

CS 485
Advanced Object Oriented Design

Unit Testing C++

in Visual Studio

Spring 2019

Reading

- Microsoft Documentation

<https://docs.microsoft.com/en-us/visualstudio/test/microsoft-visualstudio-testtools-cppunittestframework-api-reference?view=vs-2017>

- Martin Fowler's Description

<https://martinfowler.com/bliki/UnitTest.html>

Fowler's description

- Low-level, focus on small part of the system
- Written by programmers to test their own code
 - not by a separate set of testers
- Expected to be faster than other tests

Unit

- Class
- Method
- Set of Methods
- Interface
- Set of classes
- ???

Implementation Details

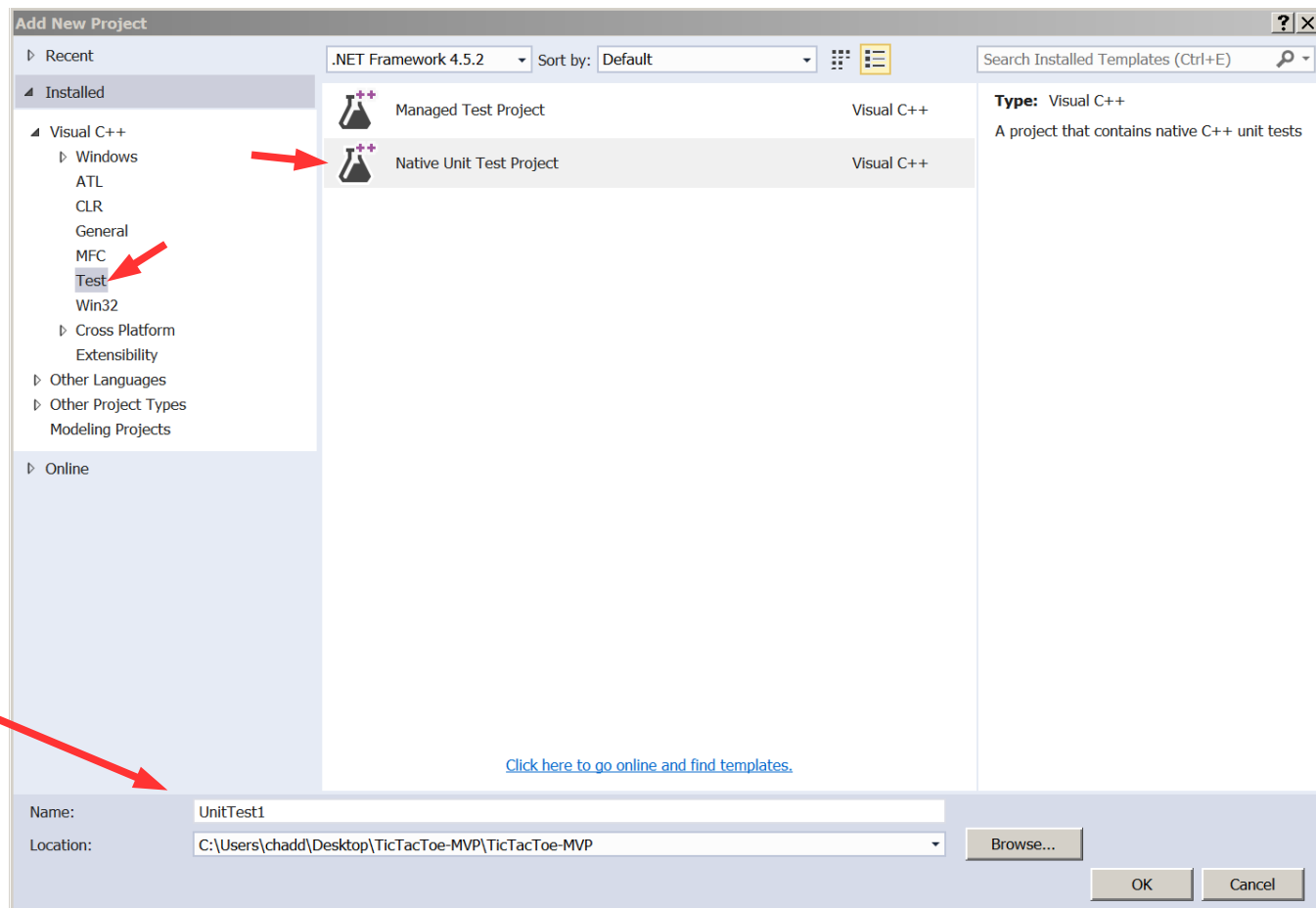
- Framework* for testing
 - easy to write many small tests
 - easy way to run the tests and track which individual test passed/failed
 - no need to build a driver
 - provide easy way to mark a test as pass/fail
 - Assert()

