

# 紫冰枫

博客园 首页 新随笔 联系 订阅 管理

随笔 - 1 文章 - 0 评论 - 0

## 公告

昵称：紫冰枫  
园龄：5年3个月  
粉丝：0  
关注：1  
[+加关注](#)

<	2018年10月						>
日	一	二	三	四	五	六	
30	1	2	3	4	5	6	
7	8	9	10	11	12	13	
14	15	16	17	18	19	20	
21	22	23	24	25	26	27	
28	29	30	31	1	2	3	
4	5	6	7	8	9	10	

## 搜索

## Qt使用gtest

近日需要使用Qt进行一个工具的编写, 恰逢最近在学习使用gtest作为测试的首选, 本着熟能生巧, 能有机会多练习就多练习, 无机会创造机会也要上(/(T o T)/~~)的积极(折腾作死)态度, 开展了Qt使用gtest的折腾.

首先, 说明一下环境: win7+qt5.8.0(MinGW), gtest最新版本, 直接从GitHub拉取(<https://github.com/google/googletest>)

其次, 要使用gtest, 先要使用对应的编译器编译, 这里由于使用的是MinGW版本的Qt, 所以gtest也要使用MinGW进行编译, 不然编译器无法正常连接gtest使用. 下面是关于Qt编译gtest的步骤:

1. 在拉取下来的googletest目录下建立一个gtest.pro(Qt project文件)[特别注明: 当前gtest将gmock集成到gtest工程下了, 所以gtest的目录下是有googletest目录的], 如下图:

谷歌搜索

## 常用链接

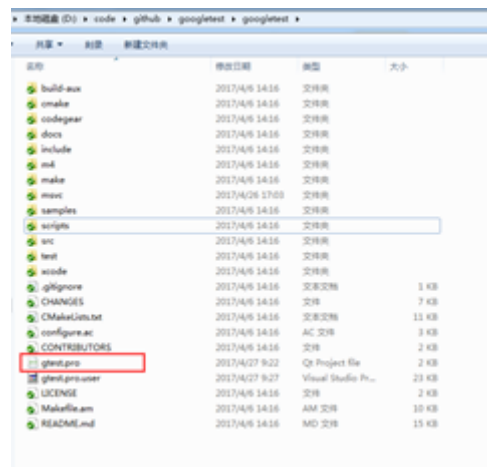
[我的随笔](#)[我的评论](#)[我的参与](#)[最新评论](#)[我的标签](#)

## 随笔档案

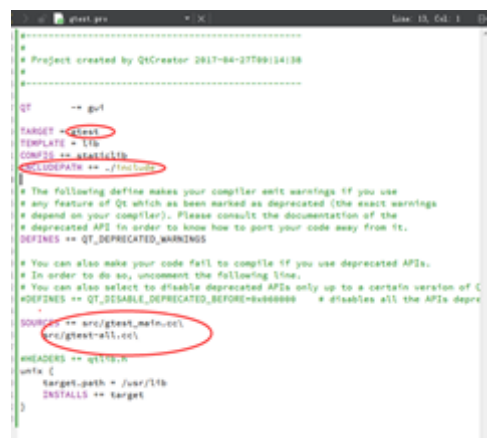
2017年4月 (1)

## 阅读排行榜

1. Qt使用gtest(470)

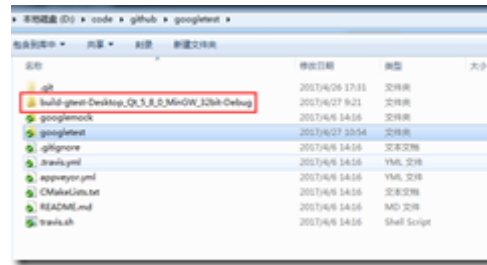


2. 编辑gtest.pro文件, 曾参考过网上一篇blog(这里)进行编辑, 但是编译生成出来的是两个.o文件和一个.exe文件, 并不是MinGW编译出来的静态库.a文件. 后来的做法, 是先使用Qt新建一个静态库的工程, 然后拷贝这个新建的静态库工程的pro的内容到gtest.pro, 然后再按需更改, gtest.pro的更改后的内容如下图:



图中圈出来的内容为需要更改的部分

3. 使用Qt打开gtest.pro工程, 构建之, Qt会在.pro的上一级目录下生成对应的编译目录和输出目录(基于shadow build), 如下图:



图中可以看到gtest工程下已经集成了gmock工程

4.在输出目录下, 可以看到MinGW编译出的gtest库文件libgtest.a

编译得到想要的gtest库后, 开始使用在Qt环境下使用之.

1.使用Qt新建一个console的验证工程gtestforqt

2.编辑gtestforqt.pro文件, 使其可以连接到我们编译的gtest库文件, 如下图:

```
QT += gtest
CONFIG += c++11

INCLUDEPATH += D:\code\work_test\for_qt\gtestforqt
win32:LIBS += -LD:\code\work_test\for_qt\gtestforqt\lib\Debug\ -l\lgtest
INCLUDEPATH += ..\gtestforqt\gtest
win32:LIBS += -L..\gtestforqt\lib\Debug\ -l\lgtest

TARGET = gtestforqt
CONFIG += console
```

INCLUDEPATH 是增加对gtest头文件的链接路径

LIBS 是增加对gtest库文件的链接路径

注释部分, 是使用绝对路径, 不建议使用, 对各个机子的环境依赖太强.

建议使用下面的相对路径(注意, 要包含pro文件在的当前文件夹在内..`..\gtestforqt\lib\Debug\`,而不能直接使用..`lib\Debug\`)

3.include gtest的文件件,并初始化gtest, 如下图:

```

#include <QCoreApplication>
#include "gtest/gtest.h"

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);
    testing::InitGoogleTest(&argc, argv);
    return RUN_ALL_TESTS();
}

```

4. 构建运行后, gtest成功执行:

```

C:\Qt\Qt5.8.0\Tools\QtCreator\bin\qtcreator_process_stub.exe
Running 0 tests from 0 test cases.
0 tests from 0 test cases ran. (0 ms total)
PASSED
0 tests.

```

5. 增加一个TEST并运行, 如下图

```

#include <QCoreApplication>
#include "gtest/gtest.h"

// Test add function
TEST(add, test_1)
{
    return 0;
}

// Test add function
TEST(add, test_2)
{
    int a = 1;
    EXPECT_EQ(add(a, 1), 2);
}

// Test main function
int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);
    testing::InitGoogleTest(&argc, argv);
    return RUN_ALL_TESTS();
}

```

```

C:\Qt\Qt5.8.0\Tools\QtCreator\bin\qtcreator_process_stub.exe
Running 2 tests from 1 test case.
Global test environment tear-down
1 test from environment and main
1 test from addTest
1 test from addTest (0 ms total)
PASSED
Global test environment tear-down
2 test from 1 test case ran. (0 ms total)
2 tests.

```

至此, Qt编译gtest并使用gtest的工作已经折腾完成, 以此记录, 以便后面自己查看和其他人查看(网上这方面的资料不多..或者是我没查到 = .!=)

好文要顶

关注我

收藏该文



紫冰枫  
关注 - 1  
粉丝 - 0

+加关注

0

0

posted @ 2017-04-27 11:51 紫冰枫 阅读(470) 评论(0) 编辑 收藏