Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

Кафедра математической
кибернетики и компьютерных наук

АВТОМАТИЗАЦИЯ ТЕСТИРОВАНИЯ ВЕБ-ПРИЛОЖЕНИЯ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

Студентки 4 курса 451 группы
направления 09.03.04 — Программная инженерия
факультета КНиИТ
Паляевой Александры Вячеславовны

Научный руководитель
доцент, к. ф.-м. н. _____ А. С. Иванова

Заведующий кафедрой
к. ф.-м. н. _____ С. В. Миронов

Саратов 2018

ВВЕДЕНИЕ

Разработка долгоживущих, крупных программных проектов подразумевает необходимость их поддержки, доработки, выпуска обновлений и новых версий. При этом большинство из них являются веб-приложениями, работающими в интернет-браузере. При внесении изменений нужно быть уверенным не только в том, что новая функциональность работает как требуется, но и в том, что старая по-прежнему работает. Поэтому при добавлении функциональности неизбежно возрастет и объем тестирования. При этом проведение одних и тех же проверок для «старого» функционала может быть утомительным для человека. Получается, что на крупных проектах особенно часто должны выполняться следующие виды тестов: дымовое тестирование, регрессионное тестирование, тестирование приемки [1]. Такие тесты выполняются как минимум для каждого нового билда, поэтому их автоматизация является естественным решением.

Автоматизированное тестирование — это набор техник, подходов и инструментальных средств, позволяющий исключить человека из выполнения некоторых задач в процессе тестирования [2]. Таким образом, автоматизация призвана упростить и ускорить процесс тестирования.

Исходя из всего вышесказанного, можно заключить, что автоматизация тестирования программного обеспечения сейчас очень актуальна.

Целью бакалаврской работы является разработка собственного фреймворка для автоматизации тестирования через GUI на примере веб-приложения «AliExpress» (https://ru.aliexpress.com/). Для ее достижения были поставлены следующие **задачи**:

1. описать преимущества и недостатки ручного и автоматизированного тестирования;
2. рассмотреть способы автоматизации тестирования;
3. составить чек-лист и тест-кейсы для автоматизированного тестирования;
4. выбрать наиболее подходящие инструменты и подходы для создания фреймворка;
5. провести автоматизированное тестирование по созданным тест-кейсам на следующих уровнях: smoke test, critical path test;
6. описать разработанный фреймворк, продемонстрировать примеры его запусков;

7. проанализировать полученные результаты.

Методологические основы теории автоматизированного тестирования и методов проектирования тестов изложены в работах С. Куликова [2], Г. Майерса [3], Р. Блэка [4], Л. Коупленда [5].

Структура и объём работы. Бакалаврская работа состоит из введения, трех разделов, заключения, списка использованных источников и двух приложений. Общий объем работы — 57 страниц, из них 50 страниц — основное содержание, включая 2 таблицы, 11 рисунков, цифровой носитель (CD-диск с исходным кодом) в качестве приложения, список использованных источников информации — 20 наименований.

Первый раздел **«Теоретические основы ручного и автоматизированного тестирования и их сравнение»** посвящен общим сведениям о ручном и автоматизированном тестировании. В нем описываются преимущества и недостатки обоих подходов, рассматриваются случаи наилучшего применения автоматизации тестирования, а также перечисляются наиболее популярные подходы к автоматизации тестирования, рассматриваются некоторые характерные особенности разработки фреймворка для тестирования веб-приложения.

Второй раздел **«Технологии и подходы, использованные при создании фреймворка»** посвящен описанию технологий и инструментов, примененных для разработки фреймворка.

Третий раздел **«Разработка фреймворка для автоматизированного тестирования интернет-магазина AliExpress»** посвящен описанию разработанного фреймворка. В нем рассматривается интернет-магазин AliExpress и его функционал, который был протестирован, приводится чек-лист, по которому производилось тестирование, перечисляются примененные виды тестирования, подробно описываются все составляющие разработанного фреймворка, описывается код, приведенный в приложении, приводятся примеры запуска приложения и сгенерированные отчеты о результатах тестирования, а в заключении приводятся выявленные в ходе выполнения работы преимущества и недостатки автоматизированного тестирования.

