

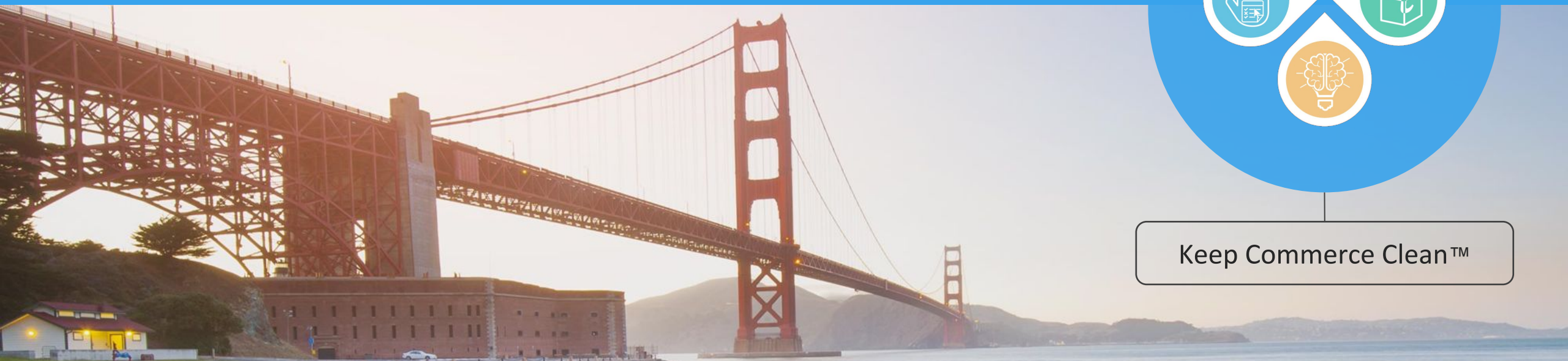


Wrap up Selenium

TEST OUR COFFEE @ Coimbra



Keep Commerce Clean™



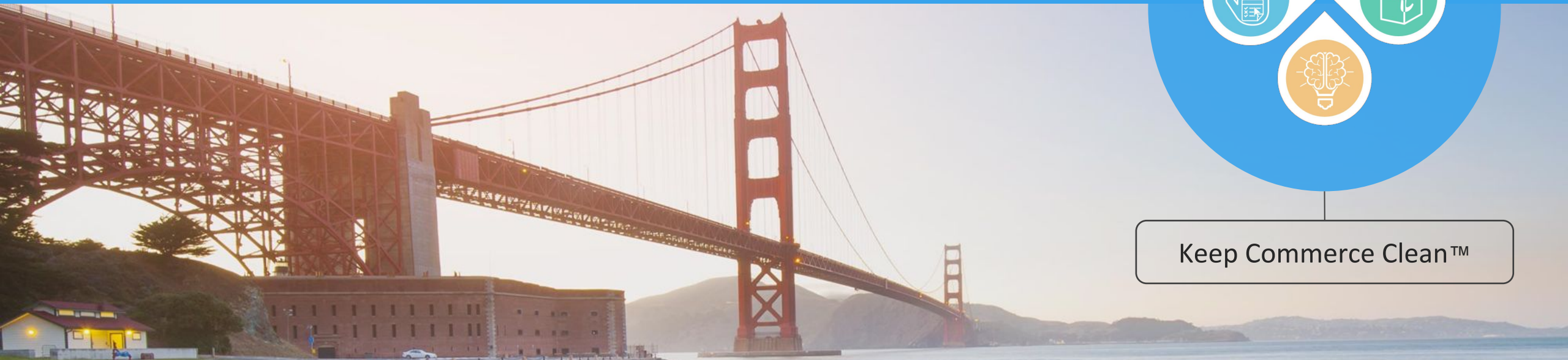


Wrap up Selenium

“The Feedzai Approach to Run Thousands of Tests in a Distributed Way”



