HEISENBUG

Flaky tests

Порядок имеет значение

Андрей Солнцев

codeborne

Обо мне

- Программяка
- Codeborne
- Юнит-тесты, TDD, чистый код
- Опен-сорс
- Создатель **Selenide**
- Таллинн, Эстония

Andrei Solntsev

twitter.com/asolntsev

asolntsev.github.io/ru/video

В предыдущих сериях

“Flaky tests”

Гейзенбаг 2017

- Примеры
- Профилактика

“Flaky tests 2.0”

DelEx 2019

- Анимация
- Видосики

https://asolntsev.github.io/ru/video/

В предыдущих сериях

“Flaky tests”

Гейзенбаг 2017

- Примеры
- Профила

“Flaky tests 2.0”

DelEx 2019

ия

ки

Сегодня:

- **Порядок событий** во время теста

Flaky test

- это тест, который падает *иногда*

- Тратите время!
- Пропускаете реальные ошибки
- Пропускаете usability issues
- Теряется доверие к тестам

Ой, 30 тестов упало.
Надо изучить!

... Впрочем,
П.....У..... !

Ой, 30 тестов упало.
Надо изучить!

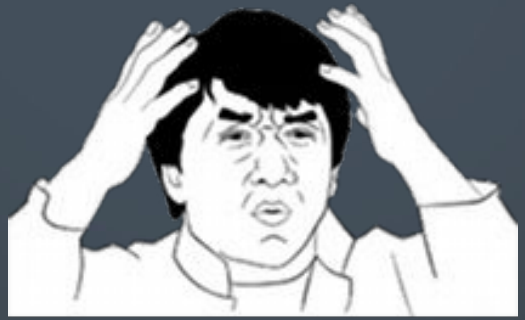
... Впрочем,
Перезапущу!

Каждый билд нужно изучать

вручную

И вы называете это автоматизацией?

Индустрия в опасности!



Мы **боимся**
флейки тестов

Ребята помогают нам закрыть глаза на проблему

Gradle

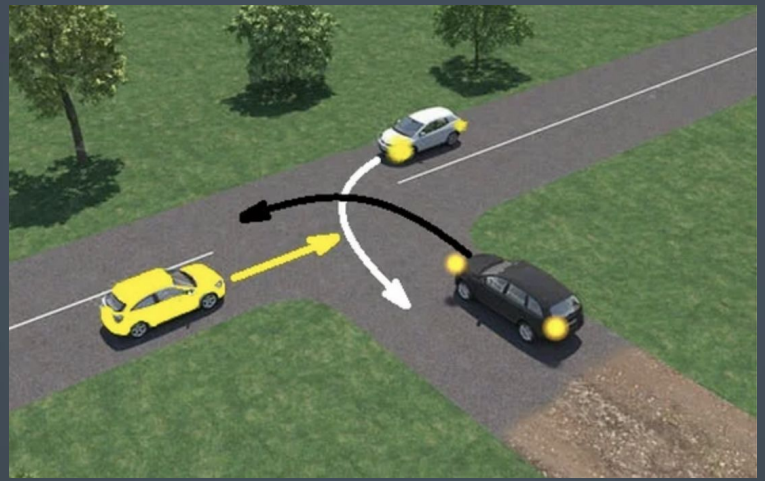
Test Retry plugin

TestNG

BDD

ReportPortal.io

Порядок событий



Как вы это протестируете?

1. Клик “Сохранить”
2. Надпись “Сохраняю...”
3. Клик “Ок”
4. Надпись пропадает

Тест:

1. “Сохранить”
2. “Сохраняю...”
3. “Ок”
4. Пропадает

```
$(“.save”).click();

$(“.saving”).should(appear);

$(“.ok”).click();

$(“.saving”).should(disappear)
```

Тест:

