Предпосылки создания новой версии фреймворка.

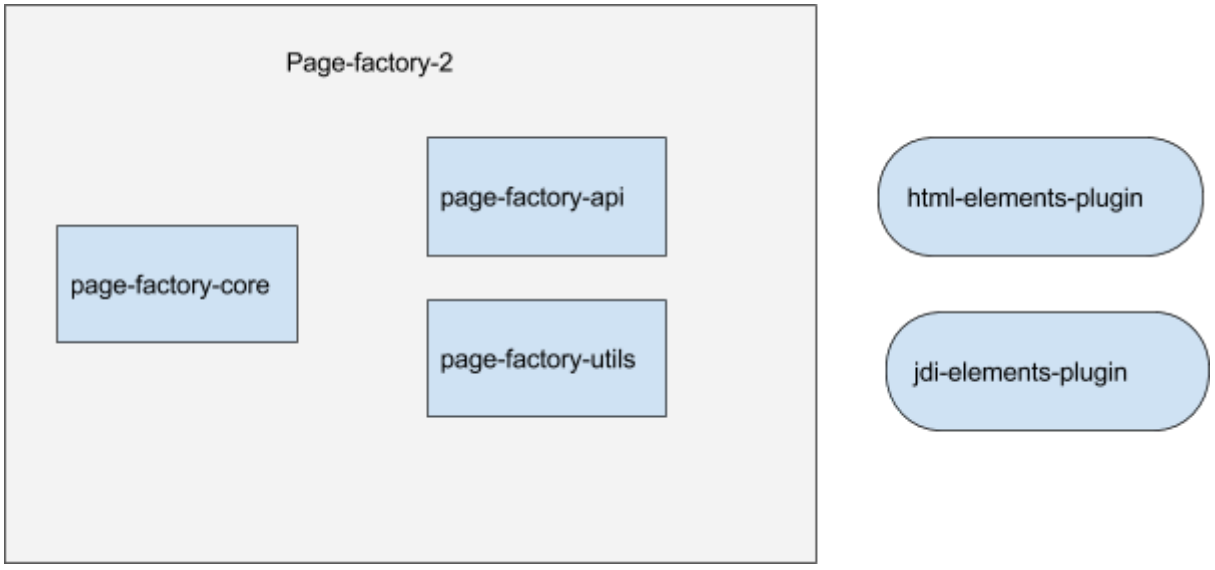
Первая версии феймворка **page-factory** имела монолитную архитектуру, что в результате сильно ограничивало возможности по ее развитию.

Одной из ключевых проблем стало отсутствие возможности выбора элементов для описания страниц (PageObject-ов).

В монолитной архитектуре поддержка yandex html-elements была “зашита” внутри фреймворка, что лишало возможности использовать какой-нибудь другой набор элементов для описания страниц вместо yandex-html-elements.

О page-factory-2

Page-factory-2 представляет собой набор модулей. Каждый модуль отвечает за определенные задачи.



На данной схеме изображены 5 модулей. Каждый модуль - отдельный мавен-артефакт.

Page-factory-api - внешний интерфейс page-factory-2. Он в себя включает основной класс **Page**, который является базовым для всех страниц и механизм получения текущей страницы из контекста теста. Также в этом модуле вынесены все аннотации и исключения.

Page-factory-utils - набор утилитных методов из page-factory (рефлексийный поиск и другие служебные методы). Этот модуль также используется в **html-elements-plugin**.

Page-factory-core - основная часть фреймворка. В ней реализованы ключевые механизмы управления исполнением теста и базовая логика по работе с WebElements.

html-elements-plugin - плагин, позволяющий описывать PageObject-ы элементами из фреймворка yandex-html-elements. Также содержит набор step definitions, управляющих поиском элементов на странице.

jdil-elements-plugin - плагин, позволяющий описывать PageObject-ы элементами из фреймворка JDI. Также содержит набор утилитных методов, управляющих поиском элементов на странице.

Ряд важных изменений

В связи с рядом причин, page-factory-2 был внесен набор изменений относительно первой версии. Самые важные из них представлены ниже:

1. Удалена фича “**RedirectsTo**”, позволяющая с помощью аннотации @RedirectsTo менять текущую страницу в контексте после нажатия на определенный элемент.

2. Action по выставлению checkBox-а из yandex html-elements фреймворка перенес из **Page** в **HTMLPage**.

3. WaitException, ранее наследуемый от AutoTestError теперь наследует AssertionError.

Миграция на page-factory-2

Для того, чтобы начать разрабатывать тесты на **page-factory-2** фреймворке или перейти на данный фреймворк необходимо выполнить следующее:

1. Подключить базовую зависимость

   ```
   <groupId>ru.sbtqa.tag.pagefactory</groupId>
   <artifactId>page-factory-core</artifactId>
   ```

2. При необходимости использовать готовые элементы для описания Page Object-ов подключить один из плагинов:

   ```
   <groupId>ru.sbtqa.tag.pagefactory</groupId>
   <artifactId>html-elements-plugin</artifactId>
   ```

   или

   ```
   <groupId>ru.sbtqa.tag.pagefactory</groupId>
   <artifactId>jdi-elements-plugin</artifactId>
   ```

3. Все PageOject-ы должны быть унаследованы от одного из базовых классов:

Page - класс из **page-factory-api.** Базовый класс для всех PageObject-ов. Не содержит никакой логики.

WebElementsPage - Класс находится в **page-factory-core**. Содержит набор реализованных методов (Action-ов) по работе с WebElement-ми. При описании PageObject-ов WebElement-ми рекомендуется наследоваться от этого класса.

HTMLPage - класс из **html-elements-plugin**. Содержит основные Action-ы по взаимодействию с элементами из yandex html-elements framework.

JDIPage - класс из **jdi-elements-plugin**. Содержит основные Action-ы по взаимодействию с элементами из JDI framework.

По факту, каждый page-класс - набор общих действий с определенным типом элементов.

4. В тестовом классе, запускающем Cucumber указать **TagCucumber** и дополнительно пути к step definitions используемых плагинов.

В следующем примере, используется элементы html-elements:

```
@RunWith(TagCucumber.class)
@CucumberOptions( glue = {
"ru.sbtqa.tag.pagefactory.stepdefs",
"ru.sbtqa.tag.pagefactory.htmlstepdefs"}, ...
```

Использование jdi-elements-plugin

Существует ряд особенностей при использовании данного плагина и jdi элементов для описания страниц:

1. Перед выполнение теста необходимо вызвать метод *JDIUtils.setJDIConfig(Supplier<WebDriver> driverSupplier)*.
   Это необходимо для того, чтобы JDI фреймворк сам инициализировал компоненты на страницах.
2. В конструкторе PageObject-а, который описан с помощью jdi элементов вызвать следующих метод инициализации *JDIUtils.initElementsOnPage(Object page)*.

Набор ограничений

Здесь приведен набор общих вопросов-ответов, определяющих ограничения при работе с данным фреймворком.

Вопрос: Можно ли использовать несколько типов элементов при описании PageObject-а?

Ответ: Нет, нельзя. Для каждого варианта, в котором используется один набор элементов, строго определенный механизм инициализации элементов на странице.