



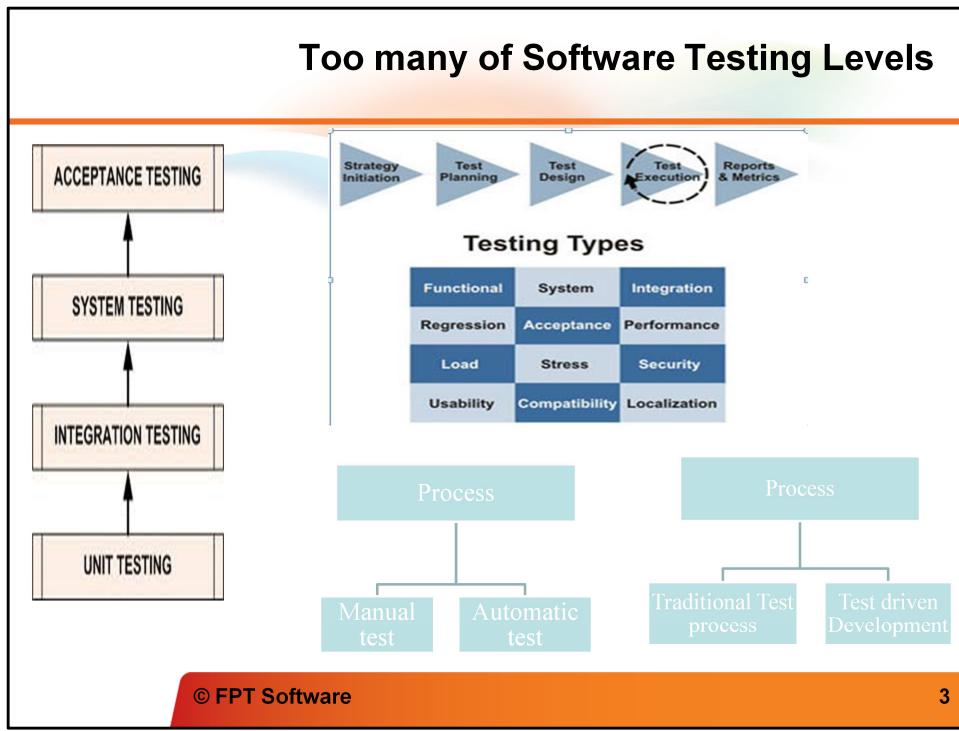
Automated Unit Testing with CPPUnit

Instructor: <Name of Instructor>

Latest updated by: HanhTT1

Agenda

- Software testing levels
- Manual unit testing
- Unit Testing based on UT cases
- Automated Unit Testing
- Automated Unit Testing with CPPUnit



Software testing view theo nhiều hướng:

- Theo chiều dọc
- Theo chiều ngang
- Theo quy trình tiến hành

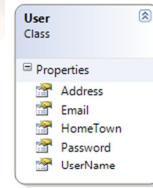
How we test this function?

Requirement :

- Write a module to add an User to Database

Business rule :

- Email can not be duplicated
- Email must be in valid form
- UserName 's length must be > 8
- UserName can not be duplicated
- Password length must be > 8



Giảng viên write a simple method to test như trên

Manual Unit Testing

- Write code
- Uploading the code to some place
- Build it
- Running the code manually (in many cases filling up forms etc step by step)
- Check Log files, Database, External Services, Values of variable names, Output on the screen etc
- If it does not work, repeat the above process

Giang vien demo traditional tesst

Manual Unit Testing - Limitation

- Developer nhớ được trường hợp nào thì test trường hợp đó
- Đến cuối dự án số lượng test case càng lúc càng nhiều, khả năng cover của lập trình viên giảm xuống!
- Nhiều test case bị trùng lắp
- Nhiều test case bị lack
- Team lead không thể review hết được
- → Kết quả dự án chỉ trông chờ vào tester!!!!
- → Rất nhiều lỗi phát sinh sau khi system test, đa phần các lỗi xuất phát do Dev test không kỹ từ lúc Unit Test!

