03 构建和链接静态库和动态库

构建和链接静态库和动态链接库

准备工作

创建message类和main.cpp文件：

Message.h 文件：

```
#pragma once
#include <iosfwd>
#include <string>
class Message {
public:
    Message(const std::string &m):m_strMessage(m){}
    friend std::ostream &operator<<(std::ostream &os, Message &obj) {
        return obj.printObject(os);
    }
private:
    std::string m_strMessage;
    std::ostream& printObject(std::ostream &os);
};
```

Message.cpp 文件：

```
#include "Message.h"
#include <iostream>
```

```
#include <string>

std::ostream& Message::printObject(std::ostream& os) {

    os << "This is my very nice message: " << std::endl;

    os << m_strMessage;

    return os;

}
```

main.cpp 文件：

```
#include "Message.h"

#include <cstdlib>

#include <iostream>

int main() {

    Message say_hello("Hello, CMake World!");

    std::cout << say_hello << std::endl;

    Message say_goodbye("Goodbye, CMake World");

    std::cout << say_goodbye << std::endl;

    return EXIT_SUCCESS;

}
```

具体实施
在根目录下创建CMakeLists.txt文件：
CMakeLists.txt：

```
 # 1.要求最低cmake版本是3.5，如果低于3.5就报错

cmake_minimum_required(

    VERSION

    3.5

    FATAL_ERROR

)

# 2.声明项目名称为hello-world，使用的编程语言是C++

project(
```

```
    hello-world
    LANGUAGES CXX
)
# 3.创建一个静态库目标：  使用的源文件Message类来创建message静态库
add_library(
    message
    STATIC
    Message.h
    Message.cpp
)
# 4.创建可执行文件目标：通过编译和链接源文件main.cpp来创建目标文件hello-world
add_executable(
    hello-world
    main.cpp
)
# 5.将message库链接到可执行文件hello-world
target_link_libraries(
    hello-world
    message
)
```

对项目进行配置和构建

```
 1.打开终端cd到项目所在的根目录。
 2.输入：cmake -H. -Bbuild 并执行。
 3.输入：cd ./build并执行。
 4.输入：cmake --build .并执行。
 5.在build目录中即可看到生成的可执行文件hello-world
 6.输入：./hello-world并执行即可在终端看到程序处理的结果。
```