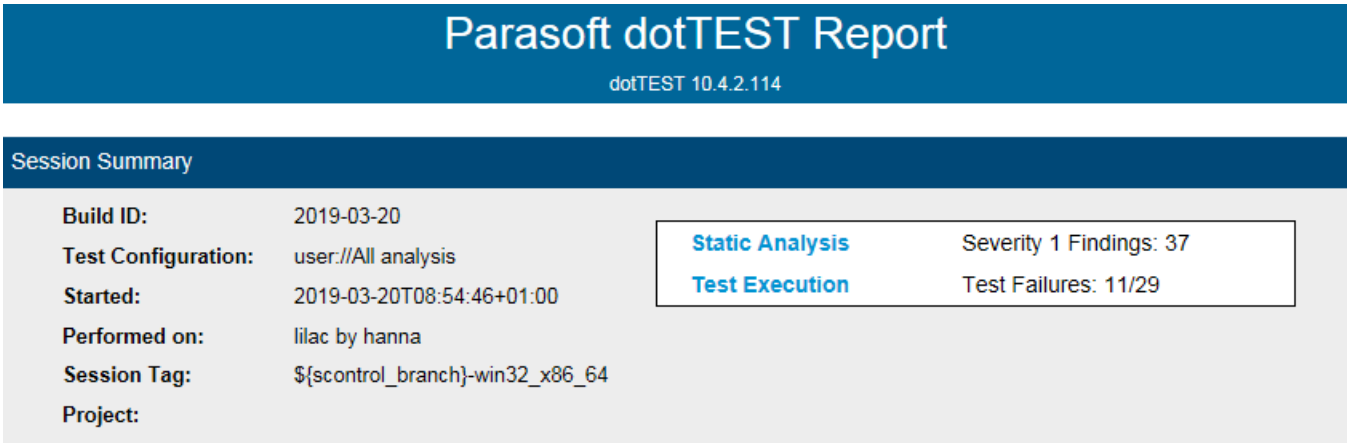
Viewing Reports

Open the report.html or report.pdf file saved to the working directory or location specified with the `-report` switch. Reports may contain different sections depending on the type of analysis, but the following sections are included in all static and flow analysis configurations.

Header

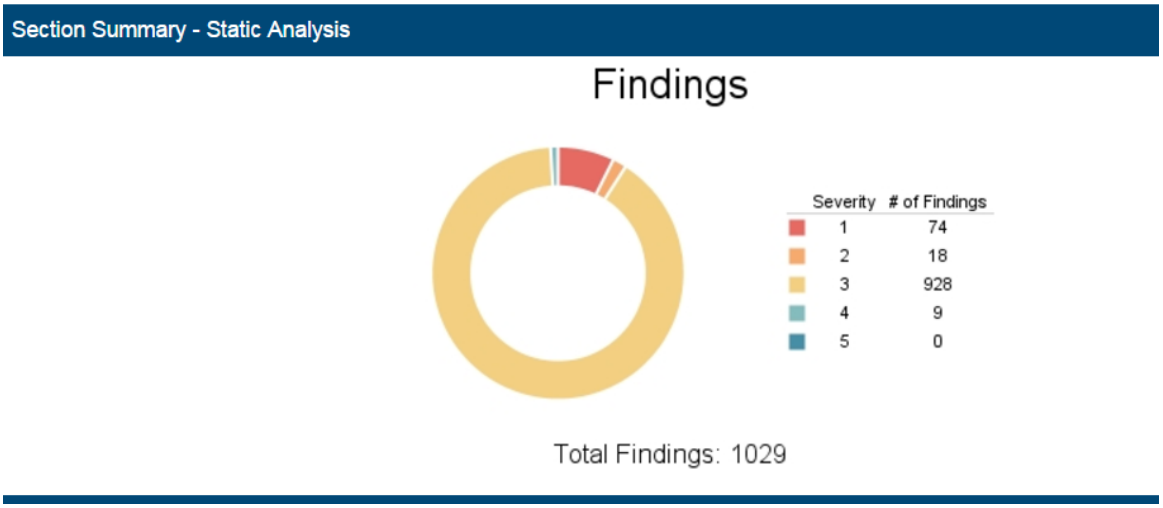


The following information is included:

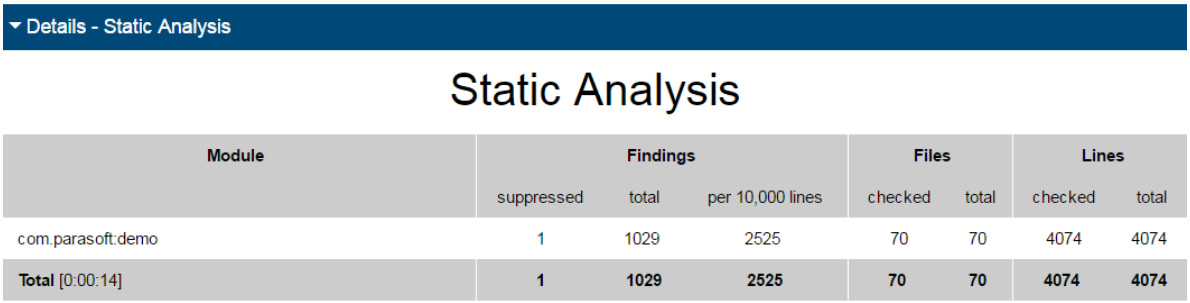
- Tool used for the analysis
- Build ID
- Test configuration
- Time stamp of the analysis
- Machine name and user name
- Session tag
- Project name
- Number of findings with the highest severity
- Number of failed tests

Static Analysis

The first part of the report covers the Static Analysis findings and is divided into two main sections.The first section is a summary which shows an overview of findings displayed as a pie chart. The colors indicate different severity types and their corresponding number of findings detected during static analysis:



The second section shows the details of static analysis findings. It starts with a table which includes static analysis results:



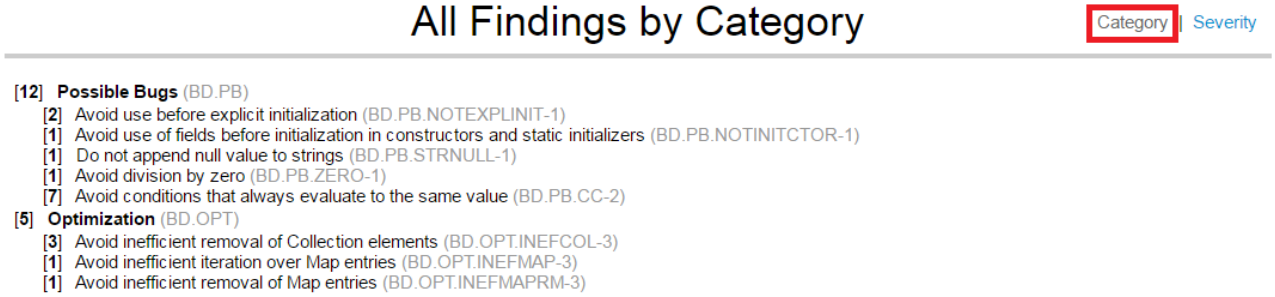
The following information is included:

- Name of module
- Number of suppressed rules
- Total number of findings
- Average number of findings per 10,000 lines
- Number of analyzed files
- Total number of files in the module
- Number of code lines analyzed
- Total number of code lines in the module

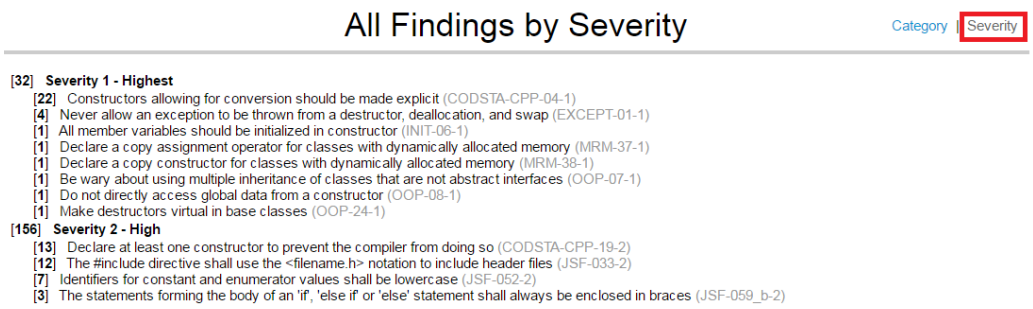
All Findings

The All Findings section displays the details of findings organized by category or severity. Click the **Severity** or **Category** link to toggle between views.

In the Category view, findings are reported by rule and grouped by category. A count of how many times each rule was violated in the scope of analysis is also shown.