Giảng viên write a simple method to test như trên

Unit Testing based on UT cases

- Để giải quyết vấn đề trên Mỗi khi developer test xong phải viết tài liệu mô tả test case trên word hoặc excel !
- Điều này giúp team rất dễ dàng review.. Tuy nhiên, cách làm này sẽ phát sinh ra rất nhiều hạn chế!

Function Code	Function I	Function Name	Function A									
Created By	<Developer Name>	Executed By	Lack of test cases -5									
Lines of code	100	Test requirement	<brief description about requirements which are tested in this function>									
Test requirement	Passed	Failed	Untested	N/A/B	Total Test Cases							
0	0	0	15	5	15							
			Module 1	Module 2	Module 3	Module 4	Module 5	Module 6	Module 7	Module 8	Module 9	Module 10
			Precondition									
	a		-2	O								
	b		-1		O	O	O	O				
	c		0		O							
			1			O	O					
			2				O	O				
			3					O	O			
			4						O	O		
			5							O	O	
			6								O	O
			7									O
			8									O
			9									O
			10									O
			11									O
			12									O
			13									O
			14									O
			15									O
			16									O
			17									O
			18									O
			19									O
			20									O
			21									O
			22									O
			23									O
			24									O
			25									O
			26									O
			27									O
			28									O
			29									O
			30									O
			31									O
			32									O
			33									O
			34									O
			35									O
			36									O
			37									O
			38									O
			39									O
			40									O
			41									O
			42									O
			43									O
			44									O
			45									O
			46									O
			47									O
			48									O
			49									O
			50									O
			51									O
			52									O
			53									O
			54									O
			55									O
			56									O
			57									O
			58									O
			59									O
			60									O
			61									O
			62									O
			63									O
			64									O
			65									O
			66									O
			67									O
			68									O
			69									O
			70									O
			71									O
			72									O
			73									O
			74									O
			75									O
			76									O
			77									O
			78									O
			79									O
			80									O
			81									O
			82									O
			83									O
			84									O
			85									O
			86									O
			87									O
			88									O
			89									O
			90									O
			91									O
			92									O
			93									O
			94									O
			95									O
			96									O
			97									O
			98									O
			99									O
			100									O
			101									O
			102									O
			103									O
			104									O
			105									O
			106									O
			107									O
			108									O
			109									O
			110									O
			111									O
			112									O
			113									O
			114									O
			115									O
			116									O
			117									O
			118									O
			119									O
			120									O
			121									O
			122									O
			123									O
			124									O
			125									O
			126									O
			127									O
			128									O
			129									O
			130									O
			131									O
			132									O
			133									O
			134									O
			135									O
			136									O
			137									O
			138									O
			139									O
			140									O
			141									O
			142									O
			143									O
			144									O
			145									O
			146									O
			147									O
			148									O
			149									O
			150									O
			151									O
			152									O
			153									O
			154									O
			155									O
			156									O
			157									O
			158									O
			159									O
			160									O
			161									O
			162									O
			163									O
			164									O
			165									O
			166									

Unit Testing based on UT cases

- Các dự án lớn thì số lượng tài liệu test case thường cũng rất lớn!
- Các dự án lớn thì requirement thường hay thay đổi
- Mỗi khi requirement thay đổi → Phải sửa code → phải cập nhật lại tài liệu testcase → và lại manual retest , rất tốn effort → Càng đến cuối dự án, lượng việc sinh ra càng nhiều , viết test case document trở thành “địa ngục “ thực sự ! → dev không còn đủ effort update test case document, tài liệu nhanh chóng bị lạc hậu, hoặc việc update chỉ là đối phó!
- Một số trường hợp không thể dùng Excel TestCase



So, what is the solution?

