

Faculty of Mathematics and Physics
Charles University in Prague
4th May 2015



C# Made Easy!

Programming II

Workshop 11 – Functional Testing

Workshop 11

Outline

1. Test
2. Functional Testing – What, Why, How
3. Homework



Test 11

Test

Find the test here (no-ads):

<http://goo.gl/bgQp7p>

Permanent link:

<https://docs.google.com/forms/d/1Cqmf-U46lo8KIYpAmF3pBLF4b8xuleesjLbiBng8lwl/viewform>

Time for the test:

8 min

Testing

What?

WHAT?

Testing

What?

- Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation.

To put it simply...

- Running a code that executes another code and compares results with pre-computed/pre-specified ones.

Testing

What?

- Simple example

```
class Calc {  
    int Add(int a1, int a2)  
}
```

```
class CalculatorTest {  
  
    public boolean TestAdd() {  
        Calc c = new Calculator();  
        if (c.Add(1, 1) == 2) return true;  
        return false;  
    }  
  
}
```

Testing

Why?

WHY?

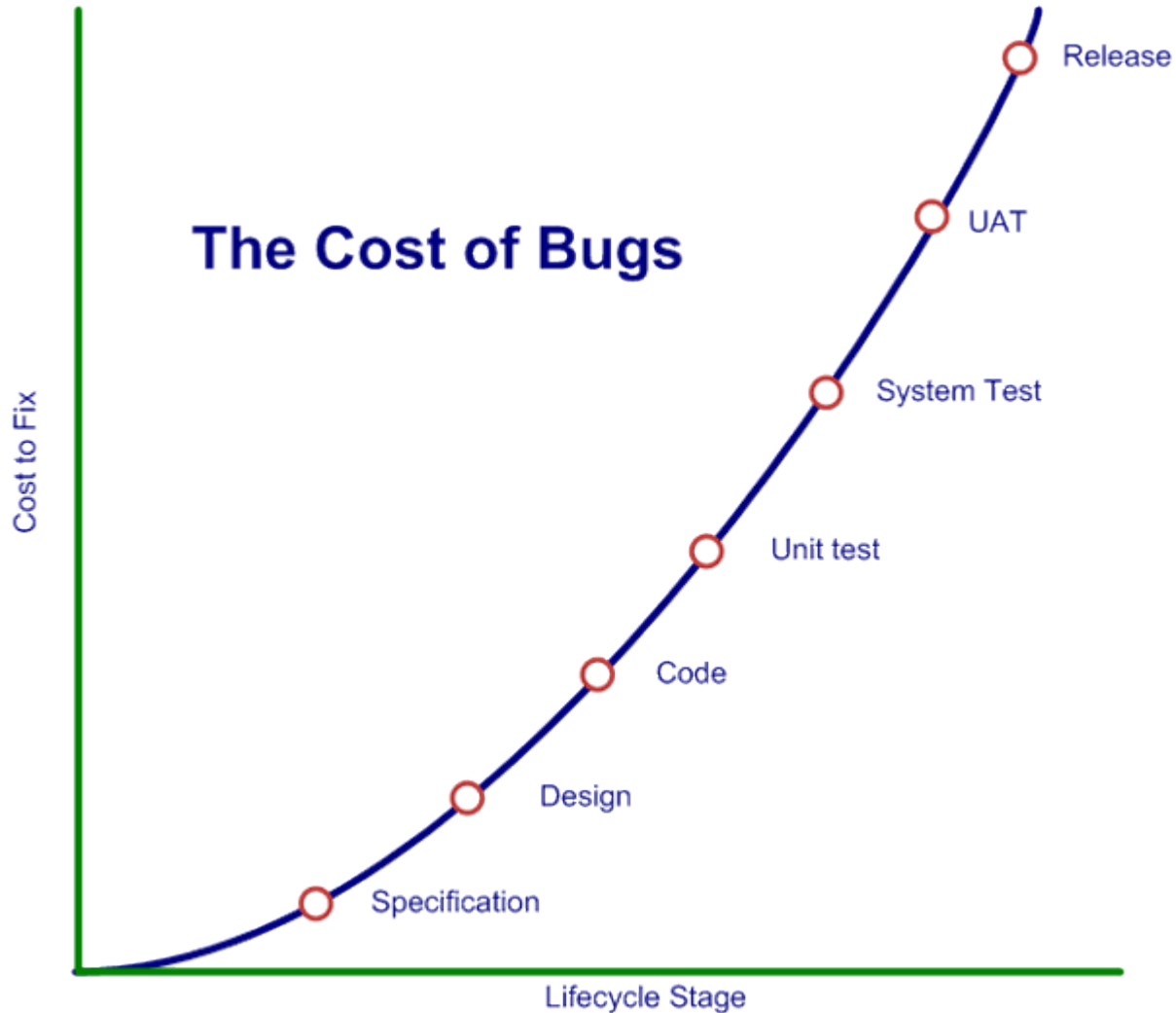
Testing

Why?

Because **TIME** (translates as **MONEY**) matters!

