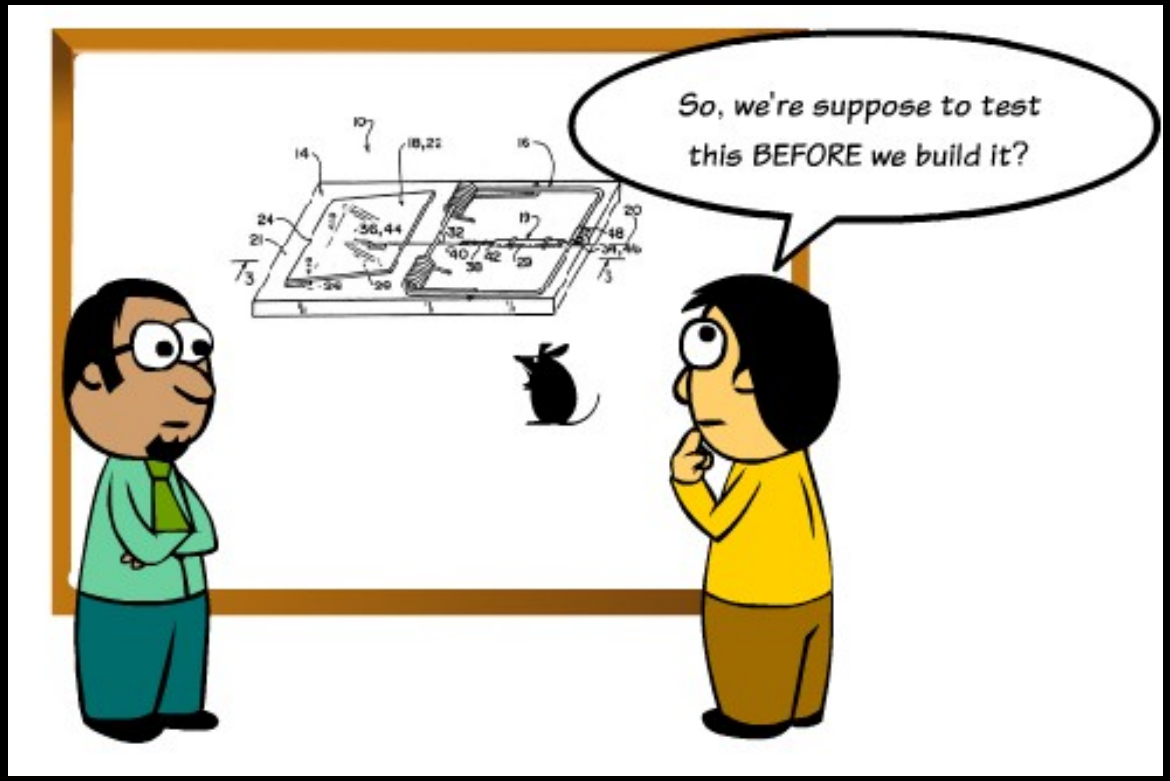
Experiences with Test Driven Development (TDD)



Ole Nielsen

How the customer explained it

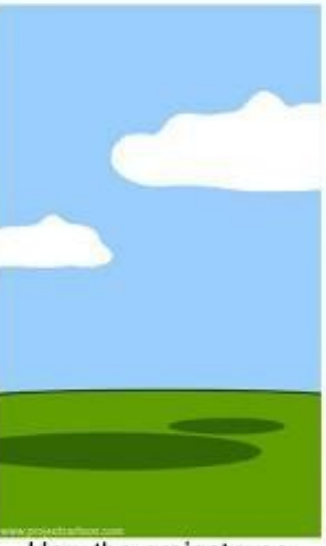
How the project leader understood it

How the analyst designed it

How the programmer wrote it

What the beta testers received

How the business consultant described it



How the project was documented

What operations installed

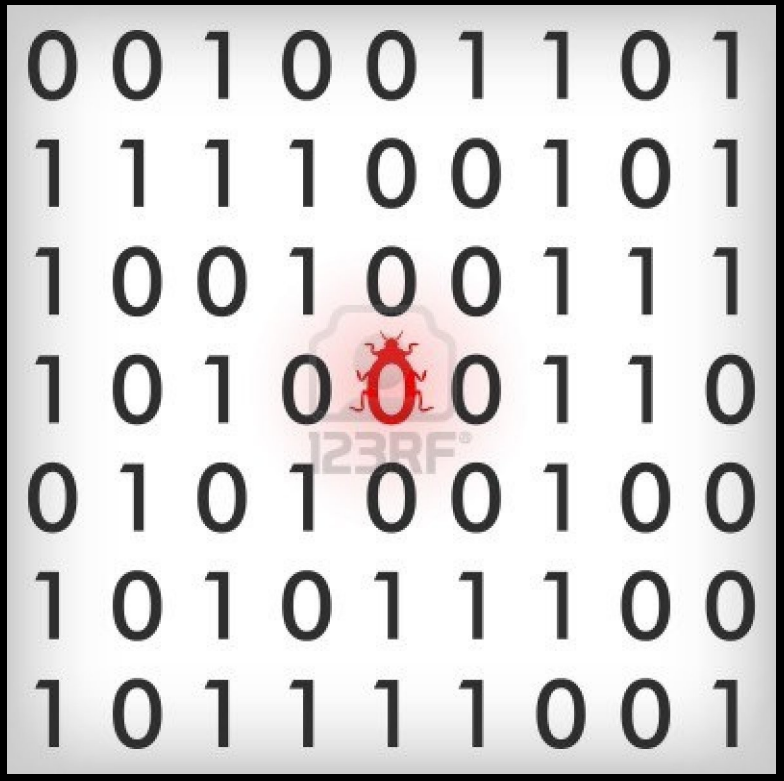
How the customer was billed

How it was supported

What marketing advertised

What the customer really needed


ERROR


0 0 1 0 0 1 1 0 1
1 1 1 1 0 0 1 0 1
1 0 0 1 0 0 1 1 1
1 0 1 0 0 0 1 1 0
0 1 0 1 0 0 1 0 0
1 0 1 0 1 1 1 0 0
1 0 1 1 1 1 0 0 1

- What is TDD
- Why is it important
- Examples
- Reflections
- Continuous Integration
- Other important practices

What is Test Driven Development

- Repetition of very short development cycle
- Test first → minimal code → refactor → deploy
- Maintaining a growing suite of tests
- Large number of individual tests enforce requirements
- Known as unit-, regression-, system- or integration-testing
- Integral to Agile and Xtreme programming
  - But universally applicable

Think of code and test as one

When writing the code, think of the test.
When writing the test, think of the code.

