



Java QA Engineer. Professional

Углубленная автоматизация тестирования на стеке Java

Длительность курса: 110 академических часов

1 **Вводный урок.
Локаторы,
Ожидания,
Capabilities**

Цели занятия:

рассказать о формате взаимодействия в OTUS и о проектной работе;
вспомнить основные определения, методы поиска;
написать простой тест.

Краткое содержание:

пишем простой тест, вспоминаем основы;
локаторы: вспоминаем про xpath, css, id, name, class;
ожидания: вспоминаем стандартные Capabilities.

2 **Многопоточность,
асинхронные
методы**

Цели занятия:

Изучение параллелизации в Java. Студент научится проектировать многопоточные приложения и запускать логические блоки в несколько потоков. Так же на занятии будет изучена разница между потоком и процессом.

Краткое содержание:

На занятии будет изучена организация многопоточности в Java и передача данных между потоками. Так же будут рассмотрены различные методики параллелизации приложения.

3 **Дженерики,
Слушатели
(listeners),
Сложные
ожидания - Часть
1**

Цели занятия:

познакомиться с дженериками;
изучить функциональные интерфейсы;
узнать что такое слушатели;
написать ожидания.

Краткое содержание:

дженерики <T>;
функциональные интерфейсы;
слушатели (listeners):
- узнаем, что это такое;
- пишем свои слушатели;
- подсвечиваем элемент после нажатия;
- убираем подсветку;
EventFiringDriver;
практика.

ожидания:
пишем свои ожидания;
а. Количество элементов больше или равно;
б. Ни один из элементов не виден.

4 **Дженерики,
Слушатели
(listeners),
Сложные
ожидания - Часть
2**

Цели занятия:

узнать что такое слушатели;

Краткое содержание:

слушатели (listeners):
- узнаем, что это такое;
- пишем свои слушатели;
- подсвечиваем элемент после нажатия;
- убираем подсветку;
- снятие скриншотов при ошибке;
EventFiringDriver;
практика.

**5 Библиотеки
Actions**

Цели занятия:

узнать, что такое Actions и для чего это.

Краткое содержание:

actions:

- что это и для чего;
 - обзор;
 - практика (выпадающие списки google).
-

**6 Генерация
тестовых данных
через Stream API**

Цели занятия:

изучить Java Stream: Filter, Map, Reducer.

Краткое содержание:

работа с коллекциями: Java Stream:

- что такое стримы;
 - Filter;
 - Map;
 - Reducer.
-

7 **Практика:**
Написание авто-
теста с нуля

Цели занятия:

выполнить ДЗ;
написать теста с нуля.

Краткое содержание:

создание проекта;
подключение библиотек (WebDriverManager,
selenium);
реализация ожиданий, слушателей;
добавление Actions;
реализация PF, WDF;
ответы на вопросы.

Домашние задания

1 ДЗ #1: Автотест со своими ожиданиями

Цель: Реализовать на практике полученные знания, с использованием Actions и своих ожиданий.

Необходимо создать проект в Maven'e и реализовать:

1. Фабрику (WebDriverFactory), которая будет получать значение из окружения и запускать соответствующий браузер

Браузеры: Chrome, Firefox, Opera

2. Реализовать подсветку элементов перед нажатием, после нажатия вернуть данные в исходное состояние

3. На главной странице Otus'a снизу найти список курсов(популярные курсы, специализации, рекомендации) и реализовать:

- Метод фильтр по названию курса

- Метод выбора курса, стартующего раньше всех/ позже всех (при совпадении дат - выбрать любой) при помощи reduce

4. Реализовать движение мыши при помощи и выбор курса при помощи библиотеки Actions

1 Подход BDD

Цели занятия:

описать BDD подход: история возникновения, какие проблемы решает;
познакомиться с DI (Dependency Injection).

Краткое содержание:

история возникновения BDD подхода;
какие проблемы решает BDD подход;
Dependency Injection;
краткий обзор популярных библиотек для BDD.

