

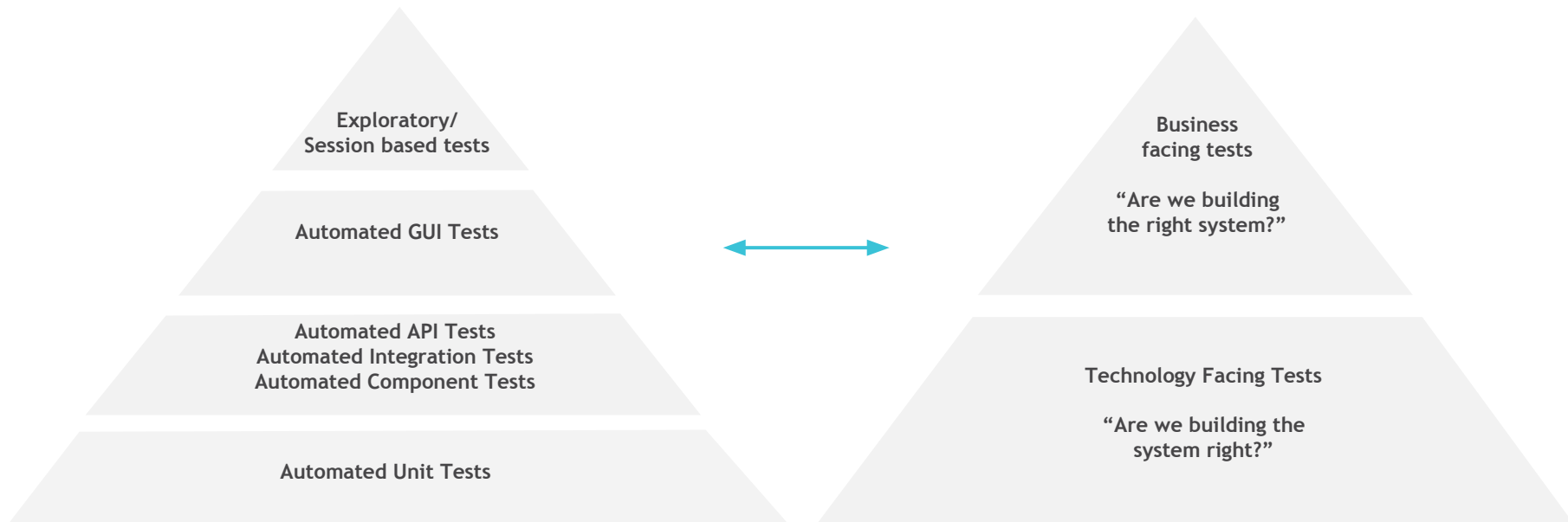


<ерат>

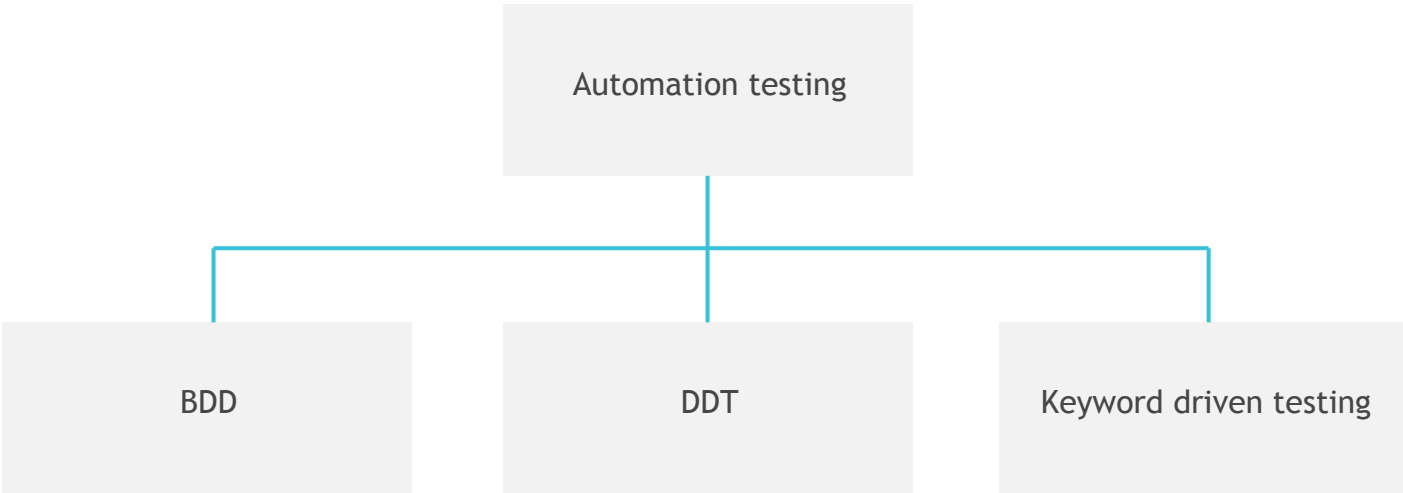
ПРИЕМОЧНОЕ  
ТЕСТИРОВАНИЕ  
СРЕДСТВАМИ **JBEHAVE**

