

## 公告

DigitalOcean优惠码

这里的博客将不再更新, 最新博客  
请移步至:

我的独立博客:

<http://blog.coderzh.com/>



微信公众号:

hacker-thinking

昵称: CoderZh

园龄: 11年3个月

粉丝: 801

关注: 10

+加关注

## 搜索

 找找看

## 随笔分类

Agile(2)

Android(3)

ASP.NET(3)

C#(20)

C/C++(24)

Cocos2d-x(1)

Emacs(2)

Google App Engine(7)

JAVA(3)

Linux(1)

Lua(2)

Python(66)

Ubuntu(9)

VBS(4)

随笔-234 文章-10 评论-2052

## 玩转Google开源C++单元测试框架Google Test系列(gtest)之二 - 断言

### 一、前言

这篇文章主要总结gtest中的所有断言相关的宏。gtest中, 断言的宏可以理解分为两类, 一类是ASSERT系列, 一类是EXPECT系列。一个直观的解释就是:

1. ASSERT\_\* 系列的断言, 当检查点失败时, 退出当前函数 (注意: 并非退出当前案例)。
2. EXPECT\_\* 系列的断言, 当检查点失败时, 继续往下执行。

### 二、示例

```
// int型比较, 预期值: 3, 实际值: Add(1, 2)
EXPECT_EQ(3, Add(1, 2))
// ...
```

假如你的Add(1, 2) 结果为4的话, 会在结果中输出:

```
g:\myproject\c++\gtestdemo\gtestdemo\gtestdemo.cpp(16): error: Value of: Add(1, 2)
      Actual: 4
      Expected: 3
```

如果是将结果输出到xml里的话, 将输出: (关于将结果输出为xml, 见: <http://www.cnblogs.com/coderzh/archive/2009/04/10/1432789.html>)

```
<?xml version="1.0" encoding="UTF-8" ?>
<testcase name="Demo" status="run" time="0" classname="AddTest">
  <failure message="Value of: Add(1, 2) Actual: 4 Expected: 3" type=""><![CDATA[g:\myproject\c++\gtestdemo\gtestdemo\gtestdemo.cpp:16
Value of: Add(1, 2)
      Actual: 4
      Expected: 3]]></failure>
</testcase>
```

如果你对自动输出的出错信息不满意的话, 你还可以通过操作符<<将一些自定义的信息输出, 通常, 这对于调试或是对一些检查点的补充说明来说, 非常有用!

- 安全性测试(9)
- 测试生活感悟(7)
- 程序人生(15)
- 代码安全(3)
- 单元测试(19)
- 公告(13)
- 每周总结(4)
- 软件测试(30)
- 设计模式
- 性能测试(7)
- 学习笔记(27)

随笔档案

- 2015年9月 (1)
- 2015年8月 (2)
- 2015年6月 (4)
- 2015年5月 (2)
- 2015年4月 (5)
- 2015年3月 (1)
- 2014年5月 (2)
- 2014年4月 (2)
- 2011年5月 (1)
- 2011年3月 (1)
- 2011年1月 (1)
- 2010年12月 (3)
- 2010年11月 (3)
- 2010年10月 (2)
- 2010年9月 (6)
- 2010年8月 (2)
- 2010年7月 (4)
- 2010年6月 (3)
- 2010年5月 (4)
- 2010年4月 (9)
- 2010年3月 (6)
- 2010年2月 (3)
- 2010年1月 (16)
- 2009年12月 (6)
- 2009年11月 (3)
- 2009年10月 (4)
- 2009年9月 (3)
- 2009年8月 (2)

下面举个例子：

如果不使用<<操作符自定义输出的话：

```
for (int i = 0; i < x.size(); ++i)
{
    EXPECT_EQ(x[i], y[i]);
}
```

看到的结果将是这样的，你根本不知道出错时 i 等于几：

```
g:\myproject\c++\gtestdemo\gtestdemo\gtestdemo.cpp(25): error: Value of: y[i]
    Actual: 4
Expected: x[i]
Which is: 3
```

如果使用<<操作符将一些重要信息输出的话：

```
for (int i = 0; i < x.size(); ++i)
{
    EXPECT_EQ(x[i], y[i]) << "Vectors x and y differ at index " << i;
}
```