In the Severity view, findings are reported and grouped by severity. A count of findings per severity is also included.



You can't toggle between these sections in the PDF versions of the report and they are published separately.

Findings by Author

This section includes a table of authors associated with the analyzed code and a count of findings per each author. Findings are segmented into findings associated with suppressed rules and findings recommended for remediation. Click on an author link to view their finding details.

Findings by Author

Back to Top

Author	Findings		
	suppressed	total	recommended
hanna	0	26	26

hanna Total Findings : 26 Back to Top

BankExample/Parasoft.Dottest.Examples.Bank/Money/Currency.cs
49: Replace equality comparison of floating-point types to comparison which handles floating-point uncertainty. PB.DNCF-2
BankExample/Parasoft.Dottest.Examples.Bank/Money/CurrencyProvider.cs
23: Provide 'default' for 'switch' statement CS.PB.DEFSWITCH-2
38: Provide 'default' for 'switch' statement CS.PB.DEFSWITCH-2

The details view includes the following information:

- File containing the finding and its location
- Violation message and rule
- Flow analysis reports also mark the cause of the violation (C), violation points (P), thrown exceptions (E), and important data flows (!)

Findings by File

You can navigate the analyzed code to the reported findings in the Findings by File section. Each node begins with a value that indicates the total number of findings in the node. The value in brackets shows the number of suppressed rules in the node. You can click nodes marked with a plus sign (+) to expand them. PDF versions of the reports are already fully expanded.{

Findings by File

Expand All Collapse All Back to Top

+ **26 (0) Total (Suppressed)**
  + **26 (0) Parasoft.Dottest.Examples.Bank**
    + **3 (0) AccountNumber.cs**
      Findings: 3
      54: Change Equals() implementation to match a standard pattern. Use: if (obj == null) { return false; } hanna OPU.CPNEQ-1
      54: Change Equals() implementation to match a standard pattern. Use: if (this.GetType() != obj.GetType()) { return false; } hanna OPU.CPTEQ-2
      57: Change bitwise operator to logical operator in method 'Equals'. hanna CS.PB.BITBOOL-2

Metrics Summary

If your test configuration includes metrics analysis, a metrics section will appear in the report. See Metrics Analysis for additional information.