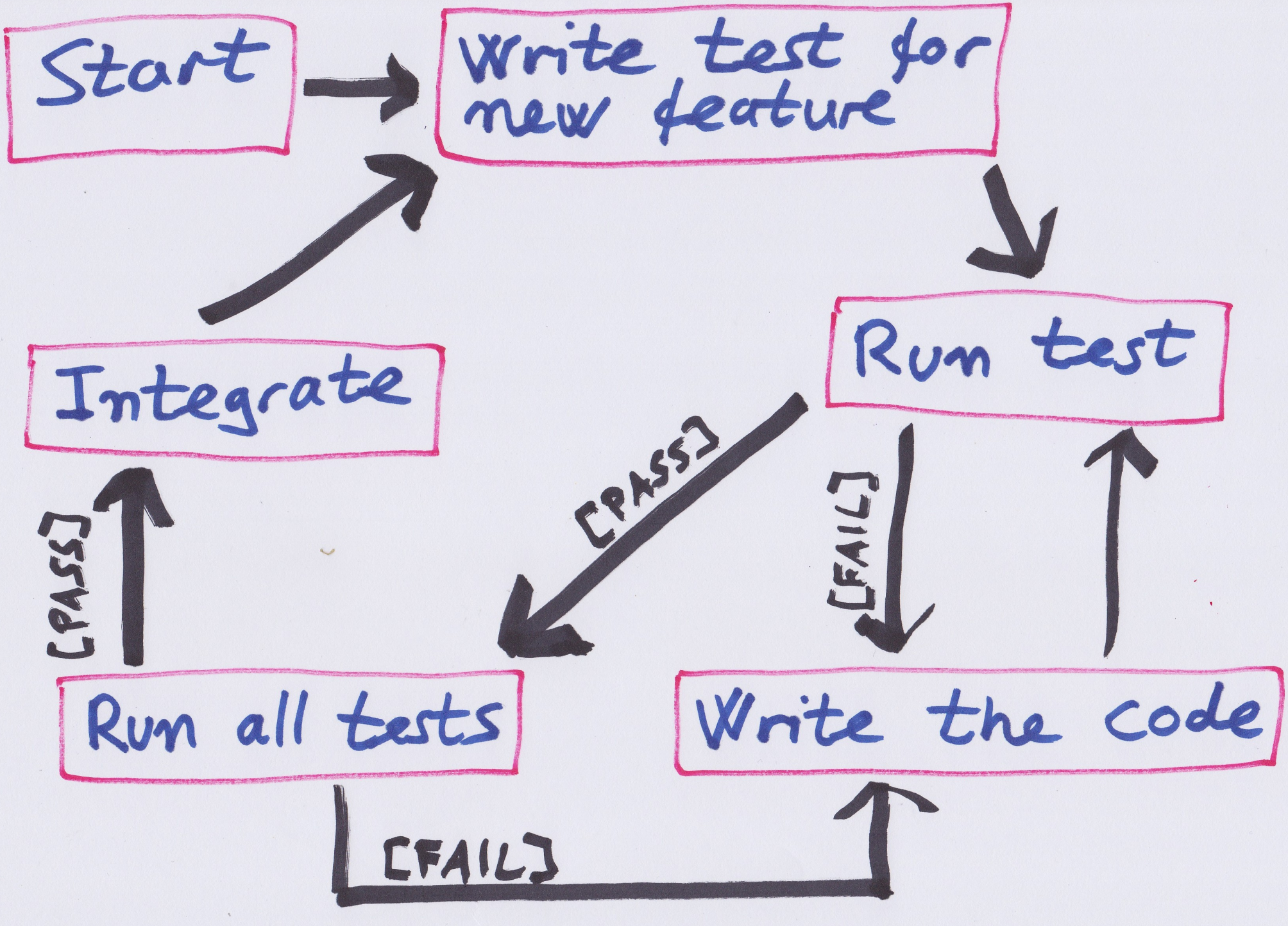
When you think of code and test as one,
testing is easy and code is beautiful.

Why Test Driven Development?

- Not possible to manually test all cases of a complex software project – must be automatic!
- Protects against regression errors
- Greatly assists in debugging
- Empowers refactoring and optimisations
- Codification of requirements!
- Reduces fear (Kent Beck 2002)
- Enables continuous integration and deployment

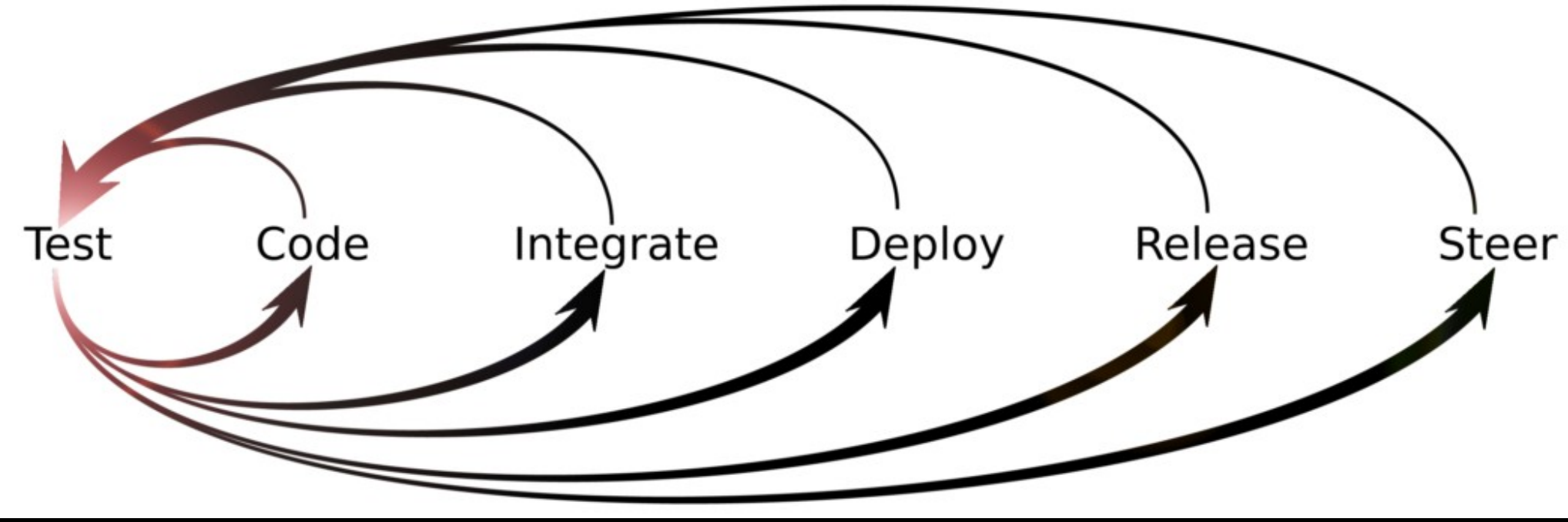
Your code is like clay.
When it's fresh, it's soft and malleable.
As it ages, it becomes hard and brittle.