从输出结果中就可以定位到在 i = 2 时出现了错误。这样的输出结果看起来更加有用，容易理解：

```
g:\myproject\c++\gtestdemo\gtestdemo\gtestdemo.cpp(25): error: Value of: y[i]
    Actual: 4
Expected: x[i]
Which is: 3
Vectors x and y differ at index 2
```

三、布尔值检查

Fatal assertion	Nonfatal assertion	Verifies
ASSERT_TRUE( <i>condition</i> );	EXPECT_TRUE( <i>condition</i> );	<i>condition</i> is true
ASSERT_FALSE( <i>condition</i> );	EXPECT_FALSE( <i>condition</i> );	<i>condition</i> is false

2009年7月 (7)  
2009年6月 (2)  
2009年4月 (12)  
2009年3月 (5)  
2009年2月 (2)  
2009年1月 (3)  
2008年12月 (7)  
2008年11月 (9)  
2008年9月 (8)  
2008年8月 (7)  
2008年7月 (8)  
2008年6月 (9)  
2008年5月 (33)  
2008年4月 (6)  
2008年2月 (1)  
2007年12月 (3)  
2007年11月 (3)  
2007年10月 (7)  
2007年9月 (1)

系列文章

Python天天美味系列  
攻击方式学习系列  
瘦客户端那些事  
玩转gtest系列

读书笔记

Python网络编程  
xUnit Test Patterns  
卓有成效的程序员

友情链接

积分与排名

积分 - 547881  
排名 - 209

最新评论

1. Re:Google Test交流与测试开发  
经验总结  
您好，可以发一份给我吗，  
1034828302@qq.com

四、数值型数据检查

Fatal assertion	Nonfatal assertion	Verifies
<code>ASSERT_EQ(<i>expected</i>, <i>actual</i>) ;</code>	<code>EXPECT_EQ(<i>expected</i>, <i>actual</i>) ;</code>	<code><i>expected</i> == <i>actual</i></code>
<code>ASSERT_NE(<i>val1</i>, <i>val2</i>) ;</code>	<code>EXPECT_NE(<i>val1</i>, <i>val2</i>) ;</code>	<code><i>val1</i> != <i>val2</i></code>
<code>ASSERT_LT(<i>val1</i>, <i>val2</i>) ;</code>	<code>EXPECT_LT(<i>val1</i>, <i>val2</i>) ;</code>	<code><i>val1</i> &lt; <i>val2</i></code>
<code>ASSERT_LE(<i>val1</i>, <i>val2</i>) ;</code>	<code>EXPECT_LE(<i>val1</i>, <i>val2</i>) ;</code>	<code><i>val1</i> &lt;= <i>val2</i></code>
<code>ASSERT_GT(<i>val1</i>, <i>val2</i>) ;</code>	<code>EXPECT_GT(<i>val1</i>, <i>val2</i>) ;</code>	<code><i>val1</i> &gt; <i>val2</i></code>
<code>ASSERT_GE(<i>val1</i>, <i>val2</i>) ;</code>	<code>EXPECT_GE(<i>val1</i>, <i>val2</i>) ;</code>	<code><i>val1</i> &gt;= <i>val2</i></code>

五、字符串检查

Fatal assertion	Nonfatal assertion	Verifies
<code>ASSERT_STREQ(<i>expected_str</i>, <i>actual_str</i>) ;</code>	<code>EXPECT_STREQ(<i>expected_str</i>, <i>actual_str</i>) ;</code>	the two C strings have the same content
<code>ASSERT_STRNE(<i>str1</i>, <i>str2</i>) ;</code>	<code>EXPECT_STRNE(<i>str1</i>, <i>str2</i>) ;</code>	the two C strings have different content
<code>ASSERT_STRCASEEQ(<i>expected_str</i>, <i>actual_str</i>) ;</code>	<code>EXPECT_STRCASEEQ(<i>expected_str</i>, <i>actual_str</i>) ;</code>	the two C strings have the same content, ignoring case
<code>ASSERT_STRCASENE(<i>str1</i>, <i>str2</i>) ;</code>	<code>EXPECT_STRCASENE(<i>str1</i>, <i>str2</i>) ;</code>	the two C strings have different content, ignoring case

