

cmake(八)Cmake定义安装

原创 wzj_110 于 2021-04-18 19:53:58 发布 阅读量632 收藏2 点赞数5
分类专栏: [cmake DSL语言](#)

版权



cmake DSL语言 专栏收录该内容

38 篇文章

已订阅

一 cmake安装汇总

目录结构

参考博客

① 语法规则

CMake中可通过**INSTALL**指令来定义安装规则，并配合**CMAKE_INSTALL_PREFIX**变量来指定安装的路径
即执行cmake命令时指定：
cmake -DCMAKE_INSTALL_PREFIX=/tmp

定义安装的规则内容可以是可执行的**二进制目标**、**动态库（共享库）**、**静态库**、**文件**、**目录**、**脚本**等

类型

② 安装规则

定义安装规则包括以下几方面：

1. **二进制目标文件**（可执行、动态库、静态库）的安装
2. **普通文件**的安装
3. **非目标可执行程序**（如**脚本**）的安装
4. **目录**的安装
5. **安装时执行cmake脚本**

③ 二进制目标文件

1. **二进制目标文件**（可执行、动态库、静态库）的安装

```
INSTALL(TARGETS targets... [[ARCHIVE|LIBRARY|RUNTIME]
[DESTINATION <dir>] [PERMISSIONS permissions...]
[CONFIGURATIONS [Debug|Release|...]]
[COMPONENT <component>] [OPTIONAL]] [...])
```

TARGETS: 后面的**targets**就是**add_executable**或**add_library**定义的目标，二进制目标或库

ARCHIVE|LIBRARY|RUNTIME: 分别指**静态库**、**动态库**、**可执行二进制目标**

DESTINATION <dir>: 定义要安装到哪个路径，结合**CMAKE_INSTALL_PREFIX**，路径为**\$(CMAKE_INSTALL_PREFIX)/DESTINATION**，如果以**/**开头，则从**根目录**开始，这时**CMAKE_INSTALL_PREFIX**就不起作用了，所以一般写相对路径。

PERMISSIONS: 指定**权限**

Debug|Release: 指定**版本**

其它: **少用，不说**

举个例子：

```
INSTALL(TARGETS exe sharedlib staticlib
  RUNTIME DESTINATION bin
  LIBRARY DESTINATION lib
  ARCHIVE DESTINATION static_lib
)
```

分门别类安装

exe安装到\${CMAKE_INSTALL_PREFIX}/bin下
sharedlib安装到\${CMAKE_INSTALL_PREFIX}/lib下
staticlib安装到\${CMAKE_INSTALL_PREFIX}/static_lib下

参数中的TARGET可以是很多种目标文件，最常见的是通过ADD_EXECUTABLE或者ADD_LIBRARY定义的目标文件，即可执行二进制、动态库、静态库：

目标文件	内容	安装目录变量	默认安装文件夹
ARCHIVE	静态库	\${CMAKE_INSTALL_LIBDIR}	lib
LIBRARY	动态库	\${CMAKE_INSTALL_LIBDIR}	lib
RUNTIME	可执行二进制文件	\${CMAKE_INSTALL_BINDIR}	bin
PUBLIC_HEADER	与库关联的PUBLIC头文件	\${CMAKE_INSTALL_INCLUDEDIR}	include
PRIVATE_HEADER	与库关联的PRIVATE头文件	\${CMAKE_INSTALL_INCLUDEDIR}	include

为了符合一般的默认安装路径，如果设置了DESTINATION参数，推荐配置在安装目录变量下的文件夹

```
1  INSTALL(TARGETS myrun mylib mystaticlib
2      RUNTIME DESTINATION ${CMAKE_INSTALL_BINDIR}
3      LIBRARY DESTINATION ${CMAKE_INSTALL_LIBDIR}
4      ARCHIVE DESTINATION ${CMAKE_INSTALL_LIBDIR}
5  )
```

与后边目标文件一一匹配

上面的例子会将：可执行二进制myrun安装到 \${CMAKE_INSTALL_BINDIR} 目录，动态库 libmylib.so 安装到 \${CMAKE_INSTALL_LIBDIR} 目录，静态库 libmystaticlib.a 安装到 \${CMAKE_INSTALL_LIBDIR} 目录。

④ 普通文件安装

2. 普通文件的安装

```
INSTALL(FILES files... DESTINATION <dir>
  [PERMISSIONS permissions...]
  [CONFIGURATIONS [Debug|Release|...]]
  [COMPONENT <component>]
  [RENAME <name>] [OPTIONAL])
```