1 Теоретические основы ручного и автоматизированного тестирования и их сравнение

1.1 Основные понятия теории тестирования

Тестированием программного обеспечения называют процесс анализа программного продукта и сопутствующей документации с целью выявления дефектов и повышения качества [6].

Дефектом (англ. bug) называют отклонение фактического результата от ожиданий наблюдателя, сформированных на основе требований, спецификаций, иной документации или опыта и здравого смысла. Следовательно, дефекты могут встречаться не только в коде приложения, но и в любой документации, архитектуре, дизайне, настройках и т. д.

Тест-кейсом называют набор входных данных, условий выполнения и ожидаемых результатов, разработанный с целью проверки того или иного свойства или поведения программного средства.

Тестом называют набор из одного или нескольких тест-кейсов.

Чек-лист — в общем случае представляет список идей по тестированию, или же набор тест-кейсов.

1.2 Ручное и автоматизированное тестирование

Ручное тестирование — тестирование, в котором тест-кейсы выполняются человеком вручную без использования средств автоматизации.

Автоматизированное тестирование — набор техник, подходов и инструментальных средств, позволяющий исключить человека из выполнения некоторых задач в процессе тестирования. При таком подходе тест-кейсы частично или полностью выполняются специальным инструментом.

Существует два основных подхода к автоматизации тестирования:

- Тестирование на уровне кода. К нему относится, в частности, модульное тестирование.
- GUI-тестирование — имитация действий пользователя с помощью специальных тестовых фреймворков.

В данной работе в качестве подхода к тестированию было выбрано автоматизированное тестирование через GUI.

1.3 Достоинства и недостатки ручного тестирования

Преимущества ручного тестирования:

— Тестировщик представляет потенциального пользователя.
— Отзыв об удобстве пользовательского интерфейса.
— В краткосрочной перспективе ручное тестирование дешевле, чем инструменты автоматизации (как по времени, так и по бюджету).
— Изменения могут быть протестированы вручную сразу.
— Возможность исследовательского тестирования.

Недостатки ручного тестирования:

— Человеческий фактор.
— Трудоемкость повторного использования.
— Невозможность нагрузочного тестирования.
— С ростом функциональности приложения объем ручного тестирования также растет.

1.4 Достоинства и недостатки автоматизации тестирования

Преимущества автоматизации тестирования:

— Скорость выполнения тест-кейсов.
— Отсутствует влияние человеческого фактора.
— Средства автоматизации способны выполнить тест-кейсы, в принципе непосильные для человека.
— Среда автоматизации может формировать небольшие понятные отчёты, графики и т.д.
— Средства автоматизации способны выполнять низкоуровневые действия с приложением, операционной системой, каналами передачи данных и т. д.
— Возможность увеличить тестовое покрытие.
— Тесты могут быть выполнены в нерабочее время.

При всех плюсах, с автоматизацией тестирования связана серия серьёзных недостатков и рисков:

— Автоматизация — это «проект внутри проекта».
— Разработка и сопровождение автоматизированных тест-кейсов занимает немало времени.
— Многие коммерческие средства автоматизации стоят дорого.
— Средств автоматизации крайне много, что усложняет проблему выбора.
— Автоматизированные тесты очень чувствительны к среде.

— АТ нельзя использовать применительно к объектам, которые может проверить только человек.