If you write tests when the code is fresh
and easy to change, testing will be easy,
and both the code and the tests will be strong.


Start
Write test for new feature
Run test
[FAIL]
[PASS]
Write the code
[FAIL]
Run all tests
[PASS]
Integrate

Tests not run waste away

Run your tests often.
Don't let them get stale.
Rejoice when they pass.
Rejoice when they fail.


Test
Code
Integrate
Deploy
Release
Steer

Simple but real test examples

```
def test_populate_polygon(self):
        """Polygon can be populated by random points
        """

        # Create non trivial polygon
        polygon = [[0, 0], [10, 10], [15, 5], [20, 10], [25, 0],
                   [30, 10], [40, -10]]

        # Call function to be tested
        points = populate_polygon(polygon, 5)

        # Check that result is as expected
        assert len(points) == 5
        for point in points:
            msg = 'Point "%s" is not inside test polygon.' % str(point)
            assert is_inside_polygon(point, polygon), msg
```

```
def test_bad_CLASS(self):
    """Check that a bad CLASS string gives RuntimeError."""

    CLASS = 'bad'
    sa_1_0 = np.array([0.125, 0.444])
    sa_0_3 = np.array([0.25, 1.655])
    skew = 40
    num_spans = 4
    model = 'LINEAR'
    self.failUnlessRaises(RuntimeError, bd.bridge_states_ModelLinear,
                          model, CLASS, sa_0_3, sa_1_0, skew, num_spans)
```

```
@skipIf(sys.platform == 'win32', "Test cannot run on Windows")
def test_full_run_qgszstats(self):
    """Aggregation results are correct using native QGIS zonal stats.

    .. note:: We know this is going to fail (hence the decorator) as
        QGIS1.8 zonal stats are broken. We expect this to pass when we
        have ported to the QGIS 2.0 api at which time we can remove the
        decorator. TS July 2013

    """

    # TODO check that the values are similar enough to the python stats
    myFileList = ['kabupaten_jakarta.shp']
    load_layers(myFileList, clear_flag=False, data_directory=BOUNDDATA)

    myResult, myMessage = setup_scenario(
        DOCK,
        hazard='A flood in Jakarta like in 2007',
        exposure='People',
        function='Need evacuation',
        function_id='Flood Evacuation Function',
        aggregation_layer='kabupaten jakarta',
        aggregation_enabled_flag=True)
    assert myResult, myMessage
```

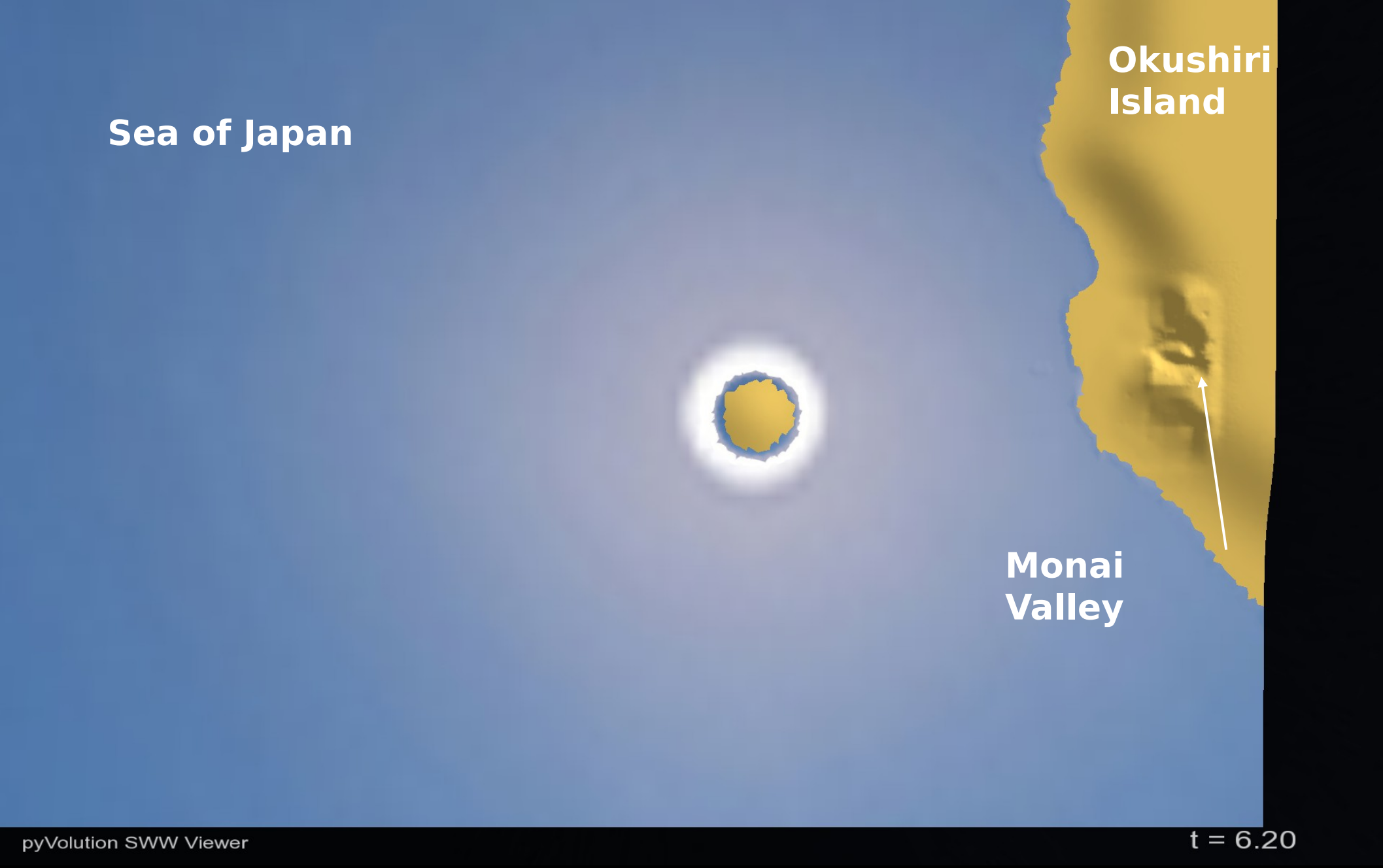
```
@Test
public void testStaticNotifySingleReceiver()
{
   final INotificationReceiver receiver = context.mock(INotificationReceiver.class);
   final INotification notification =
                Notification.create(NotificationLevel.ERROR, "A title", "Some text").build();

   NotificationManager.registerReceiver(receiver);

   context.checking(new Expectations() {{{
      oneOf(receiver).handle(with(notification));
   }}});

   NotificationManager.notify(notification);
}
```