\*STREQ\*和\*STRNE\*同时支持char\*和wchar\_t\*类型的，\*STRCASEEQ\*和\*STRCASENE\*却只接收char\*，估计是不常用吧。下面是几个例子：



```
TEST(StringCmpTest, Demo)
{
    char* pszCoderZh = "CoderZh";
    wchar_t* wszCoderZh = L"CoderZh";
    std::string strCoderZh = "CoderZh";
    std::wstring wstrCoderZh = L"CoderZh";

    EXPECT_STREQ("CoderZh", pszCoderZh);
    EXPECT_STREQ(L"CoderZh", wszCoderZh);
```

谢谢

--xuxd88

2. Re:玩转Google开源C++单元测试框架Google Test系列(gtest)(总楼主好厉害，写的通俗易懂，我等晚辈拜服！

--One\_IT\_One\_World

3. Re:Google Test交流与测试开发经验总结

您好，一直在看您的文章，写的非常不错，我最近也刚开始写Gtest，能否方便给我发一份PPT呢？

邮箱：1206050824@qq.com

谢谢您啦！

--One\_IT\_One\_World

4. Re:玩转Google开源C++单元测试框架Google Test系列(gtest)(总google test

--Big~Newbie

5. Re:PyQt4学习资料汇总

正在学习，邮箱：1464983982@qq.com 感谢分享

--jobs\_huang

6. Re:gtest参数化测试代码示例

博客园的链接改了，是这个地址：

--canbeing

7. Re:玩转Google开源C++单元测试框架Google Test系列(gtest)之一 - 初识gtest

@xiao\_1bai编译的目标就是生成lib文件，你已经成功了。现在可以在你的项目引用gtestd.lib...

--cnbloghzc

8. Re:ViEmuVS2013-3.2.1 破解安装失败，提示：所需要的.NET Framework 没有

--xiake007

9. Re:玩转Google开源C++单元测试框架Google Test系列(gtest)之七 - 深入解析gtest

```
EXPECT_STRNE("CnBlogs", pszCoderZh);
EXPECT_STRNE(L"CnBlogs", wszCoderZh);

EXPECT_STRCASEEQ("coderzh", pszCoderZh);
//EXPECT_STRCASEEQ(L"coderzh", wszCoderZh); 不支持

EXPECT_STREQ("CoderZh", strCoderZh.c_str());
EXPECT_STREQ(L"CoderZh", wstrCoderZh.c_str());
}
```

六、显示返回成功或失败

直接返回成功：SUCCEED();

返回失败：

Fatal assertion	Nonfatal assertion
FAIL();	ADD_FAILURE();

```
TEST(ExplicitTest, Demo)
{
    ADD_FAILURE() << "Sorry"; // None Fatal Asserton, 继续往下执行。

    //FAIL(); // Fatal Assertion, 不往下执行该案例。

    SUCCEED();
}
```

七、异常检查

Fatal assertion	Nonfatal assertion	Verifies
ASSERT_THROW(statement, exception_type);	EXPECT_THROW(statement, exception_type);	statement throws an exception of the given type
ASSERT_ANY_THROW(statement);	EXPECT_ANY_THROW(statement);	statement throws an exception of any type
ASSERT_NO_THROW(statement);	EXPECT_NO_THROW(statement);	statement doesn't throw any exception

谢谢，作者哥哥，这么多章，最精彩这章。领教了，谢谢

--\$JackChen

10. Re:最常用的Emacs的基本操作

如果Emacs入手都算有些难度。。。那VIM怎么办？

--震灵


阅读排行榜

- 1. 玩转Google开源C++单元测试框架Google Test系列(gtest)(总)(224598)
- 2. 玩转Google开源C++单元测试框架Google Test系列(gtest)之一 - 初识gtest(148792)
- 3. 玩转Google开源C++单元测试框架Google Test系列(gtest)之二 - 断言(107271)
- 4. 玩转Google开源C++单元测试框架Google Test系列(gtest)之三 - 事件机制(69736)
- 5. 玩转Google开源C++单元测试框架Google Test系列(gtest)之六 - 运行参数(55607)
- 6. 玩转Google开源C++单元测试框架Google Test系列(gtest)之四 - 参数化(55280)
- 7. C# 中使用JSON - DataContractJsonSerializer(54221)
- 8. 玩转Google开源C++单元测试框架Google Test系列(gtest)之七 - 深入解析gtest(50702)
- 9. PyQt4学习资料汇总(49953)
- 10. 代码覆盖率浅谈(49521)