1. “Сохранить” — `$(“.save”).click();`
2. “Сохраняю...” — `$(“.saving”).should(appear);`
3. “Ок
4. Про

selenide.org

- Автоматические ожидания
- Решает 90% флейки тестов

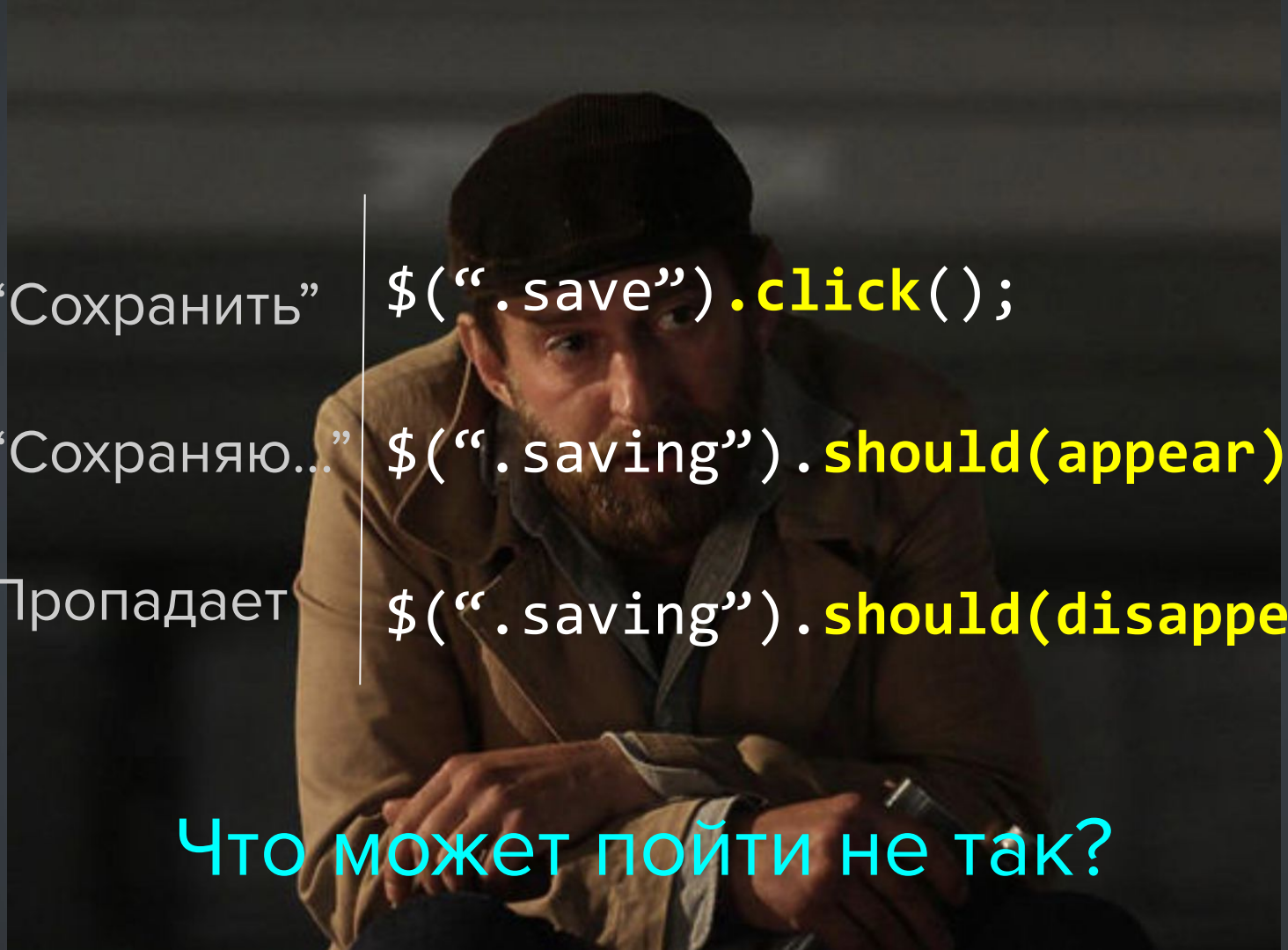
`$(“.saving”).should(disappear)`

Усложняем задачу

1. Клик “Сохранить”
2. Надпись “Сохраняю...” (N секунд)
3. Надпись пропадает САМА

Тест:

1. “Сохранить”	`$(“.save”).click();`
2. “Сохраняю...”	`$(“.saving”).should(appear);`
3. Пропадает	`$(“.saving”).should(disappear)`



1. “Сохранить” `$(“.save”).click();`
2. “Сохраняю...” `$(“.saving”).should(appear);`
3. Пропадает `$(“.saving”).should(disappear)`

Что может пойти не так?