1.5 Оценка эффективности и области применения автоматизации тестирования

Поскольку автоматизация тестирования — это дорогостоящий процесс, требующий немалого времени, прежде чем внедрять его в реальный проект, обычно проводят оценку его применимости и эффективности с помощью специальных подходов. В первую очередь следует учитывать следующие характеристики [7, 8]:

— Затраты времени на ручное и автоматизированное выполнение тестов. Чем ощутимее разница, тем более выгодной представляется автоматизация.
— Количество повторений выполнения одних и тех же тест-кейсов.
— Затраты времени на отладку, обновление и поддержку автоматизированных тест-кейсов.
— Наличие в команде соответствующих специалистов.
— Поспособствует ли внедрение АТ оптимизации процесса тестирования и разработки в том числе, есть ли у команды на это силы и средства?
— Длительность проекта. Осуществить полноценную, приносящую плоды автоматизацию можно именно на длительном проекте.
— Желание заказчика.

Случаи наибольшей эффективности при применении автоматизации тестирования:

1. Регрессионное тестирование.
2. Конфигурационное тестирование и тестирование совместимости.
3. Использование комбинаторных техник тестирования — генерация комбинаций значений и многократное выполнение тест-кейсов с использованием этих сгенерированных комбинаций в качестве входных данных.
4. Модульное тестирование.
5. Интеграционное тестирование.
6. Тестирование безопасности.
7. Тестирование производительности.
8. Дымовое тестирование.
9. Приложения (или их части) без графического интерфейса.

10. Проверка приложений и их компонентов, не предназначенных для взаимодействия с человеком (веб-сервисы, серверы, библиотеки и т. д.).
11. Длительные, рутинные, утомительные для человека проверки, например, требующие сравнения больших объёмов данных.
12. Технические задачи, например, проверки корректности протоколирования, работы с базами данных и т.д.

Случаи, в которых автоматизацию тестирования применять неэффективно или невозможно [4]:

1. Планирование.
2. Разработка тест-кейсов.
3. Написание отчётов о дефектах.
4. Анализ результатов тестирования и отчётность.
5. Функциональность, которую нужно проверить всего один или несколько раз.
6. Низкая стабильность требований — придётся очень многое переделывать.
7. Если есть проблемы с планированием и ручным тестированием.
8. Нехватка времени и угроза срыва сроков.
9. Области тестирования, требующие оценки ситуации человеком, например тестирование удобства использования.

Автоматизация при верном применении может дать ощутимую выгоду, но при неверном принесёт лишь затраты.

1.6 Технологии и подходы к автоматизации тестирования

Любая технология автоматизации тестирования базируется на определенном наборе технических решений (языки программирования, способы взаимодействия с приложением, инструментальные средства). Инструмент автоматизации тестирования — это программа или набор программ, позволяющих создавать, редактировать, отлаживать, выполнять автоматические тесты и собирать статистику после их выполнения [1].

Основные технологии автоматизации тестирования [2]:

1. **Запись и воспроизведение (Record & Playback).** Средство автоматизации записывает действия тестировщика и может воспроизвести их, управляя тестируемым приложением.
2. **Тестирование под управлением данными (англ. Data Driven Testing**

— **DDT).** Из тест-кейса вовне выносятся входные данные и ожидаемые результаты.

3. **Тестирование под управлением ключевыми словами (англ. Keyword-driven testing — KDT).** Логическим развитием идеи о вынесении данных из тест-кейса является вынесение команд (описания действий) из тест-кейса вовне.
4. **Фреймворки автоматизации тестирования** — решения, объединяющие в себе лучшие стороны других технологий и подходов. Для таких фреймворков характерны: высокая абстракция кода, универсальность и переносимость используемых подходов, высокое качество реализации. Каждый фреймворк специализируется на своём виде или уровне тестирования и наборе технологий.
5. **Тестирование под управлением поведением (англ. Behavior Driven Testing — BDT)** представляет собой развитие идей тестирования под управлением данными и ключевыми словами. Тесты концентрируются на бизнес-сценариях.

1.7 Особенности разработки фреймворка для автоматизированного тестирования

Фреймворком для автоматизации тестирования называют набор взаимодействующих между собой компонентов, облегчающих создание, выполнение автоматических тестов и последующее представление результатов.

