Polytesting

Run the Same Tests on Low and High Levels

Peter Zsoldos
@zsepi
hello@zsoldosp.eu

Test Pyramid FAIL

- All unit tests are green
- Yet production is broken
- Disclaimer: Some Can Do It Well

End-to-End Testing FAIL

- Usually duplicates unit test scenarios
- Full stack tests are slow
- Slow tests aren’t run often

Polytesting

- One test, executed with multiple drivers
- a.k.a.: Polymorphism
- We already polytest!
  - tox
  - Selenium – PhantomJS, Firefox, Chrome, etc.
  - Postgres, MySQL, SQLite, etc.

Test Scenarios Are Stable

```
def test_can_move_to_same_status_different_person(self):
    self.given_a_board(
        owners=['Alice', 'Bob'], states=['Open', 'Done'],
        with_tasks=[{
            'owner': 'Alice', 'status': 'Open',
            'name': 'task', 'href': '/task'
        }]
    )
    self.move_task('/task', to_owner='Bob', to_status='Open')
    self.assertEquals(
        [{'name': 'task', 'href': '/task'}],
        self.get_tasks_for(owner='Bob', status='Open'))
```

Pure Business Logic

```
def move_task(self, url, to_owner, to_status):
    self.board.move(url, to_owner, to_status)
```

django.test.client.Client

```
def move_task(self, url, to_owner, to_status):
    post_data = dict(
        url=url, to_owner=to_owner, to_status=to_status)
    url_to_post_to = reverse('move_task')
    response = self.client.post(url_to_post_to, post_data)
    if response.status_code != 302:
        raise Exception('expected HTTP 302 .... ' % dict(
            status=response.status_code,
            payload=post_data,
            url=url_to_post_to
        ))
```

Selenium

```
def move_task(self, url, to_owner, to_status):
    source = \
        self.selenium.find_element_by_css_selector(
            'a[href="%s"]' % url)
    target = \
        self.selenium.find_element_by_css_selector(
            'td[owner="%s"][state="%s"]' % (
                to_owner, to_status))
    ActionChains(
        self.selenium).drag_and_drop(
            source, target).perform()
```

Inspiration From

- Production failures :)
- BDD
- DDD/Ubiquitous Language
- Onion / Hexagonal / Ports & Adapters Architecture
- Page Object Pattern

Current Status / Plans

1. Finish proof of concept code for Django 1.6
   - 95% done
2. Blog/README
3. Frameworkify/Librarify

@zsepi

hello@zsoldosp.eu