论多文件的编译运行及 Google Test

本文档帮助各位从 单文件编译 运行 过渡到 多文件编译运行
我们的剧情设定如下：
Date.h Date.cpp User.h User.cpp Meeting.h Meeting.cpp Main.cpp
其中依赖关系是
Meeting -> Date
Main -> Meeting
Main -> User
我们想要的可执行文件名称为 Main
下面我们开始来讲解，如何编译出可执行文件 Main
首先是 g++ 的 -c 参数

```
g++ -c Date.cpp
```

这里，会生成 .o 文件，我们在编译的时候，不需要写进 .h 文件

为了编译出 Main，我们可以尝试这样做

```
g++ -o Main Date.cpp User.cpp Meeting.cpp Main.cpp
```

这里，cpp 文件的顺序没有关系，-o 选项 后面必须带一个可执行文件的名字，这里我们取名为 Main
这是多文件编译的一种方法，即将所有的 cpp 都放在一起

还有一种是，我们可以先编译出 .o 文件
比如，我们按照以下命令来编译

```
g++ -c Date.cpp
g++ -c User.cpp
g++ -c Meeting.cpp
```

这时我们会得到 Date.o User.o Meeting.o 文件，接下来，我们运行

```
g++ -o Main Date.o User.o Meeting.o Main.cpp
```

将我们所要编译的 Main.cpp 与 .o 文件 link 在一起。

- 为什么我们要这么做？

Makefile 的时候我们就可以用到，具体请参照参考资料中的 Makefile 编写，之后再继续往下看。

下面我们展示一个简单的 Google Test 与 Makefile 的结合。
剧情设定：假设我们要测试 Date 类中的 isValid 函数。假设当前目录下，有

```
Date.h Date.cpp DateTest.cpp Makefile Main.cpp
```

Date.h 与 Date.cpp 我们已经编写完毕，下面看看简单的 DateTest.cpp 编写

```
#include <gtest/gtest.h>
#include "Date.h"

TEST(DateTest, isValid) {
  Date date1(2013, 2, 30, 1, 1);
  EXPECT_FALSE(Date::isValid(date1));
  Date date2(2012, 2, 20, 1, 1);
  EXPECT_TRUE(Date::isValid(date2));
}
```

下面是 Main.cpp

```
#include <gtest/gtest.h>

int main(int argc, char** argv) {
   testing::InitGoogleTest(&argc, argv);
   // Runs all tests using Google Test.
   return RUN_ALL_TESTS();
}
```

现在我们来编译这里这个 Google Test
首先，我们先编译出 Date.o

```
g++ -c Date.cpp
```

我们继续编译出最终的可执行文件，假设我们想要的 可执行文件名为 AllTest.out

```
g++ -o AllTest.out Date.o DateTest.cpp Main.cpp -lgtest -lpthread
```

还记得 -o 的后面要跟 可执行文件的名字？
最后我们运行：

```
./AllTest.out
```

当然你要注意相对路径的正确性。

接下来，按照上面的剧情，我们写一个 Makefile

```
All: Date.o DateTest.cpp Main.cpp
[TAB]g++ -o AllTest.out Date.o DateTest.cpp Main.cpp -lgtest -lpthread
Date.o: Date.h Date.cpp
[TAB]g++ -c Date.cpp
```

以后，当我们修改了 DateTest.cpp 或者 Date 类，我们只需要 make 一下，就得到了可执行文件。