Главной особенностью фреймворка является отделение друг от друга разных компонент. Такими компонентами в данном случае являются: работа с драйвером браузера, взаимодействие с веб-элементами сайта, логика тестов, тестовые данные, формирование отчетности. Такое разделение функциональности позволяет добиться большей надежности, гибкости и поддерживаемости.

1.8 Взаимодействие с пользовательским интерфейсом тестируемого веб-приложения

В данной работе тестирование проводилось через пользовательский интерфейс (GUI) посредством воспроизведения действий конечного пользователя.

В данной работе каждый шаг автоматизированного теста состоит из двух частей:

1. Найти элемент интерфейса.
2. Выполнить действия с найденным элементом: клик, ввод значения в текстовое поле, считывание текстового значения.

В данной работе для взаимодействия с веб-приложением используется Selenium Webdriver. Именно он обеспечивает поиск веб-элементов в DOM-дереве веб-страницы и выполнение действий над ними.

Для того, чтобы уникально идентифицировать элемент страницы, используются специально составленные строки — локаторы.

WebDriver предоставляет несколько способов использования локаторов для поиска элементов. Но самыми популярными из них являются CSS Selectors и XPath — это сложные языки запросов, дающие большую свободу в построении локаторов.

1.9 Архитектура автоматизированных тестов

Для удобства написания автоматизированных тестов на основе уже имеющихся тест-кейсов, структура тест-скриптов должна быть аналогична структуре ручного тест-кейса. Тогда каждый тест-скрипт должен состоять из следующих частей [9]:

1. Предусловие (англ. precondition) — подготовка системы к тестированию.
2. Шаги теста (англ. steps).
3. Постусловие (англ. postcondition) — завершение работы с системой после тестирования.

2 Технологии и подходы, использованные при создании фреймворка

2.1 Java как язык разработки

В качестве языка разработки был выбран Java — объектно-ориентированный язык программирования со статической сильной типизацией. Особенность языка состоит в том, что программы при компиляции транслируются в специальный байт-код, затем они могут быть запущены на любом устройстве с виртуальной Java-машиной (Java Virtual Machine — JVM) [10].

Разработанное приложение для удобства собрано в исполняемый JAR-архив.

2.2 Среда разработки

Интегрированная среда разработки (англ. Integrated development environment — IDE) — комплекс программных средств, используемый программистами для удобства разработки программного обеспечения за счет проверки кода во время написания, упрощения сборки и запуска.

Для данной работы использовалась IntelliJ IDEA как наиболее мощная и удобная для Java.

2.3 Паттерны работы со страницами

Для отделения логики тестов от описания веб-страниц используется паттерн Page Object Model (POM). Он состоит в создании отдельного класса для каждой страницы сайта. Этот класс будет хранить элементы на странице и описывать взаимодействие с ними — например, заполнение и проверку [11].

Для более удобного использования Page Object Model также был применен паттерн Page Factory. Это оптимизированный способ работы с объектной моделью страницы, встроенный в Selenium WebDriver. Он позволяет удобно инициализировать элементы всех веб-страниц.

2.4 Инструмент управления браузером Selenium WebDriver

Для автоматизации работы с браузером Google Chrome использовался **Selenium WebDriver (Selenium 2.0)**. Это семейство драйверов для различных браузеров, а также набор клиентских библиотек для этих драйверов на разных языках программирования [12].

2.5 Фреймворк для модульного тестирования TestNG

Непосредственно тесты были написаны с помощью TestNG — фреймворка для модульного тестирования в Java. Он основан на JUnit и NUnit, поэтому предоставляет некоторые новые функции которые делают его более мощным и простым в использовании.

Для конфигурации и запуска набора тестов (англ. test suite) используется XML-файл (обычно называющийся testng.xml), где для каждого набора перечисляются классы, входящие в него.