Декабрь, 2016

# УРОВНИ ТЕСТИРОВАНИЯ



# СТРУКТУРА АВТОМАТИЗИРОВАННОГО ТЕСТИРОВАНИЯ



# ПОДХОДЫ В АВТОМАТИЗИРОВАННОМ ТЕСТИРОВАНИИ

1

## **KDT (Keyword Driven Testing)**

- тесты, управляемые ключевыми словами

2

## **DDT (Data Driven Testing)**

- позволяет данные хранить отдельно от тестов

3

## **BDD (Behavior-driven development)**

- разработка, основанная на поведении

# BDD ПОДХОД

---

**GIVEN**

<Исходное состояние>

**WHEN**

<Событие>

**THEN**

<Результат>

# BDD ФРЕЙМВОРКИ



# BDD FRAMEWORKS



```
⚙️ Feature: Product basket
  In order to buy products
  As a customer
  I need to be able to put interesting products into a basket

  Rules:
  - VAT is 20%
  - Delivery for basket under £10 is £3
  - Delivery for basket over £10 is £2

  Scenario: Buying a single product under £10
    Given there is a "Sith Lord Lightsaber", which costs £5
    When I add the "Sith Lord Lightsaber" to the basket
    Then I should have 1 product in the basket
    And the overall basket price should be £9
```

Feature: Google Test

```
Background: Open Google
  Given I open "http://www.google.com/webhp?hl=en"

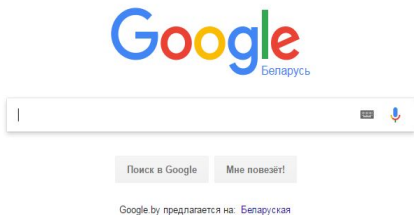
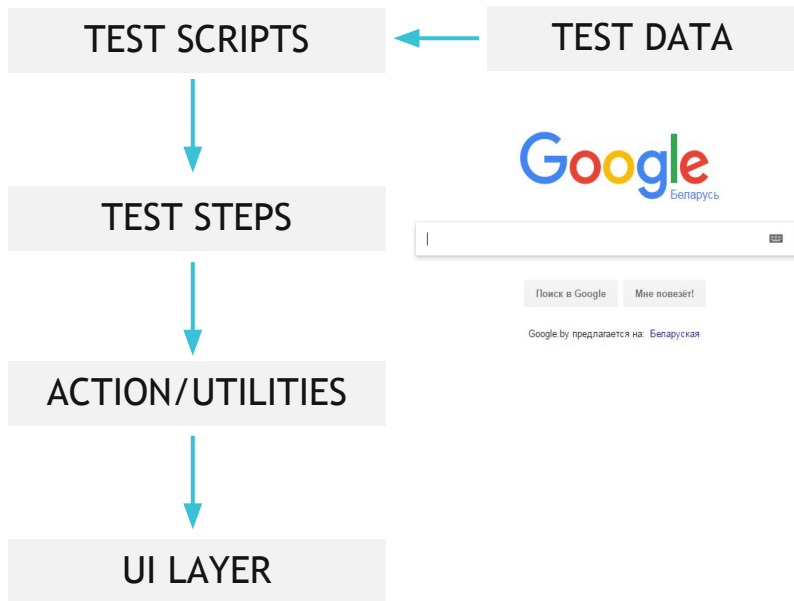
Scenario: Perform Search
  Given I don't see "Search Results Link"
  When I type "Test query" into "Search Field"
  And I click "Search Button"
  Then I see "Search Results Link"
```

Feature: Calculator

```
In order to avoid silly mistakes
As a math idiot
I want to be told the sum of two numbers

@mytag
Scenario: Add two numbers
  Given I have entered 50 into the calculator
  And I have also entered 70 into the calculator
  When I press add
  Then the result should be 120 on the screen
```

# УРОВНИ ФРЕЙМВОРКА АВТОМАТИЗАЦИИ И **PAGEOBJECT**



```
<input class="gsfi" id="lst-ib" maxlength="2048" name="q" autocomplete="off"
```

```
public class SearchPage {
```

```
System.setProperty("webdriver.chrome.driver", "/path/to/chromedriver");  
WebDriver driver = new ChromeDriver();  
driver.get("http://www.google.by");
```

```
public static By SEARCH = By.xpath("//input[@name='q'] ");
```

```
WebElement searchInput = driver.findElement(SEARCH);  
searchInput.sendKeys("Apple");  
searchInput.submit();
```

```
}
```