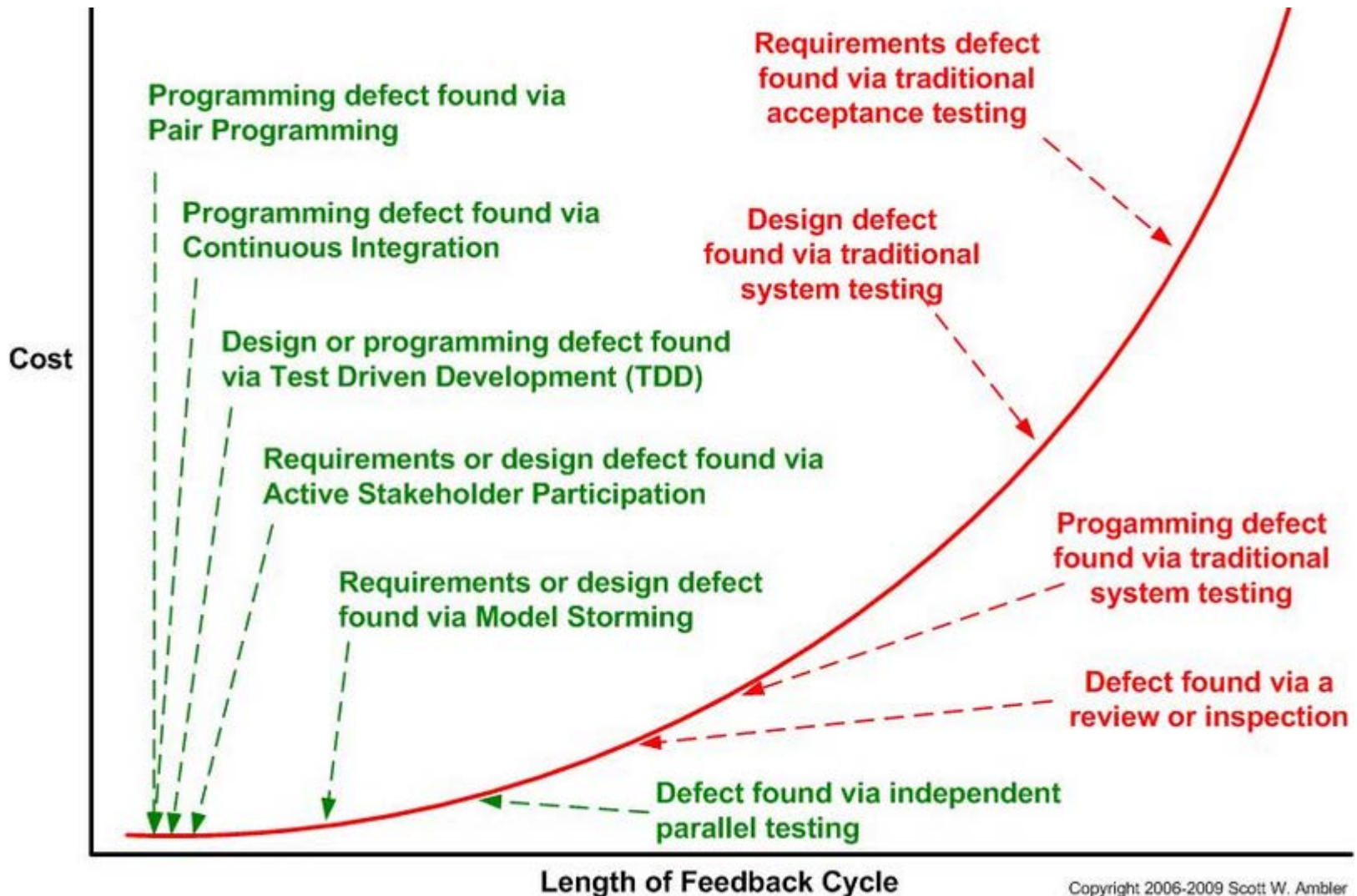
Testing

Why?



Testing

Why?



Testing

Why?

- Tests reduce bugs in existing features
- Tests reduce bugs in new features
- Tests (of complex code base) are good documentation
- Tests improve design
- Tests reduce cost of change
- Tests constrain features
- Tests reduce fear of making changes
 - For your colleagues as well as you!

Testing

Why not to?

WHY NOT TO?

Testing

Why not to write tests?

- It takes too much time to write tests
- It takes too much time to execute tests
- It's not your job to test the code
- I don't really know how the code should behave so I can't test it!

Testing

Why not to write tests?

- It takes too much time to write tests
 - But you have time to hunt bugs down?
- It takes too much time to execute tests
- It's not your job to test the code
- I don't really know how the code should behave so I can't test it!

Testing

Why not to write tests?

- It takes too much time to write tests
 - But you have time to hunt bugs down?
- It takes too much time to execute tests
 - You are running your tests manually ... and the same time you're considering yourself to be THE programmer?
- It's not your job to test the code
- I don't really know how the code should behave so I can't test it!

Testing

Why not to write tests?

- It takes too much time to write tests
 - But you have time to hunt bugs down?
- It takes too much time to execute tests
 - You are running your tests manually ... and the same time you're considering yourself to be THE programmer?
- It's not your job to test the code
 - Oh, and you expect to have customers?
- I don't really know how the code should behave so I can't test it!

Testing

Why not to write tests?

- It takes too much time to write tests
 - But you have time to hunt bugs down?
- It takes too much time to execute tests
 - You are running your tests manually ... and the same time you're considering yourself to be THE programmer?
- It's not your job to test the code
 - Oh, and you expect to have customers?
- I don't really know how the code should behave so I can't test it!
 - You should have not started writing such code from the very beginning!