Keep Commerce Clean™



AGENDA

1. Meet Feedzai
2. Selenium @ Feedzai
3. Our Wrapped Selenium
4. Main Features
5. Biggest Challenges
6. Q&A



feedzai

PSTQB
ASSOCIAÇÃO PORTUGUESA
DE TESTES DE SOFTWARE

Meet Feedzai

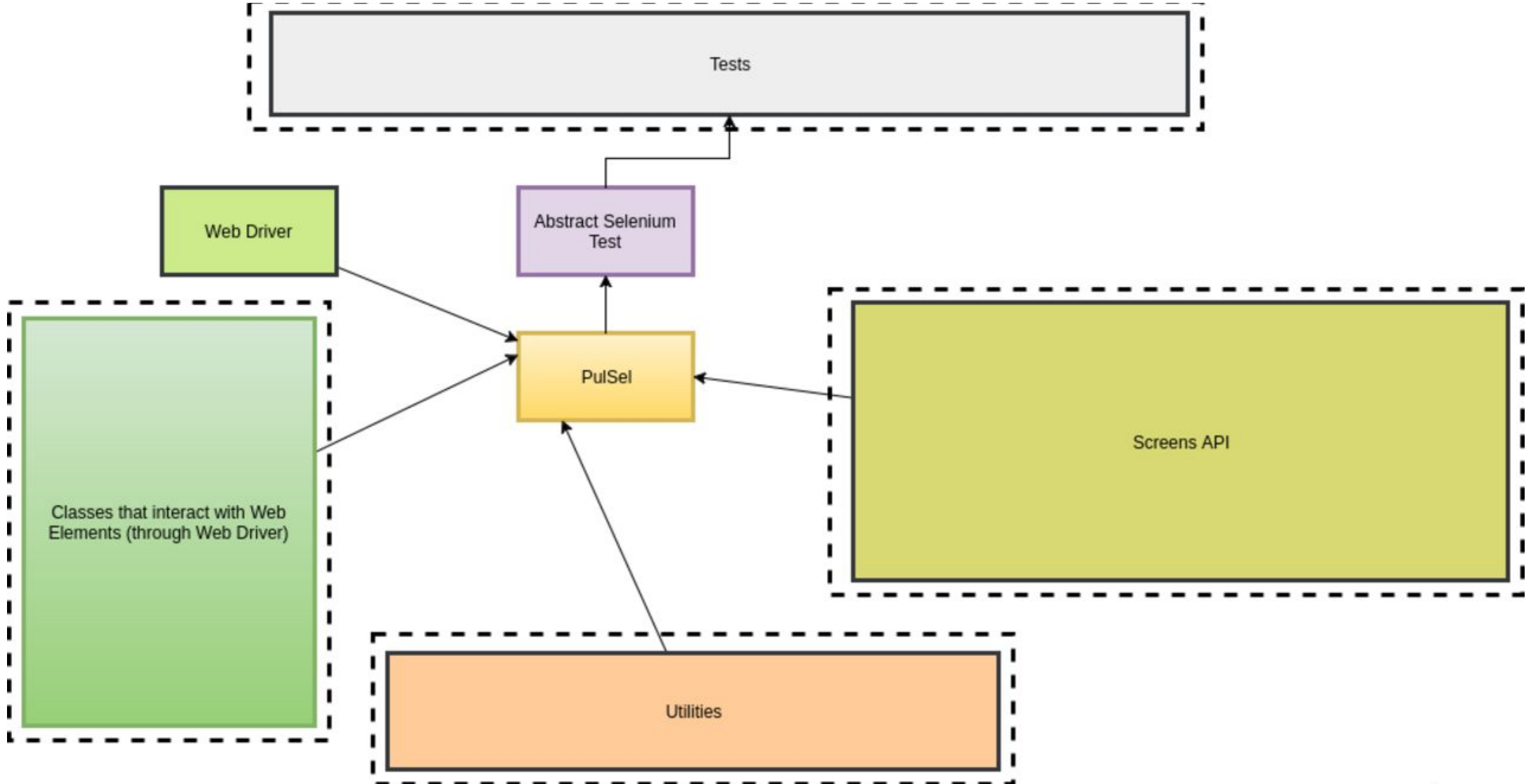


Selenium @ Feedzai

- Used since 2012
- Currently using Selenium 2.48.2
- Java + Maven + JUnit + Docker
- Number of tests: ~1100
- Around 110k lines of code
- CI Build time around 1h 45m

Our Wrapped Selenium

Selenium Architecture



Abstract Selenium Test

- Base class from each test extends.
- Start PulSel (Pulse Selenium)
- Start / Setup Environment (Local or Docker)
- Utility method for test start / stop

PulSel (PulseSelenium)

- “Most important” class used under the Selenium Tests Implementation
- Instantiate Web Driver
- Instantiate Finder, Clicker and Asserter (used to interact with web elements)
- End point to instantiate all screens testers (with the API to interact with Pulse Views pages)
- Other utilities: Screenshot taker, Javascript checker...

Finder / Asserter / Clicker

- Receive WebDriver instance
- Finder: contains methods that allow us to search for web elements
- Asserter: contains methods that allow us to perform asserts in our web elements
- Clicker: contains methods that allow us to interact with web elements: click, write, drag...
- Object Identification Strategy: jQuery

Testers

- Receive Asserter, Clicker and Finder
- Contain the methods to interact with the specific screens from our front end pages
- Has the methods to manipulate, assert or search for given screen element
- Tester classes might instantiate other smaller and more manageable testers that will have the methods to a smaller responsibility on that functionality

Handlers

- Like a Tester this object contains also the API for manipulating a particular web element or a set of elements
- A Handler is used in testers as a way to reduce code duplication and abstract some common components
- Handlers are essentially Testers used in multiple places across the screens in our front-end application

Pulse screen vs Testers Architecture

pulse 16.1 Hello pulse 🔍 ⚙️ 🔑 📄

New Model Basic Tester

1: Data Source and Features
Choose the data source, target variable and features

2: Validation Configuration
Choose the validation and sampling types

3: Algorithm
Choose the algorithm and tune the parameters

Name: Titanic Model#1 Data Source Features Tester

Data Source: Titanic Data Source

Target Variable: Survived

Breakdowns: Fields: Passenger Class (3 values) Combinations: 3 ⚙️ Pick fields and values

Fields To Select:

Name	Type
☐ Passenger Id	Numeric
☒ Passenger Class	Categorical
☒ Sex	Categorical
☒ Age	Numeric
☐ Siblings/Spouses Aboard	Categorical
☐ Parents/Children Aboard	Categorical
☐ Fare	Numeric
☐ Port of Embarkation	Categorical

Output Scored Instances: Output Type: Not configured Path: Not configured ⚙️ Configure Score Instances

⌵ Advanced Options

Next Cancel Basic Tester

Pulse screen vs Testers Architecture

pulse 16.1 Hello pulse 🔍 ⚙️ 🔧 📄

New Model

1: Data Source and Features
Choose the data source, target variable and features

2: Validation Configuration
Choose the validation and sampling types

3: Algorithm
Choose the algorithm and tune the parameters

Name: Titanic Model#1 Input Element Handler

Data Source: Titanic Data Source Select React Handler

Target Variable: Survived Select React Handler

Breakdowns: Fields Combinatorial Breakdown Fields Handler
Passenger Class (3 values) 3 Pick fields and values

Fields To Select Fields Table Handler

Name	Type
☐ Passenger Id	Numeric
☒ Passenger Class	Categorical
☒ Sex	Categorical
☒ Age	Numeric
☐ Siblings/Spouses Aboard	Categorical
☐ Parents/Children Aboard	Categorical
☐ Fare	Numeric
☐ Port of Embarkation	Categorical

