATH	CMAKE_APPBUNDLE_PATH	CMAKE_PREFIX_PATH
CMAKE_INSTALL_PREFIX	BUILD_SHARED_LIBS	

Таблица 1: влияющие на поведение CMake

CYWIN	MSVC_VERSION:	1200, 1300, ..., 1900
MINGW	CMAKE_COMPILER_IS_GNUCC	
MSVC	CMAKE_COMPILER_IS_GNUCXX	
MSVC80	UNIX	
MSVC_IDE	WIN32	

Таблица 2: описывающие систему

Стандартные переменные (окончание)

CMAKE_BINARY_DIR	CMAKE_CURRENT_BINARY_DIR
CMAKE_SOURCE_DIR	CMAKE_CURRENT_SOURCE_DIR

Таблица 3: для информации

CMAKE_INCLUDE_CURRENT_DIR	CMAKE_LIBRARY_OUTPUT_DIRECTORY
CMAKE_RUNTIME_OUTPUT_DIRECTORY	CMAKE_ARCHIVE_OUTPUT_DIRECTORY

Таблица 4: управляющие построением

Пример

Пример (CMakeLists.txt)

```
project(my_project)

set(BINARY_DIR "${CMAKE_BINARY_DIR}")
set(CMAKE_RUNTIME_OUTPUT_DIRECTORY "${BINARY_DIR}/bin")
set(CMAKE_LIBRARY_OUTPUT_DIRECTORY "${BINARY_DIR}/lib")
set(CMAKE_ARCHIVE_OUTPUT_DIRECTORY "${BINARY_DIR}/lib")

add_subdirectory(my_library_1)
add_subdirectory(my_library_2)
add_subdirectory(my_program)
# ...
```

Команды добавления целей

Добавление целей

add_executable(

 <логическое_имя_цели> [WIN32]
 <исходный_модуль₁> ... <исходный_модуль_n>)

add_library(

 <логическое_имя_цели> [STATIC | SHARED | MODULE]
 <исходный_модуль₁> ... <исходный_модуль_n>)

add_subdirectory(

 <подкаталог_проекта> [<подкаталог_построения>])

Команды добавления целей (окончание)

Добавление целей (окончание)

```
add_library(
```

```
  <логическое_имя_цели>
```

```
  <тип_библиотеки>
```

```
  IMPORTED)
```

```
<тип_библиотеки> ::= SHARED | STATIC | MODULE | UNKNOWN
```

Команды настроек каталогов подпроектов

Настройка целей

```
include_directories(  
  ⟨каталог1⟩ ... ⟨каталогn⟩)  
  
add_definitions(  
  ⟨определение1⟩ ... ⟨определениеn⟩)  
  
add_compile_options(  
  ⟨настройка1⟩ ... ⟨настройкаn⟩)
```

Примеры

```
include_directories(  
  include)  
  
add_definitions(  
  -DDEBUG -DEXTRA_TESTS=4)  
  
add_compile_options(  
  -std=c++11)
```

Команды настроек целей

Команды

```
target_link_libraries(  
  <имя_цели>  
  [ <библиотека1> ... <библиотекаn> ] )  
  
<имя_команды>(   
  <имя_цели>  
  INTERFACE | PUBLIC | PRIVATE  
  [ <настройка1,1> ... <настройка1,m> ]  
  [  
    INTERFACE | PUBLIC | PRIVATE  
    [ <настройка2,1> ... <настройка2,n> ]  
    ...  
  ] )
```

Команды

```
<имя_команды> ::=  
  target_include_directories |  
  target_compile_definitions |  
  target_compile_options |  
  target_compile_features |  
  target_link_libraries
```

Пример использования настроек транзитивных свойств

Пример (библиотека)

```
add_library(  
  my_lib  
  my_lib.cpp my_lib.h)  
  
target_include_directories(  
  my_lib  
  INTERFACE .)
```

Пример (приложение)

```
add_executable(  
  my_prog  
  my_prog.cpp)  
  
target_link_libraries(  
  my_prog my_lib)
```

