twitter.com/asoln

10 причин моей ненависти

Andrei Solntse

codeborn

План

1. Пэджъ объжекты
2. Параллелизация
3. Параметризация
4. BDD
5. Красивые отчёты
6. TestNG
7. Оверинжиниринг


Чем плох
Пэдж обжект?

Пэдж обжект -
это хорошая идея

но

плохая практика

Классический пэдж обжект

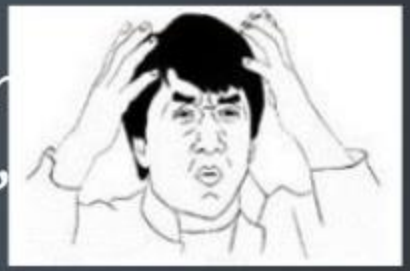
```
@Test
public void googleSearch() {
    GooglePage page = new GooglePage(webdriver);
    page.setQuery("selenide");
    page.submitQuery();
}
```

Классический пэдж обжект

```
public class GooglePage {
    @FindBy(name = “q”)
    private WebElement queue;

    public GooglePage(WebDriver webdriver) {
      PageFactory.initElements(webdriver, this);
    }

    public void setQuery(String query) {
        queue.clear();
        queue.sendKeys(query);
```

Классический пэдж обжект

```
public class GooglePage {
    @FindBy(name = “q”)
    private WebElement queue;

    public GooglePage(WebDriver webdriver
      PageFactory.initElements(webdriver,
    }
```



Бойлерплейт!

Selenide пэдж обжект

```
public class GooglePage {

    public void search(String query) {
        $(By.name(“q”))
            .val(queue)
            .pressEnter();
    }

}
```


Чем плох
@FindBy?

Чем плох @FindBy?

```
public class MyPage {
  @FindBy(xpath = "//div[1]/div[2]/input[@name=bd]")
  public WebElement birthday;
}


@Test {
  MyPage page = new MyPage(webdriver);
  birthday.sendKeys("19.03.1955");
}
```

типа

если изменится локатор,
придётся поменять только в одном месте.

но

меняется не только локатор.
Меняется поведение.

Что, если изменится не это:

```
public class MyPage {
  @FindBy(xpath = "//div[1]/div[2]/input[@name=bd]")
  public WebElement birthday;
}


@Test {
  MyPage page = new MyPage(webdriver);
  birthday.sendKeys("19.03.1955");
}
```

А вот это:

```
public class MyPage {
  @FindBy(xpath = "//div[1]/div[2]/input[@name=bd]")
  public WebElement birthday;
}

@Test {
  MyPage page = new MyPage(webdriver);
  birthday.sendKeys("19.03.1955");
}
```

1. Открыть календарь
2. Выбрать год
3. Выбрать месяц...

Упс! Придётс
менять кучу тестов

Объект - это НЕ

данные и операции с ними.

Объект - это **поведение**.

Настоящие пэдж-объек

Решение: поля -> методы

```
@Test {
  page.selectBirthday("19.03.1955");
}

public class MyPage {
  public void selectBirthday(String date) {
     ...
  }
}
```

- Публичный только метод
- Внутри - делай что хочешь

Это инкапсуляция, братан!

Можно приватное поле с @FindBy:

```
public class MyPage {
  @FindBy(xpath = "//div[1]/div[2]/input[@name=bd]")
  private WebElement birthday;

  public void selectBirthday(String date) {
      ...
  }
}
```

Можно приватное поле By:

```
public class MyPage {
  private By birthday = By.xpath(“...”);

  public void selectBirthday(String date) {
     $(birthday).sendKeys(date);
  }
}
```

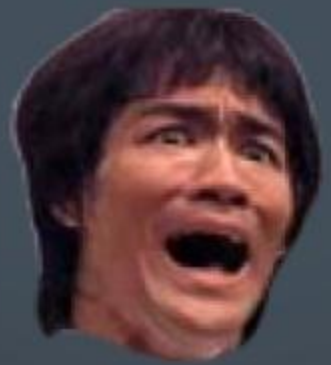
Можно приватное поле String:

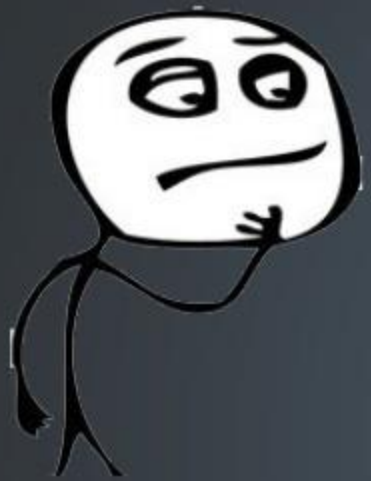
```
public class MyPage {
  private String birthday = “input[name=bday]”;

  public void selectBirthday(String date) {
     $(birthday).sendKeys(date);
  }
}
```

А можно...

ой, божечки!..

