

# 10 причин моей ненависти



Andrei Solntsev

 **codeborn**

# План

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

Чем плох  
Пэдж объект?

Пэдж обжект -  
это хорошая идея

НО

плохая практика

# Классический пэджд объект

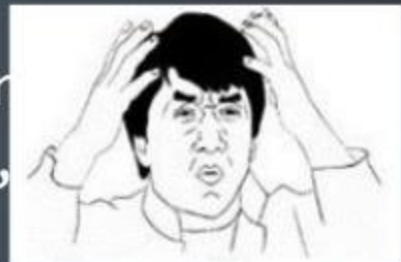
```
@Test
public void googleSearch() {
    GooglePage page = new GooglePage(webdriver);
    page.setQuery("selenide");
    page.submitQuery();
}
```

# Классический пэдждж объект

```
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 пэджд объект

```
public class GooglePage {  
    @FindBy(name = "q")  
    private WebElement queue;  
  
    public GooglePage(WebDriver webdriver,  
        PageFactory.initElements(webdriver,  
    }  
}
```



Бойлерплейт!

# Selenide пэдж объект

```
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! Придётся  
менять кучу тестов



Объект - это **НЕ**

*данные и операции с ними.*

Объект - это **поведение.**



# 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:

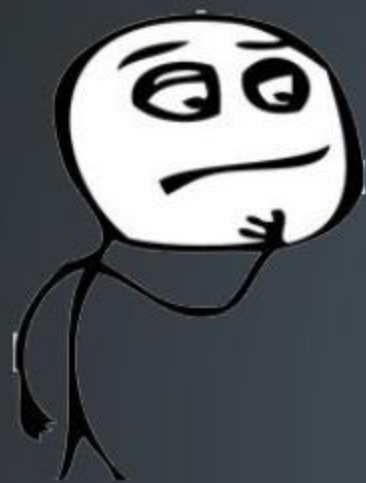
```
public class MyPage {  
    private By birthday = By.xpath("...");  
  
    public void selectBirthday(String date) {  
        $(birthday).sendKeys(date);  
    }  
}
```

# 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 локации
2. Дублирование локаций

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 разработками**



**Исправь процесс -**



**не понадобятся пэджд  
обшенты!**

# 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



"Пацан накодил  
- пацан протестил!"

**мало**

# Поговори с разработом!

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 разработом!