Пример использования настроек транзитивных свойств

Пример (библиотека)

```
add_library(  
  my_lib  
  my_lib.cpp my_lib.h)  
  
set(  
  CMAKE_INCLUDE_CURRENT_DIR_IN_INTERFACE  
  ON)
```

Пример (приложение)

```
add_executable(  
  my_prog  
  my_prog.cpp)  
  
target_link_libraries(  
  my_prog my_lib)
```

Пример использования настроек транзитивных свойств

Пример (ex-cpp14.cpp, C++14)

```
#include <iostream>

int main()
{
    auto n = 0b0'0100'1011;
    std::cout << n << std::endl;
}
```

Пример (CMakeLists.txt)

```
cmake_minimum_required(VERSION 3.2.0)

project(ex-cpp14)

add_executable(ex-cpp14 ex-cpp14.cpp)

target_compile_features(
    ex-cpp14
    PRIVATE
    cxx_auto_type
    cxx_binary_literals
    cxx_digit_separators)
```


Команды работы с переменными

Присваивание значений

```
set(  
  <имя_переменной> <значение>  
  [ [ CACHE <тип> <строка_описания> ] | PARENT_SCOPE ] )  
  
<тип> ::=  
  FILEPATH | PATH | STRING | BOOL | INTERNAL  
  
option(  
  <имя_переменной> <строка_описания> [ ON | OFF ] )
```

Команды работы с переменными

Присваивание значений (окончание)

```
set(⟨имя_переменной⟩)
```

```
unset(⟨имя_переменной⟩ [ CACHE | PARENT_SCOPE ])
```

```
set(  
  ⟨имя_переменной⟩ ⟨значение1⟩ ... ⟨значениеn⟩
```

```
math(EXPR ⟨имя_переменной⟩ ⟨выражение⟩)
```

Свойства

Объекты

- Каталоги проектов;
- Цели;
- Тесты;
- Исходные файлы;
- Переменные кэша;
- Файлы для установки.

Команда опроса свойств

Команды

```
get_property(  
  <имя_переменной>  
  <сущность>  
  PROPERTY <имя_свойства>  
  [ SET | DEFINED ] )
```

Команды

```
<сущность> ::=  
  GLOBAL |  
  DIRECTORY [ <каталог> ] |  
  TARGET <имя_цели> |  
  SOURCE <файл> |  
  INSTALL <файл> |  
  TEST <имя_теста> |  
  CACHE <имя_переменной> |  
  VARIABLE
```

Команда установки свойств

Присваивание значений (окончание)

```
set_property(  
  <сущности>  
  [ APPEND ] [ APPEND_STRING ]  
  PROPERTY <имя_свойства>  
  [ <значение1> ... <значениеm> ] )
```

Команда установки свойств (окончание)

Присваивание значений (окончание)

```
⟨сущности⟩ ::=  
GLOBAL |  
DIRECTORY [ ⟨каталог⟩ ] |  
TARGET [ ⟨имя_цели1⟩ ... ⟨имя_целиn⟩ ] |  
SOURCE [ ⟨файл1⟩ ... ⟨файлn⟩ ] |  
INSTALL [ ⟨файл1⟩ ... ⟨файлn⟩ ] |  
TEST [ ⟨имя_теста1⟩ ... ⟨имя_тестаn⟩ ] |  
CACHE [ ⟨имя_переменной1⟩ ... ⟨имя_переменнойn⟩ ]
```

Пример

Пример (CMakeLists.txt)

```
add_executable(exec main.cpp file1.cpp file1.h)
```

```
set_property(  
  TARGET exec  
  PROPERTY OUTPUT_NAME  
  my_prog)
```

Команда ветвления

Ветвление

```
if(<условие1>)  
  <команды>  
[ elseif(<условие2>)  
  <команды>  
... ]  
[ else()  
  <команды> ]  
endif()
```


Команда ветвления (окончание)