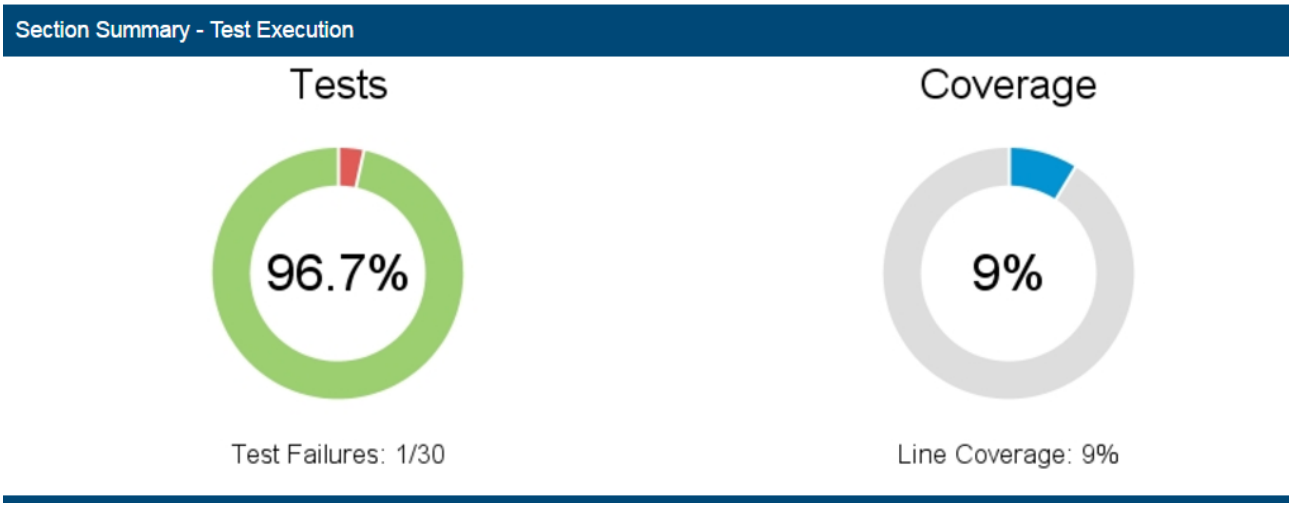
Metrics Summary

Expand All Collapse All Back to Top

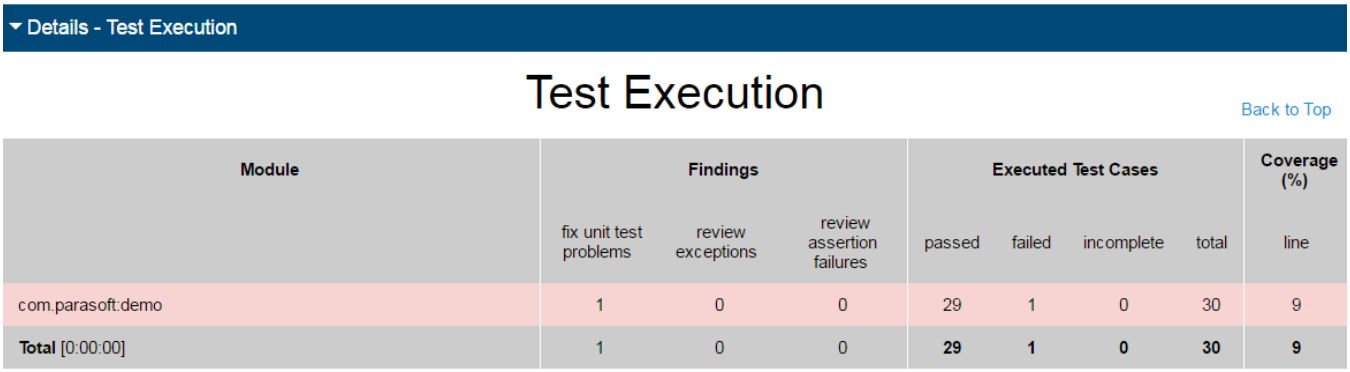
Metric name	Number of items	Average	Std. Deviation	Maximum	Minimum
McCabe Cyclomatic Complexity (METRIC.CC)	**359**	**1.507**	**1.122**	**10**	**1**
demo	359	1.507	1.122	10	1
Nested Blocks Depth (METRIC.NBD)	**359**	**0.301**	**0.711**	**8**	**0**
demo	359	0.301	0.711	8	0
Number of Physical Lines in Files (METRIC.NOPLIF)	**74**	**61.662**	**63.118**	**460**	**11**
demo	74	61.662	63.118	460	11
Number of Source Lines in Methods (METRIC.NOSLIM)	**359**	**7.036**	**5.443**	**40**	**1**
demo	359	7.036	5.443	40	1

Test Execution

The second part of the report covers the Test Execution results and is divided into two sections.The first section is a summary which shows an overview of test failures and coverage displayed as pie charts:



The second section shows the details of test execution. It starts with a table which includes test execution results and coverage information:



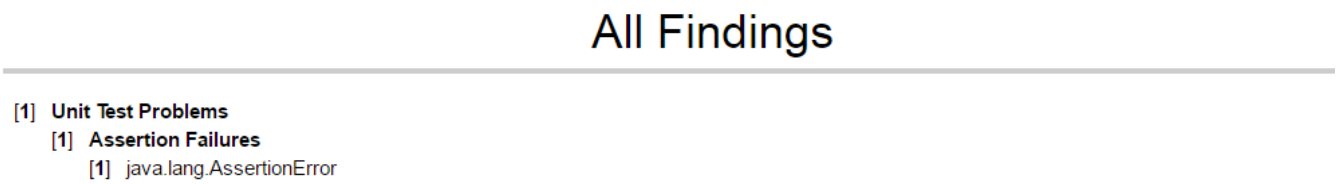
Module	Findings			Executed Test Cases				Coverage (%)
	fix unit test problems	review exceptions	review assertion failures	passed	failed	incomplete	total	line
com.parasoft:demo	1	0	0	29	1	0	30	9
Total [0:00:00]	1	0	0	**29**	**1**	**0**	**30**	**9**

The following information is included:

- Module name
- Number of unit test problems which need to be fixed
- Number of exceptions which need to be reviewed
- Number of assertion failures which need to be reviewed
- Number of unit tests successfully executed
- Number of unit tests failures
- Number of incomplete unit tests
- Total number of unit tests
- Line coverage expressed as percentage

All Findings

The All Findings section displays the details of all unit test problems detected during test execution:



Findings by Author

This section includes a table of authors associated with the analyzed code and shows the total number of findings for each author. Click on an author link to view their finding details.