вообще **без поля.**





А что,

так можно было?

```
public class MyPage {
   public void selectBirthday(String date) {
      $(“#birthday”).sendKeys(date);
   }
}
```

Какие проблемы должен решать ПО

1. Плохие локаторы

```
public class MyPage {
  @FindBy(xpath = "//div[1]/div[2]/input[@name=bd]")
  private WebElement birthday;
}
```

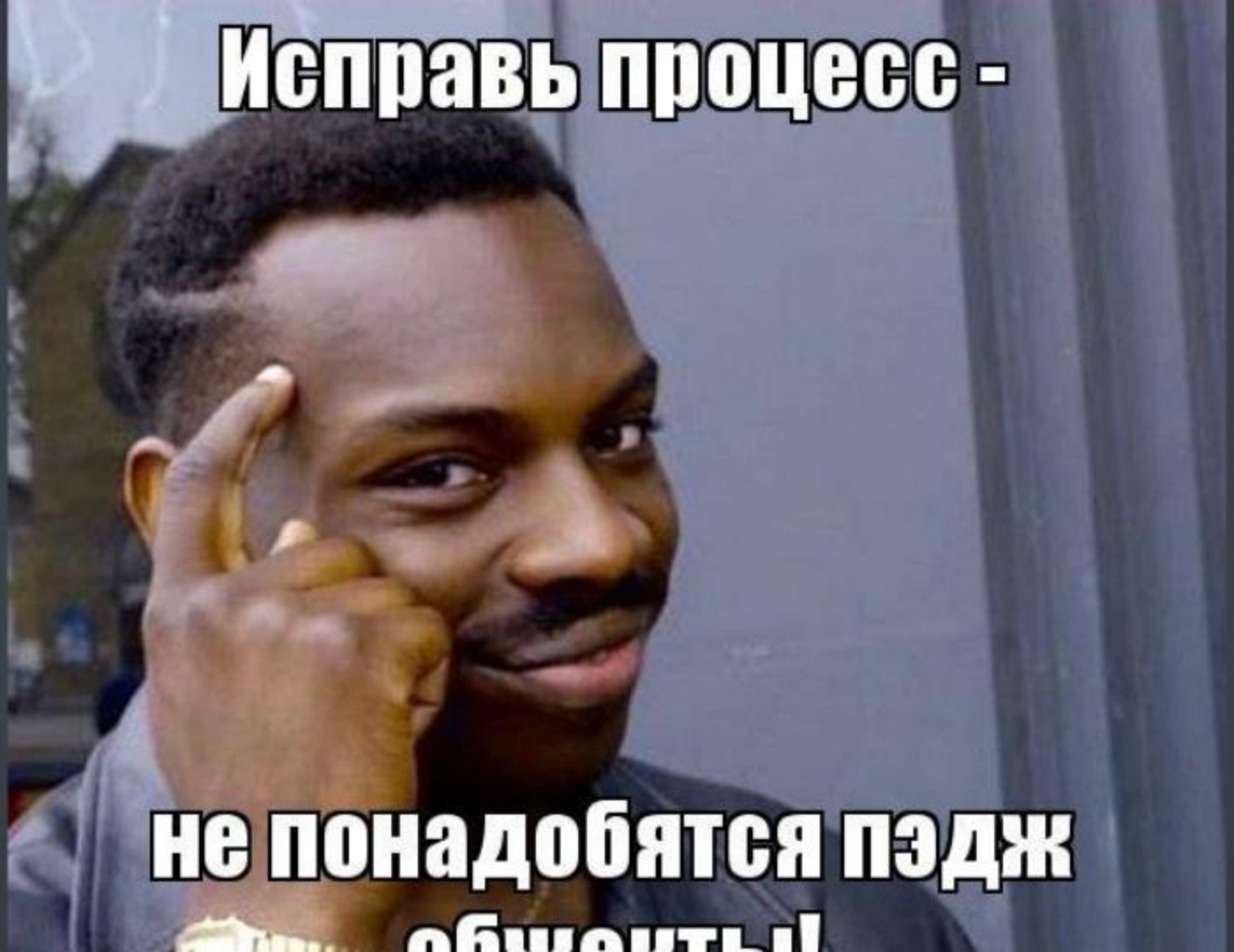
Какие проблемы должен решать ПО

1. Плохие локаторы
2. Дублирование локаторов

Но эти проблемы -
следствия
плохого процесса

Плохой процесс:

- Разрабы **не пишут тесты**
  - Нет юнит-тестов
- Через **UI** тестируется много кейсов
  - (но далеко не все!)
- Тесты медленные & нестабильные
  - (хотим отчёты и рипортпорталы)
- **QA не сотрудничают с разрабами**


Исправь процесс -
не понадобятся пэдж

При правильном процессе:

1. ~~Плохие локаторы~~
   - Разрабы делают хорошие локаторы

ПО больше не такой уж ценный:

```
public class MyPage {
  WebElement birthday = $(“#birthday”);
}
```

При правильном процессе:

1. ~~Плохие локаторы~~
   - Разрабы делают хорошие локаторы

---

2. ~~Дублирование локаторов~~
   - Большинство кейсов покрыто юнит-тестами.
   - UI-тестами покрывается только один-два сценария
   - Каждый локатор используется **мало**

"Пацан накодил - пацан протестил!"

Поговори с разрабом!