Output Scored Instances: Output Type: Not configured Path: Not configured Configure Score Instances Score Instances Tester

Advanced Options

Next Cancel

Pulse screen vs Test Implementation



1: Data Source and Features

Choose the data source, target variable and features

2: Validation Configuration

Choose the validation and sampling types

3: Algorithm

Choose the algorithm and tune the parameters

Name

Titanic Model#1

Data Source

Titanic Data Source

Target Variable

Survived

Breakdowns

Fields

Combinations

Passenger Class (3 values)

3

Pick fields and values

Fields To Select

Fields

Fields 10

Filter

Name	Type
☐ Passenger Id	Numeric
☒ Passenger Class	Categorical
☒ Sex	Categorical
☒ Age	Numeric
☐ Siblings/Spouses Aboard	Categorical
☐ Parents/Children Aboard	Categorical
☐ Fare	Numeric
☐ Port of Embarkation	Categorical

Output Scored Instances

Output Type

Path

Not configured

Not configured

Configure Score Instances

Advanced Options

Next

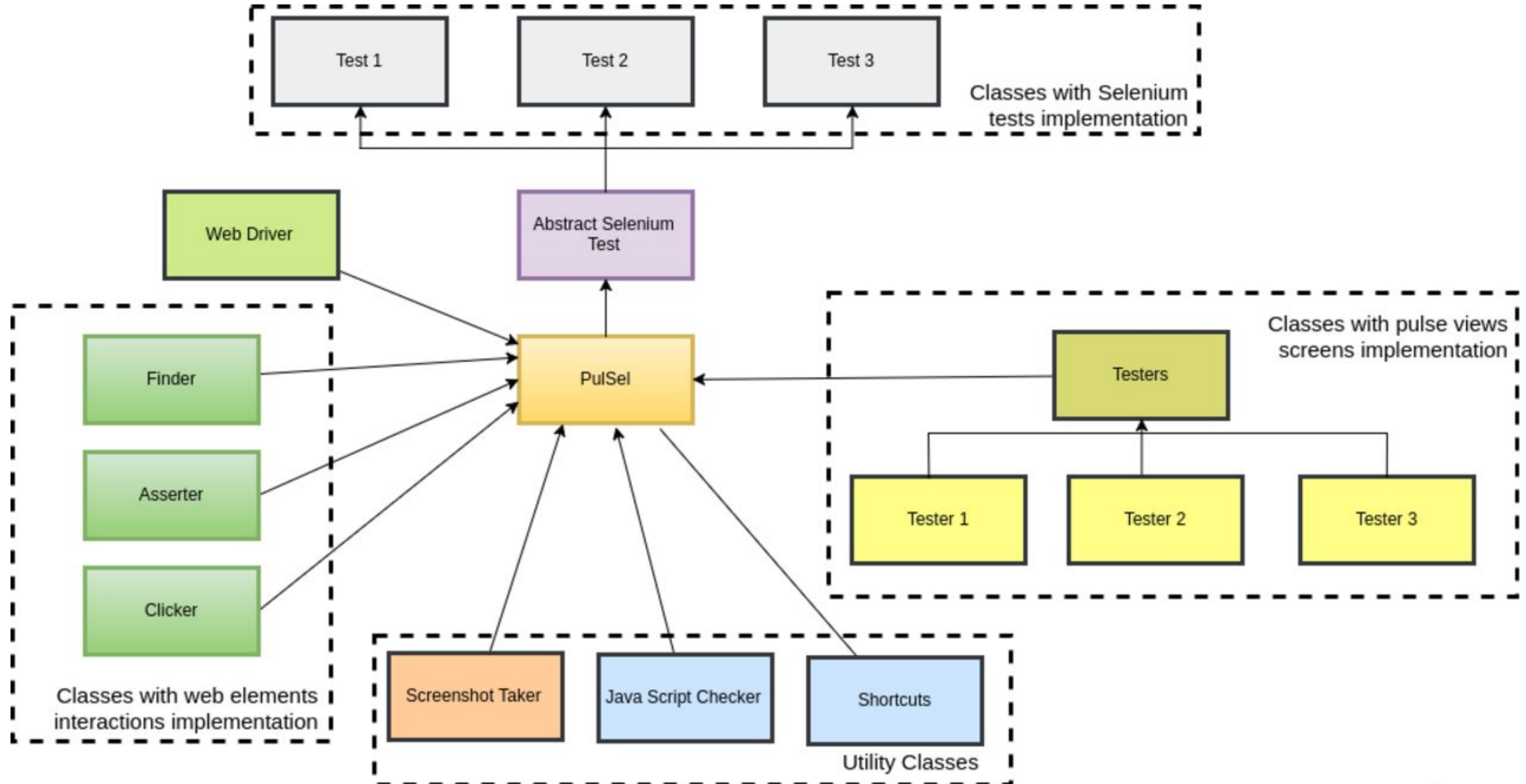
Cancel

```

tester.model().basic().wizard().assertCurrent("Data Source and Features");
tester.model().datasourceFeatures().editName("Titanic Model#1");
tester.model().datasourceFeatures().editDatasource("Titanic Data Source");
tester.model().datasourceFeatures().editTargetVariable("Survived");
tester.model().datasourceFeatures().breakdowns().clickPickFieldsAndValues();
tester.model().datasourceFeatures().breakdowns().chooseFields().selectFieldAndValues("Passenger Class", "1", "2", "3");
tester.model().datasourceFeatures().breakdowns().chooseFields().clickSaveChanges();
tester.model().datasourceFeatures().fields().uncheckField("Passenger Id");
tester.model().datasourceFeatures().fields().uncheckField("Siblings/Spouses Aboard");
tester.model().datasourceFeatures().fields().uncheckField("Parents/Children Aboard");
tester.model().datasourceFeatures().fields().uncheckField("Fare");
tester.model().datasourceFeatures().fields().uncheckField("Port of Embarkation");
tester.model().basic().clickNext();
tester.model().basic().wizard().assertCurrent("Validation Configuration");

