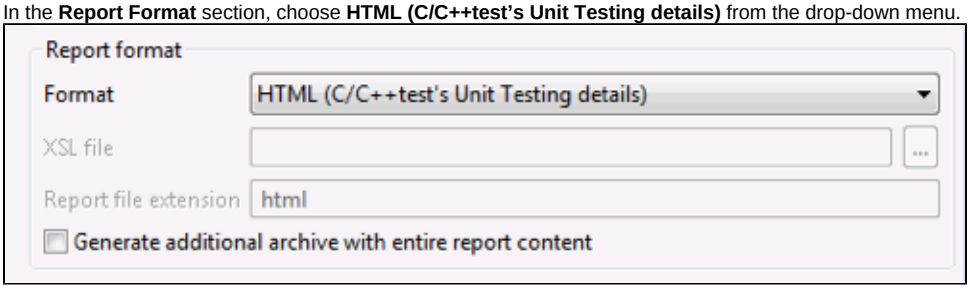
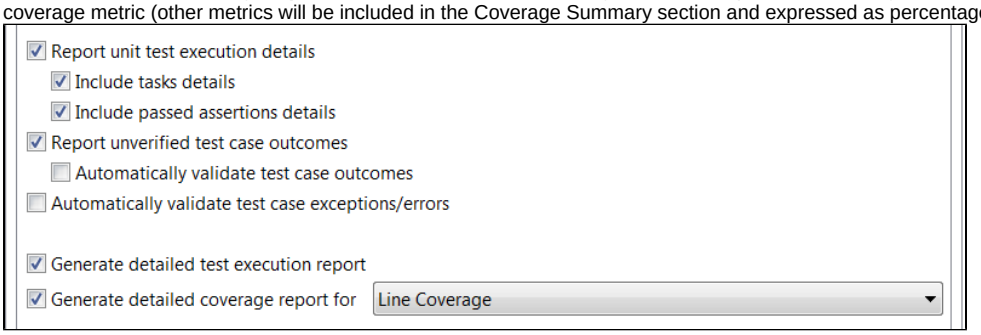
Exercise 11 - Generating Unit Test Reports

1. Choose **Parasoft> Preferences... > Reports**.
2. In the **Report Format** section, choose **HTML (C/C++test's Unit Testing details)** from the drop-down menu.

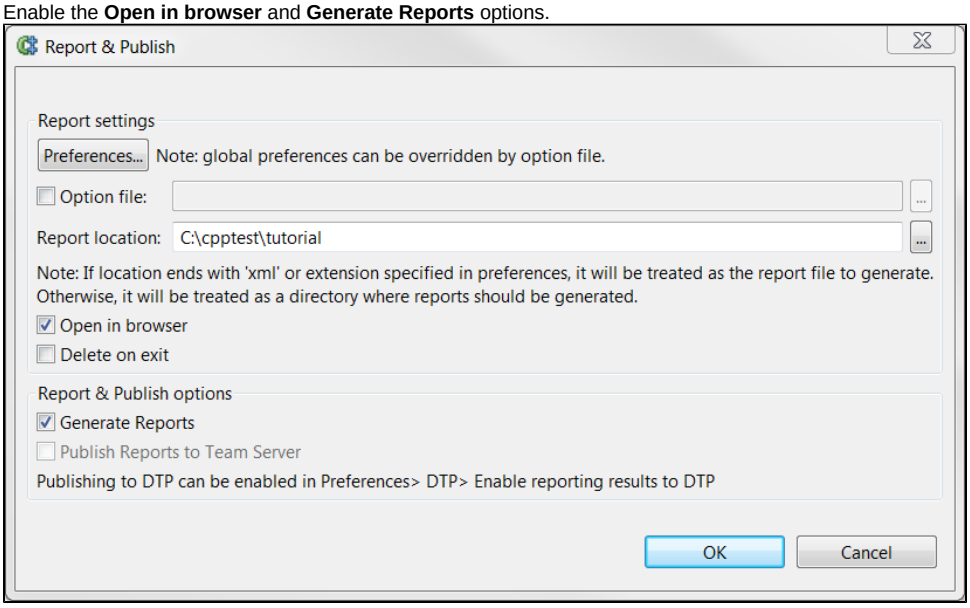


3. Click **Apply**.
4. Select the **ATM** project as the test scope and choose **Parasoft> Test Using> User-Defined> Run Unit Tests (Project Scope)**.
5. Open the **Execution> Runtime** tab and enable the following options:
   - **Report unit test execution details**
   - **Include tasks details**
   - **Include passed assertion details**
   - **Generate detailed test execution report**
   - **Generate detailed coverage report for**; then choose **Line Coverage** from the drop-down menu to generate a Detailed Report for this coverage metric (other metrics will be included in the Coverage Summary section and expressed as percentages)