评论排行榜


- 1. PyQt4学习资料汇总(153)
- 2. 开源Granados介绍 - SSH连接远程Linux服务器(C#)(66)
- 3. (原创)攻击方式学习之(1) - 跨站脚本(Cross-Site Scripting) (49)

例如：



```
int Foo(int a, int b)
{
    if (a == 0 || b == 0)
    {
        throw "don't do that";
    }
    int c = a % b;
    if (c == 0)
        return b;
    return Foo(b, c);
}

TEST(FooTest, HandleZeroInput)
{
    EXPECT_ANY_THROW(Foo(10, 0));
    EXPECT_THROW(Foo(0, 5), char*);
}
```



八、 Predicate Assertions

在使用EXPECT\_TRUE或ASSERT\_TRUE时，有时希望能够输出更加详细的信息，比如检查一个函数的返回值TRUE还是FALSE时，希望能够输出传入的参数是什么，以便失败后好跟踪。因此提供了如下的断言：

Fatal assertion	Nonfatal assertion	Verifies
ASSERT_PRED1 ( <i>pred1, val1</i> ) ;	EXPECT_PRED1 ( <i>pred1, val1</i> ) ;	<i>pred1(val1)</i> returns true
ASSERT_PRED2 ( <i>pred2, val1, val2</i> ) ;	EXPECT_PRED2 ( <i>pred2, val1, val2</i> ) ;	<i>pred2(val1, val2)</i> returns true
...	...	...

Google人说了，他们只提供<=5个参数的，如果需要测试更多的参数，直接告诉他们。下面看看这个东西怎么用。



```
bool MutuallyPrime(int m, int n)
{
    return Foo(m , n) > 1;
}
```

- 4. 三年之痒(44)
- 5. NancyBlog - 我的Google App Engine Blog(42)
- 6. CCNET+MSBuild+SVN实时构建的优化总结(40)
- 7. 创业三年来的一些感想 - 游戏篇(40)
- 8. CoderZh首款Python联机对战游戏 - NancyTetris1.0倾情发布 (一) (37)
- 9. 代码安全系列(1) - Log的注入(35)
- 10. 程序员的信仰(35)

推荐排行榜

- 1. 玩转Google开源C++单元测试框架Google Test系列(gtest)(总)(26)
- 2. 创业三年来的一些感想 - 游戏篇(14)
- 3. 程序员的共鸣 - 读《卓有成效的程序员》(12)
- 4. 代码覆盖率浅谈(12)
- 5. 《xUnit Test Patterns》学习笔记5 - xUnit基础(10)
- 6. 三年之痒(9)
- 7. 玩转Google开源C++单元测试框架Google Test系列(gtest)之一 - 初识gtest(9)
- 8. 优美的测试代码 - 行为驱动开发(BDD)(8)
- 9. Python天天美味(总)(7)
- 10. Google App Engine的14宗罪(6)

```
TEST(PredicateAssertionTest, Demo)
{
    int m = 5, n = 6;
    EXPECT_PRED2(MutuallyPrime, m, n);
}
```

当失败时，返回错误信息：

error: MutuallyPrime(m, n) evaluates to false, where  
m evaluates to 5  
n evaluates to 6

如果对这样的输出不满意的话，还可以自定义输出格式，通过如下：

Fatal assertion	Nonfatal assertion	Verifies
ASSERT_PRED_FORMAT1( <i>pred_format1</i> , <i>val1</i> );`	EXPECT_PRED_FORMAT1( <i>pred_format1</i> , <i>val1</i> );	<i>pred_format1(val1)</i> is successful
ASSERT_PRED_FORMAT2( <i>pred_format2</i> , <i>val1</i> , <i>val2</i> );	EXPECT_PRED_FORMAT2( <i>pred_format2</i> , <i>val1</i> , <i>val2</i> );	<i>pred_format2(val1, val2)</i> is successful
...	...	