Ветвление (окончание)

```
⟨условие⟩ ::=  
  ⟨переменная⟩ | (⟨условие⟩) | NOT ⟨условие⟩ |  
  ⟨условие⟩ AND ⟨условие⟩ | ⟨условие⟩ OR ⟨условие⟩ |  
  EXISTS ⟨файл_или_каталог⟩ | IS_DIRECTORY ⟨путь⟩ |  
  ⟨переменная_или_значение⟩ LESS ⟨переменная_или_значение⟩ |  
  ⟨переменная_или_значение⟩ GREATER ⟨переменная_или_значение⟩ |  
  ⟨переменная_или_значение⟩ EQUAL ⟨переменная_или_значение⟩ |  
  ⟨переменная_или_значение⟩ STRLESS ⟨переменная_или_значение⟩ |  
  ⟨переменная_или_значение⟩ MATCHES ⟨регулярное_выражение⟩ |  
  ...
```

Пример

Пример (CMakeLists.txt)

```
if(MSVC AND MSVC_VERSION GREATER 1400)
  add_definitions(/MP)
endif()
```

Команды перебора значений

Перебор значений

```
foreach(⟨имя_переменной⟩ ⟨значение1⟩ ... ⟨значениеn⟩)  
  ⟨команды⟩  
endforeach()
```

```
foreach(⟨имя_переменной⟩ IN  
  [ LISTS ⟨имя_переменной1⟩ ... ⟨имя_переменнойn⟩ ]  
  [ ITEMS ⟨значение1⟩ ... ⟨значениеn⟩ ] )
```

```
foreach(⟨имя_переменной⟩ RANGE ⟨максимум⟩)
```

```
foreach(⟨имя_переменной⟩ RANGE ⟨старт⟩ ⟨стоп⟩ [ ⟨шаг⟩ ] )
```

Включение файла или модуля

Команда `include()`

```
include(  
  ⟨файл⟩ | ⟨модуль⟩  
  [OPTIONAL] [RESULT_VARIABLE ⟨имя_переменной⟩])
```

Пример

Пример (CMakeLists.txt)

```
cmake_minimum_required(VERSION 2.8)

set(
  SUBPROJECTS
  program1 program2 super_program)

include(build.cmake)
```

Пример (продолжение)

Пример (build.cmake)

```
foreach(PROJ ${SUBPROJECTS})
  set(
    MY_BUILD_${PROJ} TRUE
    CACHE BOOL "Build the ${PROJ} subproject")
  if(MY_BUILD_${PROJ} AND
     EXISTS
       "${CMAKE_SOURCE_DIR}/${PROJ}/CMakeLists.txt")
    message(STATUS "The project ${PROJ} will be included")
    add_subdirectory(${PROJ})
  else()
    message(STATUS "The project ${PROJ} will NOT be included")
  endif()
endforeach()
```

Пример (окончание)

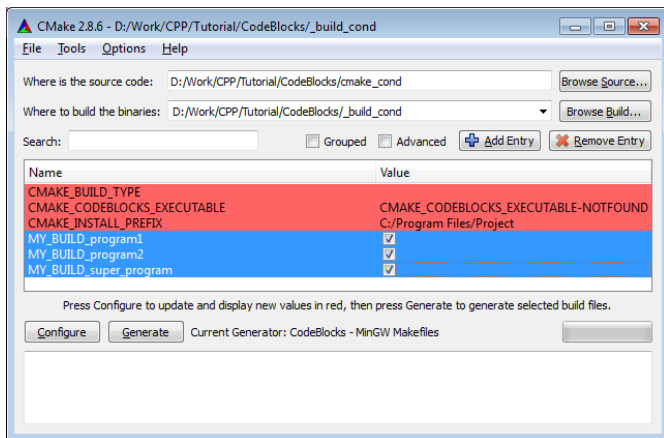


Рис. 6: генерирование проекта при помощи cmake

Команды циклов и функций

Цикл

```
while(⟨условие⟩)  
  ⟨команды⟩  
endwhile()
```

```
break()
```

```
function(⟨имя_функции⟩ [ ⟨параметр1⟩ ... ⟨параметрн⟩ ])  
  ⟨команды⟩  
endfunction()
```