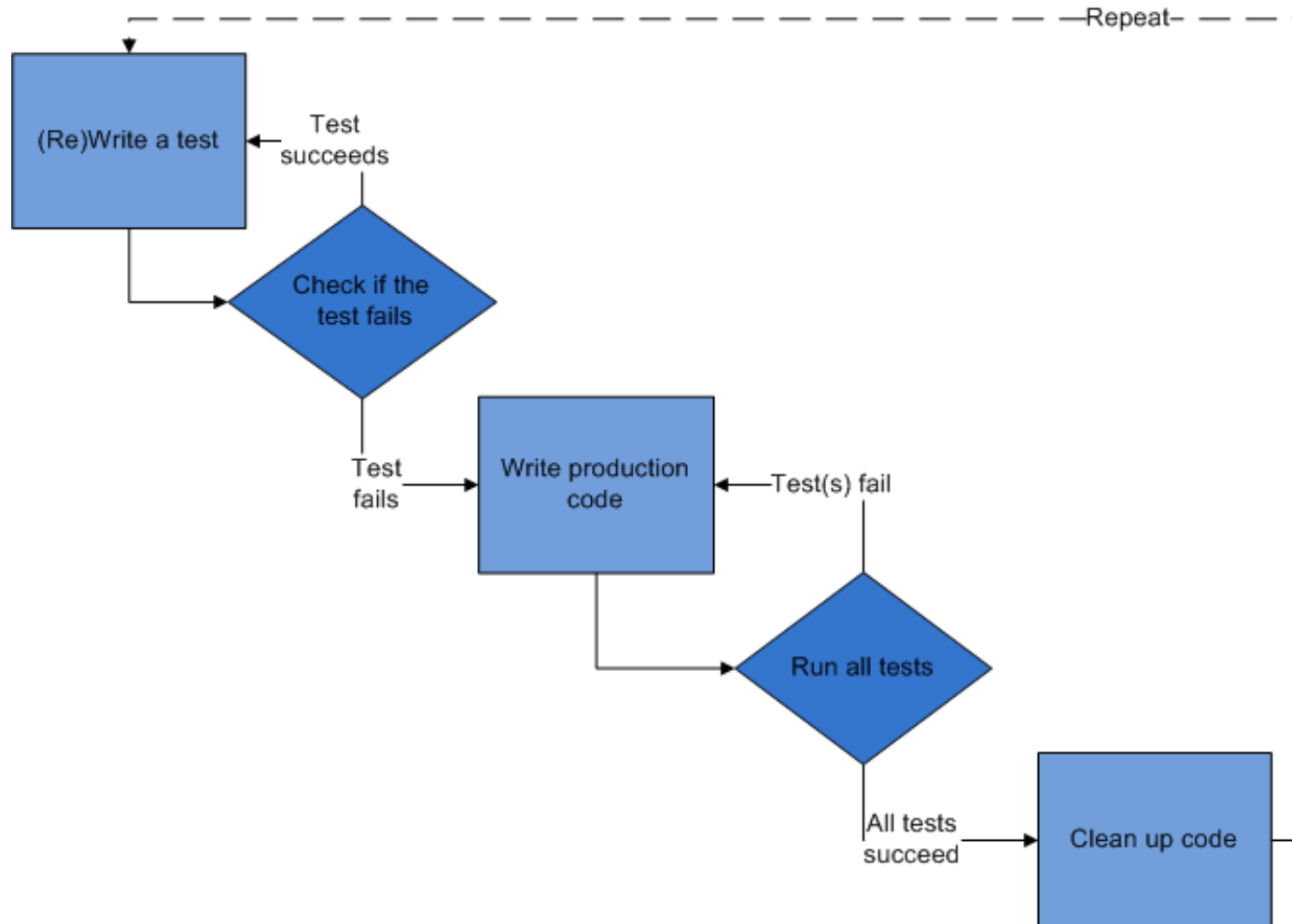
Testing

How?

HOW?

Testing

How?



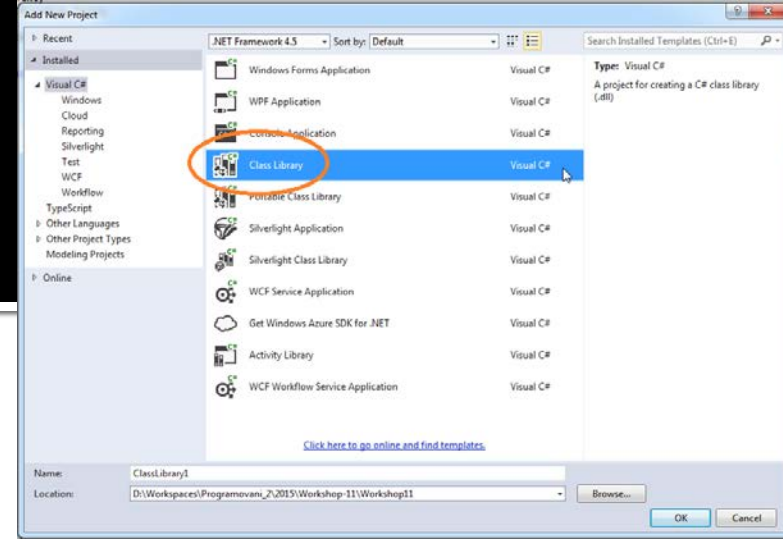
Testing

How?

Show time...

Testing

Creating projects

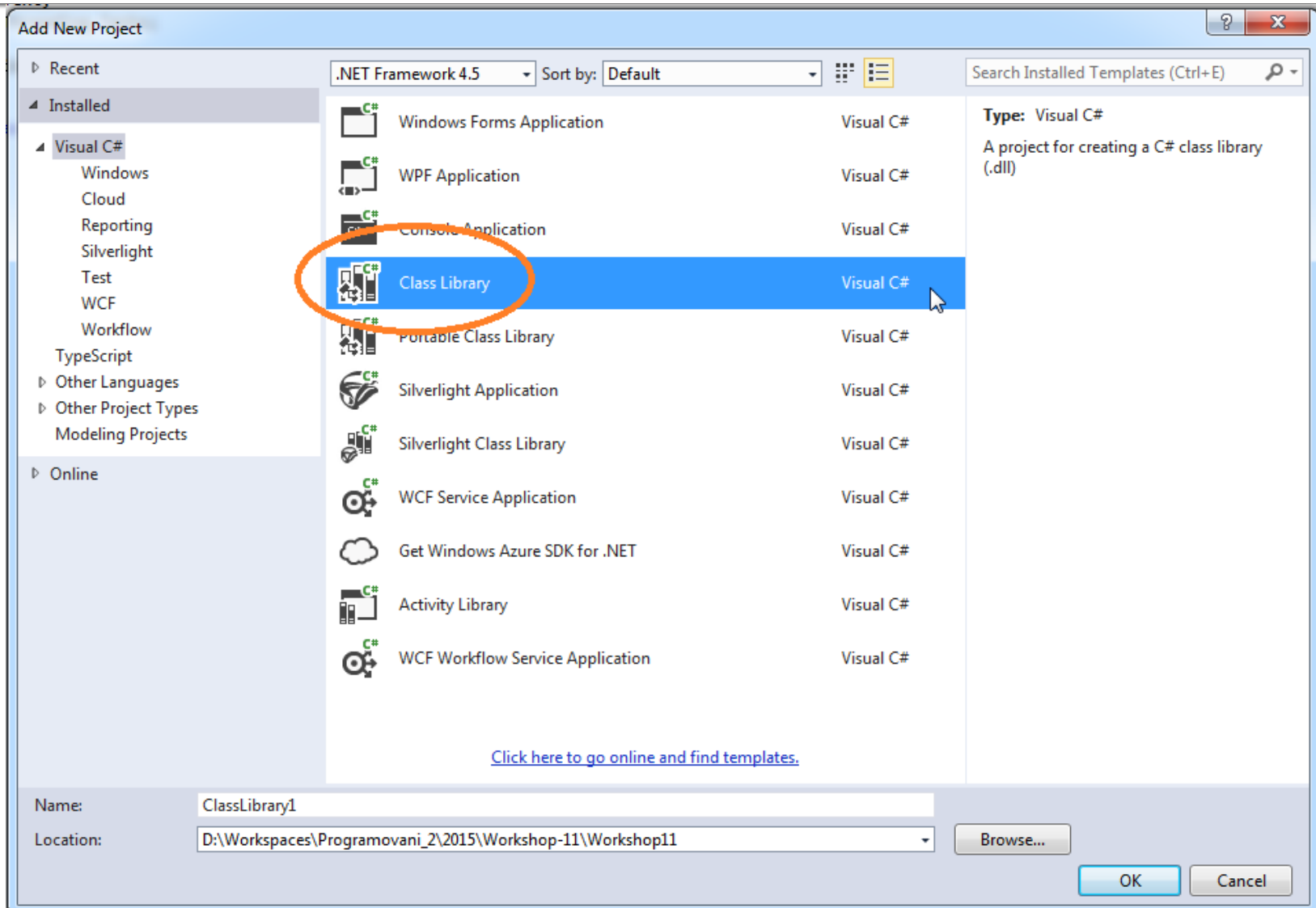


Create two (Class Library) projects

1. First will contain your classes (unit) for testing
Name it e.g.: MyLibrary
2. Second will contain TESTs that will be executed to test your first project
Name it e.g.: MyLibrary.Test