```

Selenium Architecture



Environment

Selenium Grid

- 16 Windows virtual machines
- 1 grid == 1 build at a time
- Hard to simulate grid environment locally

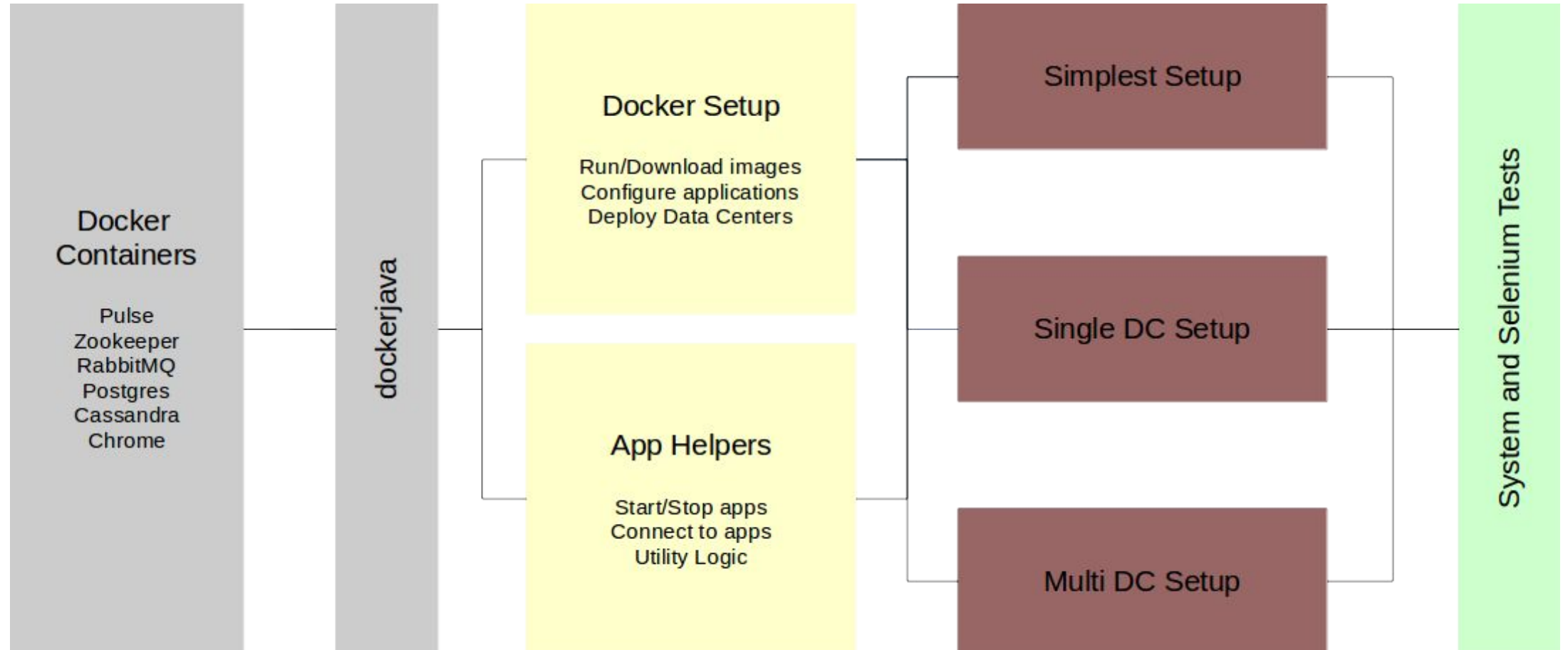


Drop selenium grid

System Test Framework

- Create test product deployments using docker containers
- Provides an api to manage the containers
- Provides an api to interact with the applications

System Test Framework



Example Test Setup

```
testSetup()  
  .withPulse(2)  
  .withCassandra(3)  
  .withZookeeper(3)  
  .withPostgres(1)  
  .withRabbitmq(3)  
  .withChrome()  
  .withPulseConfigs("pulse.example.config", "value")  
  .copyFileToPulseContainer("/some/local/file")  
  .runBeforePulse(ExampleTest::RunMigrationScripts)  
  .run();
```


Selenium on Docker

- Start and configure the containers per test class (Pulse, browser, storage)
- Run each test case in the class
- Save metrics, logs and dumps
- Tear down all containers

Main Features

Main features

- Tests are easy to write and understand since they follow a screen base logic



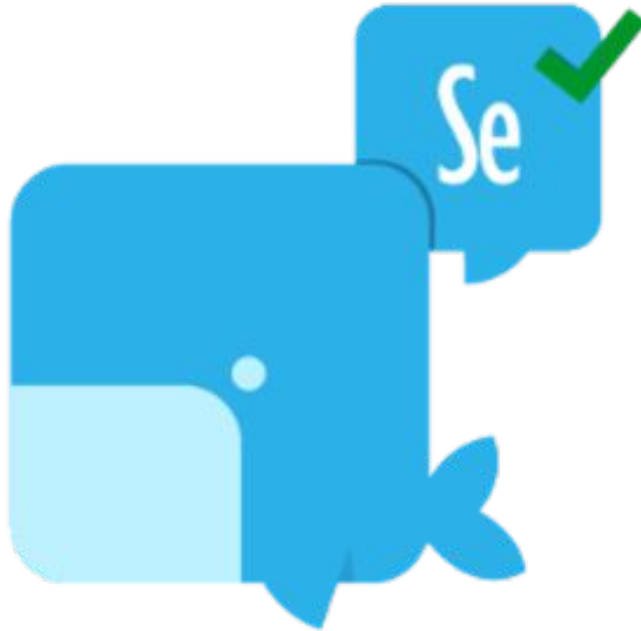
Main features

- Testers / Handlers created might be easily reused in other Testers, allowing a friendly code re-usage.