В чём он заинтересован?

- Он не любит **получать по рукам**
- Он не хочет, чтобы его **фича сломалась**
- Он не мечтает **фиксить багу** все выходные

Поговори с разрабом!

В чём он заинтересован?

- Он не любит **получать по рукам**
- Он не хочет, чтобы его **фича сломалась**
- Он не мечтает **фиксить багу** все выходные

Вы можете

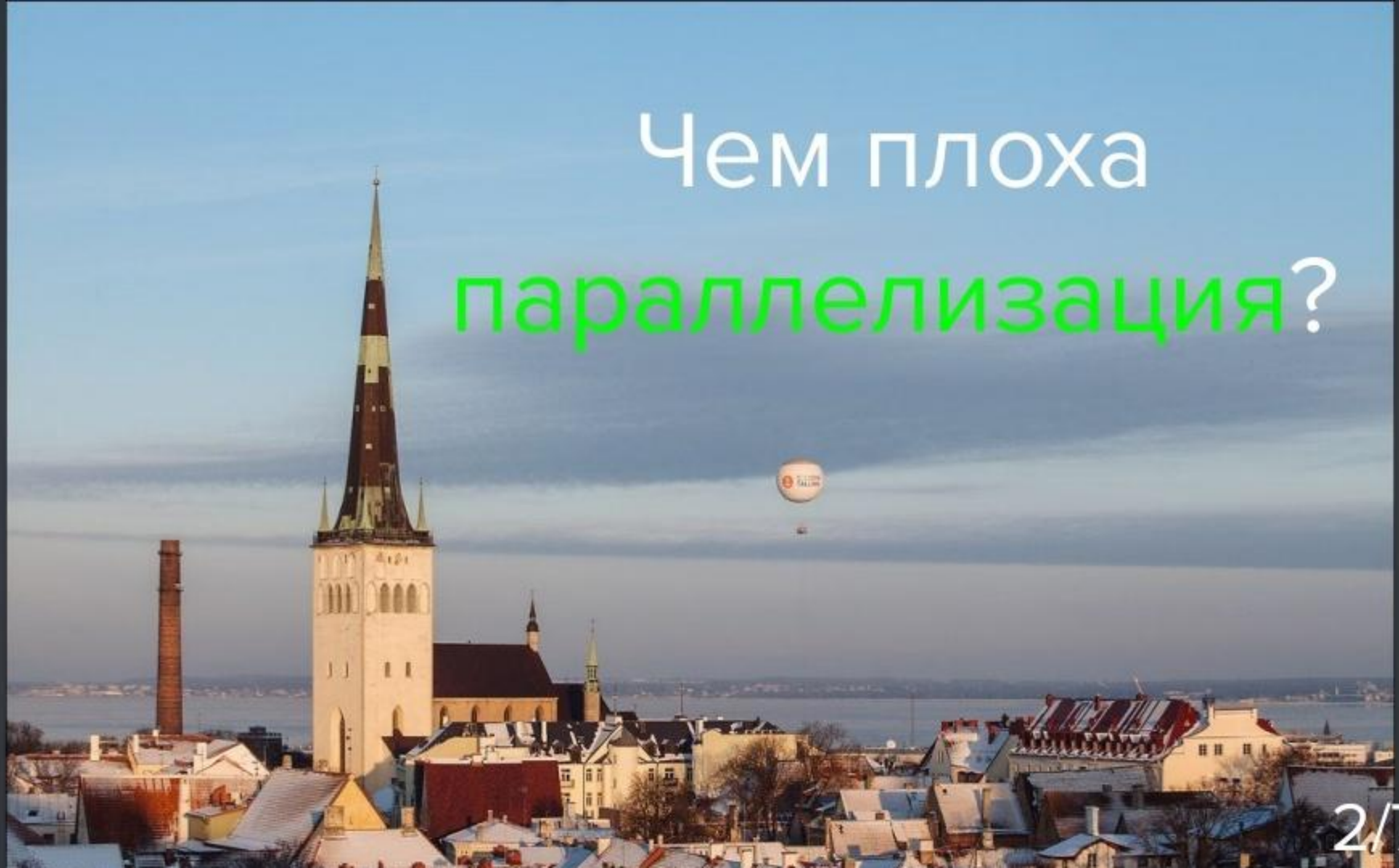
помочь

друг другу!

Крошка Ено

Пэдж обжекты -
техническое решение
для
нетехнической проблемы

А это всегда плохая идея!


Чем плоха
параллелизация?