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**

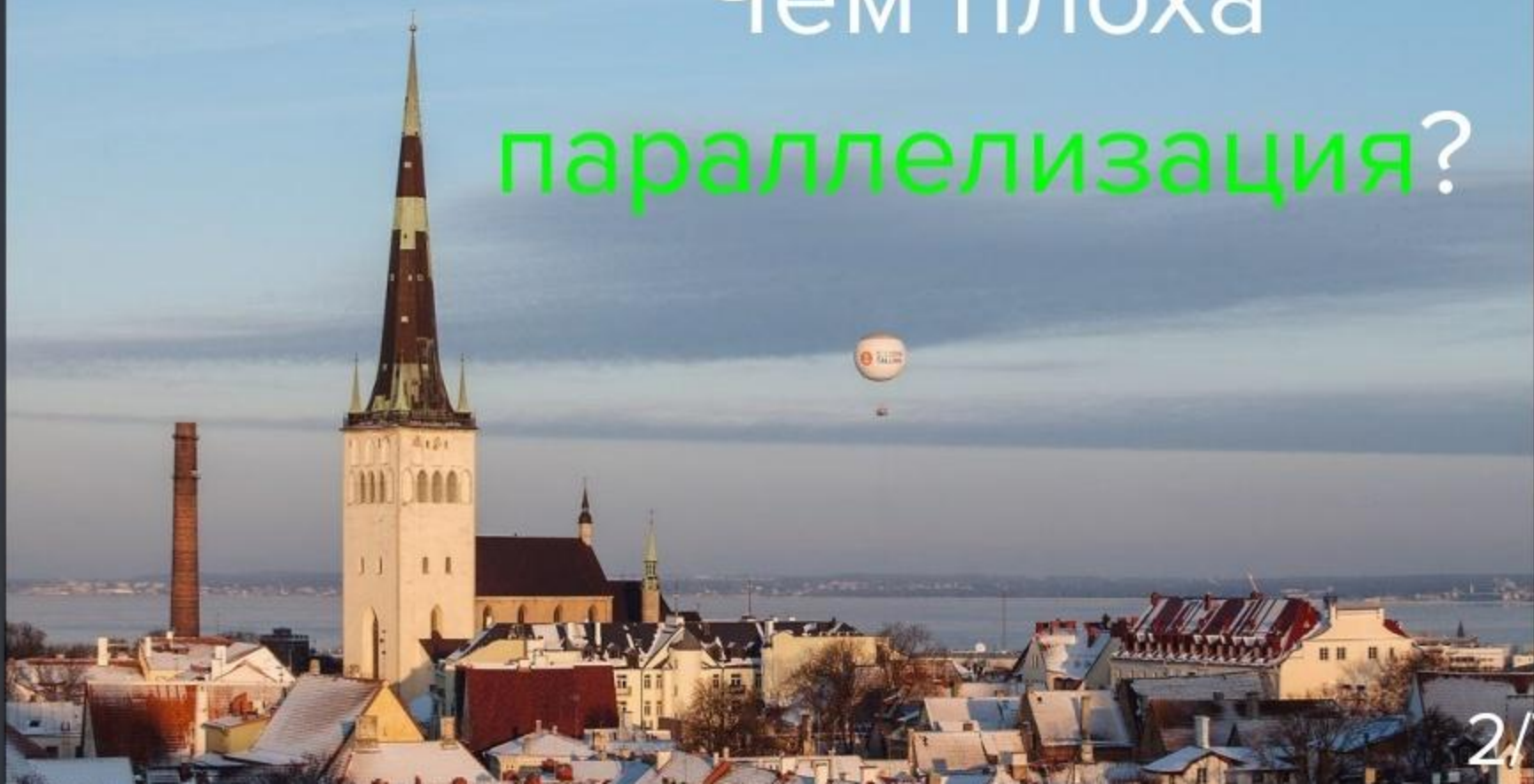
**друг другу!**



Пэдждж 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

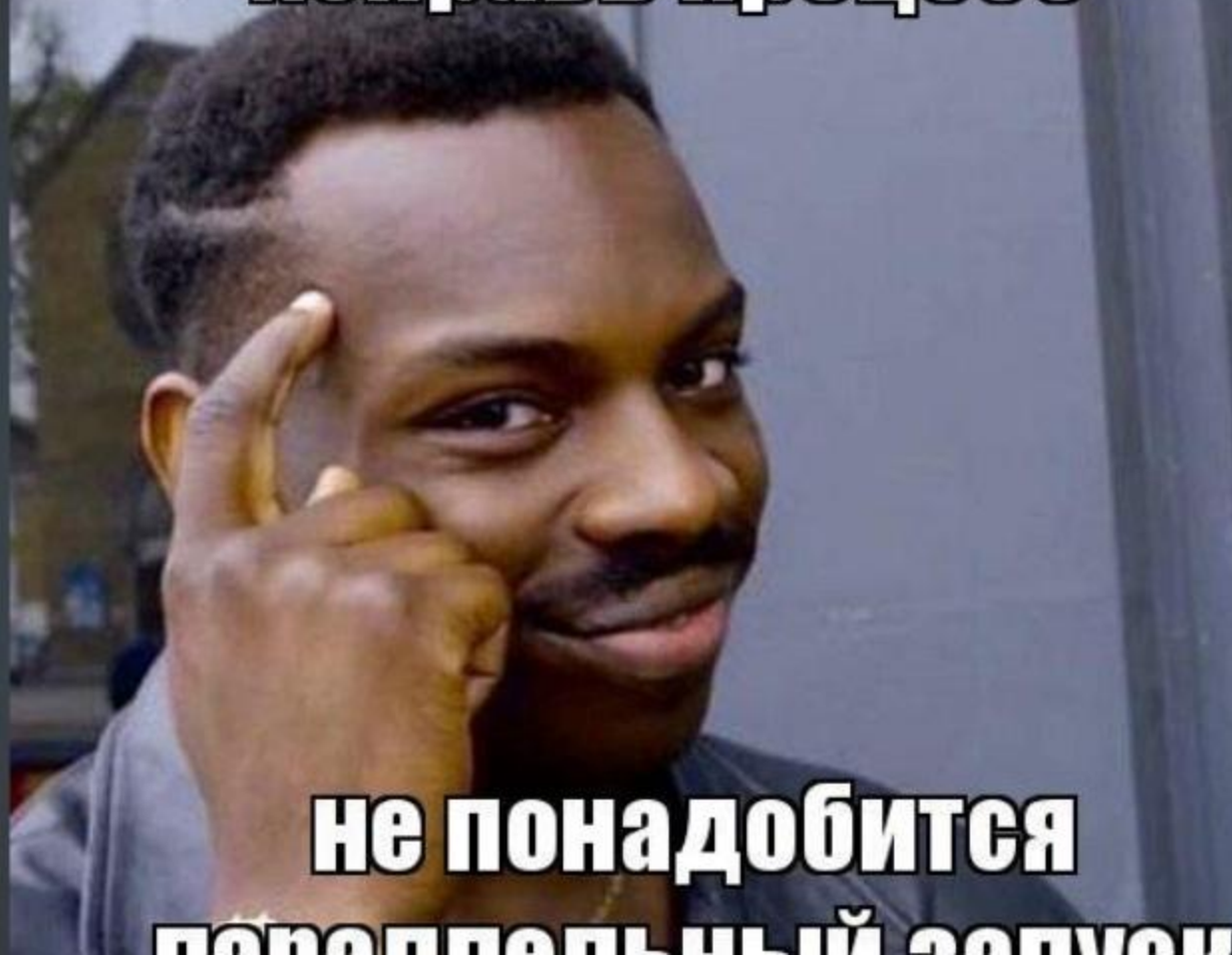


# Снова плохой процесс

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



**Исправь процесс -**



**не понадобится**

**порочный круг**

# 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  $\rightarrow$  10  
норм!

14 часов  $\rightarrow$  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?

YALINNA  
TELETORN

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



