chapter1.6 cmake设置编译方式

1. 文件构成：

```
unix
1  .
2  ├── CMakeLists.txt
3  ├── build
4  └── main.cpp
```

2. 文件填充：

2.1 main.cpp

```
#include <iostream>

int main(int argc, char *argv[])
{
   std::cout << "Hello Compile Flags!" << std::endl;

   // only print if compile flag set
#ifdef EX2
  std::cout << "Hello Compile Flag EX2!" << std::endl;
#endif

#ifdef EX3
  std::cout << "Hello Compile Flag EX3!" << std::endl;
#endif

   return 0;
}
```

2.2 CMakeLists.txt

```
cmake_minimum_required(VERSION 3.5)
#强制设置默认C++编译标志变量为缓存变量，如CMake（五） build type所说，该缓存变量被定义在文件中，相当于全局变量，源文件中也可以使用这个变量
set (CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -DEX2" CACHE STRING "Set C++ Compiler Flags" FORCE)

project (compile_flags)

add_executable(cmake_examples_compile_flags main.cpp)
#为可执行文件添加私有编译定义
target_compile_definitions(cmake_examples_compile_flags
    PRIVATE EX3
)
#命令的具体解释在二  CMake解析中，这里的注释只说明注释后每一句的作用
```

3. 文件解析：

3.1 设置每个目标的编译标志(recommended !)

1. 在现代CMake中设置C++标志的推荐方法是：

> 专门针对某个目标（target）设置标志，可以通过target_compile_definitions（）函数设置某个目标的编译标志。

```
target_compile_definitions(cmake_examples_compile_flags
    PRIVATE EX3
)
```

如果目标是一个库（cmake_examples_compile_flags），编译器在编译目标时添加定义-DEX3，并且选择了范围PUBLIC或INTERFACE，该定义-DEX3也将包含在链接此目标（cmake_examples_compile_flags）的所有可执行文件中。
注意此处！本语句使用了PRIVATE，所以编译选项不会传递。

2. 对于编译器选项，还可以使用target_compile_options（）函数：

```
target_compile_options(<target> [BEFORE]
  <INTERFACE|PUBLIC|PRIVATE> [items1...]
  [<INTERFACE|PUBLIC|PRIVATE> [items2...] ...])
```

这是给 `target` 添加编译选项：
[1] `target` 指的是由 `add_executable()` 产生的可执行文件或 `add_library()` 添加进来的库。
[2] `<INTERFACE|PUBLIC|PRIVATE>` 指的是 `[items...]` 选项可以传播的范围， `PUBLIC and INTERFACE` 会传播 `<target>` 的 INTERFACE_COMPILE_DEFINITIONS 属性， `PRIVATE and PUBLIC` 会传播 `target` 的 COMPILE_DEFINITIONS 属性。

3.2 设置默认编译标志

默认的CMAKE_CXX_FLAGS为空或包含适用于构建类型的标志。 要设置其他默认编译标志，如下使用：

```
set (CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -DEX2" CACHE STRING "Set C++ Compiler Flags" FORCE)
```

- CACHE: 强制设置默认C++编译标志变量为缓存变量，如CMake（五） build type所说，该缓存变量被定义在文件中，相当于全局变量，源文件中也可以使用这个变量。这个变量原本包含的参数仍然存在，只是添加了EX2。
- `CACHE STRING "Set C++ Compiler Flags" FORCE` 命令是为了强制将CMAKE_CXX_FLAGS变量 放到CMakeCache.txt文件中
- `"${CMAKE_CXX_FLAGS} -DEX2"` 这个字符串可以保留原有的CMAKE_CXX_FLAGS中的参数，额外添加了一个EX2参数。注意写法：空格，并且参数前加了 `-D`

类似设置CMAKE_CXX_FLAGS，还可以设置其他选项：

- 设置C编译标志： CMAKE_C_FLAGS
- 设置链接标志：CMAKE_LINKER_FLAGS.