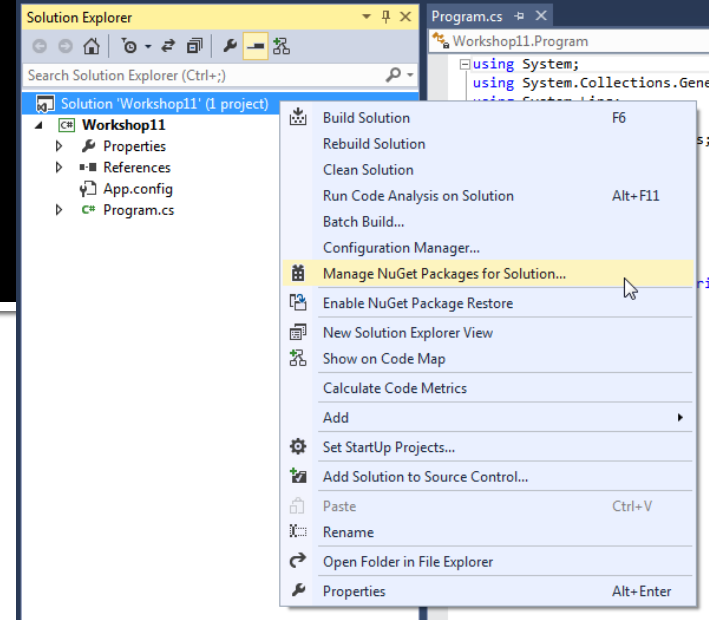
Testing

Creating projects



Testing

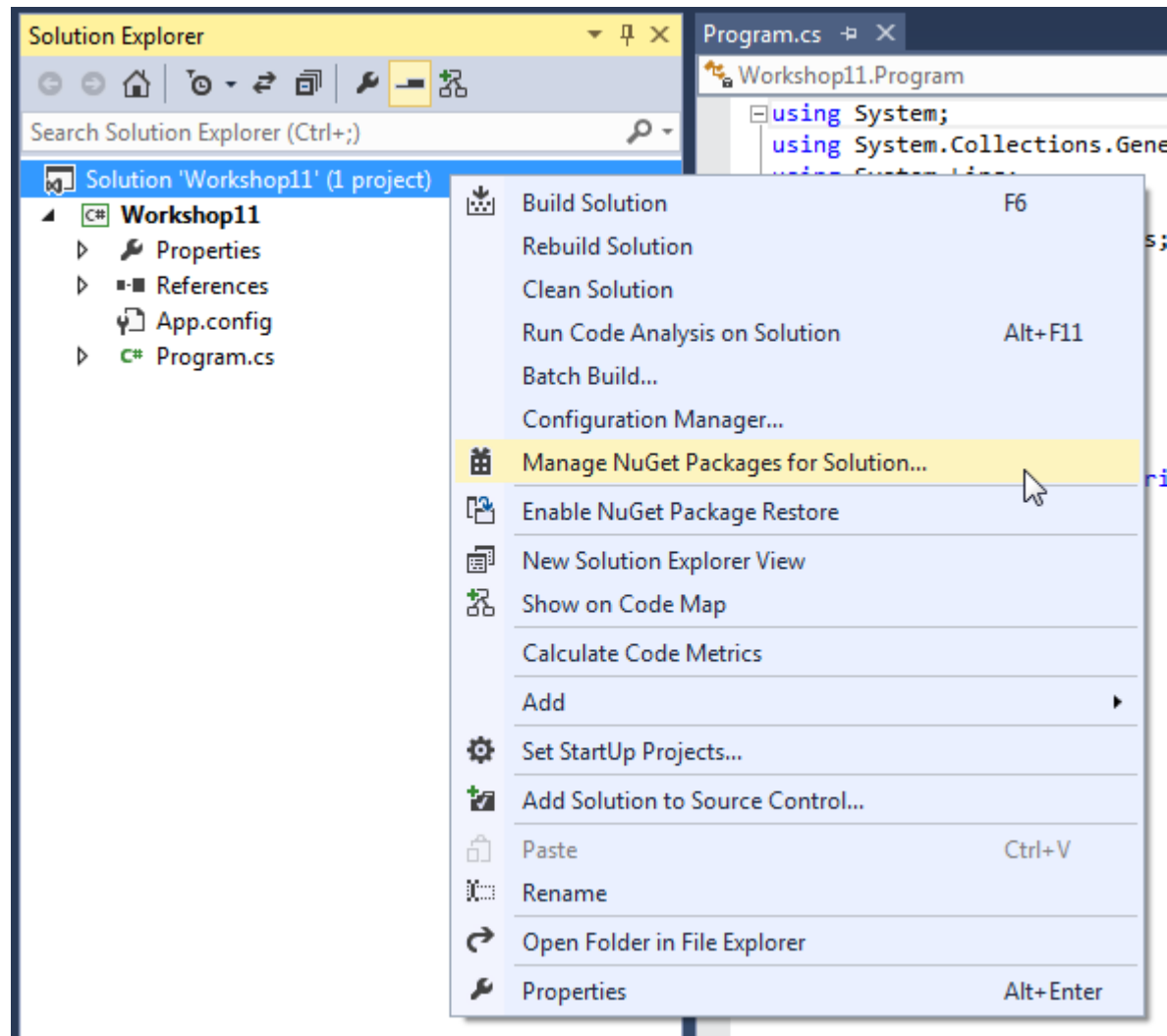
Creating projects



Import required libraries / packages that you will need to run tests, perform code coverage and generate reports.

Testing

Importing required libraries via NuGet



Testing

Importing required libraries via NuGet

The screenshot shows the 'Workshop11.sln - Manage NuGet Packages' window. The left sidebar has 'Online' and 'Search Results' circled in orange. The search bar at the top right contains 'opencover', also circled in orange. The 'Install' button for the first search result is circled in orange. A tooltip is visible over the first search result.