用法示例：

```
testing::AssertionResult AssertFoo(const char* m_expr, const char* n_expr, const char* k_expr, int m, int n, int k) {
    if (Foo(m, n) == k)
        return testing::AssertionSuccess();
    testing::Message msg;
    msg << m_expr << " 和 " << n_expr << " 的最大公约数应该是: " << Foo(m, n) << " 而不是: " << k_expr;
    return testing::AssertionFailure(msg);
}

TEST(AssertFooTest, HandleFail)
{
    EXPECT_PRED_FORMAT3(AssertFoo, 3, 6, 2);
}
```

失败时，输出信息：

error: 3 和 6 的最大公约数应该是: 3 而不是: 2

是不是更温馨呢, 呵呵。

## 九、浮点型检查

Fatal assertion	Nonfatal assertion	Verifies
<code>ASSERT_FLOAT_EQ(<i>expected</i>, <i>actual</i>);</code>	<code>EXPECT_FLOAT_EQ(<i>expected</i>, <i>actual</i>);</code>	the two float values are almost equal
<code>ASSERT_DOUBLE_EQ(<i>expected</i>, <i>actual</i>);</code>	<code>EXPECT_DOUBLE_EQ(<i>expected</i>, <i>actual</i>);</code>	the two double values are almost equal

对相近的两个数比较:

Fatal assertion	Nonfatal assertion	Verifies
<code>ASSERT_NEAR(<i>val1</i>, <i>val2</i>, <i>abs_error</i>);</code>	<code>EXPECT_NEAR(<i>val1</i>, <i>val2</i>, <i>abs_error</i>);</code>	the difference between <i>val1</i> and <i>val2</i> doesn't exceed the given absolute error

同时, 还可以使用:

```
EXPECT_PRED_FORMAT2(testing::FloatLE, val1, val2);  
EXPECT_PRED_FORMAT2(testing::DoubleLE, val1, val2);
```

## 十、Windows HRESULT assertions


Fatal assertion	Nonfatal assertion	Verifies
<code>ASSERT_HRESULT_SUCCEEDED(<i>expression</i>);</code>	<code>EXPECT_HRESULT_SUCCEEDED(<i>expression</i>);</code>	<i>expression</i> is a success HRESULT
<code>ASSERT_HRESULT_FAILED(<i>expression</i>);</code>	<code>EXPECT_HRESULT_FAILED(<i>expression</i>);</code>	<i>expression</i> is a failure HRESULT

例如:

```
CComPtr shell;  
ASSERT_HRESULT_SUCCEEDED(shell.CoCreateInstance(L"Shell.Application"));  
CComVariant empty;  
ASSERT_HRESULT_SUCCEEDED(shell->ShellExecute(CComBSTR(url), empty, empty, empty, empty));
```


## 十一、类型检查

类型检查失败时，直接导致代码编不过，难得用处就在这？看下面的例子：



```
template <typename T> class FooType {
public:
    void Bar() { testing::StaticAssertTypeEq<int, T>(); }
};

TEST(TypeAssertionTest, Demo)
{
    FooType<bool> fooType;
    fooType.Bar();
}
```



## 十二、总结

本篇将常用的断言都介绍了一遍，内容比较多，有些还是很有用的。要真的到写案例的时候，也行只是一两种是最常用的，现在时知道有这么多种选择，以后才方便查询。

系列链接：

1. [玩转Google开源C++单元测试框架Google Test系列\(gtest\)之一 - 初识gtest](#)
2. [玩转Google开源C++单元测试框架Google Test系列\(gtest\)之二 - 断言](#)
3. [玩转Google开源C++单元测试框架Google Test系列\(gtest\)之三 - 事件机制](#)
4. [玩转Google开源C++单元测试框架Google Test系列\(gtest\)之四 - 参数化](#)
5. [玩转Google开源C++单元测试框架Google Test系列\(gtest\)之五 - 死亡测试](#)
6. [玩转Google开源C++单元测试框架Google Test系列\(gtest\)之六 - 运行参数](#)
7. [玩转Google开源C++单元测试框架Google Test系列\(gtest\)之七 - 深入解析gtest](#)
8. [玩转Google开源C++单元测试框架Google Test系列\(gtest\)之八 - 打造自己的单元测试框架](#)