Main features

- Full control the machine under tests (Docker Environment)



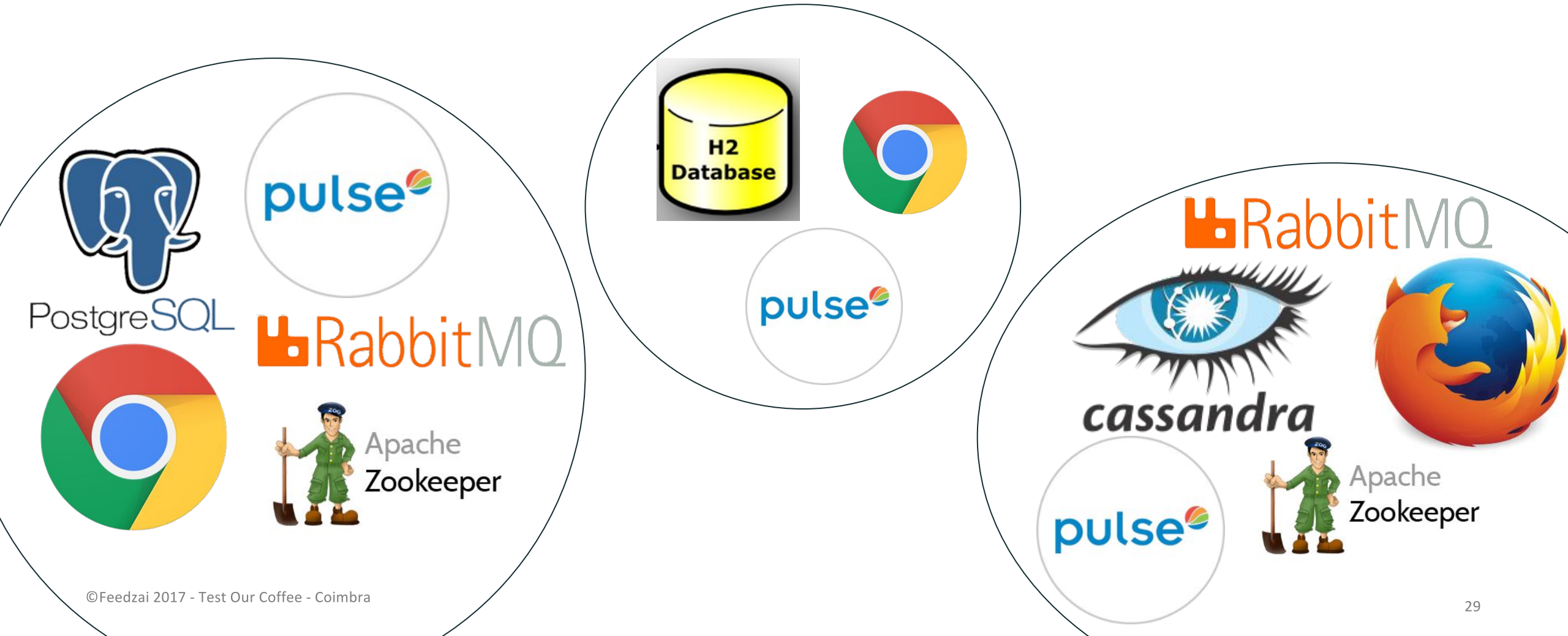
Main features

- Easy environment maintainability



Main features

- Easy to configure and parametrize deployments



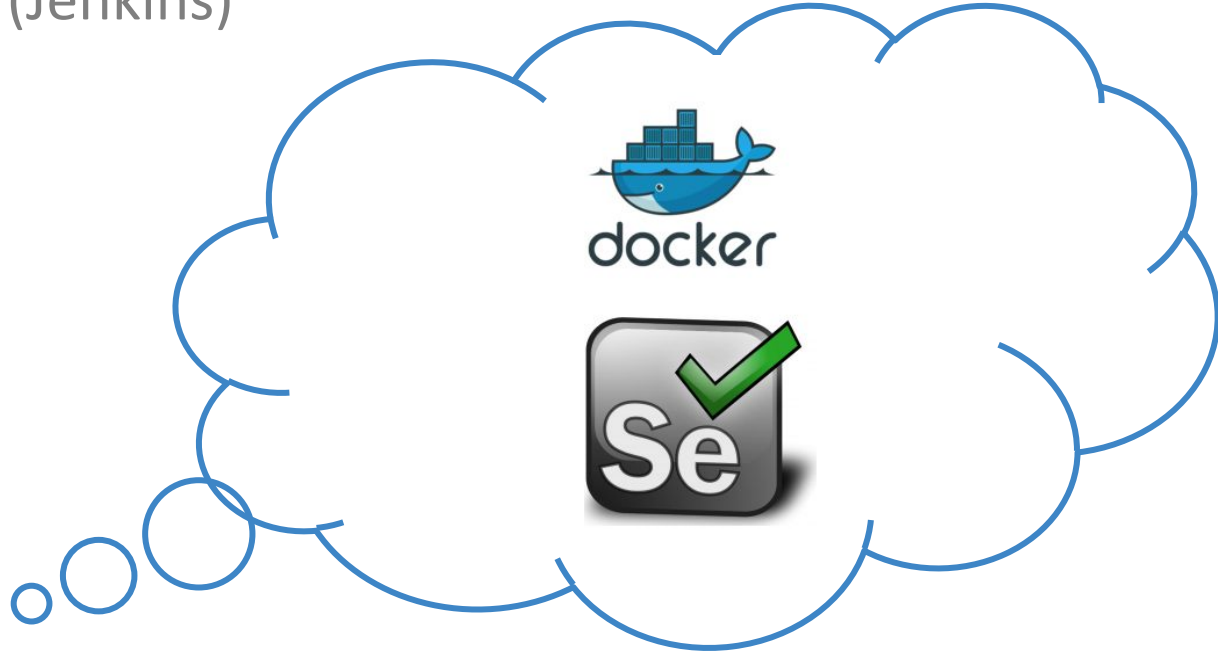
Main features

- Testers / Developers are able to easily run tests (locally, or in a Docker environment)