# 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**

A narrow, cobblestone street in a historic stone building at dusk. The street is flanked by thick stone walls. On the left, there is a large arched doorway with a wooden lattice pattern. Further down the street, there are several arches supported by stone pillars. The street is illuminated by warm, yellow lights, and the sky is a deep blue. The overall atmosphere is old and atmospheric.

Что  
не так  
с отчётами?

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

Element should be hidden {#gameWin}

Element: '</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...

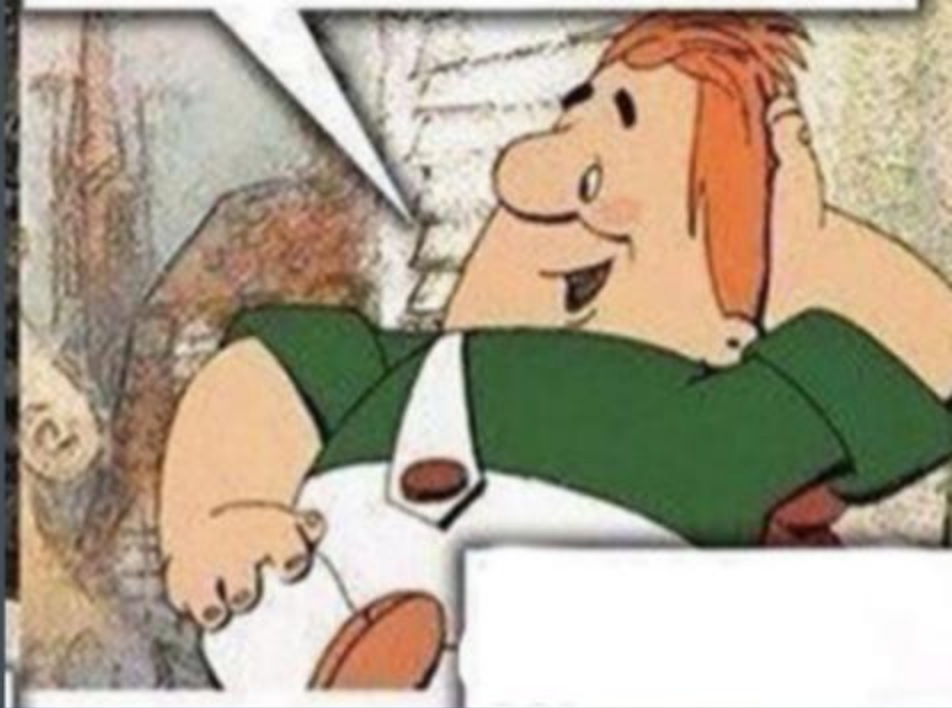


ЖОПА! Все тесты красные.