Снова плохой процесс

- Нет юнит-тестов
- UI-тестами покрываются многие комбинации
  - Сложные сценарии
  - Медленные сценарии
  - Трудновоспроизводимые сценарии
- Testability на дне
  - Медленный тестовый стенд
- Тесты бегут 12 часов

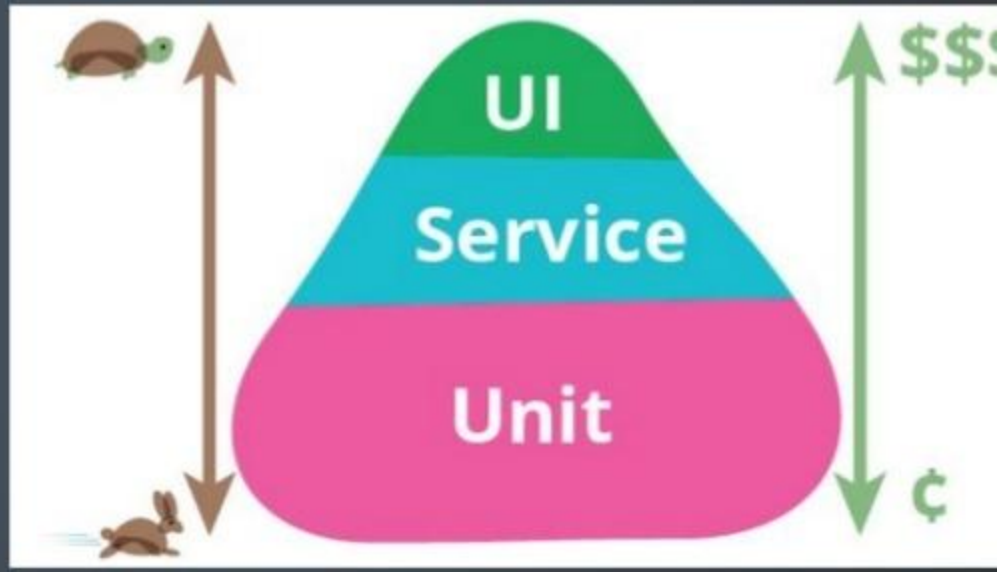
Снова плохой процесс

- Почему бы не запустить параллельно?
- В 100500 потоков?
- Ферма, йопта!
- Клауд, йопта!
- TestNG, йопта!
- Я теперь SDET, йопта!


Исправь процесс -
не понадобится

При правильном процессе:

- Юнит-тестов много
- UI-тестов мало
- Они бегут быстро



Экономически эффективный процесс тестирования

Test Pyramid

Чак Норрис

не параллелит тесты

Они и в одном потоке быстро бегут

Чак Норрис

не параллелит тесты

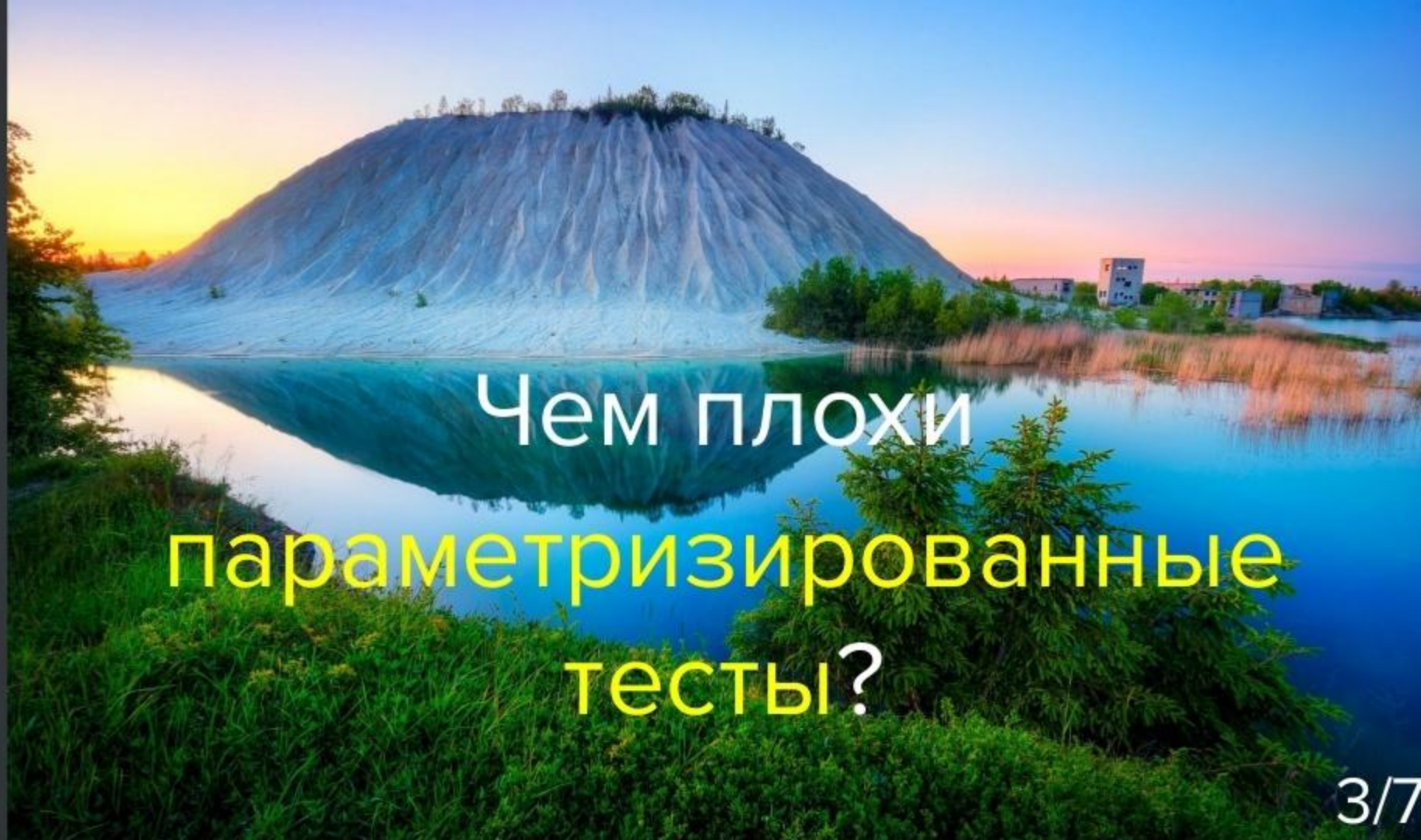
Чак Норрис

ПЕРПЕНДИКУЛЯРИТ тесты!

Параллелить, конечно, можно,
если всё остальное решено:

40 минут -> 10
норм!

14 часов -> 3 часа
не норм!


Чем плохи
параметризированные
тесты?

Болезнь индустрии

-

Слишком много через UI тесты

а

параметризированные тесты

делают это проще

Параметризированные тесты

:

Для unit-тестов

хорошо!

Для UI-тестов

плохо!

Чем плох BDD?

BDD -

это офигенная идея

но

она не работает

(в теории)

BDD - это

язык для

взаимодействия

заказчика и исполнителя

(на практике)

BDD - это

отчёты с картинками

которые

заказчик не читает

Это не про тесты

Это про разработку

Это не про отчёты

BDD

Это про взаимодействие
с заказчиком


Bullshit driven developmen

BDD: Болезни vs. симптомы

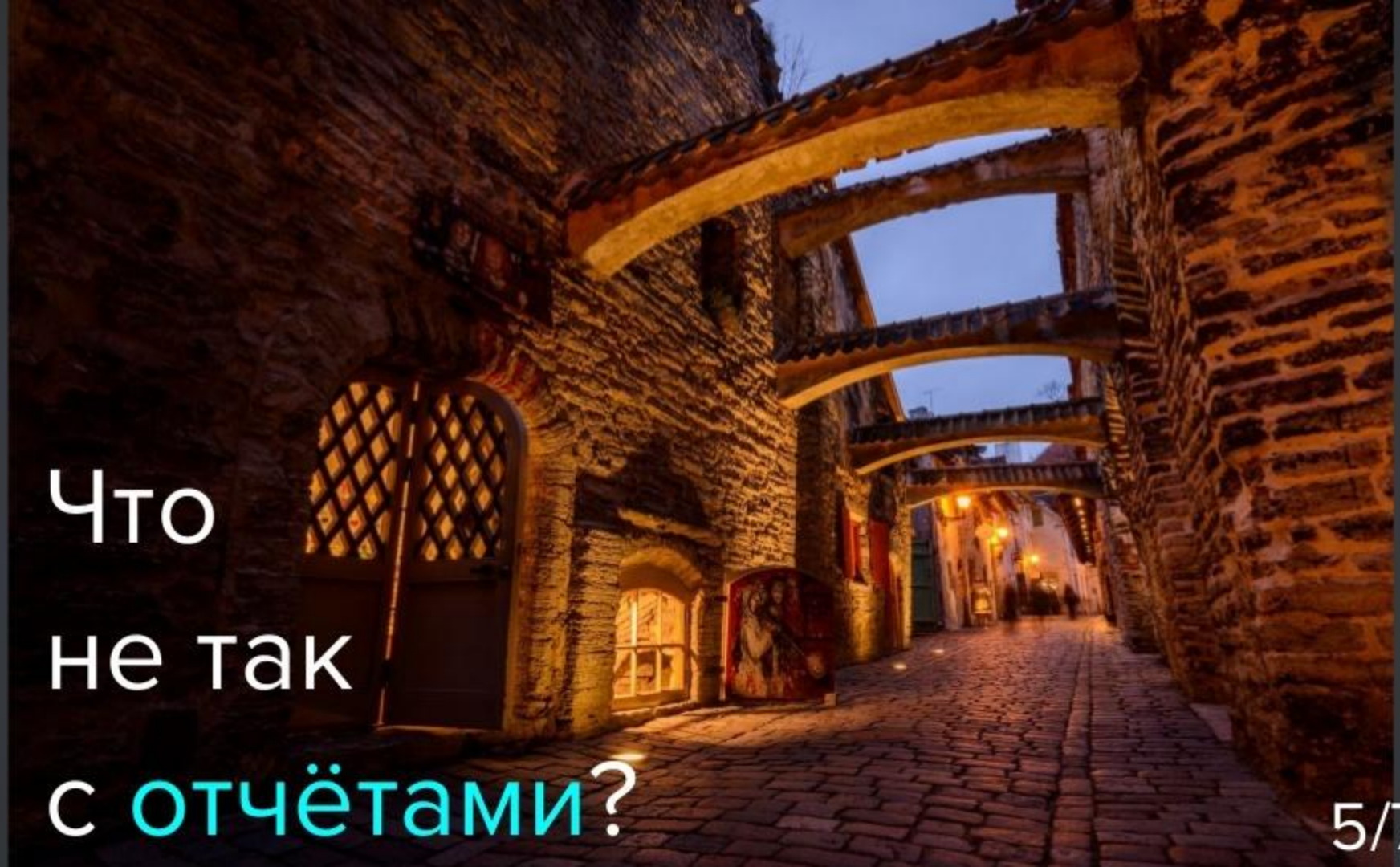
BDD обещает	Красивые отчёты
Менеджер надеется решить	Отсутствие доверия к команде. Непрозрачный рабочий процесс.
Решает ли BDD эту проблему?	Нет. Доверия как не было, так и нет.