## trader\_is\_alerted\_of\_status.story

```
Scenario: trader is not alerted below threshold  
  
Given a stock of symbol STK1 and a threshold of 10.0  
When the stock is traded at 5.0  
Then the alert status should be OFF  
  
Scenario: trader is alerted above threshold  
  
Given a stock of symbol STK1 and a threshold of 10.0  
When the stock is traded at 11.0  
Then the alert status should be ON
```

## TraderSteps.java

```
public class TraderSteps { // look, Ma, I'm a POJO!!  
  
    private Stock stock;  
  
    @Given("a stock of symbol $symbol and a threshold of $threshold")  
    public void aStock(String symbol, double threshold) {  
        stock = new Stock(symbol, threshold);  
    }  
  
    @When("the stock is traded at $price")  
    public void theStockIsTradedAt(double price) {  
        stock.tradeAt(price);  
    }  
  
    @Then("the alert status should be $status")  
    public void theAlertStatusShouldBe(String status) {  
        ensureThat(stock.getStatus().name(), equalTo(status));  
    }  
}
```

## Trader.java

```
@DefaultUrl("http://localhost:9000/admin/categories/new")  
public class EditCategoryPage extends PageObject {  
  
    @FindBy(id="object_label")  
    WebElement label;  
  
    @FindBy(id="object_code")  
    WebElement code;  
  
    @FindBy(name="save")  
    WebElement saveButton;  
  
    @FindBy(xpath = "//input[@value='Delete Category']")  
    WebElement deleteButton;  
  
    public EditCategoryPage(WebDriver driver) {  
        super(driver);  
    }  
  
    public void saveNewCategory(String labelValue, String codeValue) {  
        typeInto(label, labelValue);  
        typeInto(code, codeValue);  
        saveButton.click();  
    }  
  
    public void clickOnDelete() {  
        deleteButton.click();  
    }  
}
```

TEST SCRIPTS

TEST SCRIPTS

UI LAYER

# JBEHAVE

```
public abstract class TraderStory extends JUnitStory {

    public TraderStory() {
        configuredEmbedder().embedderControls().doGenerateViewAfterStories(true).doIgnoreFailureInStories(true)
            .doIgnoreFailureInView(true).useThreads(2).useStoryTimeoutInSecs(60);
    }

    @Override
    public Configuration configuration() {
        return new MostUsefulConfiguration();
    }

    @Override
    public InjectableStepsFactory stepsFactory() {
        return new InstanceStepsFactory(configuration(), new TraderSteps(new TradingService()));
    }
}
```

A wide-angle photograph of a busy city street at dusk. The scene is filled with a large crowd of pedestrians walking across the street. On the left, a grand, light-colored stone building with classical architectural features, including a large arched entrance and a balcony with a Union Jack flag, is visible. A sign above the entrance reads "Superary". In the center, a street sign indicates "GENT STREET W1". To the right, a modern building is under construction, partially covered in scaffolding and white protective sheeting. A red flag with the "Superary" logo is attached to the construction site. Further right, a Starbucks logo is visible on a building facade. The sky is a mix of dark blue and orange, with a bright sun low on the horizon creating a lens flare effect. The overall atmosphere is one of a bustling urban environment.

