Organizing Your Tests Using Catch

Dror Helper
AUTHOR TITLE

@dhelper blog.drorhelper.com

Overview

Naming your tests

Organizing tests with names and tags

Running Catch from the command line

```
TEST_CASE(“This is a test name”)
{
...
}
```

Reminder: Tests Names in Catch

Declared using *TEST_CASE* macro

Free formed strings

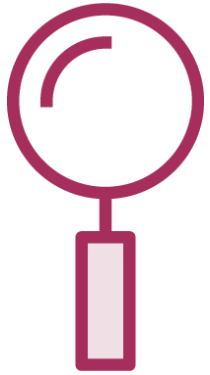
Why Do We Need a Good Test Name?

Understand

What the test does

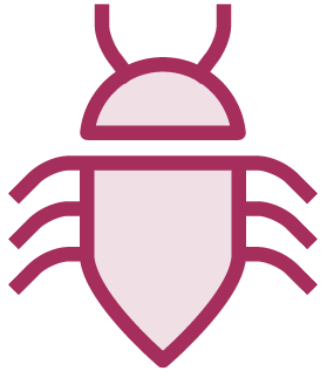
Execute

A subset of your tests



Find

The test you need

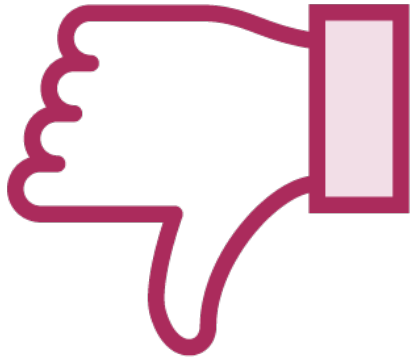


Failures

Root cause analysis

A good test name will help you write a good test

Avoid at All Costs



“Test 1_11_37”

“Customer Test Simple”

“WorkItem1234”

“Workload error exception”

Good Names

Express a specific requirement/behavior

Should include:

- The starting state
- Given input
- Expected Result
- Unless irrelevant

Should be easily found

Should not include “Test”

If Age > 18 IsAdult returns true

When called with xyz then return true

Should throw exception When invalid user

Given authenticated When invalid number used
Then transaction will fail

Called with empty list --> return nullptr

Catch & the Command Line

Catch tests are compiled into a console app (exe)

Use arguments to specify which test(s) to run T9Predict

Run Specific Test

thisTest

"This Test"

Run Group of tests

Customer*

*Customer

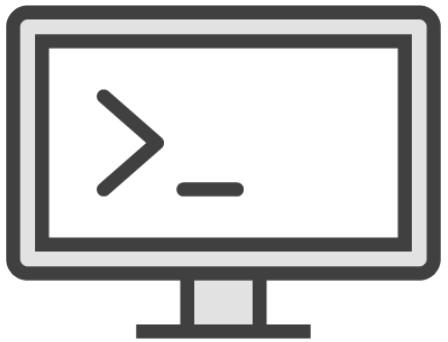
Run all tests except

Exclude:otherTest

~otherTest

~*other*

a* ~ab* abc



Test Name Related Arguments

Demo

Using catch command line arguments

- List all test names
- Execute tests by test names

```
TEST_CASE("This is a test name", "[tag1][tag2]")
{
...
}
```

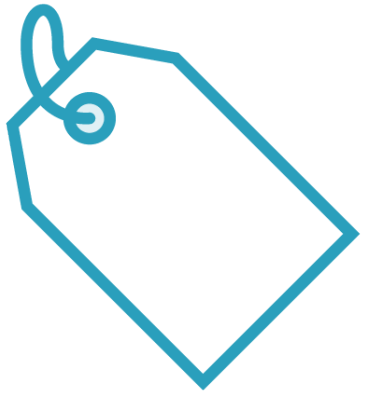
Tags

A simple way to group tests

Additional strings associated with test case

- Must begin with an alpha numeric character

Tag names are not case sensitive



Use Tags to Categorize tests

- By code under test (DB, Model, Customer)
- By test type (Unit, Integration, Scenario)
- By execution speed (Slow, Fast)

MyTests.exe “[A]”

MyTests.exe “[B]”

MyTests.exe ~[A]

MyTests.exe “[A][B]”

MyTests.exe “[A],[B]”

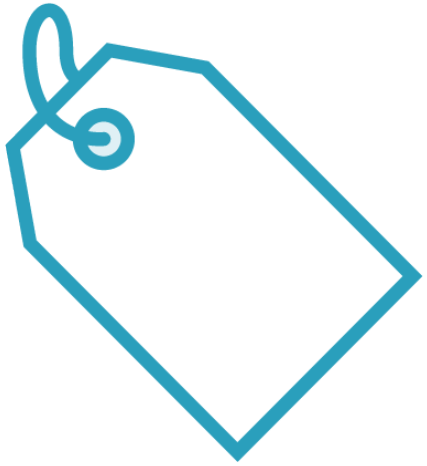
```
TEST_CASE(“Test1”, “[A]”){. . .}

TEST_CASE(“Test2”, “[A]”){. . .}

TEST_CASE(“Test3”, “[B]”){. . .}

TEST_CASE(“Test4”, “[A][B]”){. . .}
```

Special Tags



[!hide] or [.]	Skip/Ignore test
[.][integration] → [.integration]	

[!throws]	Exclude if run catch with -e or --nothrow
[!shouldfail]	Reverse failing logic (pass if fail)
[!mayfail]	Does not fail the test if assertion fails

[#<filename>]	Use -# to specify filename as tag

Demo

Using Tags to organize tests

- List all tags
- Execute tests by tags
- Execute tests by file

Create Tag Alias

```
CATCH_REGISTER_TAG_ALIAS( "[@abc]", "[a],[b]~[c]" )
```

More Command Line Arguments

-h, -?, --help

-f, --input file <filename>

-o, --out <filename>

-b, --break

-a, --abort

-x, --abortx [<threshold>]

Summary

Good test name == good test

Naming your tests

Grouping tests using Tags

- Special Tags

Using command line arguments