BDD обещает	Красивые отчёты
Менеджер надеется решить	Показать своему боссу / заказчику, что "у нас всё круто", работа кипит.
Какая проблема на самом деле	Этот менеджер - промежуточный человек, ничего полезного в проект не привносит.
Решает ли BDD эту проблему?	Нет. Гнать таких в шею - вот решение.

BDD обещает	тесты смогут писать даже нетехнические люди
Менеджер надеется решить	Технарей мало, они всё не успевают. Хочется часть работы переложить на гуманитариев.
Решает ли BDD эту проблему?	Нет. Вся работа по поддержке степов и прочей БелиБерДы по-любому

P.S. Кстати, толковых гуманитариев тоже мало

А бестолковые только мешают


Что
не так
с отчётами?

- отчёт JUnit/TestNG - достаточно хорош
- Selenide добавляет скриншот+html

```
Element should be hidden {#gameWin}
Element: '<img class="gameOver" id="gameWin" src="img/thumbs-up.jpeg"></img>'

Screenshot: file:/.../hangman/build/reports/tests/1510751914648.0.png
Page source: file:/.../hangman/build/reports/tests/1510751914648.0.html
Timeout: 4 s.
```

Этого достаточно!


ЖОПА! Все тесты красные.
Я год не видел зелёного билда...


ЖОПА! Все тесты красные.
Я год не видел зелёного билда...


Так возьми
Report Portal!


ЖОПА! Все тесты красные.
Я год не видел зелёного билда...