© FPT Software

8

Giảng viên trình bày nhược điểm

Automated Unit Testing

First Step

- Coding Process with Automated Unit Tests
 - Write code
 - Write one or more test cases script
 - Auto-compile and run
 - If tests fail -> make appropriate modifications
 - If tests pass -> repeat for next method

Automated Unit Testing First Step

DEMO

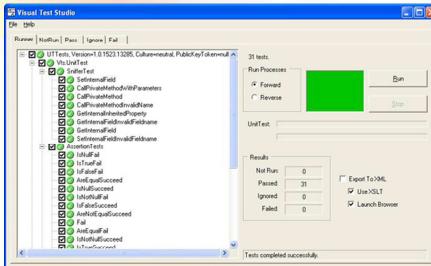
© FPT Software

10

Giảng viên demo cách viết sử dụng manual code, sử dụng code có sẵn trong thư mục “UnitTest.Demo”

Automated Unit Testing Common Tools

- UT Tools for references:
 - Java: JUnit, J2MEUnit
 - C/C++: cppUnit
 - Python: pyUnit
 - Perl: PerlUnit
 - Visual Basic: vbUnit
 - C# .NET: Nunit,csUnit



The screenshot shows the Visual Studio Test Explorer window. It displays a tree view of test cases under the category 'UTTools'. There are 31 tests listed, all of which have passed. The results summary at the bottom indicates 0 Not Run, 31 Passed, 0 Ignored, and 0 Failed. The 'Run Processes' section has 'Forward' selected. The 'Results' section includes checkboxes for 'Export To XML', 'Use XSLT', and 'Launch Browser'.

- Refferences:
 - ★ <http://www.testingfaqs.org/t-unit.html>
 - ★ www.junit.org
 - ★ <http://www.codeproject.com/gen/design/autp5.asp>

© FPT Software

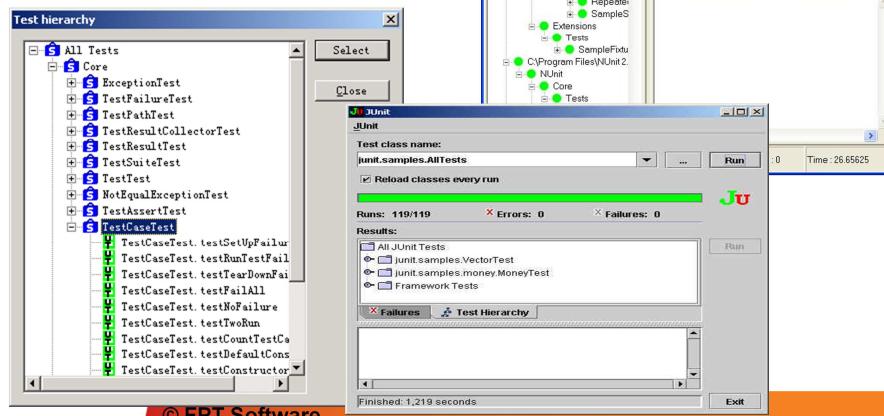
11

Automated Unit Testing Common Tools

<http://sourceforge.net/projects/cppunit/>

<http://www.nunit.org>

<http://www.junit.org/>



12

Automated Unit Testing with CPPUnit

What is CPPUnit?

Milk ? Beer or
Coffee?

- CPPUnit – an open source test tool for C/C++
- Useful for development and regression
- Leads to a design-for-test approach
- Tests can be written in C/C++

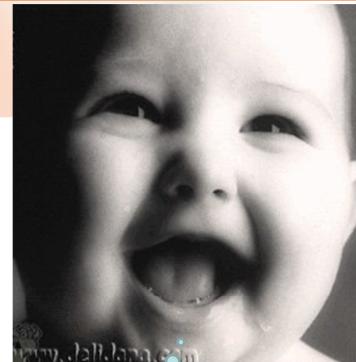


Automated Unit Testing with CPPUnit

Where to get CPPUnit?