Main features

- Integrated in CI environment (Jenkins)



Biggest Challenges

Biggest Challenges:

- Keep tests small



Biggest Challenges:

- Speed up test execution



Biggest Challenges:

- Proper error handling



Unexpected Problem

Please [try again](#) and if the problem persists contact an administrator with the following code
6f58e119-da6b-4bd2-a6ed-14c73112fc11.

Close ✕

```
DEBUG 2017-01-24 22:03:42,485 [pulse-server - 172.17.0.39_dc0@172.17.0.39:9010 *] [RMI TCP Connection(6)-172.17.0.1]
AppStateManagerImpl.transition:511 java.lang.Throwable: Application u4cgSmvFYd0nbHqf87tITg transitioning
STARTING to STARTED because successfully started application.
```

```
2 ▼ Uncaught TypeError: Cannot read property 'get' of undefined RulesP
  _buildSelectOptionFromFieldName @ RulesProjectAddEdit.jsx:211
  _onChangeSelectedTargetVariable @ RulesProjectAddEdit.jsx:84
  fireChangeEvent @ react-select.js?ts=dev:456
  setValue @ react-select.js?ts=dev:407
  selectValue @ react-select.js?ts=dev:413
  executeDispatch @ react.js?ts=dev:3303
  executeDispatch @ react.js?ts=dev:15715
  forEachEventDispatch @ react.js?ts=dev:3291
  executeDispatchesInOrder @ react.js?ts=dev:3312
  executeDispatchesAndRelease @ react.js?ts=dev:2685
  forEachAccumulated @ react.js?ts=dev:17606
  processEventQueue @ react.js?ts=dev:2892
  runEventQueueInBatch @ react.js?ts=dev:10753
  handleTopLevel @ react.js?ts=dev:10779
  handleTopLevelImpl @ react.js?ts=dev:10865
  perform @ react.js?ts=dev:16689
  batchedUpdates @ react.js?ts=dev:9205
  batchedUpdates @ react.js?ts=dev:14920
  dispatchEvent @ react.js?ts=dev:10959
```

Biggest Challenges:

- Create independent and self contained tests

```

/**...*/
@BeforeClass
public static void beforeClass() {...}

/**...*/
@AfterClass
public static void afterClass() {...}

/**...*/
@Before
public void setUp() throws Exception {...}

/**...*/
@After
public void tearDown() throws Exception {...}

/*
 *          FEATURES TESTS
 *
 */

/**...*/
@{...}
public void defaultTest() {...}

/**...*/
@Test
public void createEmptyDataSourceTest() {...}

/**...*/
@Test
public void createNoHeaderTest() {...}

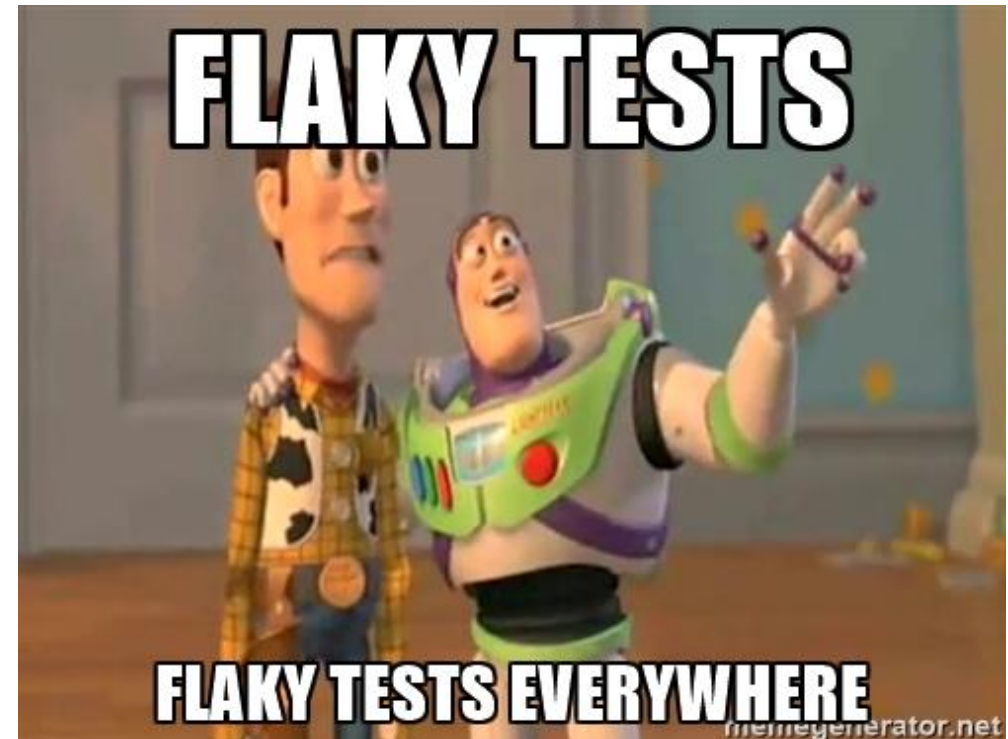
/**...*/
@{...}
public void wizardNavigationTest() {...}

```

Biggest Challenges:

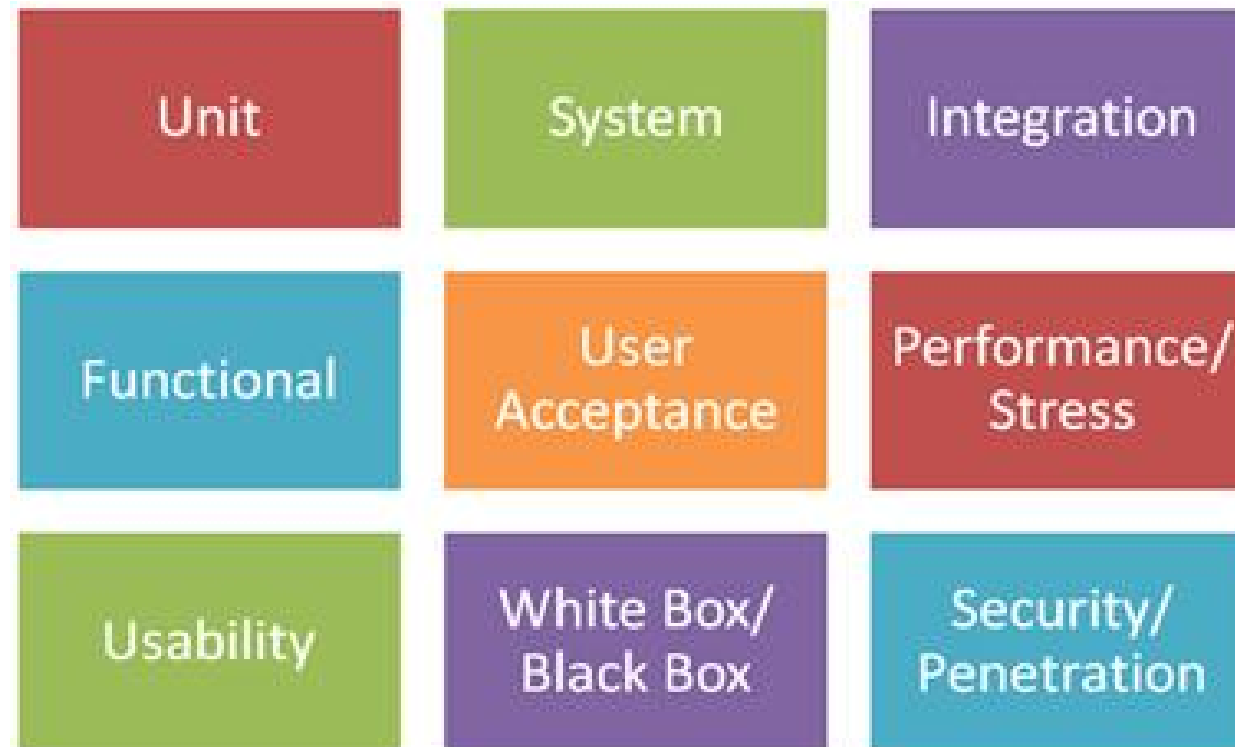
- Fight against flakiness

●	#1447	Mar 21, 2014 4:31:10 PM
●	#1446	Mar 21, 2014 3:31:10 PM
●	#1445	Mar 21, 2014 2:31:10 PM
●	#1444	Mar 21, 2014 1:31:10 PM
●	#1443	Mar 21, 2014 12:31:10 PM
●	#1442	Mar 21, 2014 11:31:10 AM
●	#1441	Mar 21, 2014 10:31:10 AM
●	#1440	Mar 21, 2014 9:31:10 AM
●	#1439	Mar 21, 2014 8:31:10 AM



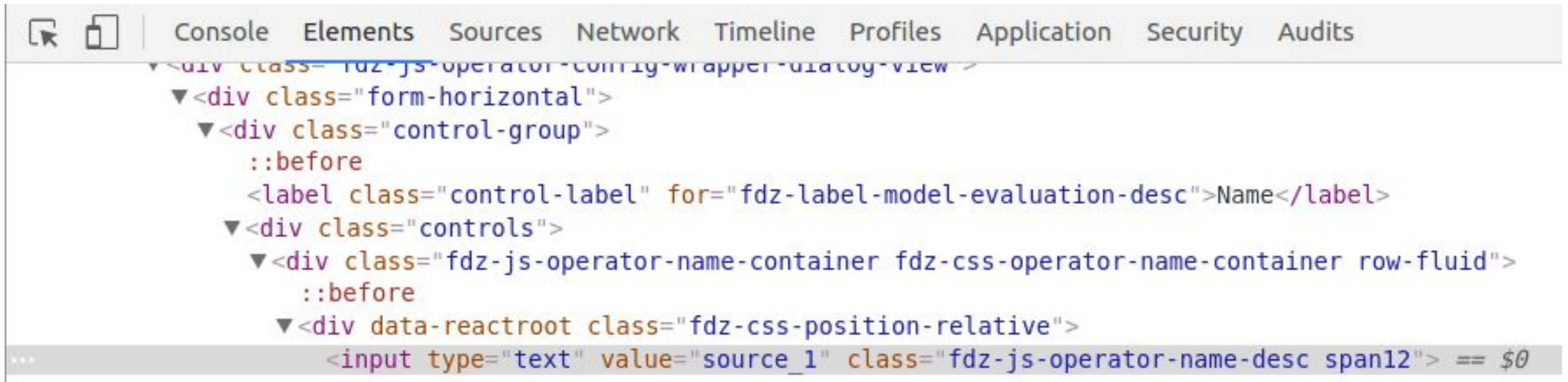
Biggest Challenges:

- Identify the correct type of test



Biggest Challenges:

- Have our own QA front-end element classes



```
▼ <div class="fdz-js-operator-config-wrapper-dialog-view">
  ▼ <div class="form-horizontal">
    ▼ <div class="control-group">
      ::before
      <label class="control-label" for="fdz-label-model-evaluation-desc">Name</label>
    ▼ <div class="controls">
      ▼ <div class="fdz-js-operator-name-container fdz-css-operator-name-container row-fluid">
        ::before
        ▼ <div data-reactroot class="fdz-css-position-relative">
          ... <input type="text" value="source_1" class="fdz-js-operator-name-desc span12"> == $0
```

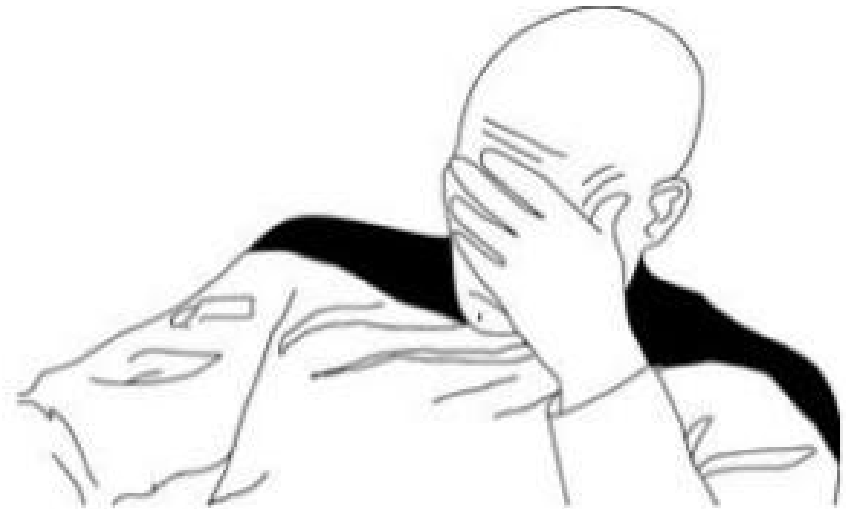
Biggest Challenges:

- Keep code base without ignored tests

“Oh, you have to invoke Maven with this flag that turns off those tests – they are tests that no longer work!”

James Coplien

Why Most Unit Testing is Waste



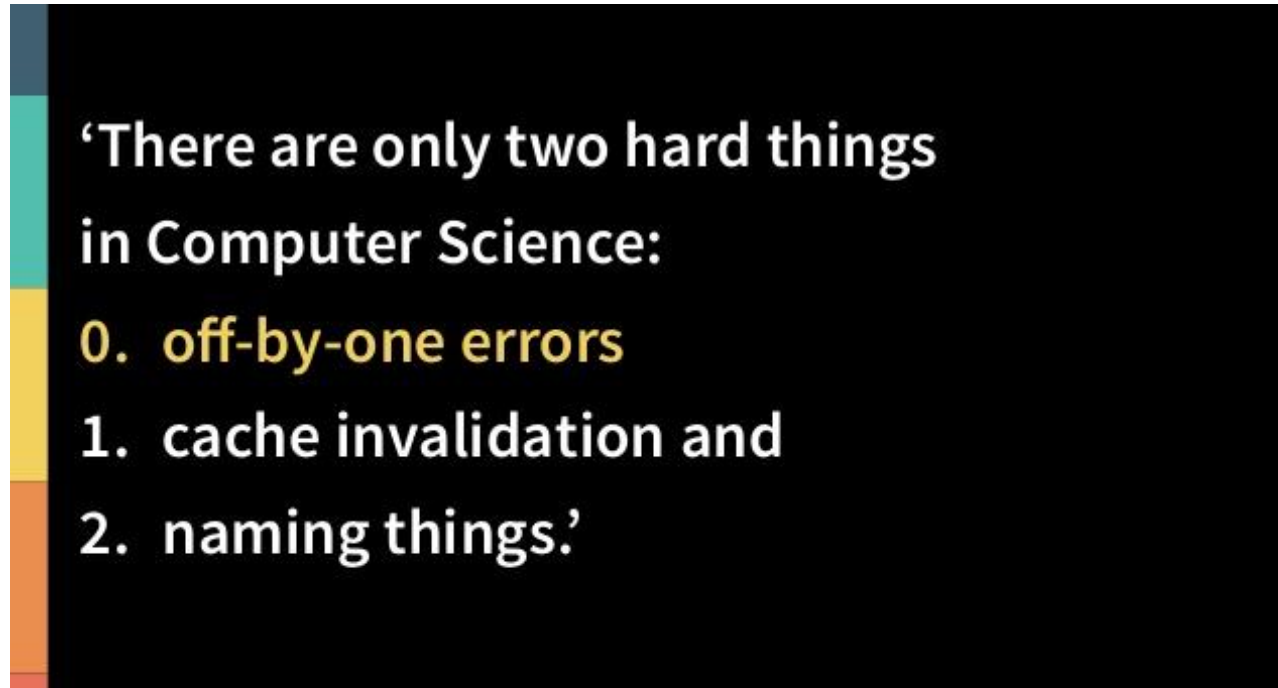
Biggest Challenges:

- Running tests in Windows



Biggest Challenges:

- Naming



Quote by Phil Karlton.

Slide taken from:

<http://www.slideshare.net/pirhilton/how-to-name-things-the-hardest-problem-in-programming>

Q&A



#feedzaihero

A scenic photograph of the Golden Gate Bridge in San Francisco at sunset. The bridge's iconic red-orange towers and suspension cables are silhouetted against a warm, orange-hued sky. The bridge spans across the water, with a large rock in the foreground and a rocky shoreline on the left. The water is calm, reflecting the soft light of the setting sun.

Thank You!

feedzai

PSTQB
ASSOCIAÇÃO PORTUGUESA
DE TESTES DE SOFTWARE

Bernardo Marques
bernardo.marques@feedzai.com

Ricardo Lopes
ricardo.lopes@feedzai.com

Test Our Coffee
31 January 2017 - Coimbra