Workshop11.sln - Manage NuGet Packages

Stable Only | Sort by: Relevance | opencover

Online

All

nuget.org

Microsoft .NET

Search Results

Updates

Each package is licensed to you by its owner. Microsoft is not responsible for, nor does it grant any licenses to, third-party packages.

1

Settings | Close

OpenCover - an open source code coverage tool
An open source code coverage tool (branch and sequence point) for all .NET Frameworks 2 and above.

coveralls.net - Coveralls.io uploader for .NET Code Coverage
Coveralls.io uploader for .NET Code Coverage.

OpenCoverToCoberturaConverter
Converts OpenCover reports to Cobertura reports.

ReportGenerator
ReportGenerator converts XML reports generated by OpenCover, PartCover, Visual Studio or NCover into a readable format.

Created by: sawilde
Id: OpenCover
Version: 1.5.3333

License:
[View License](#)

Project Information:
[Report Abuse](#)

Description:
An open source code coverage tool (branch and sequence point) for all .NET Frameworks 2 and above (including Silverlight). Also capable of handling 32 and 64 bit processes. Use ReportGenerator 1.9 for best viewing results (also available via Nuget).

Tags: Code-Coverage Reporting Testing TDD Utility

Dependencies:
No Dependencies

Testing

Importing required libraries via NuGet

The screenshot shows the Visual Studio NuGet Package Manager window for a project named 'Workshop11.sln'. The window is titled 'Workshop11.sln - Manage NuGet Packages'. On the left, there is a sidebar with 'Installed packages' and 'Online' sections. The 'Online' section is expanded to show 'Search Results'. The main area displays a list of packages. The 'ReportGenerator' package is selected, and its 'Install' button is highlighted with an orange circle. The search bar at the top right contains the text 'reportgenerator|'. The details for the 'ReportGenerator' package are shown on the right, including its version (2.1.4.0), last published date (23.3.2015), and a description. The 'OpenCover' package is also visible in the list, marked as installed with a green checkmark.

Workshop11.sln - Manage NuGet Packages

Stable Only Sort by: Relevance

reportgenerator|

ReportGenerator
ReportGenerator converts XML reports generated by OpenCover, PartCover, Visual...

Install

HtmlWarningsReportGenerator
Comand line tool generating html report from xml. It render warnings, errors and adnotations in source code. It can be ea...

OpenCover - an open source code coverage tool for .N... ✓
An open source code coverage tool (branch and sequence point) for all .NET Frameworks 2 and above. Also capable...

Created by: Daniel Palme
Id: ReportGenerator
Version: 2.1.4.0
Last Published: 23.3.2015
Downloads: 71371
License:
[View License](#)
[Project Information](#)
[Report Abuse](#)
Description:
ReportGenerator converts XML reports generated by OpenCover, PartCover, Visual Studio or NCover into a readable reports in various formats. The reports do not only show the coverage quota, but also include the source code and visualize which line has been covered.
Tags: Code Coverage Reporting Testing
Dependencies:
No Dependencies

Each package is licensed to you by its owner. Microsoft is not responsible for, nor does it grant any licenses to, third-party packages.

1

Settings Close

Testing

Importing required libraries via NuGet

The screenshot shows the Visual Studio NuGet Package Manager interface for the solution 'Workshop11.sln'. The 'Online' source is selected, and the search results are sorted by 'Relevance'. The package 'NUnit.Runners' is highlighted, and the 'Install' button is circled in orange. The right-hand pane shows the details for the 'NUnit' package, including its version (2.6.4), last published date (17.12.2014), and a description of the framework.

Workshop11.sln - Manage NuGet Packages

Installed packages

Online

All
nuget.org
Microsoft and .NET
Search Results

Updates

Stable Only Sort by: Relevance

NUnit
NUnit is a unit-testing framework for all .Net languages with a strong TDD focus.

NUnit.Runners
NUnit is a unit-testing framework for all .Net languages with a strong TDD focus. **Install**

NUnit Test Adapter for VS2012, VS2013 and VS2015
A package including the NUnit TestAdapter for Visual Studio 2012/13/15. With this package you don't need to install the V...

SpecFlow.NUnit
Combined package to setup SpecFlow with NUnit v2.6+ easily. For running tests with NUnit runners, use SpecFlow.NUnit.Ru...

SpecFlow.NUnit.Runners
Combined package to setup SpecFlow with NUnit easily for running the tests with the NUnit runners.

NQUnit.NUnit
A sample project to get you up and running with NQUnit using NUnit quickly.

MSBuild.NUnit
MSBuild.NUnit is more flexible than the NUnit support available in MSBuild Community Tasks as it allows for suppo...

Each package is licensed to you by its owner. Microsoft is not responsible for, nor does it grant any licenses to, third-party packages.

1 2 3 4 5 ▶

NUnit

Created by: Charlie Poole
Id: NUnit.Runners
Version: 2.6.4
Last Published: 17.12.2014
Downloads: 299676
License
[View License](#)
Zlib
[Project Information](#)
[Report Abuse](#)
Description:
NUnit features a fluent assert syntax, parameterized, generic and theory tests and is user-extensible. A number of runners, both from the NUnit project and by third parties, are able to execute NUnit tests.

Version 2.6 is the seventh major release of this well-known and well-tested programming tool.

This package includes the NUnit console, gui and pnunit runners and is compatible with all NUnit framework versions 2.0 through 2.6. The NUnit project editor is also included.
Tags: nunit test testing tdd runner

Settings Close

Testing

Importing required libraries via NuGet

The screenshot shows the Visual Studio NuGet Package Manager interface. The window title is "Workshop11.sln - Manage NuGet Packages". The left sidebar shows "Installed packages" and "Online" sources, with "Search Results" selected. The main area displays search results for "NUnit". The "NUnit" package is highlighted, and its "Install" button is circled in orange. The search bar at the top right contains "NUnit". The right pane shows details for the selected package, including "Created by: Charlie Poole", "Id: NUnit", "Version: 2.6.4", "Last Published: 17.12.2014", and "Downloads: 2208127". The description states: "NUnit features a fluent assert syntax, parameterized, generic and theory tests and is user-extensible. A number of runners, both from the NUnit project and by third parties, are able to execute NUnit tests." The bottom of the window has "Settings" and "Close" buttons.

Workshop11.sln - Manage NuGet Packages

Stable Only Sort by: Relevance

NUnit

NUnit is a unit-testing framework for all .Net languages with a strong TDD focus.