- Let's go to website:
<http://downloads.sourceforge.net/cppunit/cppunit-1.12.1.tar.gz>

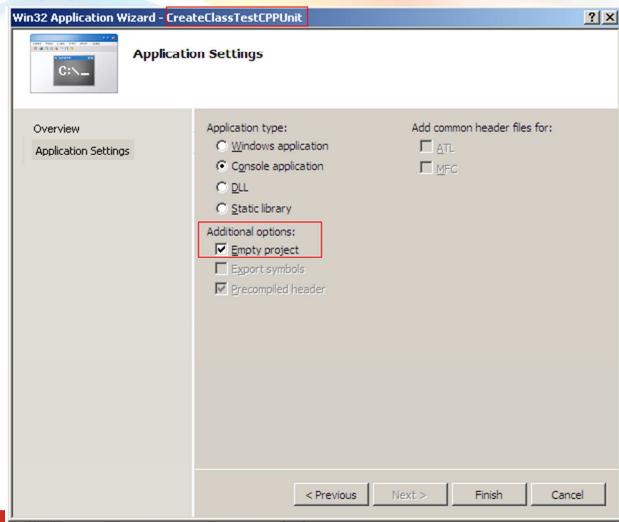
Sample: [CppUnit_Hands-on Lab.doc](#)



Automated Unit Testing with CPPUNIT

Create a test case

- Step 1: Create a project C++ dạng console



© FPT Software

15

Automated Unit Testing with CPPUNIT Create a test case

- Step 2: Create a Class
Calculator có dạng sau:
- Step 3: build. Chương trình sẽ
báo lỗi như bên dưới.

The screenshot shows the Microsoft Visual Studio interface. In the Solution Explorer, there is a project named 'CreateClassTestCPPUnit' containing a Header Files folder with 'Calculator.h' and a Source Files folder with 'Calculator.cpp'. The code editor window displays the contents of 'Calculator.h' and 'Calculator.cpp'. The 'Calculator.h' file contains a class definition for 'CCalculator' with private members 'int x;' and 'int y;', a constructor 'CCalculator(void)', a constructor 'CCalculator(int, int)', a member function 'int Addition(int, int);', and a destructor '~CCalculator(void);'. The 'Calculator.cpp' file includes 'Calculator.h', defines the constructor 'CCalculator::CCalculator(void)' and 'CCalculator::CCalculator(int, int)', and implements the member function 'int CCcalculator::Addition(int a, int b)' which returns 'a + b'. The Output window shows the build log:

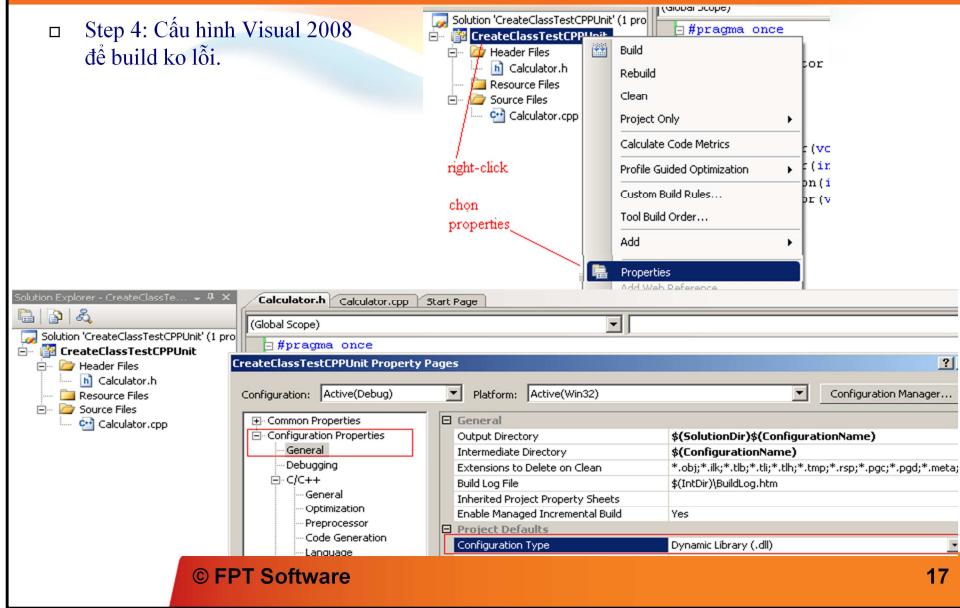
```
D:\...\00_Working\...\00_training\...\OS_FSI\...\00_CPP_Version 4.0\...\01>Error: Could not generate object file.  
1>Error: cmd:link.exe @C:\DOCUME~1\bactd1\LOCALS~1\Temp\mb7be5__cs_re  
1>Error: compiler cmd:link.exe @C:\DOCUME~1\bactd1\LOCALS~1\Temp\mb7be  
1>Build log was saved at "file:///d:/00_Working/00_training/OS_FSI/00_C  
1>CreateClassTestCPPUnit - 2 error(s), 1 warning(s)  
===== Rebuild All: 0 succeeded, 1 failed, 0 skipped =====
```