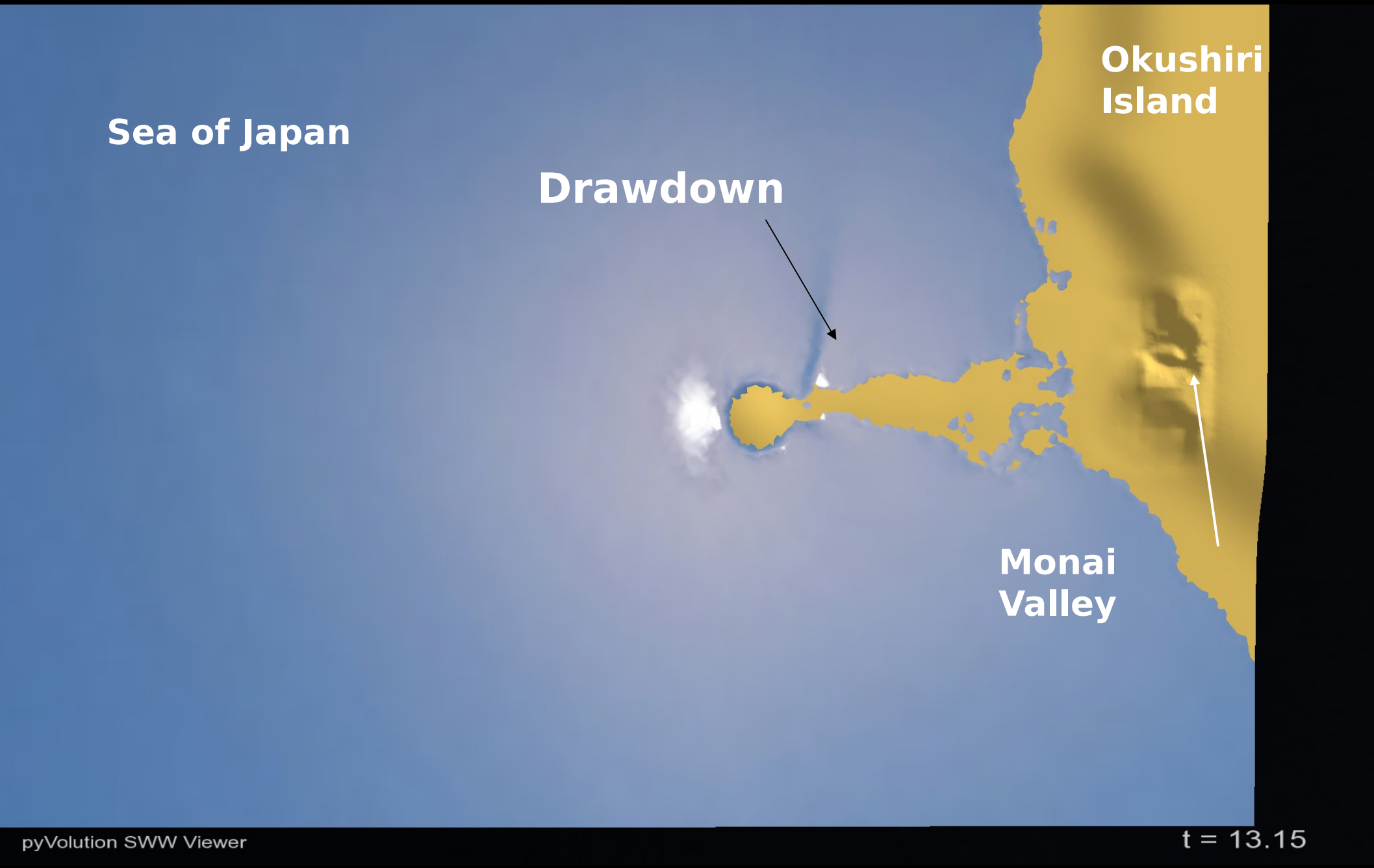
- Tests that the NotificationManager dispatches notifications to registered receivers correctly (for a single receiver case)
- Part of a larger test suite that exercises the NotificationManager
- Uses the JMock framework to mock the INotificationReceiver interface, and to set expectations on it
- The test will fail if:
  - The receiver does not receive the notification from the manager
  - The receiver receives anything other than the notification from the manager
  - The receiver receives the notification more than once