Findings by Author

Back to Top

Author	Findings	
	total	recommended
hanna	5	5

hanna Total Findings : 5 Back to Top

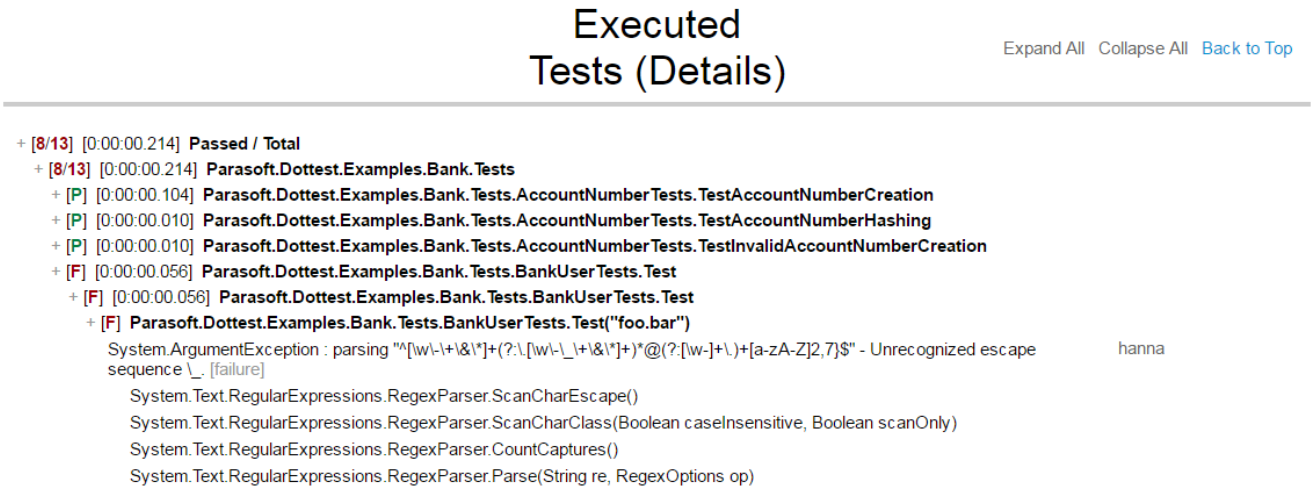
BankExample/Parasoft.Dottest.Examples.Bank/BankUser.cs [5]
Test case: Parasoft.Dottest.Examples.Bank.Tests.BankUserTests.Test("foo.bar")
System.ArgumentException : parsing "^[\w\-\+\&*]+(?:\.[\w\-_\+\&*]+)*@(?:[\w-]+\.)+[a-zA-Z]2,7}$" - Unrecognized escape sequence _. [failure]
System.Text.RegularExpressions.RegexParser.ScanCharEscape()
System.Text.RegularExpressions.RegexParser.ScanCharClass(Boolean caseInsensitive, Boolean scanOnly)

The details view includes the following information:

- Finding location
- Test name
- Failure message

Executed Tests (Details)

You can view the findings in the Executed Tests (Details) section. The nodes where all the test passed are marked with "P" in square brackets. The nodes with test failures begin with a set of values in square brackets. The first value is a count of successfully passed tests and the second indicates the total number of tests executed in the node.The letter "F" indicates the final node where the test failed. You can click nodes marked with a plus sign (+) to expand them.