© FPT Software .6

Automated Unit Testing with CPPUNIT

Create a test case

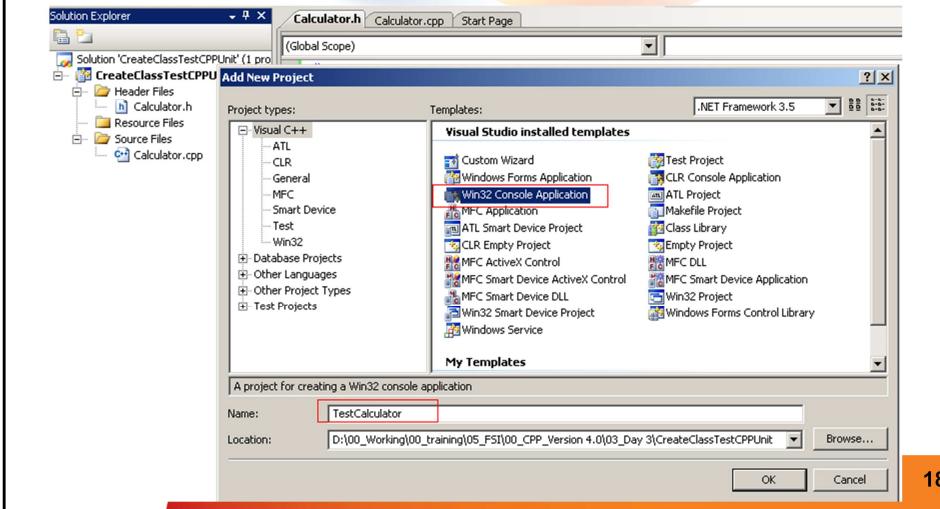
- Step 4: Cấu hình Visual 2008
để build ko lỗi.



Automated Unit Testing with CPPUNIT

Create a test case

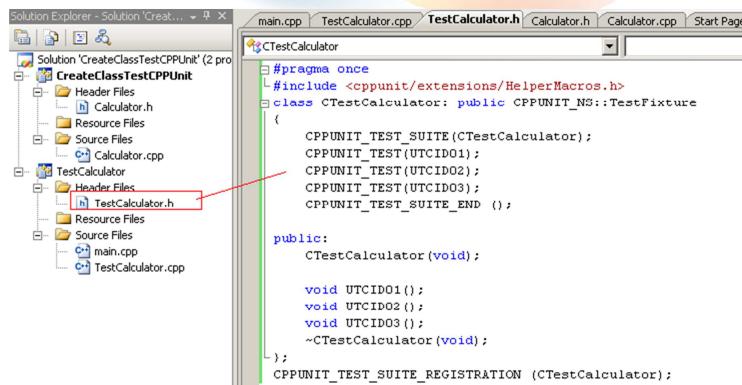
- Step 5: Add new empty project (Click phải chuột vào Solution... / Add / New Project).
- (Tạo giống bước 1)