Install

NUnit

Created by: Charlie Poole
Id: NUnit
Version: 2.6.4
Last Published: 17.12.2014
Downloads: 2208127
License
View License
Zlib
Project Information
Report Abuse
Description:
NUnit features a fluent assert syntax, parameterized, generic and theory tests and is user-extensible. A number of runners, both from the NUnit project and by third parties, are able to execute NUnit tests.
Version 2.6 is the seventh major release of this well-known and well-tested programming tool.
This package includes only the framework assembly. You will need to install the NUnit.Runners package unless you are using a third-party runner.
Tags: nunit test testing tdd framework fluent assert theory plugin addin

Each package is licensed to you by its owner. Microsoft is not responsible for, nor does it grant any licenses to, third-party packages.

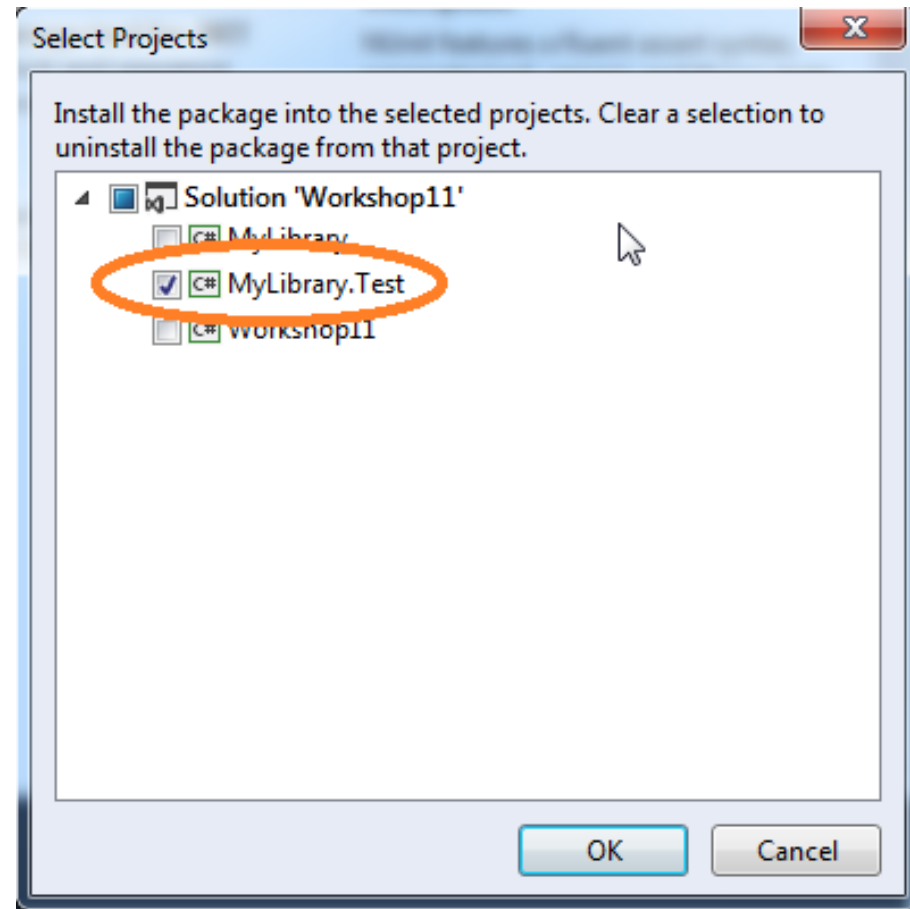
Settings Close

Testing

Reference NUnit

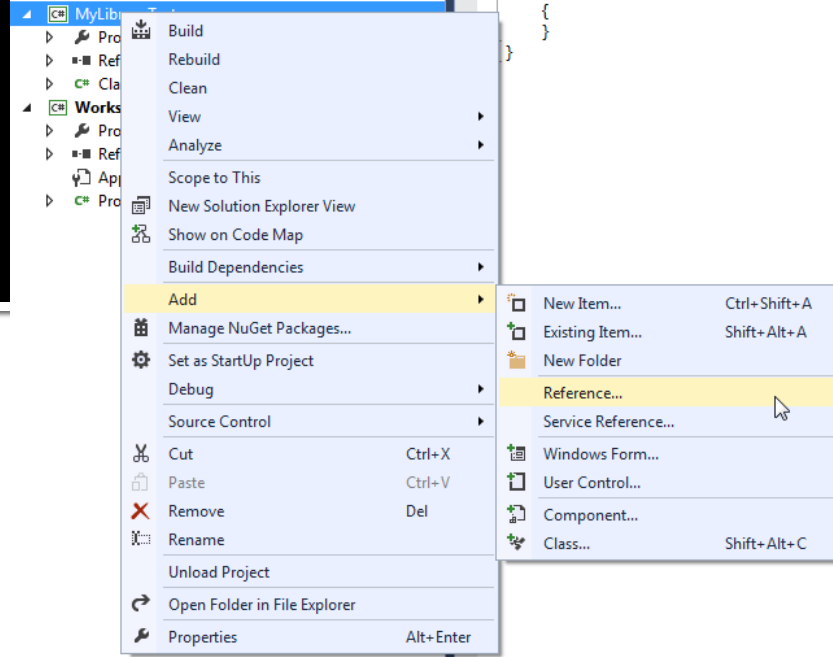
After you install NUnit, you will be asked, which projects will need to use it.

Select your “second” project, that is “one containing TESTs”. We have suffixed its name with “.Test”.



