Testing using Google Test in C++

Overview

In this tutorial, we will create some tests in C++ using the Google Test framework, which supports out-of-the-box JUnit reports.

Requirements

- cmake
- Google Test

Description

This example is based on the code from https://github.com/snikulov/google-test-examples, although native samples provided by Google test frameworks could also be used (as of Google test release 1.8.0).

Below are three tests that validate the array sorting function.

test/cpp_sorter_test.cpp

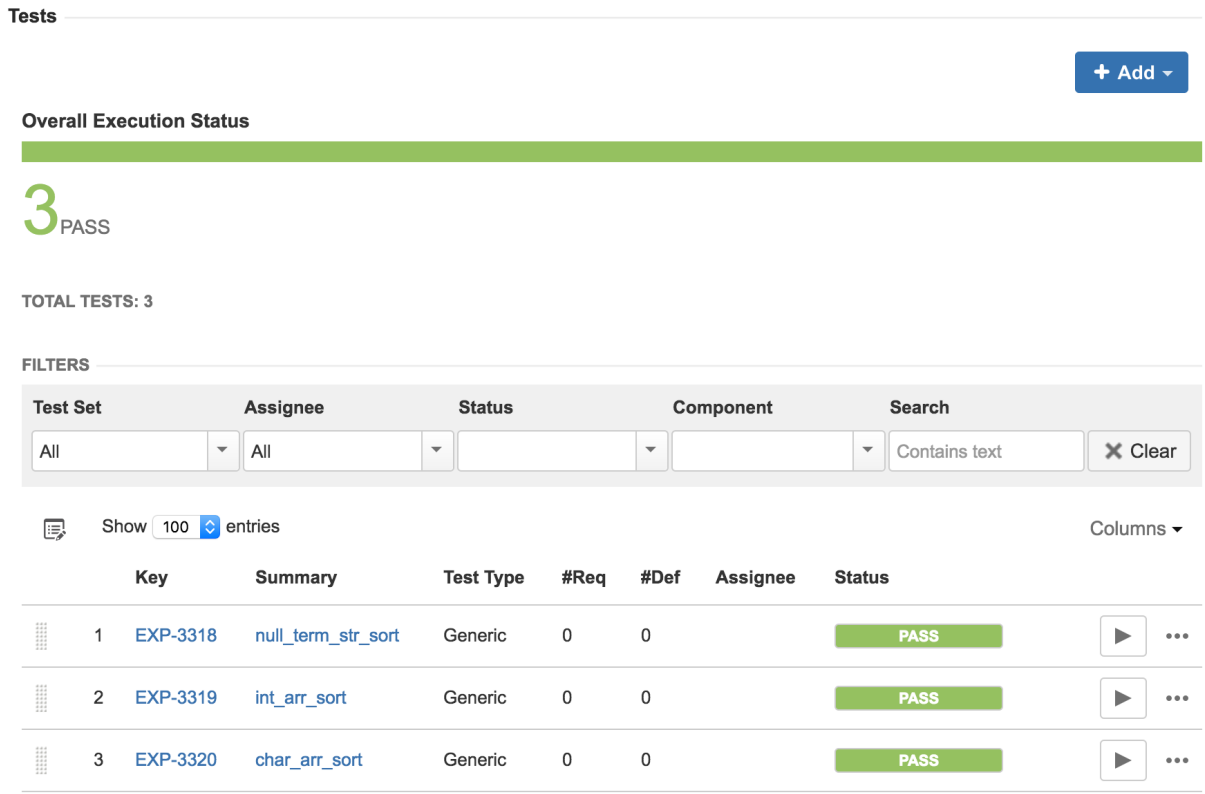
```
#include <algorithm>
#include "cpp_sorter.h"
#include "gtest/gtest.h"
TEST(cpp_sorter_test, null_term_str_sort)
{
    char arr[] = "abcdefghab";
    char eq[]  = "aabbcdefgh";
    int sz = sizeof(arr)/sizeof(arr[0]) - 1; // we need it, to avoid terminating \0 in "" definition case
    array_sort(arr, sz);
    for(int i=0; i<sz; i++)
        EXPECT_EQ(arr[i], eq[i]);
}
TEST(cpp_sorter_test, char_arr_sort)
{
    char arr[] = {'a','b','c','d','e','f','g','h','a','b'};
    char eq[]  = {'a','a','b','b','c','d','e','f','g','h'};
    int sz = sizeof(arr)/sizeof(arr[0]);
    array_sort(arr, sz);
    for(int i=0; i<sz; i++)
        EXPECT_EQ(arr[i], eq[i]);
}
TEST(cpp_sorter_test, int_arr_sort)
{
    int arr[] = {9,8,7,6,5,4,3,2,1,0};
    int eq[]  = {0,1,2,3,4,5,6,7,8,9};
    int sz = sizeof(arr)/sizeof(arr[0]);
    array_sort(arr, sz);
    for(int i=0; i<sz; i++)
        EXPECT_EQ(arr[i], eq[i]);
}
```

In order to run the tests, we need to compile the code first.

```
cd google-test-examples
mkdir build
cd build
cmake ..
cmake --build .
```

After running the tests and generating the JUnit XML reports (e.g., test_detail.xml), it can be imported to Xray (either by the REST API or through the **Import Execution Results** action within the Test Execution).

```
./google-test-examples_test  --gtest_output=xml
```



JUnit's Test Case is mapped to a Generic Test in Jira, and the **Generic Test Definition** field contains the "name of the test case" concatenated with the "name of the test within the test case".

> **About Google Test's nomenclature**
>
> As stated by Google Test,
>
> `TEST( )` arguments go from general to specific. The *first* argument is the name of the test case, and the *second* argument is the test's name within the test case. Both names must be valid C++ identifiers, and they should not contain underscore (_). A test's *full name* consists of its containing test case and its individual name. Tests from different test cases can have the same individual name.

The Execution Details of the Generic Test contains information about the Test Suite, which in this case corresponds to Google Test's "test case name".

Execution Details

Test Description

None

Test Details

Test Type:	Generic
Definition:	cpp_sorter_test.int_arr_sort

Results

Context	Error Message	Duration	Status
TestSuite cpp_sorter_test	–	0 millisec	PASS

References

- https://github.com/google/googletest
- https://github.com/google/googletest/blob/master/googletest/README.md
- https://github.com/google/googletest/blob/master/googletest/docs/Primer.md
- https://github.com/snikulov/google-test-examples
- https://github.com/google/googletest/blob/master/googletest/docs/Samples.md