ARGC

ARGV0, ARGV1, ... ARGV

ARGN

Таблица 5: переменные для доступа к параметрам

Пример

Пример (build.cmake)

```
function(my_create_projects BASE_NAME)
  foreach(PROJ ${ARGN})
    # ...
  endforeach()
endfunction()

# ...

my_create_projects(
  "my_application"
  project1 project2 superproject)
```

Команда разбора пути

Команда `get_filename_component()`

```
get_filename_component(  
  <имя_переменной> <путь_к_файлу> <компонент>  
  [CACHE])
```

```
<компонент> ::=  
  DIRECTORY | NAME | EXT | NAME_WE | ABSOLUTE | REALPATH
```

Команды поиска файлов

Команды `find_file()`, `find_library()`, `find_path()`, `find_program()`

```
<имя_команды>(
  <имя_переменной>
  <альтернативные_имена>
  [HINTS <путь1,1> ... <путь1,m> [ENV <имя_окружения1>]]
  [PATHS <путь2,1> ... <путь2,p> [ENV <имя_окружения2>]]
  [PATH_SUFFIXES <суффикс1> ... <суффиксk>]
  [DOC <строка_документации>])
```

```
<имя_команды> ::=
  find_file | find_library | find_path | find_program
```

```
<альтернативные_имена> ::=
  <имя> | NAMES <имя1> ... <имяn>
```

Каталоги поиска

Команда	Путь из CMAKE_PREFIX_PATH	Путь 2
find_file() , find_path()	⟨путь⟩/include	CMAKE_INCLUDE_PATH, CMAKE_FRAMEWORK_PATH
find_library()	⟨путь⟩/lib	CMAKE_LIBRARY_PATH, CMAKE_FRAMEWORK_PATH
find_program()	⟨путь⟩/bin, ⟨путь⟩/sbin	CMAKE_PROGRAM_PATH, CMAKE_APPBUNDLE_PATH

Таблица 6: каталоги поиска для команд

Импорт библиотеки

Пример

```
find_path(MPIR_H_DIR mpir.h)
if(NOT MPIR_H_DIR)
  message(
    SEND_ERROR
    "Could not find mpir.h")
endif()
find_library(MPIR_LIB mpir)
if(NOT MPIR_LIB)
  message(
    SEND_ERROR
    "Could not find mpir")
endif()
```

Пример (окончание)

```
add_library(mpir STATIC IMPORTED)
set_property(
  TARGET mpir
  PROPERTY
    INTERFACE_INCLUDE_DIRECTORIES
    ${MPIR_H_DIR})
set_property(
  TARGET mpir
  PROPERTY IMPORTED_LOCATION
    ${MPIR_LIB})
# ...
target_link_libraries(my_prog mpir)
```

Команда поиска пакета

Команда `find_package()`

```
find_package(  
  <имя_пакета> [<версия>] [EXACT] [QUIET] [REQUIRED]  
  [[COMPONENTS] [<компонент1> ... <компонентn>]]  
  [CONFIG | NO_MODULE]  
  [NAMES <имя1> ... <имяn>])
```

Пример

Пример (build.cmake)

```
cmake_minimum_required(VERSION 2.8)

project(test_opencv)

find_package(OpenCV REQUIRED core highgui imgproc)

add_executable(test_opencv WIN32 test_opencv.cpp)
include_directories(${OpenCV_INCLUDE_DIRS})
target_link_libraries(test_opencv ${OpenCV_LIBS})
```

Команда установки целей

Команда `install()`

```
install(  
  TARGETS <имя_цели1> ... <имя_целиn>  
  [  
    [ ARCHIVE | LIBRARY | RUNTIME ]  
    [ DESTINATION <каталог> ]  
    [ CONFIGURATIONS [ Debug | Release | ... ] ]  
    [ COMPONENT <имя_компонента> ]  
  ]  
  ...)
```