Фреймворк имеет следующие преимущества [13]:

- Тесты могут быть сгруппированы разными способами, для каждой группы могут быть настроены особенности исполнения.
- Поддержка концепции DDT (англ. Data Driven Testing) — «тестирование под управлением данными».
- Имеет множество аннотаций для так называемых setUp/tearDown методов (методы, задающие предусловия/постусловия для набора тестов) разных уровней.
- Возможность организовать параметризованные тесты.
- Возможность задать порядок выполнения разными способами.

2.6 Создание отчетов в ReportNG

Для создания отчетов использовался ReportNG — это простой плагин генерации HTML отчетов для TestNG.

2.7 Maven — средство сборки кода

По окончании разработки проекта его нужно собрать, то есть получить подготовленный для использования исполняемый файл. Сборка может содержать такие этапы, как компиляция исходного кода в бинарный код, сборка бинарного кода, выполнение тестов, генерация документации.

В данной работе в качестве системы сборки использовался Maven и с его помощью был получен исполняемый JAR-архив.

2.8 Логирование в Slf4J

В данной работе использовались следующие библиотеки для логирования: SLF4J и Log4j.

3 Разработка фреймворка для автоматизированного тестирования интернет-магазина AliExpress

Основной задачей данной работы является разработка фреймворка для автоматизированного тестирования интернет-магазина AliExpress через графический интерфейс пользователя.

3.1 Описание объекта тестирования

Объектом тестирования является глобальный интернет-магазин AliExpress (URL: https://aliexpress.com/). Данный ресурс предназначен для продажи китайских товаров в зарубежные страны.

Ключевую функциональность данного интернет-магазина условно можно разбить на компоненты, представленные в таблице 1.

Таблица 1 – Основные компоненты и функции интернет-магазина AliExpress

№	Компонент	Функции
1	Учетная запись	Создание, редактирование, авторизация
2	Корзина	Добавление, удаление товара
3	Поиск товара	—
4	Оформление заказа	Адрес доставки, оплата
5	Выбор товара	Фильтрация и сортировка товаров
6	«Мои желания»	Создание, редактирование, удаление списка

3.2 Описание тестовой стратегии и примененных подходов

Производилось тестирование следующих разделов сайта:

1. Авторизация.
2. Корзина.
3. Поиск товара.
4. Адрес доставки.
5. Раздел «Мои желания».

В данной работе применялись следующие виды тестирования:

1. Автоматизированное тестирование.
2. Системное тестирование.
3. Функциональное тестирование.
4. Дымовое тестирование.
5. Тестирование критического пути.

6. Позитивное тестирование.
7. Негативное тестирование.
8. Метод серого ящика.

3.3 Описание разработанного приложения

3.3.1 Поиск элементов на странице

Для того, чтобы из программного кода можно было обращаться к элементам интернет страниц, нужно находить эти элементы. Для этого нужно составить локатор, по которому будет найден данный элемент. Чтобы это сделать, нужно открыть инструменты разработчика в браузере и навести курсор на нужный элемент, тогда он будет найден в DOM-дереве. Далее нужно составить xpath или css локатор, и затем использовать его в программе.

3.3.2 Структура проекта

Структура проекта:

— Класс `Account` содержит регистрационные данные пользователя и методы для их задания и получения.
— Класс `ReportAppender` добавляет логи в генерируемый HTML-отчет о тестировании.
— Пакет `pageobject` содержит Page Object Model классы, моделирующие веб-страницы.
— Пакет `test` содержит непосредственно тестовые классы. Тесты написаны с помощью фреймворка модульного тестирования TestNg.
— Класс `MyChromeDriver` отвечает за инициализацию драйвера браузера.
— Класс `WebDriverUtils` содержит вспомогательные методы для работы драйвера, например, метод для переключения на последнюю открытую вкладку браузера, а также проверка того, что элемент существует на странице.
— Файл `pom.xml` — основной файл, описывающий проект. Содержит в себе все его зависимости: плагины, библиотеки и фреймворки.
— Файл `testng.xml` — конфигурационный файл для запуска тестов TestNG.
— Класс `Main` указан в `pom.xml` как главный класс. Это нужно для создания исполняемого JAR-файла. Класс Main является точкой входа приложения. Поэтому данный класс запускает тесты, заданные конфигурационным файлом `testng.xml`.

