# ASSERT_TRUE() return type does not match function type in gtest



When I am using `ASSERT_TRUE()` provided in `Gtest` I am getting below error. `return type does not match function type` with an underline in `VS 2010.` .

### abc.h

```
#include "gtest\gtest.h"

class abc {
pubilc:
    bool fun();
    private:
    bool fun1();
};
```

### abc.c

```
bool abc::fun()
{
    ASSERT_TRUE(fun1()); // Getting error: return type does not match function type
}

bool abc::fun1()
{
    return true; // True or false depanding on operation
}
```

c++ visual-studio-2010 googletest

edited Dec 29 '15 at 1:50

Bill Lynch
**53.4k** 9 81 128

asked Sep 27 '12 at 9:07

Rasmi Ranjan Nayak
**3,225** 13 53 95

## 3 Answers

There is no `return` statement specified in `fun()` but it returns a `bool` . Add a `return false;` or `return true;` to `fun()` or change its return type to `void` :

```
void abc::fun()
{
    ASSERT_TRUE(fun1());
}
```

Based on My compiler complains that a constructor (or destructor) cannot return a value. What's going on? which states (verbatim):

> Due to a peculiarity of C++, in order to support the syntax for streaming messages to an ASSERT_*, e.g.

```
ASSERT_EQ(1, Foo()) << "blah blah" << foo;
```

> we had to give up using ASSERT* and FAIL* (but not EXPECT* and ADD_FAILURE*) in constructors and destructors. The workaround is to move the content of your

> constructor/destructor to a private void member function, or switch to EXPECT_*() if that works. This section in the user's guide explains it.

the `return` type must be `void` in functions that use `ASSERT_*()` macros.

edited Mar 4 '16 at 11:46
Marvin
368 2 17

answered Sep 27 '12 at 9:09
hmjd
93k 10 129 195

I have added `return true` or `false` to `fun` it is throwing the same `error` . But by making `void` it works fine. But I would like to `return` . Then how to do it. – Rasmi Ranjan Nayak Sep 27 '12 at 9:17

@RasmiRanjanNayak, If you remove the `ASSERT_TRUE(fun1());` and have just `return true;` does it compile? – hmjd Sep 27 '12 at 9:19

Yes it does compile fine with only `return true` – Rasmi Ranjan Nayak Sep 27 '12 at 9:22

@RasmiRanjanNayak, updated answer. – hmjd Sep 27 '12 at 9:35

I think you are right. My compiler is also throwing the same error as `cannot convert from 'void' to 'BOOL'` . So, must need to use `void` instead of `BOOL or etc` – Rasmi Ranjan Nayak Sep 27 '12 at 9:38



`ASSERT_TRUE` is a macro. When expanded it will contain a branch like:

```
if (fun1() == false) {
   return;
}
```

This is how `ASSERT_TRUE` does a hard stop on failure, but it also means that your method `bool abc::fun()` now has a `void` return exit path, in conflict with its signature.

Possible fixes include don't use hard stop asserts:

```
bool abc::fun(){
    bool result = fun1();
    EXPECT_TRUE(result); //No return in expansion
                         //No hard stop!
    return result;
}
```

or change your methods return type if not needed:

```
void abc::fun(){
    ASSERT_TRUE(fun1()); //Hard stop on failure
}
```

or return by reference:

```
void abc::fun(bool &outResult){
   outResult = fun1();  //return result by reference
   ASSERT_TRUE(result);
}
```

answered Feb 18 '15 at 11:21
Downward Facing God
155 1 9

1 2.5 yrs late but this is the correct explanation ! – Some Guy Sep 23 '16 at 21:25

The `fun` method has a `bool` return type so it still needs to `return` something.

`ASSERT_TRUE` is a macro which tests that something is true, it won't call return for you. In fact, you can have many `ASSERT_TRUE` in a row, and (providing they are all true) they will all execute one after another. Think of the `ASSERT_TRUE` macro as a function call, even though it's not technically.

This should work:

```
bool abc::fun()
{
   ASSERT_TRUE(fun1());
   return true;
}
```

answered Sep 27 '12 at 9:10
Adam M-W
1,617 6 31 58

I did the same but still it is throwing same `error` . – Rasmi Ranjan Nayak Sep 27 '12 at 9:19

2 I don't think this is correct; `ASSERT_TRUE` expands into code that returns when the condition is *false*. That's how it aborts the test. – Andrew Lazarus Nov 27 '12 at 4:38