* A *software framework* provides code to solve a problem and allows the user to write small bits of code in certain spots to customize how the problem is solved.

problem → testing

Visual Studio and C++

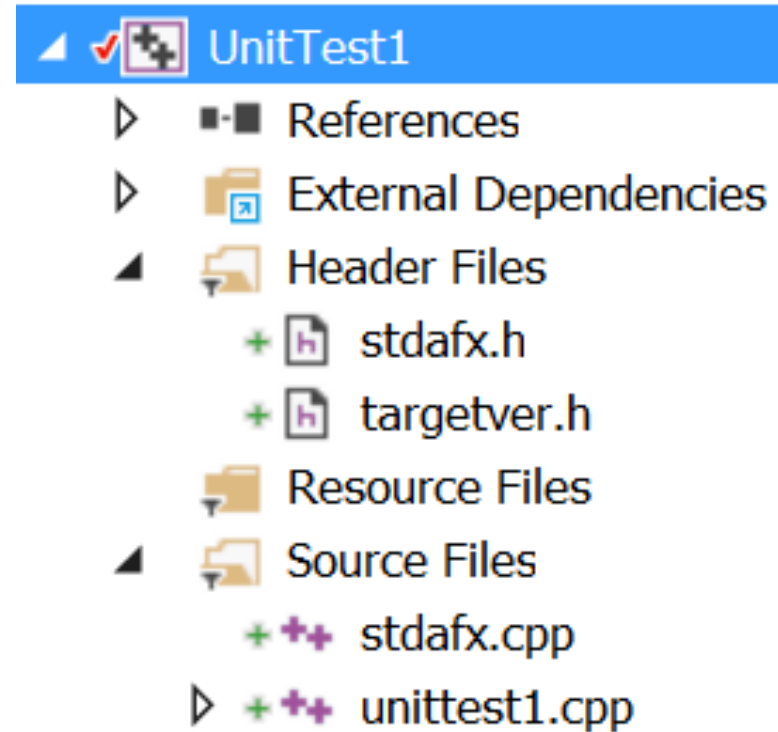
- Add a new project to your solution
- Link that project (include dir & obj files) to an existing project to test in the normal way



Give the project a decent name

The Unit Test Project

- Your tests go in the file `unittest1.cpp`
 - this file may have another, better name



Link Unit Test project back to project being tested

Include Directories
Library Directories

Linker | Input

Code

```

#include "stdafx.h"
#include "CppUnitTest.h"
#include "TicTacToeBoard.h"

using namespace Microsoft::VisualStudio::CppUnitTestFramework;

namespace TicTacToe_UnitTests
{
    TEST_CLASS(TicTacToeBoard_UnitTests)
    {
    public:

        //*****
        // Unit Test:   FailingTest
        //
        // Description: This test should fail since the initial board is not full
        //
        //*****
        TEST_METHOD (FailingTest)
        {
            TicTacToeBoard cTheBoard;

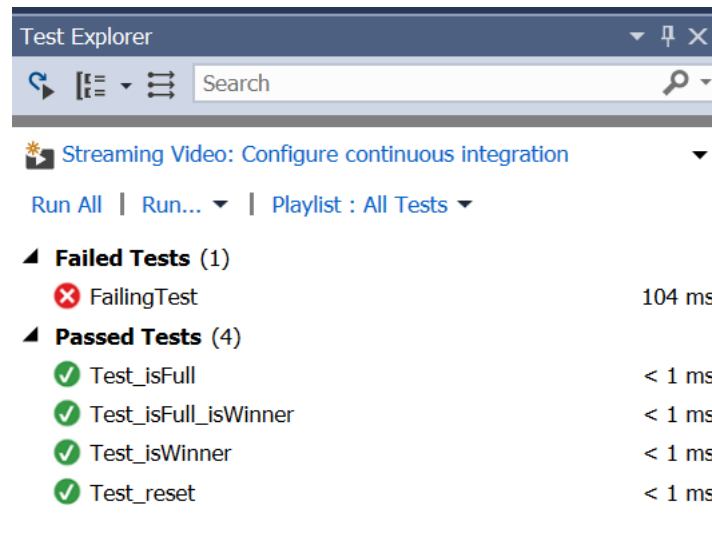
            Assert::AreEqual (true, cTheBoard.isBoardFull (),
                L"Failing Test Here is the msg!", LINE_INFO ());
        }
    }
}