Testing

Add Reference

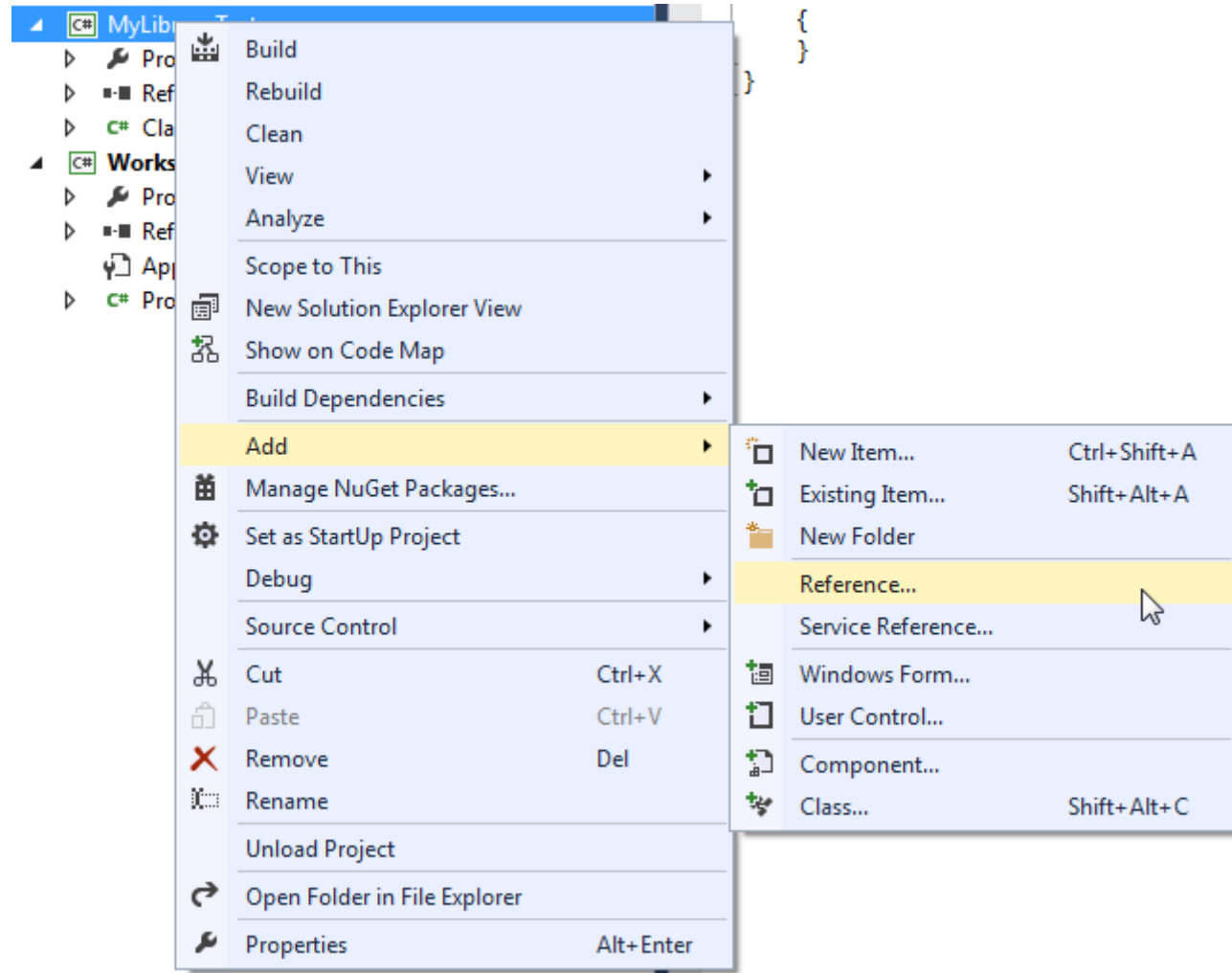


In order to be able to test code that exists in MyLibrary, we have to tell Visual Studio that project MyLibrary.Test references MyLibrary in order to be able to use namespaces from MyLibrary within MyLibrary.Test.

Testing

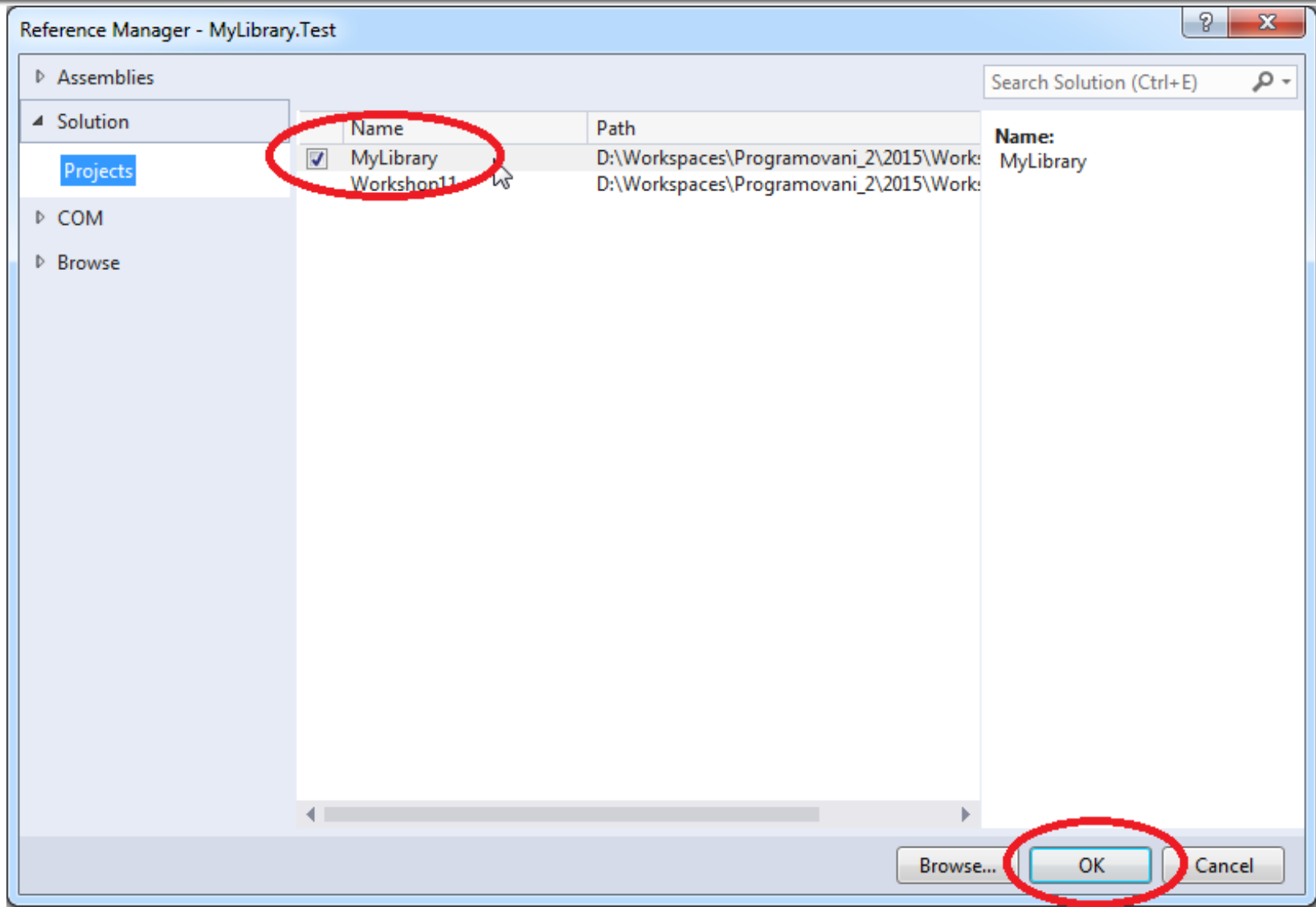
Add Reference to MyLibrary

Right-click
MyLibrary.Test
and navigate
to Add->
Reference.