Вариант 1: шустрый браузер

1. “Сохранить”
2. “Сохраняю...”
3. Пропадает

```
$(“.save”).click();
```

FAILED

```
$(“.saving”).should(appear);
```

```
$(“.saving”).should(disappear)
```

Вариант 2: тормознутый браузер

1) “Сохранить”

```
$(“.save”).click();
```

2) “Сохраняю...”

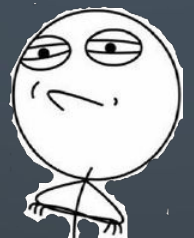
```
$(“.saving”).should(appear);
```

FAILED

```
$(“.saving”).should(disappear)
```

> 4 сек

3) Пропадает



```
Configuration.timeout = 42000;
```

Компромисс 1

1. “Сохранить”
2. “Сохраняю...”
3. Пропадает

```
$(“.save”).click();
```

FAILED

```
$(“.saving”).should(appear);
```

~~$(“.saving”).should(disappear)~~

Идём на компромисс.
Отказываемся от этой проверки.

Компромисс 2

1. “Сохранить”

```
$(“.save”).click();
```

~~`$(“.saving”).should(appear);`~~

2. “Сохраняю...”

```
$(“.saving”).should(disappear)
```

НЕ НАХОДИТ БАГУ

~~3. Пропадает~~

Идём на компромисс.
Отказываемся от этой проверки.

Итак,
как *надёжно* проверить
кейс
“Сохранить”?

Как *надёжно* проверить
кейс “Сохранить”?

Правильный ответ:

НИКАК!

(без хитрых хаков)

Почувствуй разницу:

1. **Клик** “Сохранить”
2. Надпись “Сохраняю...”
3. **Клик** “Ок”
4. Надпись пропадает

Мы контролируем

1. **Клик** “Сохранить”
2. Надпись “Сохраняю...”
3. **Проходит время**
4. Надпись пропадает

Мы НЕ контролируем

Почувствуй разницу:

1. **Клик** “Сохранить”
2. Надпись “Сохраняю...”
3. **Клик** “Ок”
4. Надпись пропадает

Тест надёжный

1. **Клик** “Сохранить”
2. Надпись “Сохраняю...”
3. **Проходит время**
4. Надпись пропадает

Тест ненадёжный

Порядок в тесте:

1. **Клик** “Сохранить”
2. Надпись “Сохраняю...”
3. **Клик** “Ок”
4. Надпись пропадает

1. **Act**
2. Assert
3. **Act**
4. Assert

...

Unit-test:

1. Act
2. Assert
3. Assert
4. Assert

UI test:

1. **Act**
2. Assert
3. **Act**
4. Assert
5. ...

Этот принцип - на каждом шагу!

Пример:

Автосохранение

видео

Логи AUT и теста:

```
10:59:12,865  UITest   click.before <a>1 Contact and Addresses</a>
10:59:13,121  UITest   click.before <button>Start filling</button>
10:59:13,686  UITest   click.before <button>Forward</button>
10:59:13,851  UITest   click.before <a>5 Preview</a>
10:59:13,905  request  POST /loanapplications/mortgage/save ...
10:59:14,010  UITest   click.after <a>5 Preview</a>
```

Когда тест зелёный:

```
click.before <a>1 Contact and Addresses</a>
click.before <button>Start filling</button>
click.before <button>Forward</button>
click.before <a>5 Preview</a>
POST /loanapplications/mortgage/save ...
click.after <a>5 Preview</a>
```

Когда тест красный:

```
click.before <a>5 Preview</a>
click.after <a>5 Preview</a>
POST /loanapplications/mortgage/save ...
```

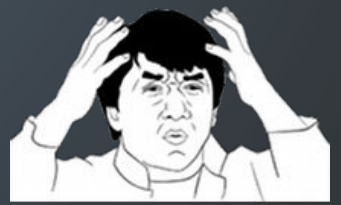
Хорошая идея -
совместить логи
теста и приложения

Но как же решить
нашу задачу
с автосохранением?

Как-то контролировать порядок!

1. Клик “страница 1”
2. *Дождаться автосохранения*
3. Клик “страница 2”
4. *Дождаться автосохранения*
5. …

Но как?