用于安装一般文件，可指定访问权限，文件名是此指令所在路径下的相对路径。如果不定义权限PERMISSIONS，安装后的权限为：OWNER_WRITE OWNER_READ|GROUP_READ|WORLD_READ，即644权限

⑤ 非目标可执行程序

备注：一般是 'sh'、'python' 脚本

3. 非目标可执行程序（如脚本）的安装

```
INSTALL(PROGRAMS files... DESTINATION <dir>
    [PERMISSIONS permissions...]
    [CONFIGURATIONS [Debug|Release|...]]
    [COMPONENT <component>]
    [RENAME <name>] [OPTIONAL])
```

与普通文件安装差不多，只是默认权限不一样：

```
OWNER_EXECUTE OWNER_WRITE OWNER_READ|
GROUP_EXECUTE GROUP_READ|WORLD_EXECUTE
WORLD_READ，即755权限——加上执行权限
```

https://blog.csdn.net/wzj_110

⑥ 目录的安装

4. 目录的安装

```
INSTALL(DIRECTORY dirs... DESTINATION <dir>
    [FILE_PERMISSIONS permissions...]
    [DIRECTORY_PERMISSIONS permissions...]
    [USE_SOURCE_PERMISSIONS]
    [CONFIGURATIONS [Debug|Release|...]]
    [COMPONENT <component>]
    [[PATTERN <pattern> | REGEX <regex>]
    [EXCLUDE] [PERMISSIONS permissions...]] [...])
```

DIRECTORY: 后面接的是所在源目录的相对路径，目录后有没有/，区别很大，如dir与dir/是不一样的，dir是将dir这个目录安装为目标路径下的dir，而dir/是将这个目录下的内容安装到目标目录，但不包括这个目录本身

PATTERN: 使用正则表达式进行过滤

PERMISSIONS: 用于指定PATTERN过滤后的文件权限

https://blog.csdn.net/wzj_110

举个栗子：

```
INSTALL(DIRECTORY samples modules/ DESTINATION
share
    PATTERN "TXT" EXCLUDE
    PATTERN "modules/*"
    PERMISSIONS OWNER_EXECUTE OWNER_WRITE
OWNER_READ GROUP_EXECUTE GROUP_READ)
```

将samples目录安装到\${CMAKE_INSTALL_PREFIX}/share目录下，将modules/中的内容安装到\${CMAKE_INSTALL_PREFIX}/share目录下。不包含目录名为TXT的目录，对modules/目录下的文件指定权限为OWNER_EXECUTE OWNER_WRITE OWNER_READ GROUP_EXECUTE GROUP_READ

https://blog.csdn.net/wzj_110

⑦ 安装时执行的cmake脚本

```
1 | find / -name *.cmake
2 |
3 | 备注： 注意'cmake'后缀文件的'编写'语法
4 |
5 | 备注： []是'非必须'
```

5. 安装时执行cmake脚本

INSTALL([[SCRIPT <file>] [CODE <code>]] [...])

SCRIPT: 用于在安装时调用cmake脚本文件, 即xxxx.cmake文件

CODE: 行cmake指令, 要用双引号, 如:

INSTALL(CODE "MESSAGE(\"Sample install message.\")")

https://blog.csdn.net/wzj_110

参考博客

二 实践

① 项目初始化

```
1  ++++++'执行步骤'+++++
2
3  1) 创建'项目'目录,'进入'该'项目'目录
4
5  2) 创建一个'src'子目录 -->存放'源文件目录'
6
7  3) 创建一个'doc'子目录 -->存放'文档文件'
8
9  4) 创建'copyright'版权文件'、'readme'操作文件'
```

```
kiosk@k8s CmakeProjects $ ls
HelloCmake HelloLibrary OutputPath SubDirectory
kiosk@k8s CmakeProjects $ mkdir CustomizeInstall
kiosk@k8s CmakeProjects $ cd CustomizeInstall/
kiosk@k8s CustomizeInstall $ mkdir {src,doc}
kiosk@k8s CustomizeInstall $ touch {copyright,readme}
kiosk@k8s CustomizeInstall $ tree .
```

```
.
├── copyright
├── doc
├── readme
└── src
```

2 directories, 2 files

https://blog.csdn.net/wzj_110

② Main.cpp编写

```
1 #include <iostream>
2 #include <stdlib.h>
3
4 int main(int argc, char** argv) {
5     std::cout << "Customize Install:自定义安装" << std::endl;
6     return EXIT_SUCCESS;
7 }
```