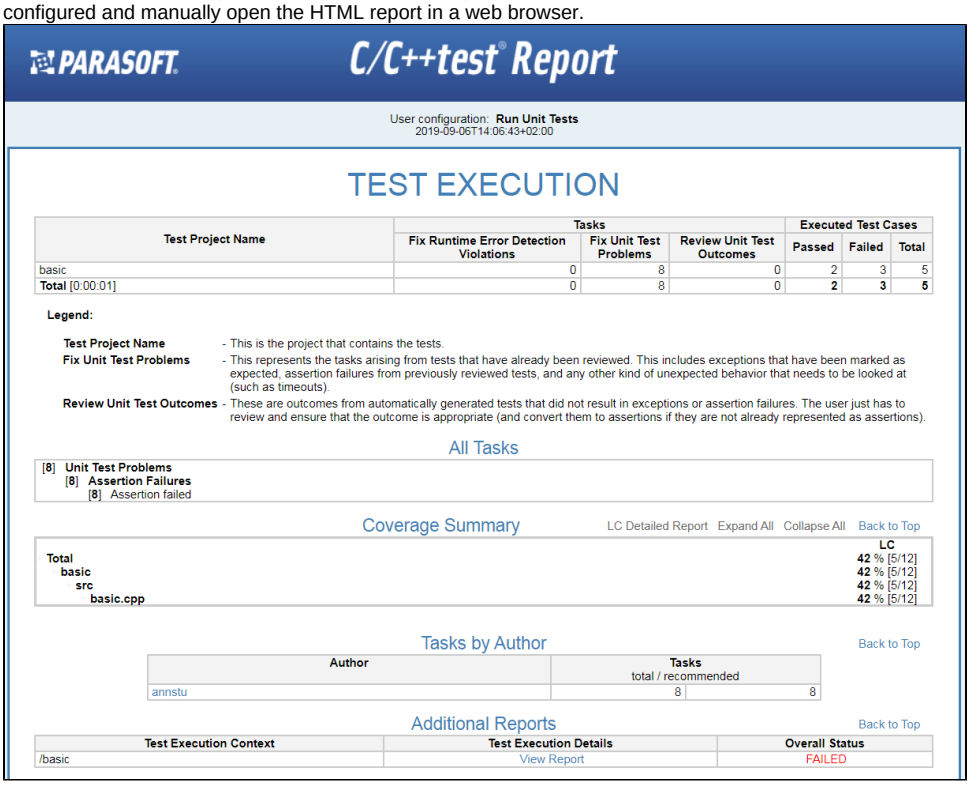


6. Click **Run Test**.
7. Go to the test progress tab (the tab has the same name as the test configuration you're running).
8. Click the **Generate Report** button in the upper right corner of the test progress tab to open the Report & Publish dialog.

9. In the Report & Publish dialog, configure the **Report location** provide the path to the location where the report will be stored.
10. Enable the **Open in browser** and **Generate Reports** options.



11. Click **OK**. A summary report will be generated and opened in the editor window for review. Alternatively, you can go to the report location you configured and manually open the HTML report in a web browser.



PARASOFT C/C++test Report

User configuration: **Run Unit Tests**
2019-09-06T14:06:43+02:00

TEST EXECUTION

Test Project Name	Tasks			Executed Test Cases		
	Fix Runtime Error Detection Violations	Fix Unit Test Problems	Review Unit Test Outcomes	Passed	Failed	Total
basic	0	8	0	2	3	5
Total [0:00:01]	0	8	0	**2**	**3**	**5**

Legend:

Test Project Name - This is the project that contains the tests.
Fix Unit Test Problems - This represents the tasks arising from tests that have already been reviewed. This includes exceptions that have been marked as expected, assertion failures from previously reviewed tests, and any other kind of unexpected behavior that needs to be looked at (such as timeouts).
Review Unit Test Outcomes - These are outcomes from automatically generated tests that did not result in exceptions or assertion failures. The user just has to review and ensure that the outcome is appropriate (and convert them to assertions if they are not already represented as assertions).

All Tasks

[8] **Unit Test Problems**
[8] **Assertion Failures**
[8] Assertion failed

Coverage Summary

LC Detailed Report Expand All Collapse All Back to Top

	LC
Total	**42** % [5/12]
basic	**42** % [5/12]
src	**42** % [5/12]
basic.cpp	**42** % [5/12]

Tasks by Author

Back to Top

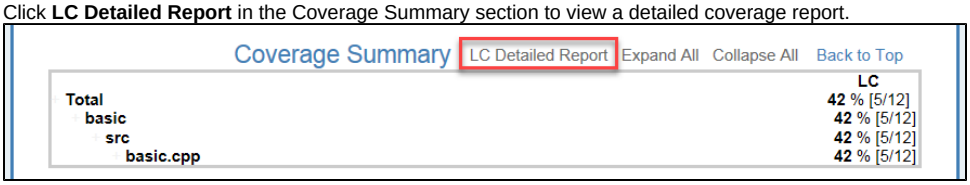
Author	Tasks total	/ recommended
annstu	8	8