[DigitalOcean](#)的VPS主机，稳定、速度快、价格也实惠。可以在上面部署独立网站或各种实用工具。

我用了很久了，确实不错，极力推荐。

使用这个链接购买可获得10美元优惠。





优惠链接: [DigitalOcean优惠码](#)



微信扫一扫交流

作者: [CoderZh](#)

公众号: hacker-thinking (一个程序员的思考)

独立博客: <http://blog.coderzh.com>

博客园博客将不再更新, 请关注我的「微信公众号」或「独立博客」。

作为一个程序员, 思考程序的每一行代码, 思考生活的每一个细节, 思考人生的每一种可能。

文章版权归本人所有, 欢迎转载, 但未经作者同意必须保留此段声明, 且在文章页面明显位置给出原文连接, 否则保留追究法律责任的权利。

分类: [单元测试, C/C++](#)

标签: [Google Test](#)

好文要顶

关注我

收藏该文



CoderZh

关注 - 10

粉丝 - 801

+加关注

4

0

« 上一篇: [玩转Google开源C++单元测试框架Google Test系列\(gtest\)之一 - 初识gtest](#)

» 下一篇: [玩转Google开源C++单元测试框架Google Test系列\(gtest\)之三 - 事件机制](#)

posted @ 2009-04-06 18:17 CoderZh 阅读(107271) 评论(9) 编辑 收藏

### 评论列表

#1楼 2010-06-20 16:52 纯一狼

>>\*STRCASEEQ\*和\*STRCASENE\*却只接收char\*, 估计是不常用吧

- 比如汉字等就不区分大小写的, 基于这个原因, 所以不提供针对wchar\_t的忽略大小写版本。

支持(0) 反对(0)

#2楼 2011-07-04 19:06 xhuren

最后一个测试用例编译出错 error C2514: "testing::internal::StaticAssertTypeEqHelper<T1,T2>" : 类没有构造函数

求大虾指教?

支持(0) 反对(0)

#3楼 2011-08-31 14:51 lyly0904

请问 如果想比较数组是否相等 或者比较内存是否相等 应该用什么宏?

#4楼 2011-08-31 14:52 lyly0904

我的邮箱lyly0904@163.com 谢谢

#5楼 2011-12-31 15:53 ~波

如果断言失败了，如何查看是哪行代码导致的失败？

支持(0) 反对(0)

#6楼 2012-12-29 10:10 dEMON\_hUNTER

写的很不错，能转载吗（不为别的，只是在自己需要的时候方便查看）

谢谢。

支持(0) 反对(0)

#7楼 2013-07-20 14:04 czjone

此框架是否支持 覆盖率 $\square$ 率的体现呢？

支持(0) 反对(0)

#8楼 2014-05-14 15:58 xincpp

SUCCEED宏是什么意思呢，没看明白

Generates a success. This does NOT make the overall test succeed. A test is considered successful only if none of its assertions fail during its execution.

支持(0) 反对(0)

#9楼 2015-09-29 19:04 mollywml

@ xhuren

就是报错才说明类型检查失败，如果把TEST里的bool换成int程序不会报错。  
这点博客里说了的：“直接导致代码编不过”

支持(0) 反对(0)

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

【推荐】50万行VC++源码: 大型组态工控、电力仿真CAD与GIS源码库

【调查】有奖调研即刻参与，你竟然是酱紫程序猿！

【推荐】Vue.js 2.x 快速入门，大量高效实战示例

【活动】腾讯云 学生专属优惠套餐 多规格选择

**最新IT新闻:**

- 飞一天不成问题? 微软测试无人驾驶AI滑翔机
  - 谷歌: 黑客攻击是用户凭证泄露最常见方式 但是网络钓鱼威胁最大
  - 阿里巴巴本月发行美元债券 最高融资70亿美元
  - 苏宁减持阿里巴巴股份: 预计不超过550万股
  - Mozilla推出速度更快的全新Firefox 57 (Quantum) 浏览器
- » 更多新闻...

**最新知识库文章:**

- 关于编程, 你的练习是不是有效的?
  - 改善程序员生活质量的 3+10 习惯
  - NASA的10条代码编写原则
  - 为什么你参加了那么多培训, 却依然表现平平?
  - 写给初学前端工程师的一封信
- » 更多知识库文章...