3.3.3 Иерархия классов, моделирующих страницы тестируемого приложения

Классы, моделирующие веб-страницы, реализованы согласно паттерну Page Object Model.

Поля классов представляют собой веб-элементы данной страницы (например, кнопки), а методы — набор действий над ними.

Родителем для всех классов является класс `BasePage`. В своем конструкторе он инициализирует драйвер браузера, а также вызывает метод `PageFactory.initElements`, тем самым реализуя паттерн Page Factory. Потомки данного класса могут вызывать родительский конструктор, и благодаря этому инициализировать свои поля, содержащие веб-элементы.

3.3.4 Иерархия тестовых классов

Каждый тестовый класс выполняет проверку функциональности определенной страницы сайта. Все классы наследуются от одного родителя. Он создает предусловия и постусловия для следующих тестов. Перед всем набором тестов создается драйвер браузера, открывается главная страница сайта. После всего набора тестов выполняется выход из аккаунта, закрываются все окна браузера. В каждом тесте аналогично могут быть заданы предусловия и постусловия.

Каждый тестовый метод в конце выполняет какую-либо проверку с помощью класса `Assert`. Это нужно для того, чтобы убедиться, что ожидаемый и действительный результаты совпадают.

3.3.5 Сборка и запуск приложения

Сборка проекта производилась с помощью Maven. Полученный JAR-файл можно запустить из консоли.

3.3.6 Отчет о результатах тестирования

После завершения всех тестов с помощью ReportNG генерируется удобный отчет о результатах тестирования в формате HTML. Отчет показывает текущего пользователя, имя компьютера, версию операционной системы, версию Java, дату и время запуска тестов. Также для каждого набора тестов представлено общее время выполнения, количество пройденных, пропущенных, проваленных тестов и процент успешного прохождения. Таким образом, из

представленного отчета можно видеть, что тестирование проводилось под ОС Windows 10, всего был исполнен 31 тест, процентный показатель успешного прохождения тестов составил 90%.

Во время работы приложения информация о всех действиях записывается в отчет о результатах тестирования и в лог-файл.

Если нажать на название конкретного набора тестов в отчете, можно просмотреть более детальную информацию по каждому тестовому методу: название, время его выполнения, аргументы, логи, а также текст ошибки и исключений, если тест провалился или был пропущен. Информация сгруппирована по успешности выполнения тестов.

3.3.7 Анализ полученных результатов

На практике были выявлены следующие преимущества автоматизированного тестирования перед ручным:

1. Автоматизированные тесты выполняются достаточно быстро. Скорость выполнения всего набора разработанных тестов составляет в среднем около 4 минут.
2. От тестировщика требуется только запустить скрипт.
3. При каждом запуске тестов формируется отчет.
4. Входные данные для тестов легко изменять.
5. Один и тот же тест может быть запущен с разными входными данными.
6. При наличии хорошо структурированного фреймворка становится легче добавлять новые тесты.

Также в ходе разработки были замечены и некоторые недостатки автоматизированного тестирования:

1. Разработка автоматизированных тестов требует значительного времени, как и поддержка. Например, при обновлении браузера требуется скачать новую версию драйвера для него. Также если изменится интерфейс сайта, тесты придется изменить.
2. Чувствительность к среде.
3. Не все тест-кейсы могут быть автоматизированы.