Automated Unit Testing with CPPUNIT

Create a test case

- Step 6: Tạo nội dung của TestCalculator giống như sau:



```
#pragma once
#include <cppunit/extensions/HelperMacros.h>
class CTestCalculator: public CPPUNIT_NS::TestFixture
{
    CPPUNIT_TEST_SUITE(CTestCalculator);
    CPPUNIT_TEST(UTCID01);
    CPPUNIT_TEST(UTCID02);
    CPPUNIT_TEST(UTCID03);
    CPPUNIT_TEST_SUITE_END();

public:
    CTestCalculator(void);

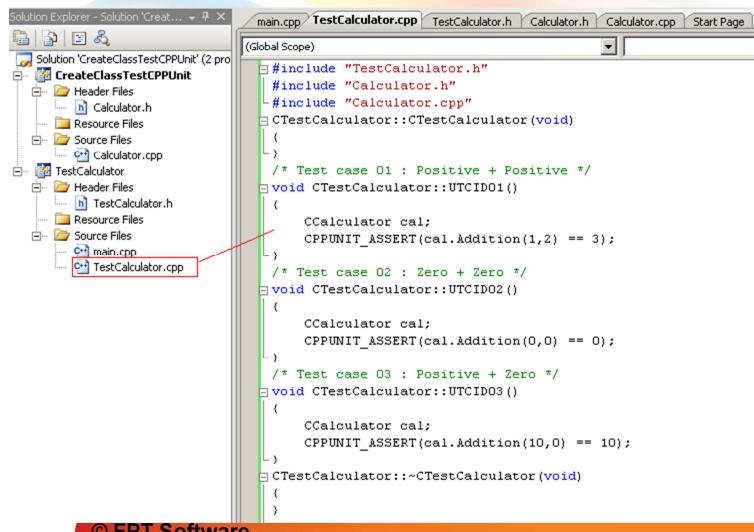
    void UTCID01();
    void UTCID02();
    void UTCID03();
    ~CTestCalculator(void);
};

CPPUNIT_TEST_SUITE_REGISTRATION (CTestCalculator);
```

Automated Unit Testing with CPPUNIT

Create a test case

- Step 6: Tạo nội dung của TestCalculator giống như sau:



```
#include "TestCalculator.h"
#include "Calculator.h"
#include "Calculator.cpp"

CTestCalculator::CTestCalculator(void)
{
}

/* Test case 01 : Positive + Positive */
void CTestCalculator::UTCID01()
{
    CCalculator cal;
    CPPUNIT_ASSERT(cal.Addition(1,2) == 3);
}

/* Test case 02 : Zero + Zero */
void CTestCalculator::UTCID02()
{
    CCalculator cal;
    CPPUNIT_ASSERT(cal.Addition(0,0) == 0);
}

/* Test case 03 : Positive + Zero */
void CTestCalculator::UTCID03()
{
    CCalculator cal;
    CPPUNIT_ASSERT(cal.Addition(10,0) == 10);
}

CTestCalculator::~CTestCalculator(void)
{}
```

© FPT Software

20

Automated Unit Testing with CPPUNIT

Create a test case

- Step 6: Tạo nội dung của TestCalculator giống như sau:



```
#include "cconio.h"
#include <iostream>
#include <iomanip>
#include <cppunit/CompilerOutputter.h>
#include <cppunit/extensions/TestFactoryRegistry.h>
#include <cppunit/TestResult.h>
#include <cppunit/TestResultCollector.h>
#include <cppunit/TestRunner.h>
#include <cppunit/BriefTestProgressListener.h>
#include <cppunit/XmlOutputter.h>
using namespace std;

int main()
{
    CPPUNIT_NS :: TestResult TestResult;
    // register listener for collecting the test-results
    CPPUNIT_NS :: TestResultCollector CollectedResults;
    TestResult.addListener(&CollectedResults);

    // register listener for per-test progress output
    CPPUNIT_NS :: BriefTestProgressListener progress;
    TestResult.addListener(&progress);

    // insert test-suite at test-runner by registry
    CPPUNIT_NS :: TestRunner Runner;
    Runner.addTest(CPPUNIT_NS :: TestFactoryRegistry :: getRegistry().makeTest());
    Runner.run(TestResult);

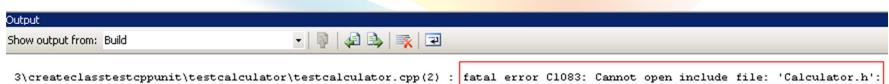
    // output results in compiler-format
    CPPUNIT_NS :: CompilerOutputter Outputter(&CollectedResults, std::cerr);
    Outputter.write();

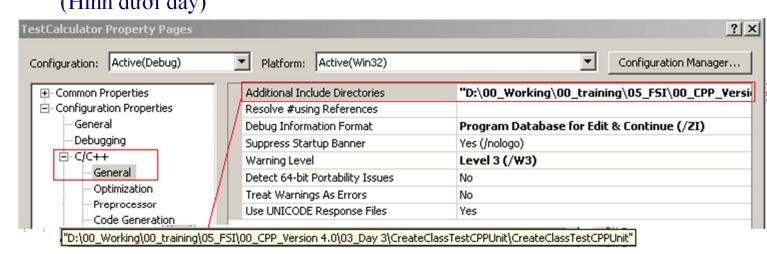
    // writing results on a XML file
    std::ofstream xmlFileOut("testresults.xml");
    CppUnit::XmlOutputter xmlOut(&CollectedResults, xmlFileOut);
    xmlOut.write();

    /* wait user to press any key */
    system("pause");
    // return 0 if tests were successful
    return CollectedResults.wasSuccessful() ? 0 : 1;
}
```

