

OMERO testing framework

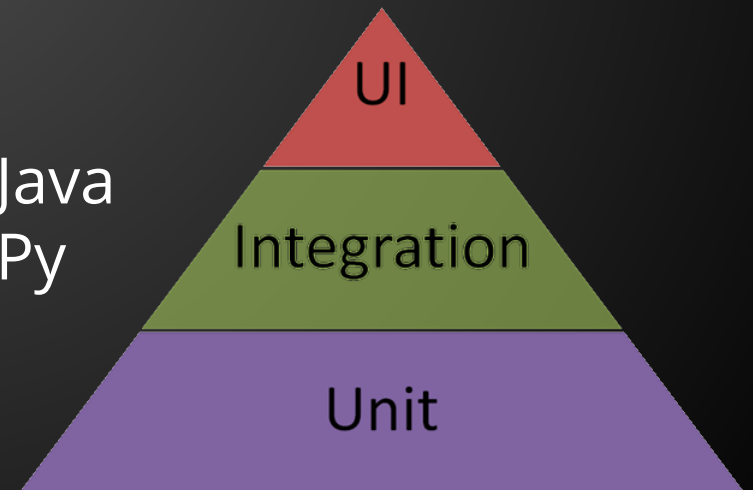
Status and future plans

Agenda

- ❑ Overview
- ❑ Running tests
- ❑ TestNG (Java)
- ❑ py.test (Python)
- ❑ Future plans

Overview

- ❑ Unit and integration tests
- ❑ Python and Java tests (and C++, Matlab, JS...)
- ❑ Start by looking at
 - ❑ components/tools/OmeroJava
 - ❑ components/tools/OmeroPy
 - ❑ components/<name>/test



Running tests

- ❑ `build.py` targets
 - ❑ Unit tests
 - ❑ `test-unit`
 - ❑ `-f components/<component>/build.xml`
`test`
 - ❑ Integration tests
 - ❑ `test-integration`
 - ❑ `-f components/<component>/build.xml`
`integration`
 - ❑ Remember to set `ICE_CONFIG`

See [developers/testing.html](#)

Running tests - cont.

- ❑ Jenkins jobs (**-integration-**)
- ❑ Travis builds for unit tests
 - ❑ Time limit excludes integration tests

		OMERO-5.0-merge-integration	3 hr 6 min - #235
		OMERO-5.0-merge-integration-broken	1 hr 32 min - #187
		OMERO-5.0-merge-integration-java	1 hr 32 min - #193
		OMERO-5.0-merge-integration-python	1 hr 32 min - #198
		OMERO-5.0-merge-integration-web	1 hr 32 min - #18

TestNG (Java)

- ❑ Driven by a suite-defining XML file
- ❑ Used for both unit and integration tests
- ❑ Issues independent of framework
 - ❑ No easy way to mock server services
 - ❑ No way of using an in-memory DB
 - ❑ Race conditions/timeout on remote calls
(LockTimeout)
- ❑ Eclipse plug-in (w/ICE_CONFIG)



See testng.org

TestNG (Java) - Omerojava

- ❑ Tests based use annotations
- ❑ Test class is still a class, so can extend `test/integration/AbstractServerTest`

```
@Test(expectedExceptions = ApiUsageException.class)
public void testSlice0Rows() throws Exception;
```

```
@Test
public void testSlice1Rows() throws Exception;
```

TestNG (Java) - example

❑ Data provider

```
@DataProvider(name = "testStream")
public static Object[][] testStreamProvider() {
    return new Object[][] {
        {streamFromFilename(TEST_FILE)}
    };
}
```

❑ and consumer

```
@Test(dataProvider = "testStream")
public void testCtor(RandomAccessInputStream stream) {
    ...
}
```


py.test

- ❑ unittest (PyUnit)
 - ❑ standard library, but limited
 - ❑ some 3rd party extensions
- ❑ py.test
 - ❑ easy to use - uses native assert
 - ❑ tests discovered by naming convention
 - ❑ fixtures/teardown - decorators & unittest-style
 - ❑ parametrizable - tests and fixtures
 - ❑ skip, xfail and other markers
 - ❑ extendible - plug-ins
 - ❑ runs unittest tests - migration
 - ❑ (too) many ways to do the same thing!

See pytest.org

Writing tests using py.test (1)

```
# test_example1.py
import test.integration.library as lib
class TestExample1(lib.ITest):
    def setup_method(self, method):
        lib.ITest.setup_method(self, method)
        # ...

    # @pytest.fixture(autouse=True)
    # def setup(self):

    def teardown_method(self, method):
        # ...
        lib.ITest.teardown_method(self, method)

    def testOne(self, arg1, arg2):
        assert arg1 is not None
        assert arg1 != arg2
```

Writing tests using py.test (2)

```
import pytest

class TestExample2(object):

    @pytest.mark.xfail(reason="See ticket #12345")
    def testFailing(self):
        assert False, "with a reason for failure"

    @pytest.mark.long_running
    def testLong(self):
        # ...

    @pytest.mark.skipif(sys.platform != 'darwin', reason="Mac-only test")
    @pytest.mark.parametrize("count", range(10))
    @pytest.mark.parametrize("arg", [1, 4, 9, 16])
    def testMacOnly(self, arg, count):
        with pytest.raises(Exception):
            self.somethingThatShouldFail(arg)
```

Running tests directly

- ❑ `./setup.py test -s path/to/tests`
 - ❑ used by top-level builds
 - ❑ can set `PYTHONPATH` & `ICE_CONFIG`
 - ❑ passes some `py.test` options (obscures one!)
- ❑ `py.test path/to/tests`
 - ❑ `PYTHONPATH` & `ICE_CONFIG` must be set
 - ❑ many more options available

```
py.test -v -s -m "not long_running" --runxfail test/integration
```

```
py.test --pastebin failed test/integration/test_admin.py
```

```
py.test -k Group --collect-only test/integration
```

Future plans

- ❑ Java vs Python
 - ❑ Systematic tests
 - ❑ Gateway-specific tests
 - ❑ Bug fix tests
- ❑ Location of Java tests
 - ❑ Omerojava, components, elsewhere?
- ❑ Tidying, duplication
- ❑ Broken tests
- ❑ Unit tests and Travis
- ❑ Questions & comments