Как дождаться автосохранения?

1. Надпись “сохраняю...”
2. Иконка типа “дискета”
3. Через JavaScript “jQuery.active”

Нет.

Всё это мы **не контролируем**.

В AUT: добавляем признак

```
function autosaveDraft() {
    self.form.removeClass('autosaved');
    self.form.addClass('autosaving');

   $.post(...)
    .then(() => {
       self.form.addClass('autosaved');
   })
```

В тесте: проверяем признак

А вдруг он остался от предыдущего автосохранения?

```
// заполняем форму, кликаем “Дальше”

$("#form").shouldHave(cssClass("autosaving"));
$("#form").shouldHave(cssClass("autosaved"));
```

В тесте: проверяем признак

```
// Arrange:
executeJavaScript("document.getElementById('form
').classList.remove('autosaving')");

// Act: заполняем форму, кликаем “Дальше”

$("#form").shouldHave(cssClass("autosaving"));
$("#form").shouldHave(cssClass("autosaved"));
```

Unit-test

1. Arrange
2. Act
3. Assert

AAA -
на каждом шаге!

UI test

1. **Arrange**
2. **Act**
3. Assert

Шаг 1

4. **Arrange**
5. **Act**
6. Assert

Шаг 2

7. ...

https://seleniumcamp.com/talk/arrange-mazafaka/

Arrange!

Перед каждым шагом:

1. А всё ли готово?
2. А всё ли закончилось на предыдущем шаге?
3. А обнулилось ли состояние?
4. А не грузится ли там в углу какая-нибудь предыдущая хреновина?

Хочешь поймать
флейки тест
-
думай как флейки тест!

Пример:

Три-четыре

видео

Есть flaky тест

```
$$("#devices .device").shouldHave(size(4));
```

- В 99.9% запусков он зелёный.
- Но иногда падает.
- Потому, что элементов не 4, а 3.

Изредка тест ломается:

```
List size mismatch: expected: = 4, actual: 3

Elements: [
  <tr class="device">BlackBerry</tr>,
  <tr class="device">Lumia</tr>,
  <tr class="device">HTC</tr>
]
```

Смотрим тест вдумчиво:

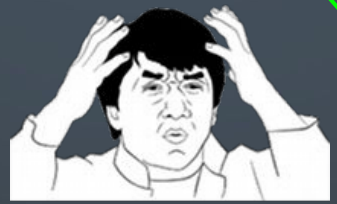
```
$(".remove-device").click();

$$("#devices .device").shouldHave(size(4));
$$("#devices .device").shouldHave(
  texts("BlackBerry", "Lumia", "HTC")
);
```

Смотрим тест вдумчиво:

```
$(".remove-device").click();

$$("#devices .device").shouldHave(size(4));
$$("#devices .device").shouldHave(
  texts("BlackBerry", "Lumia", "HTC")
);
```



Почему строк 4, но текста 3?

Исправленный тест:

```
$$("#devices .device").shouldHave(size(4));

$(".remove-device").click();

$$("#devices .device").shouldHave(size(3));
$$("#devices .device").shouldHave(
  texts("BlackBerry", "Lumia", "HTC")
);
```

Пример:

Ожидается -1 строк

Правильный тест (вроде бы?)

```
@Test
public void closeDeposit() {
  fastLogin("/deposits");
  int openDeposits = $$("#deposits tbody tr").size();

  // bla-bla-bla закрыть депозит

  $$("#deposits tbody tr").shouldHave(size(openDeposits - 1));
}
```

Правильный тест (вроде бы?)

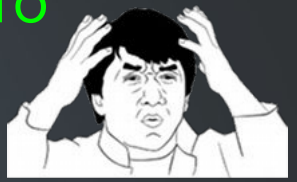
```
@Test
public void closeDeposit() {
  fastLogin("/deposits");                                    ARRANGE
  int openDeposits = $$("#deposits tbody tr").size();

  // bla-bla-bla закрыть депозит                             ACT

  $$("#deposits tbody tr").shouldHave(                       ASSERT
    size(openDeposits - 1)
  );
}
```

Но иногда он падает:

Но тогда что это за хрень?



```
List size mismatch: expected: = -1, actual: 3, collection: #deposits tbody tr
Elements: [
    <tr>73456 810 0 3331 0007779 Topupable 1 year 2.35% 25.04.2020 Active</tr>,
    <tr>Deferred Deferred 6 months 3.67% 01.07.2017 Deferred</tr>,
    <tr>Deferred Deferred 6 months 3.67% 01.07.2017 Deferred</tr>
]
```

Deposit closing successfully initiated

Их и правда 3:

Account	Type	Period	Interest	Amount	Until	State
73456 810 0 3331 0007779	Topupable	1 year	2.35%	123 456.00 ₽	25.04.2020	Active
Deferred	Deferred	6 months	3.67%	100 000.00 ₽	01.07.2017	Deferred
Deferred	Deferred	6 months	3.67%	122 222.00 ₽	01.07.2017	Deferred

Это халтура, а не ARRANGE!

```
@Test
public void closeDeposit() {
    fastLogin("/deposits");                                    ARRANGE
    int openDeposits = $$("#deposits tbody tr").size();

    // bla-bla-bla закрыть депозит

    $$("#deposits tbody tr").shoul
      size(openDeposits - 1)
    );
}
```

Список ещё не успел подгрузиться

Вот правильный ARRANGE:

```
@Test
public void closeDeposit() {
  fastLogin("/deposits");                                    ARRANGE
  int openDeposits = DB.select(“count(*) from deposits”);
  $$("#deposits tbody tr").shouldHave(size(openDeposits));

  // bla-bla-bla закрыть депозит

  $$("#deposits tbody tr").shouldHave(
    size(openDeposits - 1)
  );
```

Вот правильный ARRANGE:

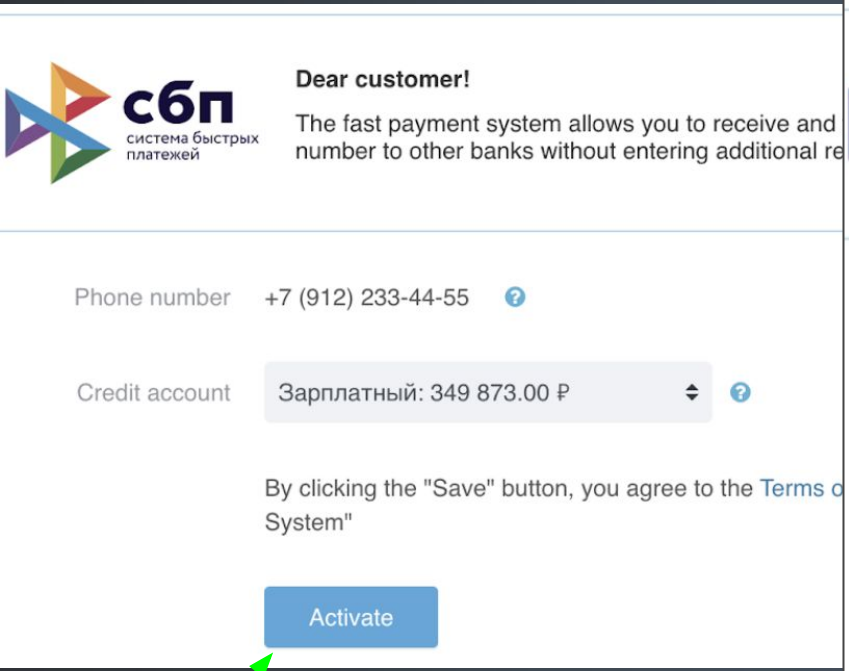
```
@Test
public void closeDeposit() {
  fastLogin("/deposits");
  int openDeposits = DB.select(“count(*) from deposits”);
  $$("#deposits tbody tr").shouldHave(size(openDeposits));
```

- Доверенный источник
  - Test data
  - SQL
  - API
  - Что угодно
- только НЕ ЧЕРЕЗ UI

Пример:

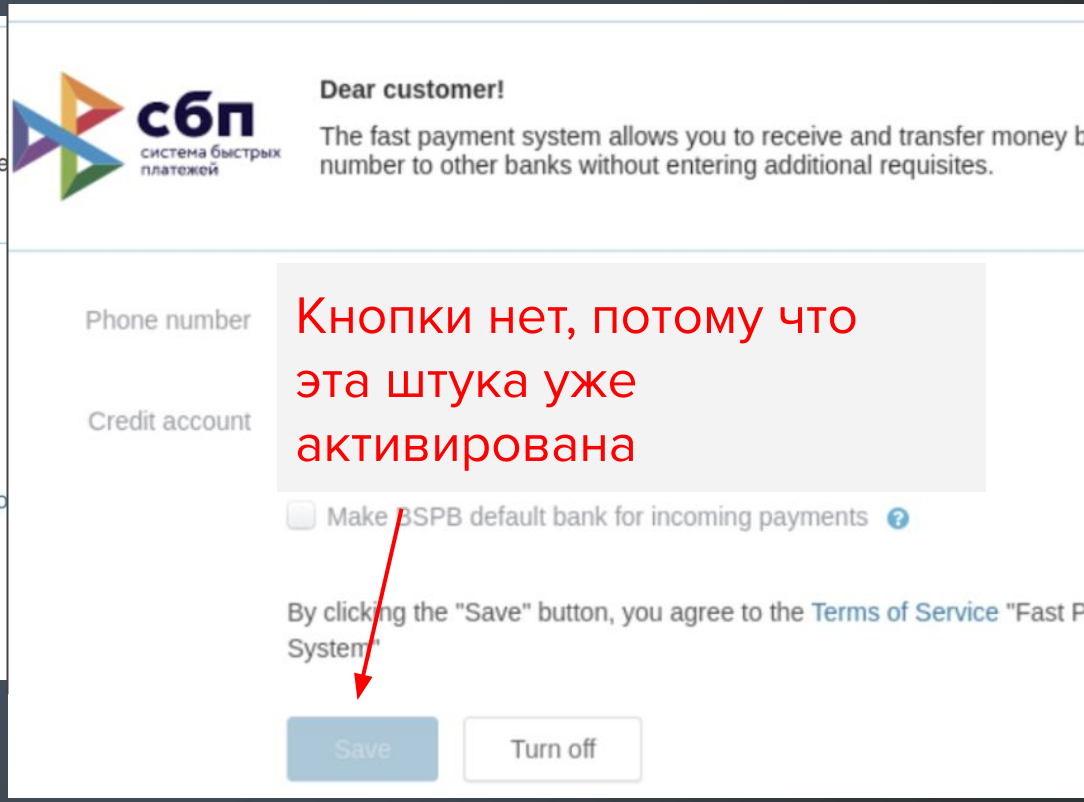
Сотри за мной нежно

Тест ок



Кнопка есть. Ок.

Тест failed



Предыдущий тест:

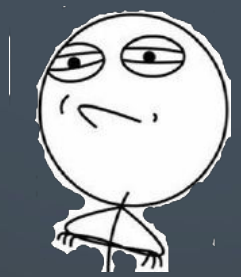
```
| ...
|.alert-success                  |should have(text 'done successfully') |PASS |
|#sbp-turnoff-confirmation-button |is(visible)                          |PASS |
|#sbp-turnoff-confirmation-button |click()                              |PASS |
|#sbp-turnoff-button              |click()                              |PASS |
+--------------------------------+---------------------+----------+----------|
```

А когда наш тест упал:

```
| ...
|.alert-success                  |should have(text 'done successfully') |PASS |
|#sbp-turnoff-confirmation-button |is(visible)                          |PASS |
+--------------------------------+---------------------+----------+----------|
```

Версия

Осталось
от предыдущего теста?



Как подчищаются данные:

```
@After
public void deleteRegistration() {
  if ($("#turnoff").is(visible)) {
    $("#turnoff").click();
  }
}
```

Неправильно:

```
@After
public void delete() {
   ...
}
```

Правильно:

```
@Before
public void delete() {
   ...
}
```

Неправильно:

```
@After
public void delete() {
  if ($(...)
      .is(visible)) {
    ...
  }
}
```

Правильно:

```
@Before
public void delete() {
  DB.executeSQL(
    “delete from orders”
  );
}
```

- Доверенный способ
  - SQL
  - API
  - Что угодно
- только НЕ ЧЕРЕЗ UI

Когда тестируешь UI

-

ты не можешь доверять UI

Андрей Солнцев
@asolntsev
asolntsev.github.io

ru.selenide.org

codeborne