Additional Reports

Back to Top

Test Execution Context	Test Execution Details	Overall Status
/basic	View Report	FAILED

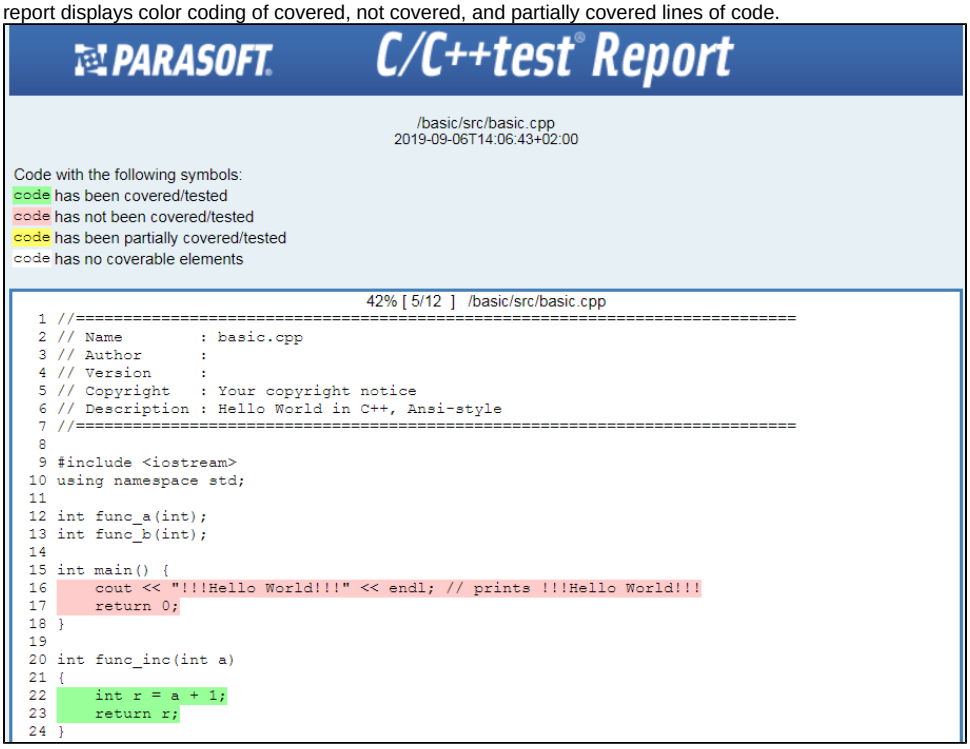
12. Click on a user name in the Tasks by Author section to view a detailed report for that user.
13. Click **LC Detailed Report** in the Coverage Summary section to view a detailed coverage report.



Coverage Summary | LC Detailed Report | Expand All | Collapse All | Back to Top

	LC
Total	**42** % [5/12]
basic	**42** % [5/12]
src	**42** % [5/12]
basic.cpp	**42** % [5/12]

14. Review the coverage information. You can use the tree in the left panel to navigate to the code of interest by clicking on a filename. The detailed report displays color coding of covered, not covered, and partially covered lines of code.



PARASOFT C/C++test Report

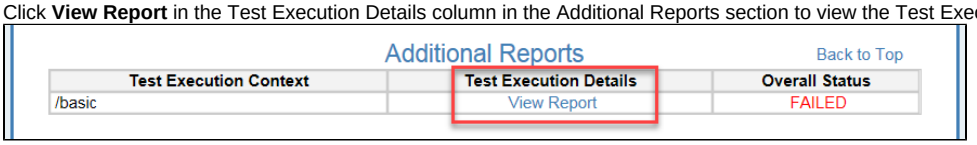
/basic/src/basic.cpp
2019-09-06T14:06:43+02:00

Code with the following symbols:
code has been covered/tested
code has not been covered/tested
code has been partially covered/tested
code has no coverable elements