Model validation as part of the test suite

Okushiri Island 1993 Tsunami

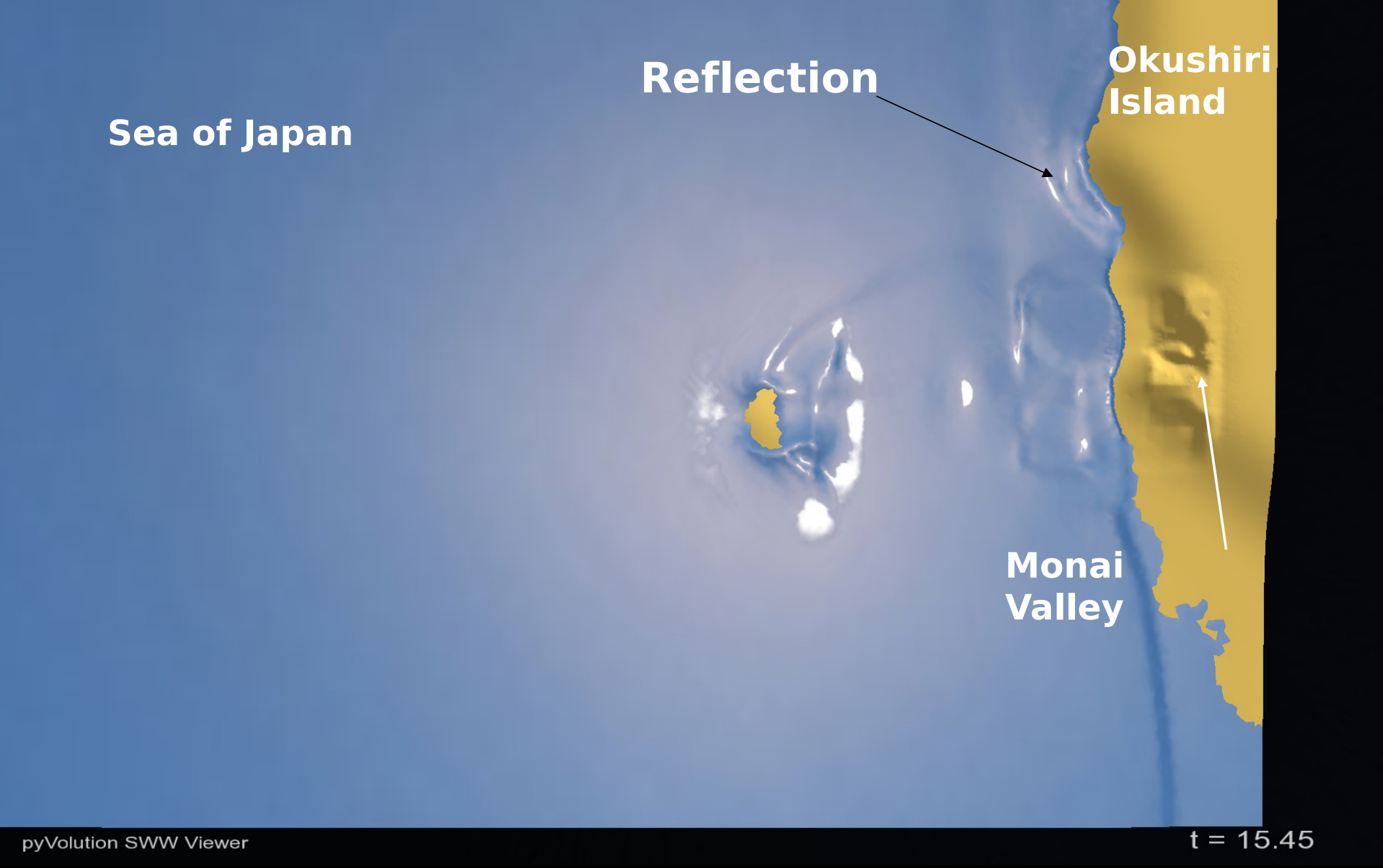
- **Magnitude 7.8 earthquake**
- **32 m run up height**
- **Numerical Simulation of wave tank experiment shows why**

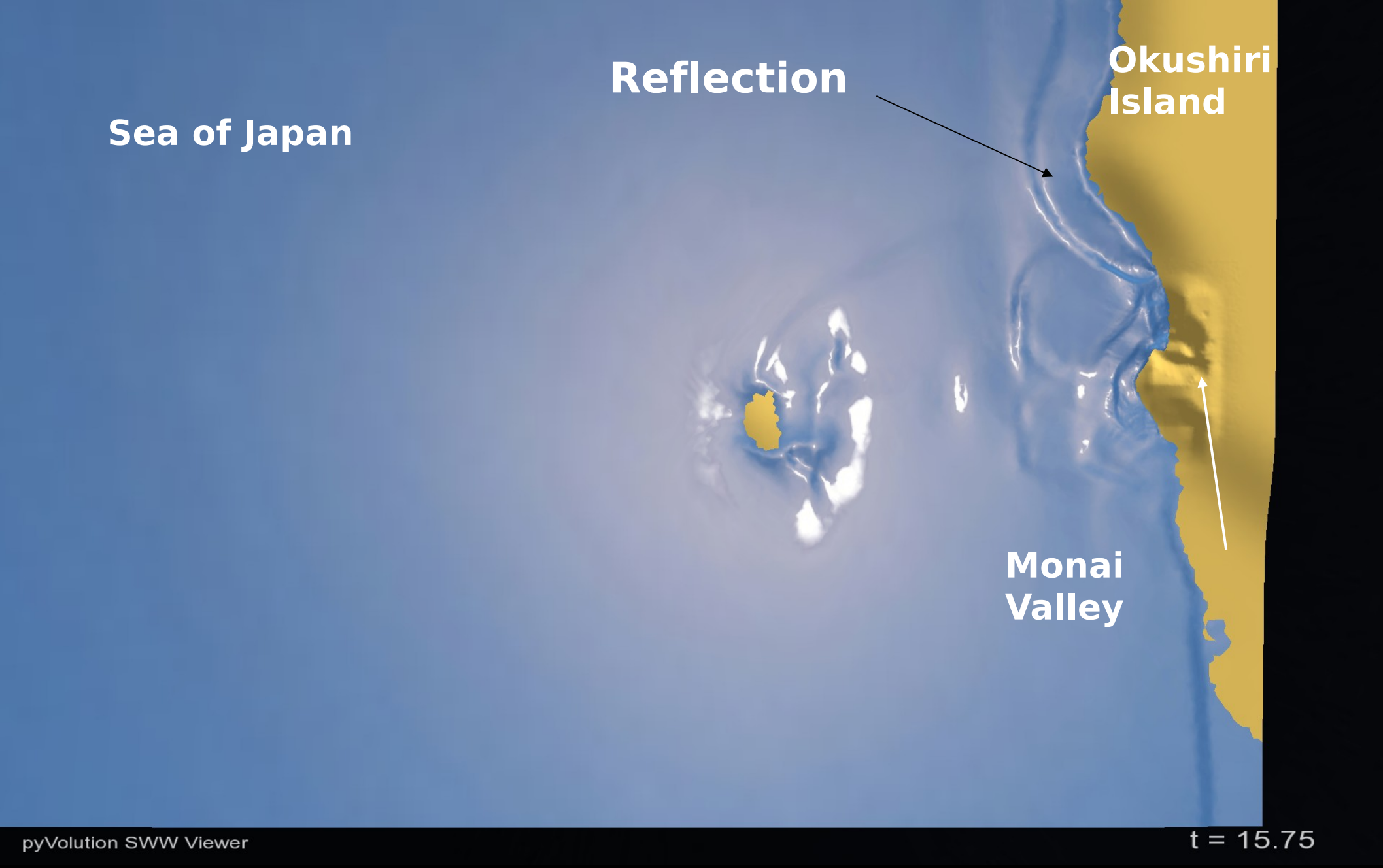

Sea of Japan
Okushiri
Island
Monai
Valley
pyVolution SWW Viewer
t = 6.20


Sea of Japan
Drawdown
Okushiri
Island
Monai
Valley
pyVolution SWW Viewer
t = 13.15