Automated Unit Testing with CPPUnit

Create a test case

- Step 7: Set TestCalculator as startup project
- Step 8: build. Chương trình sẽ thông báo lỗi như sau.


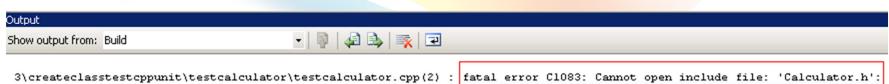
```
3\createclasstestcppunit\testcalculator\testcalculator.cpp(2) : fatal error C1083: Cannot open include file: 'Calculator.h':
```
- Step 9: Cấu hình tool để build thành công.
 - Click chuột phải project TestCalculator chọn Properties
 - Chọn Configuration Properties/ C/C++/ General
 - Tại phần Additional Include Directories: Chỉ đến đường dẫn file nguồn của project.
(Hình dưới đây)

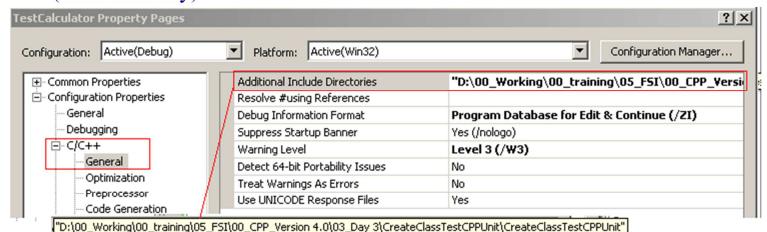
Additional Include Directories	"D:\00_Working\00_training\05_FSI\00_CPP_Version 4.0\03_Day 3\CreateClassTestCPPUnit\CreateClassTestCPPUnit"
Resolve #using References	
Debug Information Format	Program Database for Edit & Continue (/ZI)
Suppress Startup Banner	Yes (/nologo)
Warning Level	Level 3 (/W3)
Detect 64-bit Portability Issues	No
Treat Warnings As Errors	No
Use UNICODE Response Files	Yes

22

Automated Unit Testing with CPPUnit

Create a test case

- Step 7: Set TestCalculator as startup project
- Step 8: build. Chương trình sẽ thông báo lỗi như sau.


```
3\createclasstestcppunit\testcalculator\testcalculator.cpp(2) : fatal error C1083: Cannot open include file: 'Calculator.h':
```
- Step 9: Cấu hình tool để build thành công.
 - Click chuột phải project TestCalculator chọn Properties
 - Chọn Configuration Properties/ C/C++/ General
 - Tại phần Additional Include Directories: Chỉ đến đường dẫn file nguồn của project.
(Hình dưới đây)

Additional Include Directories	"D:\00_Working\00_training\05_FSI\00_CPP_Version 4.0\03_Day 3\CreateClassTestCPPUnit\CreateClassTestCPPUnit"
Resolve #using References	
Debug Information Format	Program Database for Edit & Continue (/ZI)
Suppress Startup Banner	Yes (/nologo)
Warning Level	Level 3 (/W3)
Detect 64-bit Portability Issues	No
Treat Warnings As Errors	No
Use UNICODE Response Files	Yes