"Main.cpp" [Modified][New file] 7 lines --85%--

https://blog.csdn.net/wzj_110 6,24 All

③ src目录下CMakeLists.txt文件编写


```

1 # 1) 设置可执行目标输出的路径
2 # 备注: bin目录会自动创建
3 set(EXECUTABLE_OUTPUT_PATH ${PROJECT_BINARY_DIR}/bin)
4 # 2) 生成可执行的二进制文件
5 add_executable(customize_install Main.cpp)
6 # 3) 指定这个可执行二进制要安装到的目录
7 # 备注: 会安装到 ${CMAKE_INSTALL_PREFIX}/CustomizeInstall/bin
8 install(TARGETS customize_install DESTINATION CustomizeInstall/bin)

```

CMakeLists.txt

安装的前缀: 可命令行或文件中指定

src目录自身的

"CMakeLists.txt" 8 lines --100%--

8,1 All

```

kiosk@k8s CmakeProjects $ tree CustomizeInstall/
CustomizeInstall/
├── copyright
├── doc
├── readme
└── src
    ├── CMakeLists.txt
    └── Main.cpp

```

子目录自身的

当前阶段的文件预览

2 directories, 4 files

kiosk@k8s CmakeProjects \$

④ 项目主目录编写CMakeLists.txt文件

项目主目录下的CMakeLists.txt文件

```

1 cmake_minimum_required(VERSION 3.8)
2 project(CmakeProjects)
3
4 #1) 把src源文件子目录加入到项目中
5 add_subdirectory(src bin)
6
7 #2) 把copyright、readme这两个文件安装到目录-->${CMAKE_INSTALL_PREFIX}/CustomizeInstall下
8 #备注: FILES是关键字 -->普通文件类型
9 install(FILES copyright readme DESTINATION CustomizeInstall)
10
11 #3) 非目标的'可执行程序'的安装
12 #备注1: PROGRAMS是关键字 -->非目标的可执行程序
13 #备注2: 把run_customize_install.sh安装到目录: ${CMAKE_INSTALL_PREFIX}/CustomizeInstall/bin下
14 install(PROGRAMS run_customize_install.sh DESTINATION CustomizeInstall/bin)
15
16 #4) 把doc/目录下的文件(由于/的原因-->不包含这个目录自身)安装到目录: ${CMAKE_INSTALL_PREFIX}/CustomizeInstall/share/doc下
17 #备注1: DIRECTORY是关键字
18 #备注2: 等价-->doc DESTINATION CustomizeInstall/share/
19 install(DIRECTORY doc/ DESTINATION CustomizeInstall/share/doc)

```

"CMakeLists.txt" 19 lines --5%--

1,1 All

```

1 项目的'主目录下'编写'run_customize_install.sh'文件
2
3 vim run_customize_install.sh

```

```

kiosk@k8s CustomizeInstall $ cat run_customize_install.sh
./customize_install
kiosk@k8s CustomizeInstall $ 脚本作用: 简单执行customize_install命令

```

⑤ 项目主目录创建doc目录

```

kiosk@k8s CustomizeInstall $ mkdir doc
kiosk@k8s CustomizeInstall $ cd doc/
kiosk@k8s doc $ echo 'CustomizeInstall项目的文档说明' > customize_install_doc.txt
kiosk@k8s doc $ ls
customize_install_doc.txt
kiosk@k8s doc $

```

简单来个文件表示文档文件

⑥ 项目主目录创建build目录

说明: build目录来进行'外部构建'

```
kiosk@k8s doc $ cd ../
kiosk@k8s CustomizeInstall $ mkdir build
kiosk@k8s CustomizeInstall $ tree .
```

```

.
├── build
├── CMakeLists.txt
├── copyright
├── doc
│   └── customize_install_doc.txt
├── readme
├── run_customize_install.sh
└── src
    ├── CMakeLists.txt
    └── Main.cpp
```

3 directories, 7 files

cmake之前项目最终的结构

https://blog.csdn.net/wzj_110

⑦ 执行cmake

CMake命令行参数

```
-D <var>[:<type>=<value>], -D <var>=<value>
```

Create or update a CMake CACHE entry.

When CMake is first run in an empty build tree, it creates a CMakeCache.txt file and populates it with customizable settings for the project. This option may be used to specify a setting that takes priority over the project's default value. The option may be repeated for as many CACHE entries as desired.

