Cmake

mingzailao

<2016-11-01 Tue>

Outline

Cmake

初识 cmake

背景知识

官网: www.cmake.org

特点

1. 开放源代码。
2. 跨平台, 并可生成 native 编译配置文件, 在 Linux/Unix 平台, 生成 makefile, 在苹果平台, 可以生成 xcode, 在 Windows 平台, 可以生成 MSVC 的工程文件。
3. 能够管理大型项目。
4. 简化编译构建过程和编译过程。
5. 高效虑。
6. 可扩展。

初试 cmake

问题

1. cmake 很简单, 但绝对没有听起来或者想象中那么简单。
2. cmake 编写的过程实际上是编程的过程不过你需要编写的是 CMakeLists.txt(每个目录一个), 使用的是”cmake 语言和语法”。

建议

1. 如果你的工程只有几个文件, 直接编写 Makefile 是最好的选择。
2. 如果使用的是 C/C++/Java 之外的语言, 请不要使用 cmake。
3. 结合具体的项目使用

Cmake

安装 cmake

下载地址

在这个页面, 提供了源代码的下载以及针对各种不同操作系统的二进制下载, 可以选择适合自己操作系统的版本下载安装:
www.cmake.org/HTML/Download.html

Cmake

初试 cmake

准备工作

```
mkdir ~/cmake
cd ~/cmake
mkdir project1
cd project1
touch main.cpp
touch CMakeList.txt
emacs CMakeList.txt
```

注意: 所有的 cpp,hpp 文件均为写好的。

初试 cmake

CMakeList.txt 写法 1

```
# 工程名
PROJECT (HELLO) cmake_minimum_required(VERSION 3.6) add_com
# 设定2 hello.o 由hello.cpp构建 ADD_EXECUTABLE(hello hello.
```

初试 cmake

开始构建

```
ls ~/PAPERS/project1/
```

CMakeLists.txt
hello.cpp

```
d ~/PAPERS/project1
cmake .
```

– Configuring done
– Generating done
– Build files have been written to: /Users/apple/PAPERS/project1

初试 cmake

问题

```
ls ~/PAPERS/project1
```

CMakeCache.txt
CMakeFiles
CMakeLists.txt
Makefile
cmake_install.cmake
hello
hello.cpp

无法达到分离式编译的效果。

初试 cmake

改进

上面的方式称为内部构建看到生成的临时文件比代码文件还要多, 强迫症接受不了。估计这辈子你都不希望再使用内部构建。下面介绍外部构建。

1. 首先, 请清除 project1 目录中除 hello.cpp CmakeLists.txt 之外的所有中间文件。
2. 在当前目录下新建 build 目录然后进入 build 目录运行 cmak ..

初试 cmake

语法

```
PROJECT(projectname [CXX] [C] [Java])
```

你可以用这个指令定义工程名称, 并可指定工程支持的语言, 支持的语言列表是可以忽略的, 默认情况表示支持所有语言。

初试 cmake

语法

```
SET(VAR [VALUE] [CACHE TYPE DOCSTRING [FORCE]])
#现阶段,你只需要了解 SET 指令可以用来显式的定义变量即可。
#比如我们用到的是SET(SRC_LIST hello.cpp),
#如果有多个源文件,也可以定义成:
# SET(SRC_LIST main.c t1.c t2.c)。
```

初试 cmake

语法

```
ADD_EXECUTABLE(hello ${SRC_LIST})
```

定义了这个工程会生成一个文件名为 hello 的可执行文件, 相关的源文件是 *SRC_LIST* 中定义的源文件列表, 本例中你也可以直接写成 *ADD_EXECUTABLE*(hello main.c)。

初试 cmake

基本语法规则

1. 变量使用 ${} 方式取值, 但是在 IF 控制语句中是直接使用变量名。
2. 指令 (参数 1 参数 2...)
3. 指令是大小写无关的, 参数和变量是大小写相关的。但, 推荐你全部使用大写指令。

Cmake

More better

需求

1. 为工程添加一个子目录 src, 用来放置工程源代码;
2. 添加一个子目录 doc, 用来放置这个工程的文档 hello.txt;
3. 在工程目录添加文本文件 COPYRIGHT, README;
4. 在工程目录添加一个 runhello.sh 脚本, 用来调用 hello 二进制
5. 将构建后的目标文件放入构建目录的 bin 子目录;
6. 将 hpp 文件全部放在 lib 文件夹。

More better

Solve

```
cd ~/PAPERS/project1
mkdir src
cd src
touch CMakeLists.txt
```

project1/CMakeLists 应该修改为如下

```
PROJECT(HELLO)
ADD_SUBDIRECTORY(src bin)
```

More better

Solve

project1/src/CMakeLists 应该如下

```
# for Q6
INCLUDE_DIRECTORIES(${CMAKE_SOURCE_DIR}/lib)
# for Q5
SET(EXECUTABLE_OUTPUT_PATH  ${CMAKE_SOURCE_DIR}/bin)

add_executable(hello hello.cpp)
#target.cpp 需要用到新写的一个库
# Student.hpp(#include "Student.hpp")
add_executable(target target.cpp)
```

More better

Solve

project1/lib/CMakeLists 应该如下

```
SET(LIBRARY_OUTPUT_PATH ${CMAKE_SOURCE_DIR}/lib)
SET(LIBSTUDENT_SRC Sales_item.h)
ADD_LIBRARY(Student SHARED ${LIBSTUDENT_SRC})
```

ALL DONE!

Cmake

More

THANKS
其实还有挺多没有讲到，不过这就是一个基本的 CMake 工程的构造，里面涉及到的路径没有细说，不过网上都有详细的文档说明。还有就是关于动态库和静态库的部分没有说。虽说都是细节，但是用到的时候还是需要自己查找的。