Sea of Japan
Okushiri Island
Monai Valley
pyVolution SWW Viewer
t = 14.25


Sea of Japan
Okushiri Island
Monai Valley
pyVolution SWW Viewer
t = 15.15


Sea of Japan
Reflection
Okushiri
Island
Monai
Valley
pyVolution SWW Viewer
t = 15.45


Reflection
Okushiri
Island
Sea of Japan
Monai
Valley
pyVolution SWW Viewer
t = 15.75

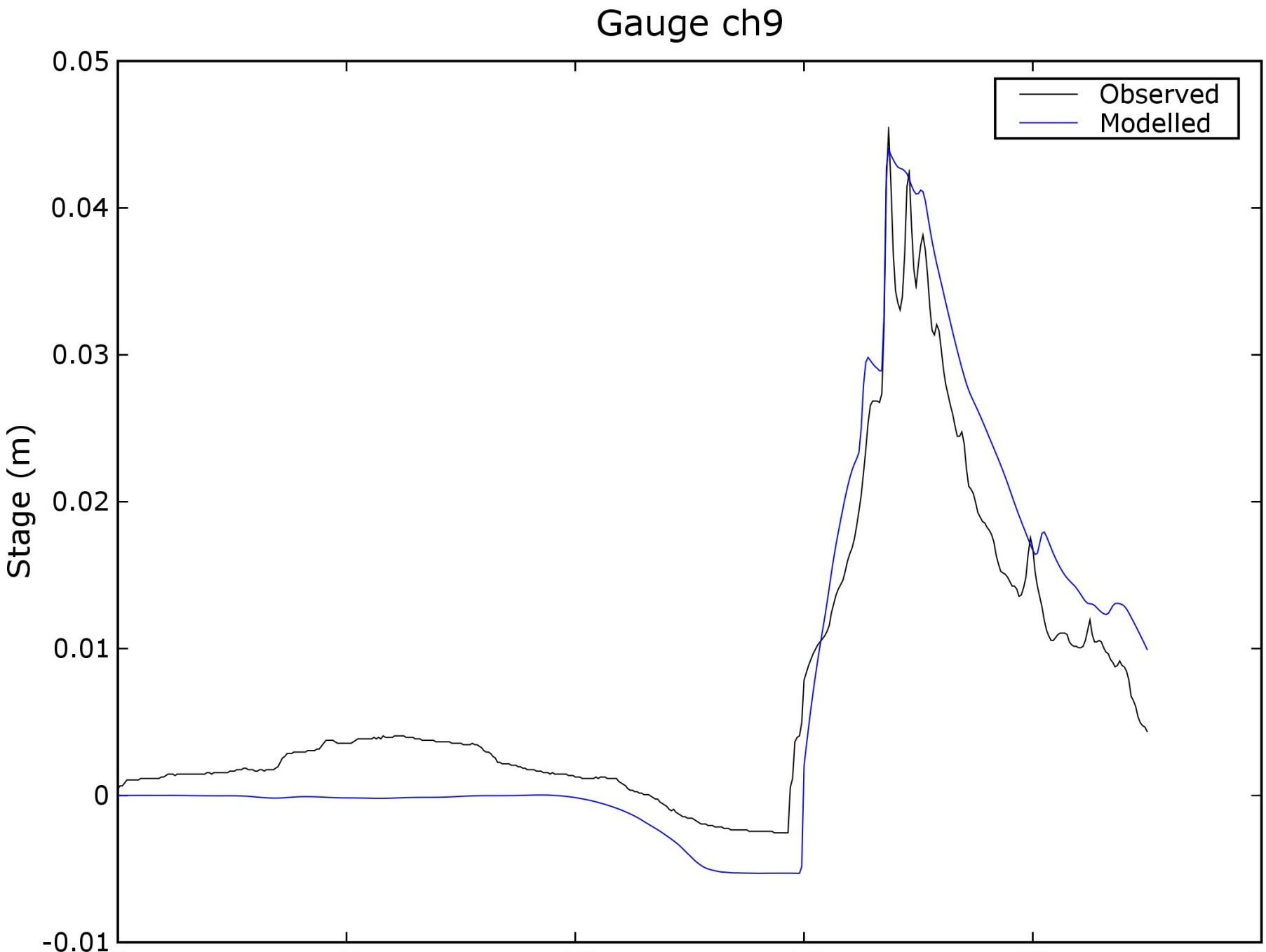

Sea of Japan
Shoaling
Okushiri
Island
Monai
Valley
pyVolution SWW Viewer
t = 16.05


Sea of Japan
Okushiri
Island
Run
up
Monai
Valley
pyVolution SWW Viewer
t = 16.25


Sea of Japan
Okushiri
Island
Run
up
Monai
Valley
pyVolution SWW Viewer
t = 16.50


Sea of Japan
Okushiri Island
Run up
Monai Valley
pyVolution SWW Viewer
t = 16.55

Model validation written as “unit” test



Reflections on testing

- A development culture!
- Have tests for the functionality you care about – future protection
- The extra time invested comes back 100 fold when debugging, refactoring and moving to new platforms
- Hard to start with existing project, so write half a dozen high level tests
- Crucial when optimising ("*Faster and faster – but wrong*"!)
- Crucial when refactoring (ripped out dependency in hours)
- Also serves as trusted documentation of APIs
- "*I don't understand how people program without tests*" (Duncan Gray)

- **The single most important practice – I'd rather keep my tests than the source code!**

- InaSAFE has over 300 tests - ANUGA almost 1000,

An imperfect test today is better than a perfect test someday

The perfect is the enemy of the good.
Don't wait for best to do better.
Don't wait for better to do good.
Write the test you can today.