23

Automated Unit Testing with CPPUNIT

Create a test case

- Step 10: re-build. Chương trình sẽ thông báo lỗi như sau.

```
Output  
Show output from: Build  
  
1>main.obj : error LNK2001: unresolved external symbol "public: static class CppUnit::TestFixtureRegistry & __cdecl CppUnit::TestFixtureRegistry::Create()" (??0TestFixtureRegistry@CppUnit@@CAGAAVTestFixtureRegistry@1@XZ)  
1>TestCalculator.obj : error LNK2019: unresolved external symbol "public: virtual __thiscall CppUnit::TestSuiteBuilderContextBase::TestSuiteBuilderContextBase(void)" (??0TestSuiteBuilderContextBase@TestSuiteBuilderContextBase@CppUnit@@CAGQAVTestSuiteBuilderContextBase@1@XZ)  
1>TestCalculator.obj : error LNK2019: unresolved external symbol "public: __thiscall CppUnit::TestSuite::TestSuite(class std::basic_string<char>,class std::basic_string<char>)" (??0TestSuite@CppUnit@@CAGQAVTestSuite@1@AQBVA2std@1@AQBVA2std@1@XZ)  
1>TestCalculator.obj : error LNK2019: unresolved external symbol "public: __thiscall CppUnit::TestName::TestName(class type_info const &)" (??0TestName@CppUnit@@CAGQAVTestName@1@AQBVRB2std@1@XZ)  
1>TestCalculator.obj : error LNK2019: unresolved external symbol "void __thiscall CppUnit::TestSuiteBuilderContextBase::AddTest(class std::basic_string<char>,class std::basic_string<char>)" (??4TestSuiteBuilderContextBase@TestSuiteBuilderContextBase@CppUnit@@CAGQAVTestSuiteBuilderContextBase@1@AQBVA2std@1@AQBVA2std@1@XZ)  
1>TestCalculator.obj : error LNK2019: unresolved external symbol "protected: class CppUnit::TestFixture & __thiscall CppUnit::TestFixture::Create()" (??0TestFixture@CppUnit@@CAGAAVTestFixture@1@XZ)
```

- ☐ Step 11: Cấu hình tool để build thành công.

- Tiếp theo chọn Configuration Properties/ Linker/ Input
 - Tại mục Additional Dependencies, nhập vào: cppunitd.lib
 - Tiếp theo chọn “Release” tại combobox Configuration
 - Tại mục Additional Dependencies, nhập vào: cppunit.lib
 - Click Ok để hoàn tất.
 - Re-build.

```
Output  
Show output from: Build  
  
1>Embedding manifest...  
1>Microsoft (R) Windows (R) Resource Compiler Version 6.0.5724.0  
1>Copyright (C) Microsoft Corporation. All rights reserved.  
1>LINK : warning LNK4044: unrecognized option '/ERROREREPORT:prompt' ignored  
1>Build log was saved at "file:///d|/00/Working/00_training/05_FSI/00_CPI/00_CPI/Debug/BuildLog.htm"  
1>TestCalculator - 0 error(s), 31 warning(s)  
===== Rebuild All: 1 succeeded 0 failed 0 skipped =====
```

24

Automated Unit Testing with CPPUNIT

Create a test case

Step 12: F5

```
cmd d:\00_Working\00_training\05_FSI\00_CPP_Version 4.0\03_Day 3\CreateClassTestCPPUnit\Debug\T...
CtestCalculator::UTCI_D91 : OK
CtestCalculator::UTCI_D92 : OK
CtestCalculator::UTCI_D93 : OK
OK <3>
Press any key to continue . . .
```

Note:

Trong trường hợp build không được vui lòng đọc hướng dẫn cách build CPPUNIT ở thư mục : [Huong Dan Cai Dat](#).

Có thể tham khảo cách tạo CPPUNIT dựa theo [Video_CPPUnitStepByStep.mp4](#)

Automated Unit Testing with CPPUnit

DEMO

CPPUnit
is
Number
One.



© FPT Software

26



© FPT Software

27