Testing

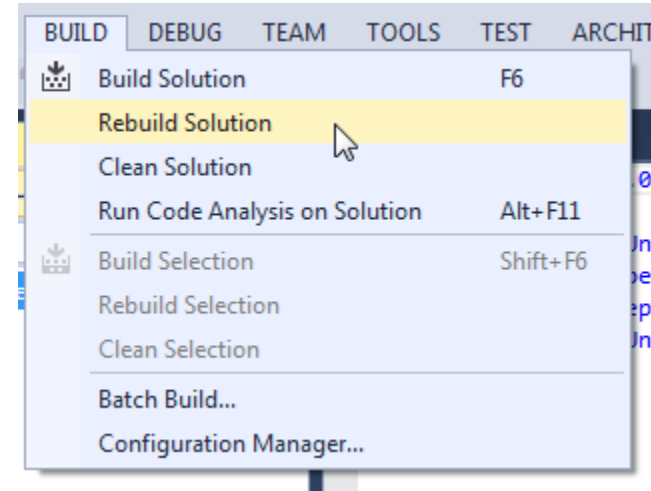
Add Reference to MyLibrary



Testing

Build the solution

When “building” the solution Visual Studio will create .dll files of your libraries. We will need both to perform tests and generate reports about code coverage.

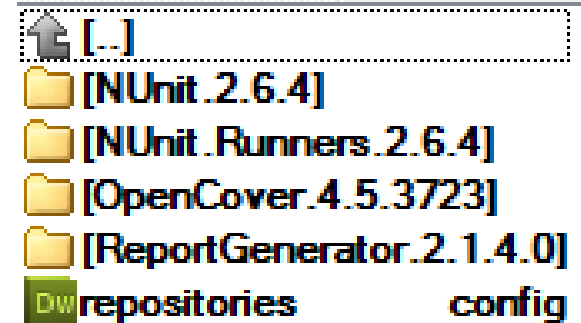


Testing

Running the tests

Navigate to the folder of your solution.
There you will find subdirectory
'packages'.

The content should look like the picture
to the right.



Testing

Running the tests

Now you will need to create 3 batch files that will

1. Run tests
2. Perform code coverage
3. Generate HTML report

Put those batch files into the folder of your solution.

Testing

Running the tests

Batch File 1: test.bat

2 lines

```
del TestResult.xml  
.\packages\NUnit.Runners.2.6.4\tools\nunit-console.exe .\MyLibrary.Test\bin\Debug\MyLibrary.Test.dll /noshadow
```

```
del TestResult.xml
```

```
.\packages\NUnit.Runners.2.6.4\tools\nunit-console.exe  
.\MyLibrary.Test\bin\Debug\MyLibrary.Test.dll /noshadow
```

You might need to adjust texts in red to match your configuration

Explanation: here we're running NUnit that executes code within your ".Test" project producing "TestResult.xml" file with the report.

Testing

Running the tests

Batch File 2: test-cover.bat

2 lines

```
del results.xml  
.\packages\OpenCover.4.5.3723\OpenCover.Console.exe -target:test.bat -register:user -filter:+[MyLibrary]*
```

```
del results.xml
```

```
.\packages\OpenCover.4.5.3723\OpenCover.Console.exe -target:test.bat -  
register:user -filter:+[MyLibrary]*
```

You might need to adjust texts in red to match your configuration

Explanation: here we're running tests again but now under observation of OpenCover that will generate Code Coverage report for namespace "MyLibrary" (that's why you might need to change that...).

Testing

Running the tests

Batch File 3: test-cover-report.bat

3 lines

```
call test-cover.bat
.\packages\ReportGenerator.2.1.4.0\reportgenerator.exe -reports:results.xml -targetdir:coverage
start firefox file://%CD%/coverage/index.htm
```

```
call test-cover.bat
.\packages\ReportGenerator.2.1.4.0\reportgenerator.exe -
    reports:results.xml -targetdir:coverage
start firefox file://%CD%/coverage/index.htm
```

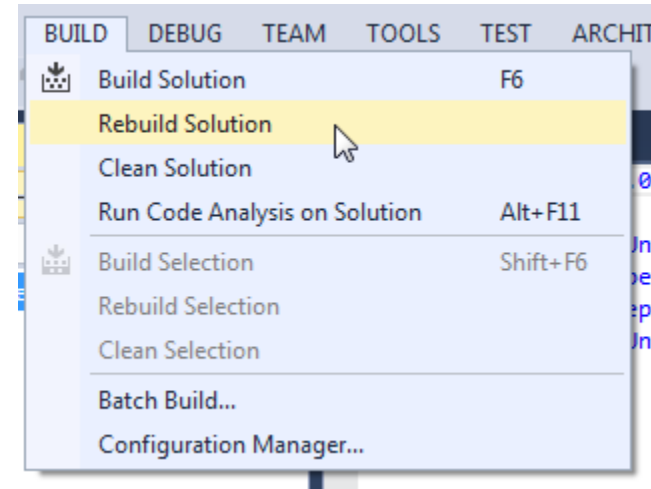
You might need to adjust text in red to match your configuration.

Explanation: Here we run ReportGenerator on the report generated by the OpenCover producing HTML pages visualizing the report.

Testing

Be sure to rebuild the solution

Don't forget to rebuild your solution every time you do any changes to any of your projects!



Extreme Programming

Task – Visualization of Binary Search Tree

- Download the template: <http://altur1.com/vebg2>
 - <http://artemis.ms.mff.cuni.cz/gemrot/lectures/prg2/2015/Workshop11-Homework.zip>
- Code Heap tests to provide complete code coverage!

Assignment 11

Send me an email

- Email: jakub.gemrot@gmail.com
- Subject: **Programming II – 2015 – Assignment 11**
- Zip up the whole solution and send it
- You WILL NOT find the assignment in CoDex!
- Deadline:
 - **10.5.2015 23:59**
- Points: 10 + 3 (meeting the deadline)

Questions?

I sense a soul in search of answers...

- Sadly, I do not own the patent for perfection (and will never do)
- In case of doubts about the assignment or some other problems don't hesitate to contact me!
 - Jakub Gemrot
 - gemrot@gamedev.cuni.cz