```

Note what is
autogenerated
by the
framework

To run

- Visual Studio Menu:
 - Test | Run | All Tests
- Test Explorer



More elaborate test

```
//*****  
// Unit Test:   Test_isWinner  
//  
// Description: Test if isWinner() can find a winner  
//  
//*****  
TEST_METHOD (Test_isWinner)  
{  
    TicTacToeBoard cTheBoard;  
  
    cTheBoard.makeMove (0, 0, TicTacToeBoard::Player::ONE);  
    cTheBoard.makeMove (0, 1, TicTacToeBoard::Player::ONE);  
    cTheBoard.makeMove (0, 2, TicTacToeBoard::Player::ONE);  
  
    Assert::AreEqual (true,  
        cTheBoard.isWinner (TicTacToeBoard::Player::ONE),  
        L"Win not detected!", LINE_INFO ());  
}
```

One More

```
//*****  
// Unit Test:  Test_reset  
//  
// Description: Test if reset works  
//  
//*****  
TEST_METHOD (Test_reset)  
{  
    TicTacToeBoard cTheBoard;  
  
    cTheBoard.makeMove (0, 0, TicTacToeBoard::Player::ONE);  
    cTheBoard.makeMove (0, 1, TicTacToeBoard::Player::ONE);  
    cTheBoard.makeMove (0, 2, TicTacToeBoard::Player::ONE);  
  
    cTheBoard.reset ();  
  
    Assert::AreEqual (false,  
        cTheBoard.isWinner (TicTacToeBoard::Player::ONE),  
        L"Reset did not work", LINE_INFO ());  
}
```

Handled an expected exception

- Call a function in the UnitTest that should throw an exception
- Make sure that Exception is thrown
- ExpectException()

```
template<typename _EXPECTEDECEPTION, typename _FUNCTOR>
static void ExpectException(
    _FUNCTOR functor,
    const wchar_t* message= NULL,
    const __LineInfo* pLineInfo= NULL)
```

```
Assert::ExpectException<std::range_error>(func,
    L"Exception not thrown", LINE_INFO());
```

https://msdn.microsoft.com/en-us/library/hh694604.aspx#BKMK_Exception_Asserts

<https://blogs.msdn.microsoft.com/dgartner/2012/04/22/using-assertexpectexception-with-native-unit-testing-in-vs11/>

Unit Test

- The unit test is in a class, so you can have private data members!
- Each test method runs in its own instance of the class
- No state is shared between test methods

“When the tests are run, an instance of each test class is created. The test methods are called in an unspecified order. You can define special methods that are invoked before and after each module, class, or method. For more information, see *Organizing C++ Tests*.”

<https://msdn.microsoft.com/en-us/library/hh270864.aspx>

Other functions

```
// runs before each test method is run
TEST_METHOD_INITIALIZE(methodName)
{
    // method initialization code
}
```

```
// runs after each test method is run
TEST_METHOD_CLEANUP(methodName)
{
    // test method cleanup code
}
```

```
// runs after each test class is created
TEST_CLASS_INITIALIZE(methodName)
{
    // test class initialization code
}
```

```
TEST_CLASS_CLEANUP(methodName)
{
    // test class cleanup code
}
```

Console Output

`std::cout` does not work in the unit test

```
Logger::WriteMessage("running test");
```

Your Task

- Download MVP_TicTacToe_Public_unit_test
- Add a new cpp file to TicTacToe_UnitTests

TicTacToeModel_UnitTests.cpp

- Add a unit test that uses makeMove() to test if isWinner() is correct