Я год не видел зелёного билда...



Так возьми  
Report Portal!

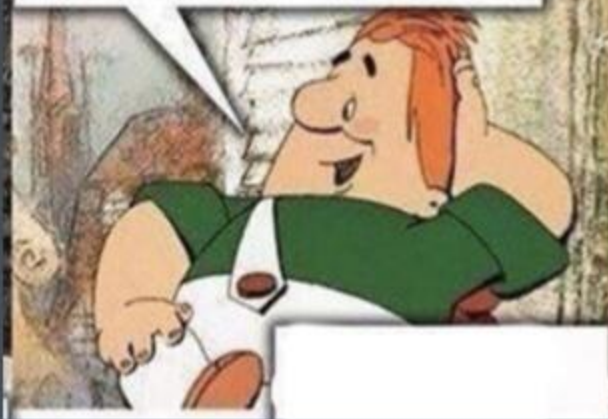


**ЖОПА! Все тесты красные.**

Я год не видел зелёного билда...



**Так возьми  
Report Portal!**



**Так это же всего  
лишь  
психологическая  
помощь**

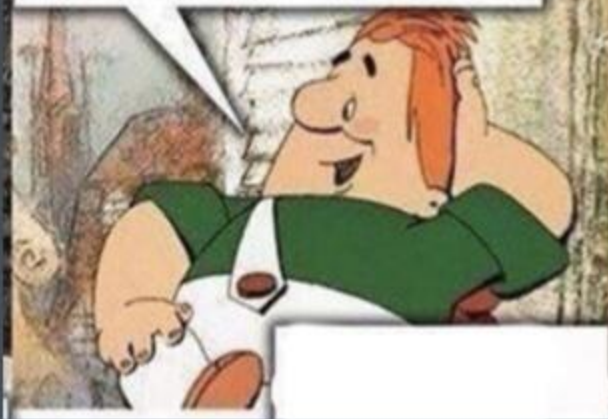


ЖОПА! Все тесты красные.

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...

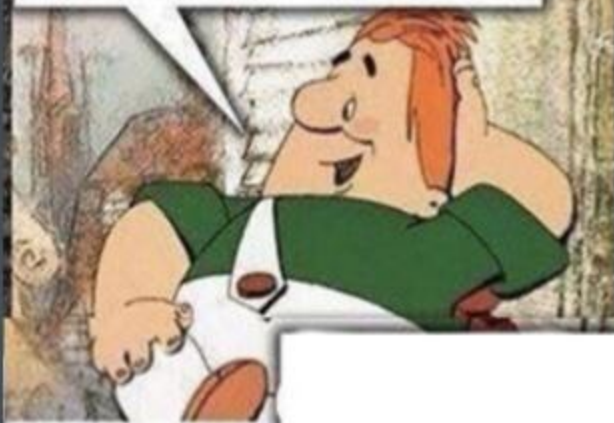


Так это же всего лишь  
психологическая  
помощь

В смысле?  
Там же крутые  
отчёты



Так возьми  
Report Portal!



Отчёты всего лишь  
убеждают тебя, что твоя  
жопа - не такая уж жопа.



ЖОПА! Все тесты красные.

Я год не видел зелёного билда...



Так это же всего лишь  
психологическая  
помощь

В смысле?  
Там же крутые  
отчёты



Так возьми  
Report Portal!



Да ты чо, пёс!  
Это  
МАШИН ЛЁНИНГ!

Отчёты всего лишь  
убеждают тебя, что твоя  
жопа - не такая уж жопа.





Что не так с  
TestNG?

# Разница между JUnit и TestNG

JUnit

Новый инстанс перед  
каждым тестом

TestNG

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

Created MyTest

Created MyTest

Created MyTest

TestNG

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;  
  
    @Before  
    void setUp() {  
        foo = 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();
```

A close-up, low-angle shot of a man's face, looking upwards. He is wearing glasses that reflect a city skyline. The text "FabFable" is visible on the reflection. The lighting is warm and dramatic, highlighting the contours of his face and the texture of his skin.

Винтовка -

Оружие новичка

Выбор профессионала -

НОЖ!





Самая **опасная** идея  
та,  
которая **кажется хорошей.**



Андрей Солнцев  
[@asolntsev](#)  
[ru.seleniumide.org](http://ru.seleniumide.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/>