Команда `install()` файлов

Команда `install()` (продолжение)

```
install(  
  FILES | PROGRAMS <файл1> ... <файлn>  
  DESTINATION <каталог>  
  [ CONFIGURATIONS [ Debug | Release | ... ] ]  
  [ COMPONENT <имя_компонента> ]  
  [ RENAME <имя> ] )
```

Команда установки содержимого каталогов

Команда `install()` (окончание)

```
install(  
  DIRECTORY [ <каталог1> ... <каталогn> ]  
  DESTINATION <каталог>  
  [ CONFIGURATIONS [ Debug | Release | ... ] ]  
  [ COMPONENT <имя_компонента> ]  
  [ FILES_MATCHING ]  
  [  
    [ PATTERN <маска> | REGEX <регулярное_выражение> ]  
    [ EXCLUDE ]  
  ]  
  [ ... ] )
```

Установка библиотеки

Пример

```
add_library(  
  my_library_1  
  f.cpp f.h)  
  
get_property(  
  LIB_TYPE  
  TARGET my_library_1  
  PROPERTY TYPE)
```

Пример (продолжение)

```
if(LIB_TYPE STREQUAL SHARED_LIBRARY)  
  install(  
    TARGETS my_library_1  
    COMPONENT user  
    RUNTIME  
    DESTINATION bin  
    LIBRARY  
    DESTINATION lib)  
endif()
```

Установка библиотеки (окончание)

Пример (продолжение)

```
install(  
  TARGETS my_library_1  
  COMPONENT developer  
  RUNTIME  
    DESTINATION bin  
  LIBRARY  
    DESTINATION lib  
  ARCHIVE  
    DESTINATION lib)
```

Пример (окончание)

```
install(  
  DIRECTORY .  
  DESTINATION include  
  COMPONENT developer  
  FILES_MATCHING  
    PATTERN "*.h")
```

Сценарий построения и установки библиотеки

Пример (build.cmd)

```
cmake^
  -G "MinGW Makefiles"^
  -D CMAKE_INSTALL_PREFIX=D:\Install\my_library_1^
  D:\Work\Source\my_library_1

mingw32-make

cmake -D COMPONENT=developer -P cmake_install.cmake
```

Команда добавления фальшивой цели

Команда `add_custom_target()`

```
add_custom_target(  
  <логическое_имя_цели>  
  [ALL]  
  [<путь_к_команде_1> [<аргумент_1,1> ... <аргумент_1,m>]]  
  [  
    COMMAND  
    <путь_к_команде_2> [<аргумент_2,1> ... <аргумент_2,n>]  
    ...  
  ]  
  [DEPENDS <файл_1> ... <файл_k>]  
  [WORKING_DIRECTORY <каталог>]  
  [VERBATIM]  
  [SOURCES <исходный_модуль_1> ... <исходный_модуль_p>])
```

Пример

Пример (7zip.cmake)

```
set(BINDIR32_ENV_NAME "ProgramFiles(x86)")

find_program(
  7ZIP_EXECUTABLE
  NAMES
    7z 7za
  PATHS
    "$ENV{ProgramFiles}/7-Zip"
    "$ENV{${BINDIR32_ENV_NAME}}/7-Zip"
    "C:/Program Files/7-Zip"
    "C:/Program Files (x86)/7-Zip"
)
```

Пример (окончание)

Пример (7zip.cmake, окончание)

```
if(7ZIP_EXECUTABLE)
  add_custom_target(
    create_archive
    COMMAND
      "${7ZIP_EXECUTABLE}"
      a "${PROJECT_NAME}.7z" "${PROJECT_SOURCE_DIR}"
    WORKING_DIRECTORY
      "${PROJECT_BINARY_DIR}"
  )
else()
  message(
    WARNING "7-zip not found")
endif()
```