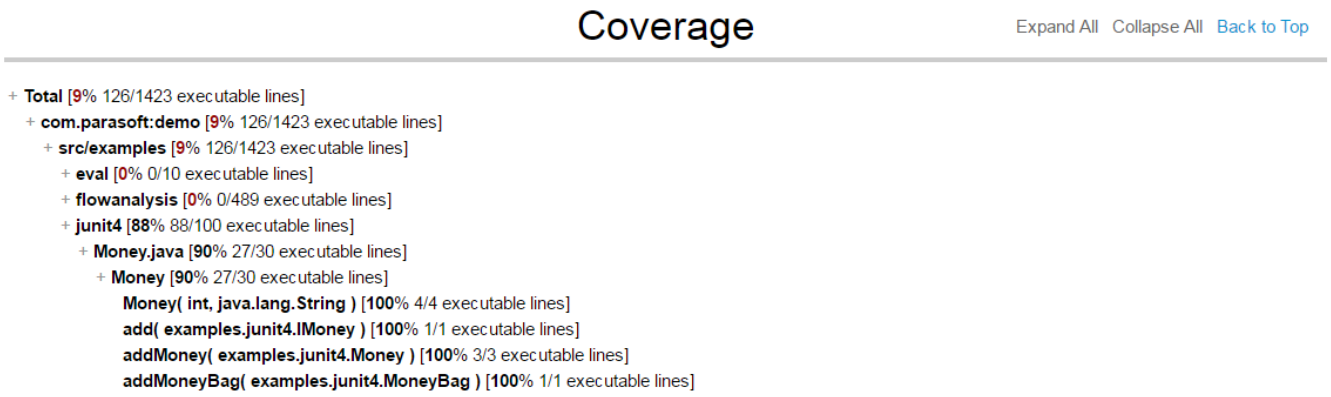
Executed Tests (Details)

Expand All Collapse All Back to Top

+ **[8/13]** [0:00:00.214] **Passed / Total**
  + **[8/13]** [0:00:00.214] **Parasoft.Dottest.Examples.Bank.Tests**
    + **[P]** [0:00:00.104] **Parasoft.Dottest.Examples.Bank.Tests.AccountNumberTests.TestAccountNumberCreation**
    + **[P]** [0:00:00.010] **Parasoft.Dottest.Examples.Bank.Tests.AccountNumberTests.TestAccountNumberHashing**
    + **[P]** [0:00:00.010] **Parasoft.Dottest.Examples.Bank.Tests.AccountNumberTests.TestInvalidAccountNumberCreation**
    + **[F]** [0:00:00.056] **Parasoft.Dottest.Examples.Bank.Tests.BankUserTests.Test**
      + **[F]** [0:00:00.056] **Parasoft.Dottest.Examples.Bank.Tests.BankUserTests.Test**
        + **[F]** **Parasoft.Dottest.Examples.Bank.Tests.BankUserTests.Test("foo.bar")**
          System.ArgumentException : parsing "^[\w\-\+\&*]+(?:\.[\w\-_\+\&*]+)*@(?:[\w-]+\.)+[a-zA-Z]2,7}$" - Unrecognized escape sequence _. [failure] hanna
            System.Text.RegularExpressions.RegexParser.ScanCharEscape()
            System.Text.RegularExpressions.RegexParser.ScanCharClass(Boolean caseInsensitive, Boolean scanOnly)
            System.Text.RegularExpressions.RegexParser.CountCaptures()
            System.Text.RegularExpressions.RegexParser.Parse(String re, RegexOptions op)

Coverage

This section shows the details of coverage collected during the test execution. Each node starts with a set of values. The first value shows coverage expressed as a percentage. The second value is a count of the number of lines in the node which were covered during the test execution. The third value indicates the total number of lines in the node. You can click nodes marked with a plus sign (+) to expand them.



Coverage

Expand All Collapse All Back to Top

+ **Total** [**9**% 126/1423 executable lines]
  + **com.parasoft:demo** [**9**% 126/1423 executable lines]
    + **src/examples** [**9**% 126/1423 executable lines]
      + **eval** [**0**% 0/10 executable lines]
      + **flowanalysis** [**0**% 0/489 executable lines]
      + **junit4** [**88**% 88/100 executable lines]
        + **Money.java** [**90**% 27/30 executable lines]
          + **Money** [**90**% 27/30 executable lines]
            Money(int, java.lang.String) [**100**% 4/4 executable lines]
            add(examples.junit4.IMoney) [**100**% 1/1 executable lines]
            addMoney(examples.junit4.Money) [**100**% 3/3 executable lines]
            addMoneyBag(examples.junit4.MoneyBag) [**100**% 1/1 executable lines]

Test Parameters

The arguments specified during analysis are shown in the Test Parameters section

Test Parameters

dottestcli -solution C:\dotTEST\Examples_10.2.3.137\VS2013\030_NUnit\BankExample\BankExample.sln -config user://Run NUnit Tests with coverage -report C:\dotTEST\Examples_10.2.3.137\VS2013\030_NUnit\020_Run_NUnit_Tests_With_Coverage_Report -out C:\dotTEST\Examples_10.2.3.137\VS2013\030_NUnit\020_Run_NUnit_Tests_With_Coverage_Output.txt