froglogic

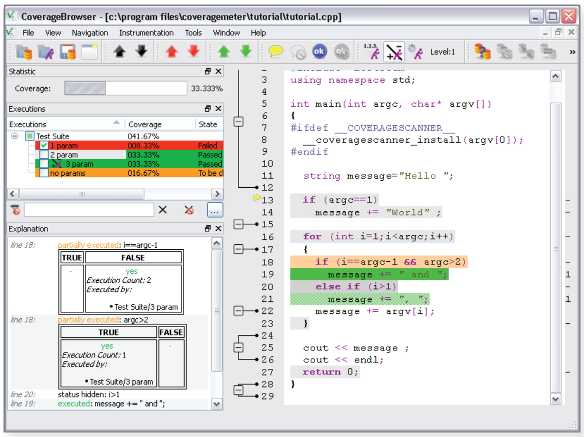
Squish Coco

Code Coverage

A complete, cross-platform code coverage tool chain for C and C++ applications.

Squish Coco provides wrappers for standard compilers to build instrumented versions of applications – with no source code changes necessary. Executing a test suite against an instrumented application produces data that Squish Coco can analyze. This analysis can be used to ensure that complete test coverage is achieved since it:

- ☑ Finds untested code sections.
- ☑ Finds redundant tests (i.e., tests that merely duplicate others).
- ☑ Finds dead code (i.e., code that is never executed).
- ☑ Computes the optimal order of test execution that will maximize the overall test coverage.
- ☑ Compares the test coverage of two applications (e.g., two versions of an application) to identify the differences in test coverage – this makes it straightforward to add new tests to the new version to maintain the level of coverage.



Coverage: 33.333%

Executions	Coverage	State
Test Suite	041.67%	
☑ 1 param	008.33%	Failed
□ 2 param	033.33%	Passed
□ 3 param	033.33%	Passed
□ no params	016.67%	To be c

line 18: partially executed: i==argc-1

TRUE	FALSE
-	yes Execution Count: 2 Executed by: • Test Suite/3 param

line 18: partially executed: argc>2

TRUE	FALSE
yes Execution Count: 1 Executed by: • Test Suite/3 param	-

line 20: status hidden: i>1

line 19: executed: message += " and ";

```
 3  using namespace std;
 4
 5  int main(int argc, char* argv[])
 6  {
 7  #ifdef __COVERAGESCANNER__
 8    __coveragescanner_install(argv[0]);
 9  #endif
10
11    string message="Hello ";
12
13    if (argc==1)                          -
14      message += "World" ;                -
15
16    for (int i=1;i<argc;i++)              -
17    {
18      if (i==argc-1 && argc>2)            -
19        message += " and ";               1
20      else if (i>1)                       -
21        message += ", ";                  1
22      message += argv[i];
23    }                                     -
24
25    cout << message ;
26    cout << endl;
27    return 0;                             -
28  }
29
```