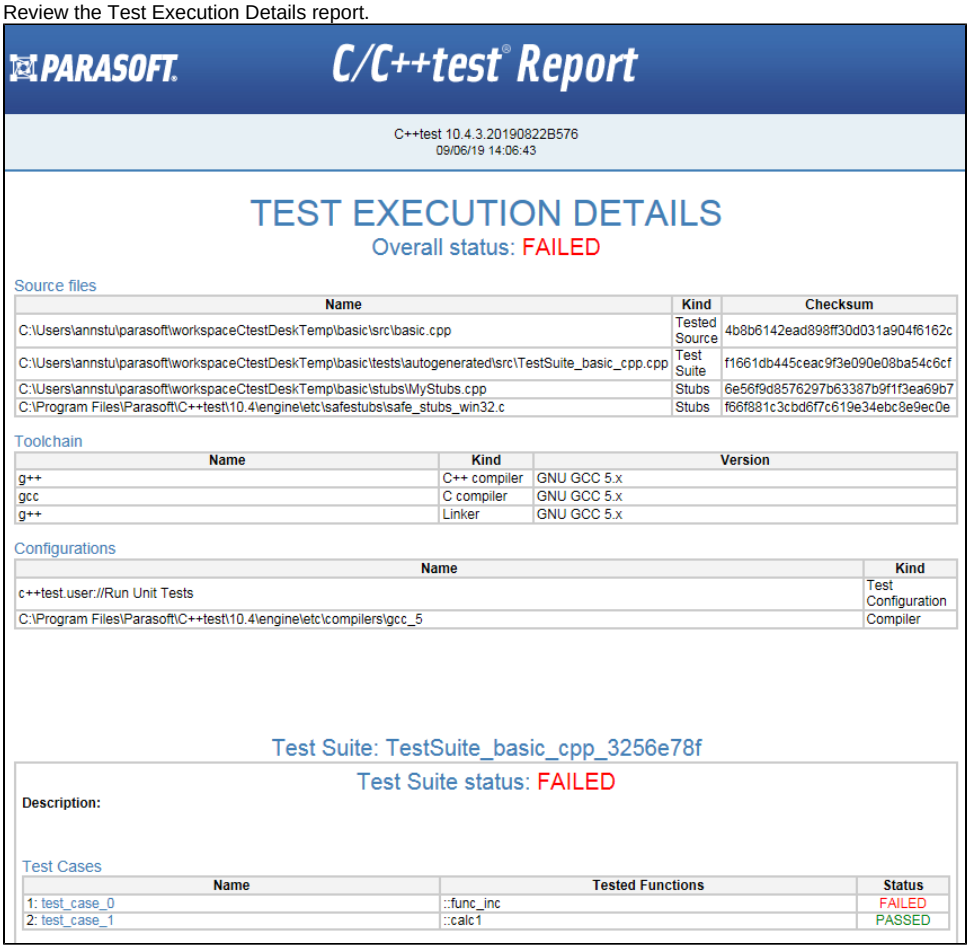
42% [5/12] /basic/src/basic.cpp

```
 1 //============================================================================
 2 // Name        : basic.cpp
 3 // Author      :
 4 // Version     :
 5 // Copyright   : Your copyright notice
 6 // Description : Hello World in C++, Ansi-style
 7 //============================================================================
 8
 9 #include <iostream>
10 using namespace std;
11
12 int func_a(int);
13 int func_b(int);
14
15 int main() {
16     cout << "!!!Hello World!!!" << endl; // prints !!!Hello World!!!
17     return 0;
18 }
19
20 int func_inc(int a)
21 {
22     int r = a + 1;
23     return r;
24 }
```

15. Click **View Report** in the Test Execution Details column in the Additional Reports section to view the Test Execution Details report.



16. Review the Test Execution Details report.



PARASOFT C/C++test® Report

C++test 10.4.3.20190822B576
09/06/19 14:06:43

TEST EXECUTION DETAILS

Overall status: FAILED

Source files

Name	Kind	Checksum
C:\Users\annstu\parasoft\workspaceCtestDeskTemp\basic\src\basic.cpp	Tested Source	4b8b6142ead898ff30d031a904f6162c
C:\Users\annstu\parasoft\workspaceCtestDeskTemp\basic\tests\autogenerated\src\TestSuite_basic_cpp.cpp	Test Suite	f1661db445ceac9f3e090e08ba54c6cf
C:\Users\annstu\parasoft\workspaceCtestDeskTemp\basic\stubs\MyStubs.cpp	Stubs	6e56f9d8576297b63387b9f1f3ea69b7
C:\Program Files\Parasoft\C++test\10.4\engine\etc\safestubs\safe_stubs_win32.c	Stubs	f66f881c3cbd6f7c619e34ebc8e9ec0e

Toolchain

Name	Kind	Version
g++	C++ compiler	GNU GCC 5.x
gcc	C compiler	GNU GCC 5.x
g++	Linker	GNU GCC 5.x

Configurations

Name	Kind
c++test.user://Run Unit Tests	Test Configuration
C:\Program Files\Parasoft\C++test\10.4\engine\etc\compilers\gcc_5	Compiler

Test Suite: TestSuite_basic_cpp_3256e78f

Test Suite status: FAILED

Description:

Test Cases

Name	Tested Functions	Status
1: test_case_0	::func_inc	FAILED
2: test_case_1	::calc1	PASSED