ЗАКЛЮЧЕНИЕ

В ходе выполнения бакалаврской работы были решены все поставленные задачи, что позволило достигнуть заявленной цели — разработать фреймворк для автоматизированного тестирования на примере веб-приложения AliExpress. Фреймворк может быть запущен из командной строки.

В основе фреймворка лежит паттерн Page Object для работы с веб-страницами.

Для его разработки были изучены и использованы следующие технологии:

- Язык программирования Java.
- Среда разработки Intellij IDEA.
- Инструмент управления браузером Selenium WebDriver.
- Фреймворк для модульного тестирования TestNG.
- Система сборки проектов Maven.
- Библиотека для генерации отчетов ReportNG.
- Интерфейс логирования Slf4J и библиотека логирования Log4J.

Был исполнен 31 тест, процентный показатель успешного прохождения тестов составил 90%.

Большая часть тест-кейсов была также проверена в летней производственной практике вручную, что позволяет сравнить ручное и автоматизированное тестирование. Время выполнения автоматизированных тестов значительно меньше, также они могут многократно повторяться, выполняться в нерабочее время, входные данные для таких тестов легко изменить. Однако разработка и поддержка автоматизированных тестов требует серьезных временных затрат. При этом ручное тестирование, напротив, требует небольшого времени на составление тест-кейсов, но большого времени на их выполнение. Также периодическое повторение тестов вручную может быть весьма утомительным.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 *Kaner, C.* Testing computer software / C. Kaner, J. Falk, H. Q. Nguyen. — Wiley, 1999.

2 *Куликов, С. С.* Тестирование программного обеспечения. Базовый курс / С. С. Куликов. — EPAM Systems, 2015.

3 *Майерс, Г.* Искусство тестирования программ / Г. Майерс, Т. Баджетт, К. Сандлер. — Диалектика, 2012.

4 *Блэк, Р.* Ключевые процессы тестирования / Р. Блэк. — Лори, 2011.

5 *Copeland, L.* A Practitioner's Guide to Software Test Design / L. Copeland. — Artech House, 2004.

6 Тестирование программного обеспечения [Электронный ресурс]. — URL: https://social.msdn.microsoft.com/Forums/ru-ru/e750a78b-0c1f-4766-81a2-7cea9b4b3ea2/-?forum=fordesktopru (Дата обращения 10.03.2018) Загл. с экр. Яз. рус.

7 *Munch, S.* The Return of Investment (ROI) of Test Automation [Электронный ресурс] / S. Munch, P. Brandstetter, K. Clevermann. — URL: https://pdfs.semanticscholar.org/df6d/c6f718817eb528154f0718bcd0392451fc0b.pdf (Дата обращения 12.04.2018) Загл. с экр. Яз. англ.

8 *Dustin, E.* Implementing Automated Software Testing: How to Save Time and Lower Costs While Raising Quality / E. Dustin, T. Garrett, B. Gauf. — Addison-Wesley Professional, 2009.

9 Архитектура Автоматических Тестов (Test Tools Architecture) [Электронный ресурс]. — URL: http://www.protesting.ru/automation/functional/toolarchitecture.html (Дата обращения 13.03.2018) Загл. с экр. Яз. рус.

10 *Шилдт, Г.* Java 8. Руководство для начинающих / Г. Шилдт. — Вильямс, 2017.

11 Page Object Model (POM) & Page Factory in Selenium: Complete Tutorial [Электронный ресурс]. — URL: https://www.guru99.com/

page-object-model-pom-page-factory-in-selenium-ultimate-guide.html (Дата обращения 01.05.2018) Загл. с экр. Яз. англ.

12 SELENIUM / WEBDRIVER автоматизация веб-приложений через браузер [Электронный ресурс]. — URL: https://selenium2.ru/docs/webdriver.html (Дата обращения 10.05.2018) Загл. с экр. Яз. рус.

13 TestNG [Электронный ресурс]. — URL: http://testng.org/doc/documentation-main.html (Дата обращения 10.05.2018) Загл. с экр. Яз. англ.