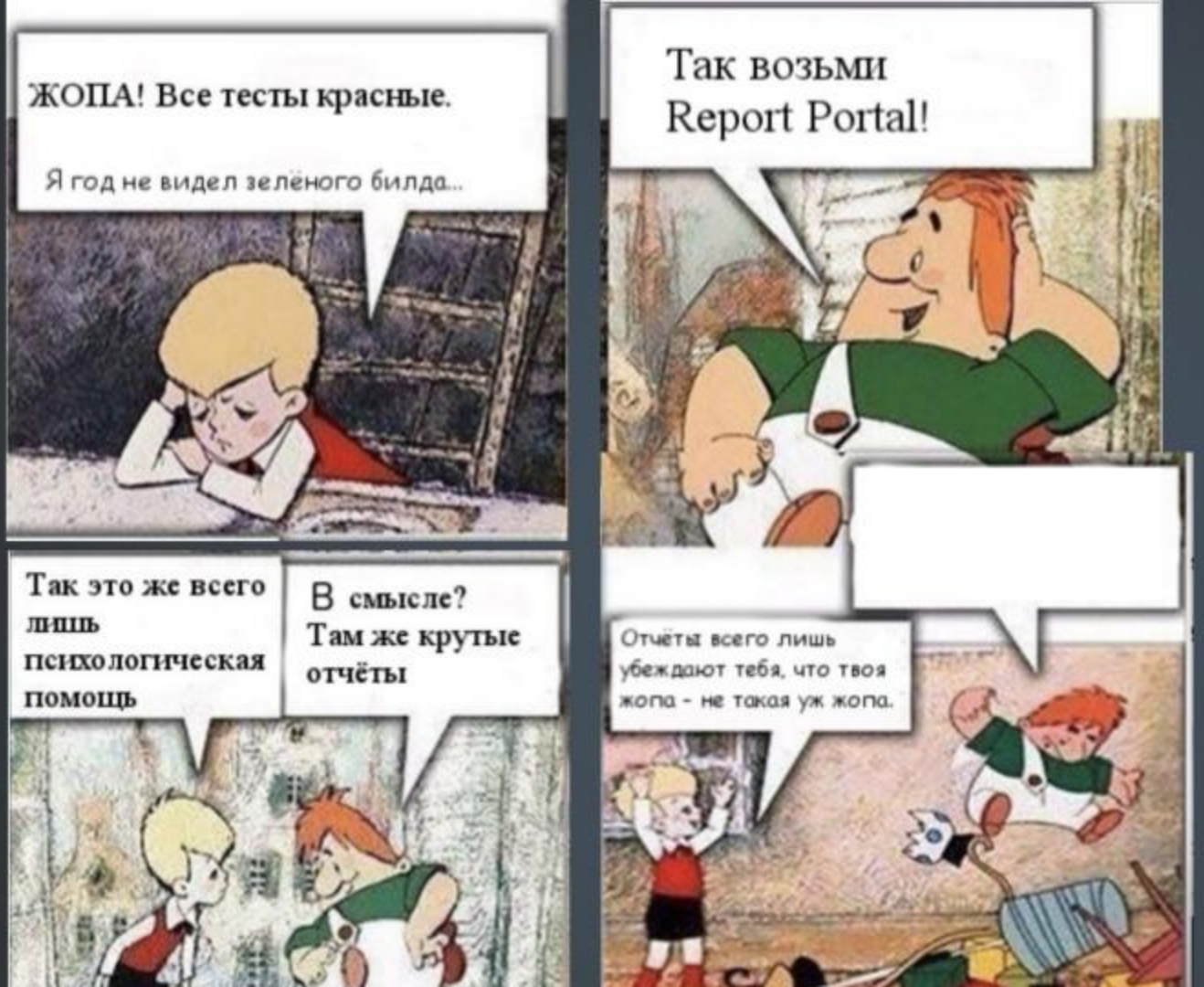

Так возьми
Report Portal!


Так это же всего
лишь
психологическая
помощь


ЖОПА! Все тесты красные.
Я год не видел зелёного билда...


Так возьми
Report Portal!


Так это же всего лишь психологическая помощь
В смысле? Там же крутые отчёты


ЖОПА! Все тесты красные.
Я год не видел зелёного билда...
Так возьми Report Portal!
Так это же всего лишь психологическая помощь
В смысле? Там же крутые отчёты
Отчёты всего лишь убеждают тебя, что твоя жопа - не такая уж жопа.


ЖОПА! Все тесты красные.
Я год не видел зелёного билда...
Так возьми Report Portal!
Да ты чо, пёс! Это МАШИН ЛЁНИНГ!
Отчёты всего лишь убеждают тебя, что твоя жопа - не такая уж жопа.
Так это же всего лишь психологическая помощь
В смысле? Там же крутые отчёты


Что не так с
TestNG?

Разница между JUnit и TestNG

JUnit	TestNG
Новый инстанс перед каждым тестом	Один инстанс на все тест-методы

```
public class MyTest {
  public MyTest() {
    System.out.println("Created MyTest");
  }

  @Test public void testA() {}
  @Test public void testB() {}
  @Test public void testC() {}
}
```

JUnit	TestNG
Created MyTest Created MyTest Created MyTest	Created MyTest

Почему это важно?

JUnit

```
public class MyTest {
  Foo foo = mock(Foo.class);
  Counter cnt = new Counter(0);
}
```

TestNG

```
public class MyTest {
  Foo foo;
  Counter cnt;

  @Befo
        setUp() {
      = mock(Foo.class);
    cnt = new Counter(0);
  }
}
```

Риск зависимых тестов!

Фичи TestNG (которых нет в JUnit)

- Конфигурация в XML
- Порядок тестов
- Зависимые тесты

Эти фичи - ЗЛО!

Не используй их, братан!

TestNG

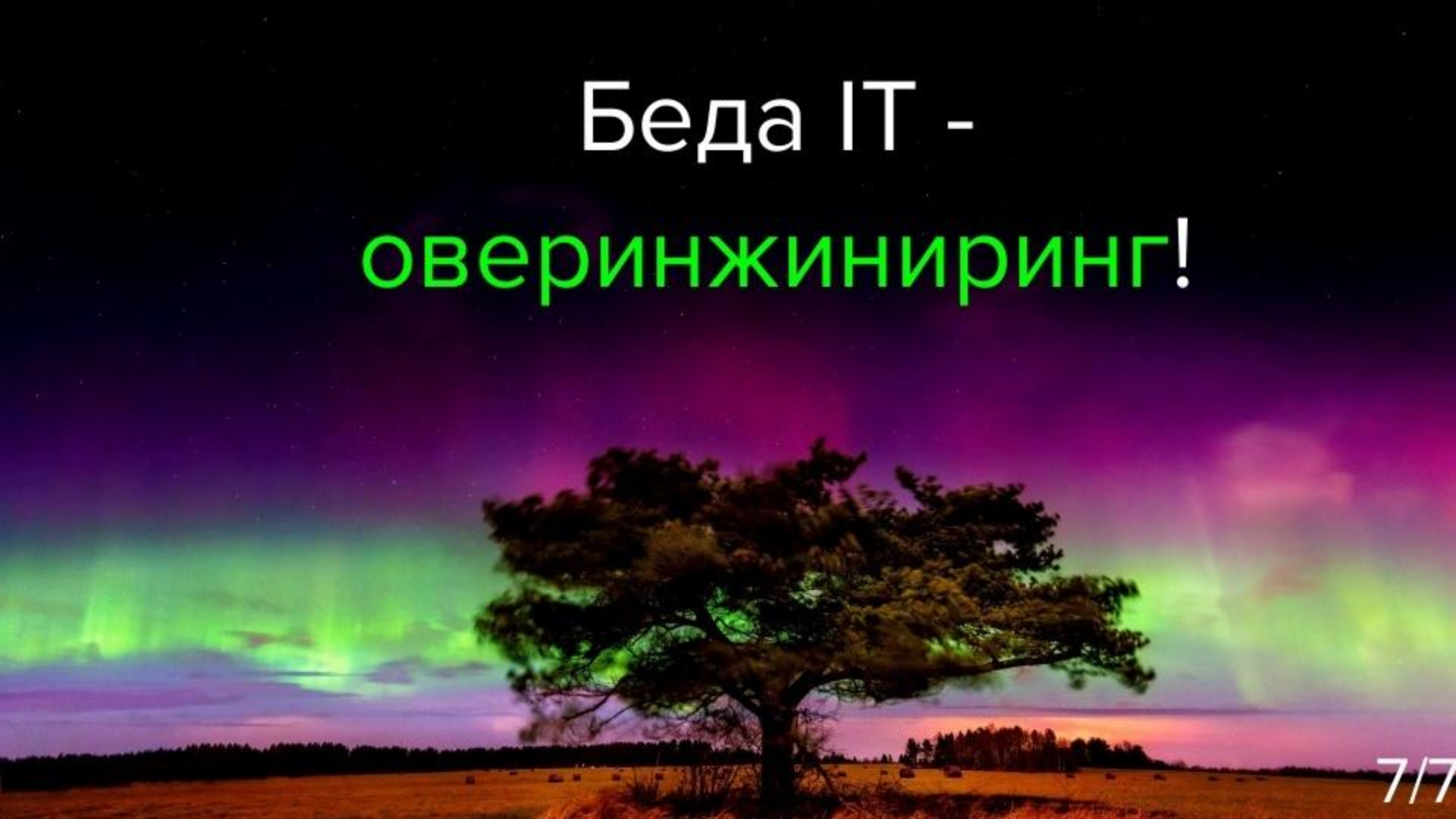
“Не хочешь злые фичи - не используй. Никто ж не заставляет.”