If the :<type> portion is given it must be one of the types specified by the set() command documentation for its CACHE signature. If the :<type> portion is omitted the entry will be created with no type. If it does not exist with a type already. If a command in the project sets the type to PATH or FILEPATH then the <value> will be converted to an absolute path.

This option may also be given as a single argument: -D<var>[:<type>=<value>] or -D<var>=<value>.

部分

省略

https://blog.csdn.net/wzj_110

- 1 一般： 我们会在'CMakeLists.txt'文件指定默认的'PREFIX',如果需要'自定义',则命令行进行'覆盖'
- 2
- 3 cmake_install.cmake '验证'是否'覆盖'

```
kiosk@k8s build $ pwd
/var/ftp/pub/pub/cmake/test/CmakeProjects/CustomizeInstall/build
kiosk@k8s build $ ls
kiosk@k8s build $ cmake3 -DCMAKE_INSTALL_PREFIX=/tmp/install ..
```

```
-- The C compiler identification is GNU 4.8.5
-- The CXX compiler identification is GNU 4.8.5
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc - works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ - works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /var/ftp/pub/pub/cmake/test/CmakeProjects/CustomizeInstall/build
```

```
kiosk@k8s build $ ls
bin  CMakeCache.txt  CMakeFiles  cmake_install.cmake  Makefile
kiosk@k8s build $ ll bin/
total 20
drwxrwxr-x 3 kiosk kiosk 4096 Apr 18 19:38 CMakeFiles
-rw-rw-r-- 1 kiosk kiosk 2147 Apr 18 19:38 cmake_install.cmake
-rw-rw-r-- 1 kiosk kiosk 8393 Apr 18 19:38 Makefile
kiosk@k8s build $ make
```

```
Scanning dependencies of target customize_install
[ 50%] Building CXX object bin/CMakeFiles/customize_install.dir/Main.cpp.o
[100%] Linking CXX executable customize_install
[100%] Built target customize_install
```

```
kiosk@k8s build $ ll bin/
total 36
drwxrwxr-x 3 kiosk kiosk 4096 Apr 18 19:38 CMakeFiles
-rw-rw-r-- 1 kiosk kiosk 2147 Apr 18 19:38 cmake_install.cmake
-rwxrwxr-x 1 kiosk kiosk 12712 Apr 18 19:39 customize_install
-rw-rw-r-- 1 kiosk kiosk 8393 Apr 18 19:38 Makefile
```

```
kiosk@k8s build $ bin/customize_install
Customize Install:自定义安装
```

通过命令名指定

二进制可执行文件

测试

https://blog.csdn.net/wzj_110

⑧ make install 之后观察/tmp/install目录

备注： 符合'预期'

```
kiosk@k8s build $ ll /tmp/install/
total 0
kiosk@k8s build $ make install
[100%] Built target customize_install
Install the project...
-- Install configuration: ""
-- Installing: /tmp/install/CustomizeInstall/copyright
-- Installing: /tmp/install/CustomizeInstall/readme
-- Installing: /tmp/install/CustomizeInstall/bin/run_customize_install.sh
-- Installing: /tmp/install/CustomizeInstall/share/doc
-- Installing: /tmp/install/CustomizeInstall/share/doc/customize_install_doc.txt
-- Installing: /tmp/install/CustomizeInstall/bin/customize_install
kiosk@k8s build $ tree /tmp/install/
/tmp/install/
├── CustomizeInstall
│   ├── bin
│   │   ├── customize_install
│   │   └── run_customize_install.sh
│   ├── copyright
│   ├── readme
│   └── share
│       └── doc
│           └── customize_install_doc.txt
4 directories, 5 files
```

提示信息可知：按照CMakeList.txt指定路径进行安装

安装之后的结构

https://blog.csdn.net/wzj_110

```
kiosk@k8s build $ tree /tmp/install/
/tmp/install/
├── CustomizeInstall
│   ├── bin
│   │   ├── customize_install
│   │   └── run_customize_install.sh
│   ├── copyright
│   ├── readme
│   └── share
│       └── doc
│           └── customize_install_doc.txt
4 directories, 5 files
kiosk@k8s build $ cd /tmp/install/CustomizeInstall/bin/
kiosk@k8s bin $ cat run_customize_install.sh
./customize_install
kiosk@k8s bin $ ./run_customize_install.sh
Customize Install:自定义安装
```

符合预期

https://blog.csdn.net/wzj_110

显示推荐内容