# VDD ПОДХОД НА ПРОЕКТЕ

# ФУНКЦИОНАЛ ПРОЕКТА

1

Создание  
тренингов

- обеспечивает полное описание курсов, которые доступны в каталоге

2

Расписание

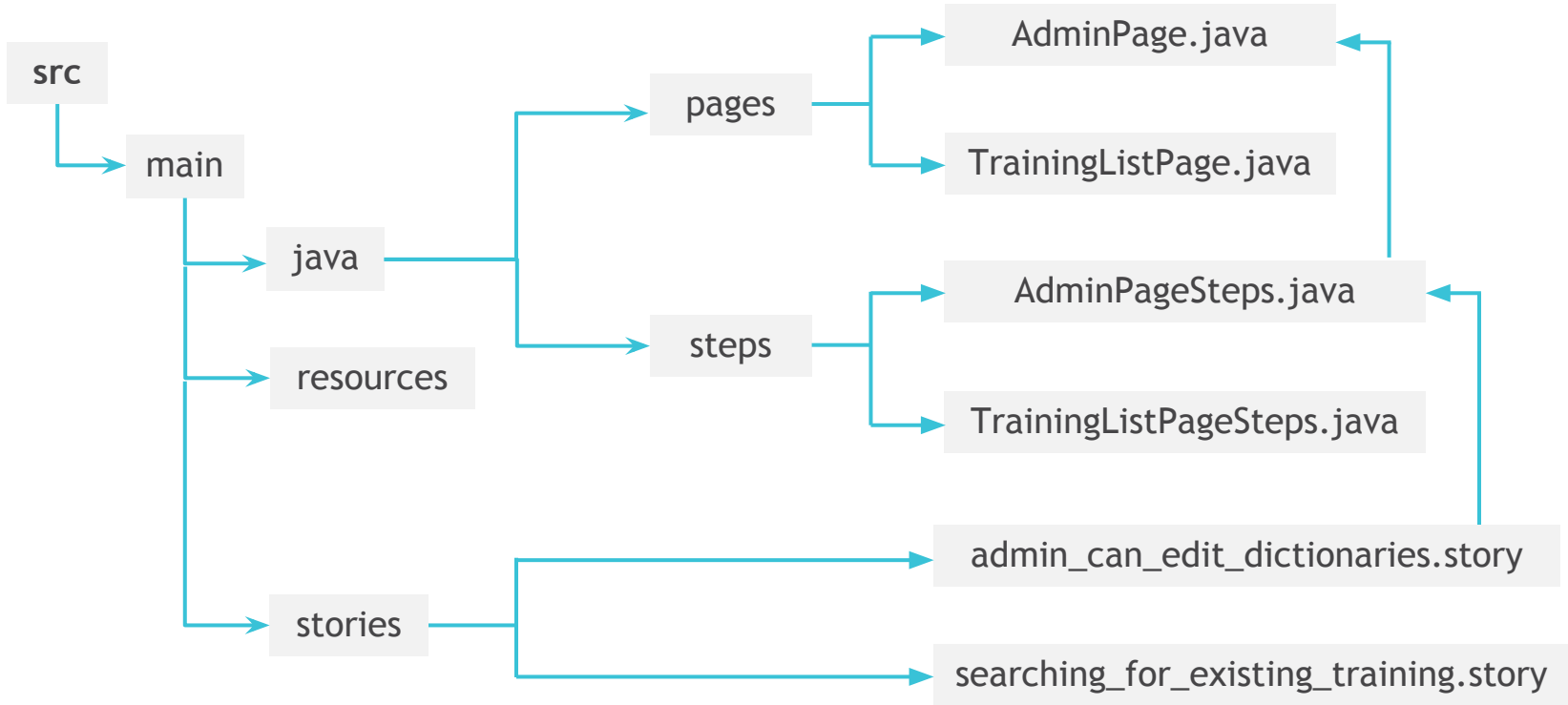
- запланированный тренинг с определенным тренером, расписанием и местоположением

3

Обработка  
запросов

- управление регистрацией пользователей на тренинг: перемещение в запланированные события, отметка посещаемости, сбор фидбеков

# СТРУКТУРА ТЕСТОВОГО ПРОЕКТА



# ТЕСТОВЫЙ СЦЕНАРИЙ

admin\_can\_edit\_dictionaries.story

## Meta:

@testType ui

@user //users/admin/xUsername

@xmlFile testdata/smoke/smokeTestData.xml

*Scenario: Administrator can edit Categories*

**Given** user logs in with 'Administrator' role

**When** user clicks 'Admin' tab

**And** user clicks Add button

**And** user types random records name

**And** user types 'TestName' records short name

**And** user clicks Ok button

**And** user clicks Save dictionary button

**And** user clicks edit icon on new added record

**And** user types 'newTestName' records short name

**And** user clicks Ok button

**And** user clicks Save dictionary button

**Then** new record Short Name is 'newTestName'

# ТЕСТОВЫЙ КЛАСС

## AdminPageSteps.java

```
public class AdminPageSteps {

    @When("user clicks '$tabName' tab")
    public void goToTab(String tabName) {
        getPage().goToTab(tabName);
    }

    @When("user clicks Add button")
    public void clickAddButton() {
        getPage().clickAddButton();
    }

    @Then("new record exists in list")
    public void checkNewRecordExists() {
        String recordName = RecordsStorage.getLastCreatedRecord().getName();
        Assert.assertTrue(String.format(Messages.NEW_RECORD_NOT_DISPLAYED, recordName), getPage().isRecordExist(recordName));
    }
    .....
}
```



# PAGE OBJECT

## AdminPage.java

```
public class AdminPage extends CustomAbstractPage {

    protected static final By ADD_BUTTON = By.cssSelector(".ui-button-add-label.ui-button-add-label--green");
    protected static final By MODAL_BODY = By.className("ui-modal__body");
    protected static final String RECORD_ROW_XPATH = "//td[1]//div[text()='0']/../../..";

    public void clickAddButton() {
        _findElement(ADD_BUTTON).click();
        _findElement(MODAL_BODY);
    }

    public boolean isRecordExist(String name) {
        return isElementVisible(formatXPath(RECORD_ROW_XPATH, name), PropertyConstants.LOWEST_WAIT);
    }

    .....
```



СПАСИБО ЗА ВНИМАНИЕ!

---