3.3 设置CMake标志

与构建类型类似，可以使用以下方法设置全局C 编译器标志。
在cmake命令行中：

```
cmake .. -DCMAKE_CXX_FLAGS="-DEX3"
```

3.4 区别

- 3.2方法的设置CMAKE_C_FLAGS和CMAKE_CXX_FLAGS将为该目录或所有包含的子目录中的所有目标全局设置一个编译器标志。 现在不建议使用该方法，首选使用target_compile_definitions函数。
- 3.1方法是被建议的，只为这个目标设置编译选项 。
- 3.3设置的也是全局编译器选项。

4. 总览：

```
huluobo@huluobodeMacBook-Pro ▶ ~/cmake-examples/myCmake/chapter1.6 ▶ ᛘ main ± ▶ cd build
 huluobo@huluobodeMacBook-Pro ▶ ~/cmake-examples/myCmake/chapter1.6/build ▶ ᛘ main ± ▶ cmake ..
-- The C compiler identification is AppleClang 15.0.0.15000040
```

```
-- The CXX compiler identification is AppleClang 15.0.0.15000040
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: /Library/Developer/CommandLineTools/usr/bin/cc - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: /Library/Developer/CommandLineTools/usr/bin/c++ - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done (0.4s)
-- Generating done (0.0s)
-- Build files have been written to: /Users/huluobo/cmake-examples/myCmake/chapter1.6/build
 huluobo@huluobodeMacBook-Pro ▶ ~/cmake-examples/myCmake/chapter1.6/build ▶ ᛘ main ± ▶ make VERBOSE=1
/Applications/CMake.app/Contents/bin/cmake -S/Users/huluobo/cmake-examples/myCmake/chapter1.6 -
B/Users/huluobo/cmake-examples/myCmake/chapter1.6/build --check-build-system CMakeFiles/Makefile.cmake 0
/Applications/CMake.app/Contents/bin/cmake -E cmake_progress_start /Users/huluobo/cmake-
examples/myCmake/chapter1.6/build/CMakeFiles /Users/huluobo/cmake-
examples/myCmake/chapter1.6/build//CMakeFiles/progress.marks
/Library/Developer/CommandLineTools/usr/bin/make  -f CMakeFiles/Makefile2 all
/Library/Developer/CommandLineTools/usr/bin/make  -f CMakeFiles/cmake_examples_compile_flags.dir/build.make
CMakeFiles/cmake_examples_compile_flags.dir/depend
cd /Users/huluobo/cmake-examples/myCmake/chapter1.6/build && /Applications/CMake.app/Contents/bin/cmake -E
cmake_depends "Unix Makefiles" /Users/huluobo/cmake-examples/myCmake/chapter1.6 /Users/huluobo/cmake-
examples/myCmake/chapter1.6 /Users/huluobo/cmake-examples/myCmake/chapter1.6/build /Users/huluobo/cmake-
examples/myCmake/chapter1.6/build /Users/huluobo/cmake-
examples/myCmake/chapter1.6/build/CMakeFiles/cmake_examples_compile_flags.dir/DependInfo.cmake "--color="
/Library/Developer/CommandLineTools/usr/bin/make  -f CMakeFiles/cmake_examples_compile_flags.dir/build.make
CMakeFiles/cmake_examples_compile_flags.dir/build
[ 50%] Building CXX object CMakeFiles/cmake_examples_compile_flags.dir/main.cpp.o
/Library/Developer/CommandLineTools/usr/bin/c++ -DEX3  -DEX2 -arch arm64 -isysroot
/Library/Developer/CommandLineTools/SDKs/MacOSX14.0.sdk -MD -MT
CMakeFiles/cmake_examples_compile_flags.dir/main.cpp.o -MF
CMakeFiles/cmake_examples_compile_flags.dir/main.cpp.o.d -o
CMakeFiles/cmake_examples_compile_flags.dir/main.cpp.o -c /Users/huluobo/cmake-
examples/myCmake/chapter1.6/main.cpp
[100%] Linking CXX executable cmake_examples_compile_flags
/Applications/CMake.app/Contents/bin/cmake -E cmake_link_script
CMakeFiles/cmake_examples_compile_flags.dir/link.txt --verbose=1
/Library/Developer/CommandLineTools/usr/bin/c++  -DEX2 -arch arm64 -isysroot
/Library/Developer/CommandLineTools/SDKs/MacOSX14.0.sdk -Wl,-search_paths_first -Wl,-
headerpad_max_install_names CMakeFiles/cmake_examples_compile_flags.dir/main.cpp.o -o
cmake_examples_compile_flags
[100%] Built target cmake_examples_compile_flags
/Applications/CMake.app/Contents/bin/cmake -E cmake_progress_start /Users/huluobo/cmake-
examples/myCmake/chapter1.6/build/CMakeFiles 0
 huluobo@huluobodeMacBook-Pro ▶ ~/cmake-examples/myCmake/chapter1.6/build ▶ ᛘ main ± ▶
```