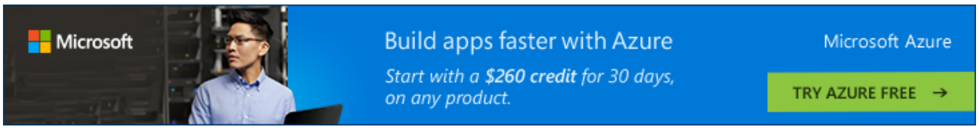
Google Test: Parameterized tests which use an existing test fixture class?

Ask Question


Microsoft
Build apps faster with Azure
Start with a $260 credit for 30 days, on any product.
Microsoft Azure
TRY AZURE FREE →

I have a test fixture class which is currently used by many tests.

```
#include <gtest/gtest.h>
class MyFixtureTest : public ::testing::Test {
  void SetUp() { ... }
};
```

I would like to create a parameterized test which also uses all that MyFixtureTest has to offer, without needing to change all my existing tests.

How do I do that?

I have found similar discussions on the web, but have not fully understood their answers.

c++ unit-testing googletest

edited Jun 30 '10 at 18:52

asked Jun 30 '10 at 18:40
des4maisons
723 3 11 19

3 Answers

The problem is that for regular tests your fixture has to be derived from testing::Test and for parameterized tests, it has to be derived from testing::TestWithParam<>.

In order to accommodate that, you'll have to modify your fixture class in order to work with your parameter type

```
template <class T> class MyFixtureBase : public T {
  void SetUp() { ... };
  // Put the rest of your original MyFixtureTest here.
};

// This will work with your non-parameterized tests.
class MyFixtureTest : public MyFixtureBase<testing::Test> {};

// This will be the fixture for all your parameterized tests.
// Just substitute the actual type of your parameters for MyParameterType.
class MyParamFixtureTest : public MyFixtureBase<
    testing::TestWithParam<MyParameterType> > {};
```

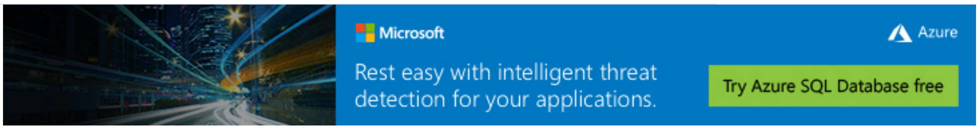
This way you can keep all your existing tests intact while creating parameterized tests using

```
TEST_P(MyParamFixtureTest, MyTestName) { ... }
```

edited Jul 15 '10 at 18:22

answered Jul 14 '10 at 21:55
VladLosev
4,226 1 16 34

Wow, thanks a lot! – des4maisons Jul 15 '10 at 14:22


Microsoft
Azure
Rest easy with intelligent threat detection for your applications.
Try Azure SQL Database free

This question is now answered in the Google Test documentation at https://github.com/google/googletest/blob/master/googletest/docs/AdvancedGuide.md#how-to-write-value-parameterized-tests (the answer from VladLosev is technically correct, but perhaps slightly more work)

Specifically, when you want to add parameters to a pre-existing fixture class, you can do

```
class MyFixtureTest : public ::testing::Test {
  ...
};
class MyParamFixtureTest : public MyFixtureTest,
                           public ::testing::WithParamInterface<MyParameterType>
{
```

```
TEST_P(MyParamFixtureTest, MyTestName) { ... }
```

edited May 23 '17 at 11:46
Community♦
1 1

answered Jan 14 '14 at 16:34
mabraham
1,259 17 20

2 This definitely seems cleaner and less work compared to what VladLosev proposed (which is technically correct and perhaps from a time when the ParamInterface was not available) – sanchitarora Jan 9 '15 at 20:41

Don't forget that if you define `SetUp()` in `MyParamFixtureTest` it should call `MyFixtureTest::SetUp()`, otherwise that setup of the base class will never run. – DBedrenko Jan 2 at 15:45

If you create a new fixture that is derived from this common one and than create your parameterized tests on that derived class - would that help you and solve your problem?

From Google Test wiki page: "In Google Test, you share a fixture among test cases by putting the shared logic in a base test fixture, then deriving from that base a separate fixture for each test case that wants to use this common logic."

answered Jul 2 '10 at 4:51
ratkok
622 8 13

Thanks for the answer. My issue is that a parameterized test fixture needs to inherit from ::testing::TestWithParam, whereas my current fixture only inherits from ::testing::Test. Also, tests which use parameterized tests are declared with TEST_P instead of TEST_F. Any ideas how to resolve the differences? – des4maisons Jul 14 '10 at 19:34