What makes good testing

- Sets up all conditions, call the test, verify the result.
- Verify result using known invariant conditions (e.g. conservation of mass, linearly interpolated points must lie within certain bounds, inverse transforms reproduce original inputs, read back what was written, retain double precision...)
- Not only “happy path” but also corner cases (None, NaN, 0, max, min, %, &, $): E.g. intersection (Parallel lines, coinciding lines, intersecting at endpoints, ...)
- Input that causes error conditions
- If a new bug is found, write new test revealing it “*It is a lot easier to debug a test than running the application*” (Tim Sutton)
- Independent of context, time and other tests
- Named after what is being tested: (see next slide)
- Good coverage (>80%) - but not 100%!
- Programmers need GUTs

Google: Klaus P. Berg – Good Unit Tests

#148 Linear and quantile bins are correct ... ok
#149 Bounding boxes can be converted between list and string ... ok
#150 Intersections of bounding boxes work ... ok
#151 Bounding box can be buffered ... ok
#152 Centroid point data can be derived from polygon data ... ok
#153 Projections that are compatible but not identical are recognised ... ok
#154 Donut polygon can be read, interpreted and written correctly ... ok
#155 Empty keywords can be handled ... ok
#156 Bounding box can be extracted from geotransform ... ok
#157 Resolution can be extracted from geotransform ... ok
#158 Bounding box is correctly extracted from file. ... ok
#160 Vector and Raster objects can be instantiated with None ... ok
#161 Keywords can be written and read ... ok
#162 Keywords and values with colons raise error messages ... ok
#163 Vector and Raster instances have a similar API ... ok
#164 Points along line are computed correctly ... ok
#165 Bounding box minimal size can be controlled ... ok
#166 Multipart polygons are be converted to singlepart ... ok
#167 Raster layers with no projection causes Exception to be raised ... ok
#168 NODATA value is correctly handled for GDAL layers ... ok
#169 Ordering of polygon vertices is preserved when writing and reading ... ok
#170 Polygon areas are computed correctly ... ok
#171 Polygon centroids are computed correctly ... ok
#172 Polygons with inner rings can be written and read ... ok
#173 Projection information can be correctly compared ... ok
#174 Raster extrema (including NAN's) are correct. ... ok
#175 Raster layers can be converted to vector point layers ... ok

TDD Project examples in GA

- Pypar (2002) – Python for parallel computing
- ANUGA (2006) – Hydrodynamic modelling
- EQRM (2007) – Earthquake risk modelling
- PF3D (2010) – Volcanic ashload modelling
- InaSAFE (2012) – Impact from natural disasters
- HazIMP (2013) – Assess building vulnerability
- EarthSci (2013) - Worldwind 4D visualisation

What doesn't easily lend itself to automated testing

Design
Performance
UI's
Error messages
Real time
Interdependent Systems (use Mock!)

Other types of testing

- Style checking (e.g. PEP8 for Python)
- Static code analysis (e.g. Pylint)
- Customised checks:
  - Scan for unwanted strings, e.g. assert, settrace, …
  - IP status of bundled data (is it ok to publish?)
  - Missing translations
  - Passwords, connection strings, security issues
  - - anything you care about, that can be automated

Continuous Integration Server

Jenkins

search log in

Jenkins › InaSAFE-QGIS1 ENABLE AUTO REFRESH

- Back to Dashboard
- Status
- Changes
- GitHub
- Coverage Report
- Violations
- SLOCCount
- GitHub Hook Log

Build History (trend)

#685	Dec 16, 2012 6:49:55 AM
#684	Dec 14, 2012 7:45:38 AM
#683	Dec 14, 2012 6:42:11 AM
#682	Dec 14, 2012 6:07:30 AM
#681	Dec 13, 2012 8:35:31 AM
#680	Dec 13, 2012 7:51:34 AM
#679	Dec 12, 2012 4:02:11 PM
#678	Dec 12, 2012 2:54:21 PM
#677	Dec 12, 2012 8:01:09 AM
#676	Dec 11, 2012 11:00:39 PM
#675	Dec 7, 2012 9:54:49 AM
#674	Dec 7, 2012 5:23:39 AM
#673	Dec 6, 2012 6:30:55 PM
#672	Dec 6, 2012 3:44:43 PM
#671	Dec 6, 2012 3:30:30 PM
#670	Dec 6, 2012 3:18:00 PM
#669	Dec 6, 2012 3:11:14 PM
#668	Dec 6, 2012 7:07:54 AM
#667	Dec 5, 2012 12:43:30 PM
#666	Dec 5, 2012 10:26:58 AM
#665	Dec 5, 2012 9:53:20 AM
#664	Dec 5, 2012 9:20:40 AM
#663	Dec 4, 2012 12:31:51 PM
#662	Dec 4, 2012 11:09:32 AM
#661	Dec 4, 2012 10:35:55 AM
#660	Dec 4, 2012 8:35:14 AM
#659	Nov 30, 2012 6:23:53 PM
#658	Nov 30, 2012 3:46:06 PM
#657	Nov 30, 2012 2:22:25 PM
#656	Nov 30, 2012 9:35:18 AM
#655	Nov 30, 2012 9:00:46 AM

More ...

RSS for all RSS for failures

Project InaSAFE-QGIS1

Automated testing for InaSAFE against QGIS version 1.8 Because realtime work has a QGIS 2 dependency, it is excluded from these tests.

Coverage Report

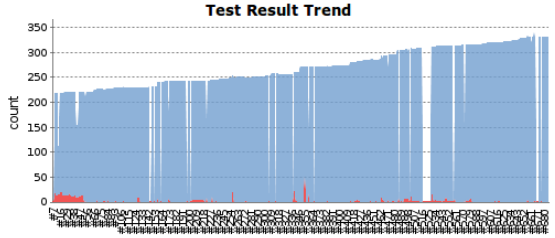
Recent Changes

Latest Test Result (no failures)

Permalinks

- Last build (#685), 1 day 21 hr ago
- Last stable build (#685), 1 day 21 hr ago
- Last successful build (#685), 1 day 21 hr ago
- Last failed build (#669), 11 days ago
- Last unstable build (#667), 12 days ago
- Last unsuccessful build (#676), 6 days 4 hr ago

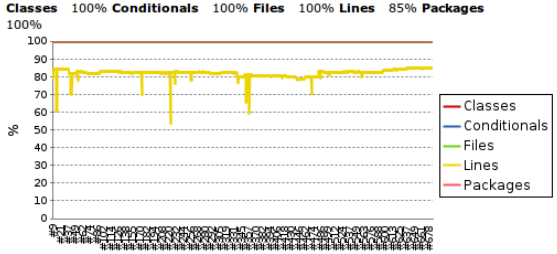
Test Result Trend



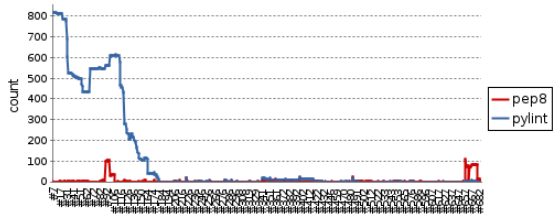
(just show failures) enlarge

Code Coverage

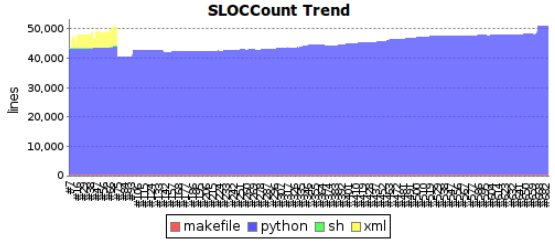
Classes 100% Conditionals 100% Files 100% Lines 85% Packages