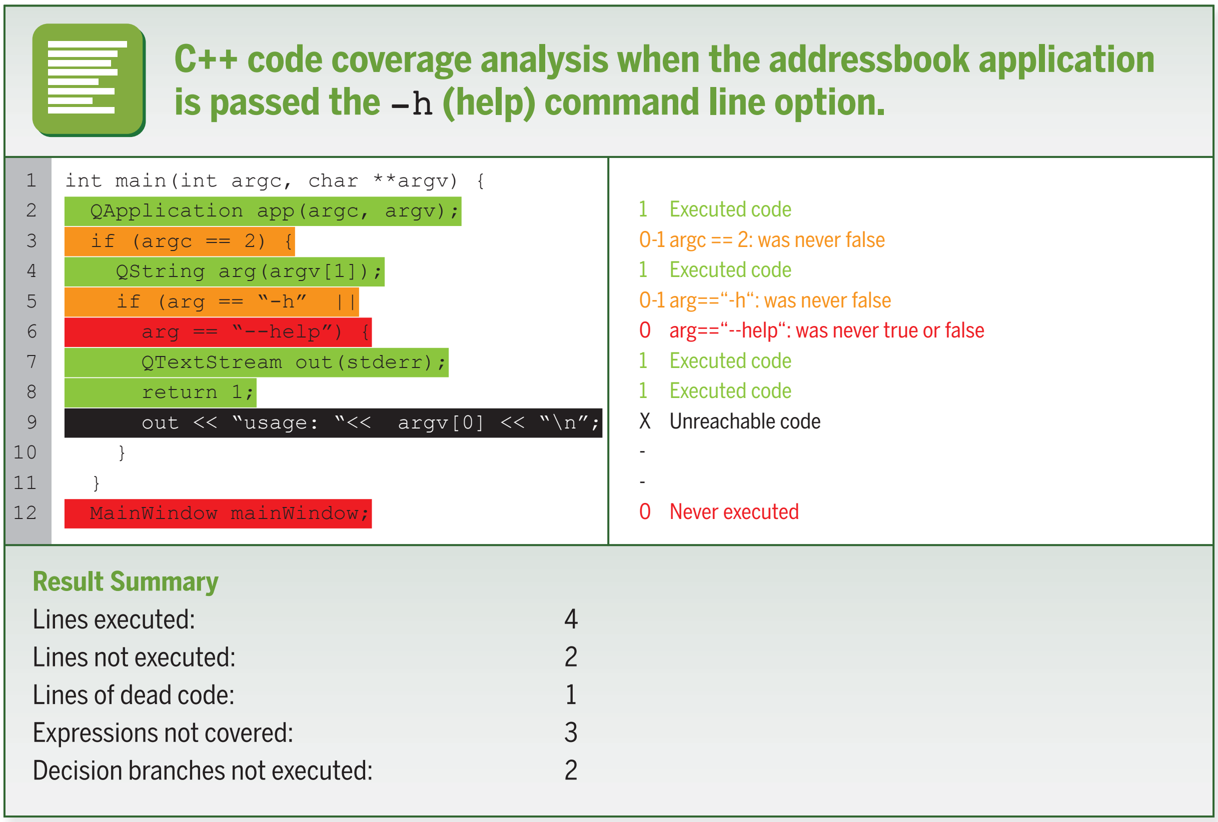
> *“Squish Coco is a test engineer’s equivalent to a programmer’s debugger, only instead of helping to **understand** bugs, Squish Coco helps to **reveal** bugs by highlighting untested statements.”*
>
> Sébastien Fricker, Lead Developer

Supported Platforms:

- Windows (32- and 64-bit)
- Linux (32- and 64-bit)
- Mac OS X (32- and 64-bit)
- Embedded Operating Systems
- UNIX UNIX (Solaris, AIX, …)

In addition, Squish Coco can be made available and supported on other platforms and integrated with custom tool chains.

Squish Coco can be used at every stage of testing and with every testing method (unit tests, automated tests, manual tests, etc.). Furthermore, Squish Coco can merge multiple execution reports to provide advanced analysis.



C++ code coverage analysis when the addressbook application is passed the `-h` (help) command line option.

```
int main(int argc, char **argv) {
  QApplication app(argc, argv);
  if (argc == 2) {
    QString arg(argv[1]);
    if (arg == “-h”  ||
      arg == “--help”) {
      QTextStream out(stderr);
      return 1;
      out << “usage: “<<  argv[0] << “\n”;
    }
  }
  MainWindow mainWindow;
```

1 Executed code
0-1 argc == 2: was never false
1 Executed code
0-1 arg==“-h“: was never false
0 arg==“--help“: was never true or false
1 Executed code
1 Executed code
X Unreachable code
-
-
0 Never executed

Result Summary

Lines executed:	4
Lines not executed:	2
Lines of dead code:	1
Expressions not covered:	3
Decision branches not executed:	2

The Squish Coco Tools:

- **CoverageScanner** this analyzes and instruments C and C++ applications.
- **CoverageBrowser** a sophisticated GUI tool which displays and manages coverage analysis data and results. (Available to commercial licensees only.)
- **Microsoft® Visual Studio Add-in** this generates code coverage configurations for all available C and C++ projects directly inside the Visual Studio IDE.

Supported Coverage Levels:

☑ Function coverage

☑ Line coverage

☑ Decision coverage

☑ Condition coverage

☑ Condition/decision coverage

More Information:

http://www.froglogic.com/squish/coco
coco@froglogic.com