Бог с вами, хотите - используйте TestNG.

Но не называйте его “тулом для *продвинутого* интеграционного тестирования”.

Он для неэффективного, неуправляемого, медленного, нестабильного интеграционного тестирования.

Беда IT - оверинжиниринг!



Оверинжиниринг

```
protected void waitForElementClickable(By locator, Integer... timeOutInSeconds) {
  int attempts = 0;
  while (attempts < 2) {
    try {
       waitFor(elementToBeClickable(locator),
           (timeOutInSeconds.length > 0 ? timeOutInSeconds[0] : null));
       break;
    } catch (StaleElementReferenceException e) {
  }
  attempts++;
}

void waitFor(ExpectedCondition<WebElement> condition, Integer timeOutInSeconds) {
    timeOutInSeconds = timeOutInSeconds != null ? timeOutInSeconds : 30;
    WebDriverWait wait = new WebDriverWait(driver, timeOutInSeconds);
    wait.until(condition);
}
```

Оверинжиниринг

```
protected void waitForElementClickable(By locator, Integer... timeOutInSeconds) {
  int attempts = 0;
  while (attempts < 2) {
    try {
       waitFor(elementToBeClickable(locator),
           (timeOutInSeconds.length > 0 ? timeOutInSeconds[0] : null));
       break;
    } catch (StaleElementReferenceException e) {
  }
  attempts++;
}

void waitFor(ExpectedCondition<WebElement> condition, Integer timeOutInSeconds) {
    timeOutInSeconds = timeOutInSeconds != null ? timeOutInSeconds : 30;
    WebDriverWait wait = new WebDriverWait(driver, timeOutInSeconds);
    wait.until(condition);
}
```

Оверинжиниринг

```
protected void waitForElementClickable(By locator, Integer... timeOutInSeconds) {
  int attempts = 0;
  while (attempts < 2) {
    try {
       waitFor(elementToBeClickable(locator),
           (timeOutInSeconds.length > 0 ? timeOutInSeconds[0] : null));
       break;
    } catch (StaleElementReferenceException e) {
  }
  attempts++;
}

void waitFor(ExpectedCondition<WebElement> condition, Integer timeOutInSeconds) {
    timeOutInSeconds = timeOutInSeconds != null ? timeOutInSeconds : 30;
    WebDriverWait wait = new WebDriverWait(driver, timeOutInSeconds);
    wait.until(condition);
}
```

KISS : Keep it scuko simple!

```
protected void waitForElementClickable(By locator, int timeout)
  new WebDriverWait(driver, timeout).until(condition);
}

protected void waitForElementClickable(By locator) {
  waitForElementClickable(locator, 30);
}
```

KISS by Selenide

```
$(locator).click();

$(locator).shouldBe(visible).click();

$(locator).waitUntil(visible, 30_000).click();
```

Винтовка -

Оружие новичка

Выбор профессионала -

НОЖ!


Самая опасная идея
та,
которая кажется хорошей.
Douglas Crockford
JavaScript: The Good Parts

devclub.eu

Андрей Солнцев
@asolntsev
ru.selenide.org

codeborne

Спасибо за фотки:

1. https://stories.genvagula.com/my-magical-estonia-500aafd5b2c0
2. https://www.facebook.com/stan.vasilyev
3. https://i-love-tallinn.livejournal.com/306474.html
4. https://www.facebook.com/lyosha.razin
5. https://www.facebook.com/ttrk19/