pep8 0 pylint 0



SLOCCount Trend



Help us localize this page

Page generated: Dec 18, 2012 3:57:15 AM REST API Jenkins ver. 1.492

Live Jenkins server for InaSAFE release branch

http://jenkins.linfiniti.com/job/InaSAFE-Release-Branch-QGIS1/

Example of failing test and its resolution

http://jenkins.linfiniti.com/job/InaSAFE-QGIS1/754/

http://jenkins.linfiniti.com/job/InaSAFE-QGIS1/756/

Another set of fixes

http://jenkins.linfiniti.com/job/InaSAFE-Release-Branch-QGIS1/32/

Back to Project
Status
Changes
Console Output
View Build Information
Polling Log
Git Build Data
Test Result
Coverage Report
Violations
SLOCCount
Previous Build
Next Build

Build #754 (Jan 29, 2013 3:28:05 AM)

Started 7 mo 6 days a
Took 34 min

Changes

1. Added unit test for #372 (commit: 9d9df8a9498254a082ed0957e0465f25b1286471) (detail)
2. IDE update - more config options for OSX (commit: bd78335f3be466c469929ad000fefe8c300e3ac8) (detail)

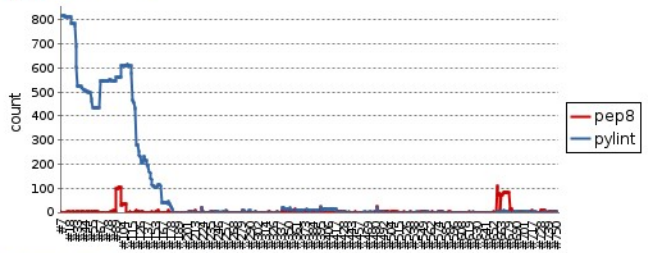
Started by an SCM change

Revision: bd78335f3be466c469929ad000fefe8c300e3ac8

- origin/master

Test Result (1 failure / +1)
workspace.safe_qgis.test_dock.DockTest.test_InsufficientOverlapIssue372

pep8 0 pylint 0



52,213 (+19) lines in 150 (+0) files and 3 (+0) languages.

- python : 52,148 (+19) lines in 147 (+0) files.
- sh : 28 (+0) lines in 2 (+0) files.
- makefile : 37 (+0) lines in 1 (+0) files.

Back to Project
Status
Changes
Console Output
View Build Information
Polling Log
Git Build Data
Test Result
Coverage Report
Violations
SLOCCount
Previous Build
Next Build

Build #756 (Jan 29, 2013 4:45:05 AM)

Started 7 mo 6 days a
Took 33 min

Changes

1. Fix for failing test for #372 (commit: c76a61d257f077ada46eb0e9442f072d31aab444) (detail)

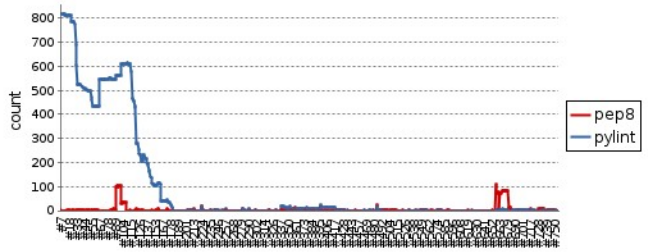
Started by an SCM change

Revision: 1745ad3078b8ee6b979a24249361c59986ef8be7

- origin/master

Test Result (no failures)

pep8 1 (+1) pylint 0



52,213 (+1) lines in 150 (+0) files and 3 (+0) languages.

- python : 52,148 (+1) lines in 147 (+0) files.
- sh : 28 (+0) lines in 2 (+0) files.
- makefile : 37 (+0) lines in 1 (+0) files.

Fix for failing test for #372

Browse code

master version-1_2_0 realtime-version-2

timlinux authored 7 months ago 1 parent bd78335 commit c76a61d257f077ada46eb0e9442f072d31aab444

Showing **1 changed file** with **6 additions** and **5 deletions**. Show Diff Stats

11 safe_qgis/test_dock.py View file @ c76a61d

```
...  ...  @@ -863,18 +863,19 @@ def test_InsufficientOverlapIssue372(self):
863  863
864  864          # Enable on-the-fly reprojection
865  865          setCanvasCrs(GEOCRS, True)
866       -        # Zoom to an area where there is no overlaop with layers
867       -        myRect = QgsRectangle(106.61001188831219, -6.130614191176471,
868       -                              106.67188745972703, -6.080190955882353)
     866  +        # Zoom to an area where there is no overlap with layers
     867  +        myRect = QgsRectangle(106.635434302702, -6.101567666986,
     868  +                              106.635434302817, -6.101567666888)
869  869          CANVAS.setExtent(myRect)
870  870
871  871          # Press RUN
872       -        QTest.mouseClick(myButton, QtCore.Qt.LeftButton)
     872  +        DOCK.accept()
873  873          myResult = DOCK.wvResults.page().currentFrame().toPlainText()
874  874
875  875          # Check for an error containing InsufficientOverlapError
876       -        myMessage = 'Result not as expected: %s' % myResult
877  876          myExpectedString = 'InsufficientOverlapError'
     877  +        myMessage = 'Result not as expected %s not in: %s' % (
     878  +            myExpectedString, myResult)
878  879          # This is the expected impact number
879  880          self.assertIn(myExpectedString, myResult, myMessage)
880  881
```

0 notes on commit c76a61d

Show line notes below

Please sign in to comment.

Dream?

- One day all our software systems will
  - Be under version control
  - Have regression test suites
  - Be managed by a continuous integration server
  - Be Open Source
  - Be easily deployed to dev, test, prod and elsewhere
  - …. and more

7 noble steps to software that works

No news is good news

If it can't install – it doesn't exist

Without docs – it doesn't exist

Without testing – it doesn't exist

No future without a past

Sleep in the bed you make

Get it right – then fast

Thank You!

Links

http://www.agitar.com/downloads/TheWayOfTestivus.pdf

http://www.javaworld.com/javaworld/jw-03-2009/jw-03-good-unit-tests-1.html

http://www.javaworld.com/javaworld/jw-04-2009/jw-04-good-unit-tests-2.html