2 Cucumber

Цели занятия:

познакомиться с библиотекой Cucumber;
написать несколько тестов с данной библиотекой.

Краткое содержание:

библиотека Cucumber;
написание тестов с данной библиотекой.

Цели занятия:

получить ответы на возникшие вопросы;
реализовать сценарий на Cucumber

Краткое содержание:

реализация сценария на Cucumber;
ответы на вопросы;
объяснение ДЗ.

Домашние задания

1 ДЗ #2: Реализовать BDD подход

Цель: Реализовать BDD фичи с использованием cucumber'a.

В существующий проект добавить библиотеку BDD, реализовать тесты с прошлого дз:

- Реализовать выбор фабрики через фичу ("Я открываю браузер Chrome)
- Поиск указанного курса (название курса задается в фиче) и его выбора (в случае если несколько, то выбирается случайный)
- Поиск курсов, стартующих в указанную дату или позже указанной даты и вывод информации о них в консоль (название, дата старта)
- Перейти в раздел Курсы > Подготовительные курсы, выбрать самый дорогой и самый дешевый курс при помощи filter и вывод информации о нем в консоль.

1 Основы API

Цели занятия:

узнать основные архитектурные решения при разработке backend сервера;
узнать о Rest, SOAP и их отличия;
познакомиться с основными инструментами для работы с Rest/SOAP.

Краткое содержание:

SOAP/ Rest(ful);
различия, подходы;
Curl, postman/newman, SoapUI.

2 RestAssured

Цели занятия:

познакомиться с библиотекой RestAssured для тестирования Rest серверов (как самых распространенных);
познакомиться с принципами валидации ответов от сервера на примере <http://jsonstub.com/>.

Краткое содержание:

написание тестов;
JsonSchemaValidator;
JsonPath;
практика: написание тестов.

Домашние задания

1 Rest-assured

Цель: Написать автотесты с использованием Rest-assured.

Написать тест кейсы в excel или в комментариях к классу:

- создание пользователя
- получение пользователя по имени

Для данных тест кейсов написать автотесты.
Описание API - <https://petstore.swagger.io>

3 Contract Testing Spring

Цели занятия:

познакомиться с возможностями тестирования через Spring Contract Testing;
реализовать Spring Contract Testing на примере <http://jsonstub.com/>.

Краткое содержание:

Spring Contract Testing.

4 **Citrus Framework:
Теория, первый тест**

Цели занятия:

познакомиться с Citrus Framework;
создание тестового проекта и запуск теста.

Краткое содержание:

теория Citrus Framework;
структура тестового проекта;
написание и запуск первого теста;
создание Endpoints;
работа с переменными.

5 **Citrus Framework:
Углубленное
изучение,
ApplyBehavior,
Параметризованные
тесты**

Цели занятия:

описать особенности Citrus Framework;
создавать параметризованные тесты;
применять ApplyBehavior в тестах;
создание Mock-объектов;
тестирования SOAP;
описать специфические функции Citrus Framework.

Краткое содержание:

Citrus framework;
Parametrized tests;
Citrus functions;
ApplyBehavior;
Testing SOAP
Mock;

**1 Selenium Grid,
многопоточное
выполнение
JUnit/TestNG**

Цели занятия:

Познакомить студентов с Selenium Grid;
научиться настраивать Selenium Grid, агентов;
научиться запускать тесты при помощи Selenium
Grid,
параллельный запуск тестов через JUnit/TestNG

Краткое содержание:

Знакомство с Selenium Grid;
настройка Selenium Grid, настройка агентов;
запуск тестов при помощи Selenium Grid;
запуск тестов через JUnit/TestNG с помощью
Selenium Grid.

**2 Современный
способ
распараллеливания
Selenoid**

Цели занятия:

познакомиться с Selenoid, основными командами
Docker'a;
поднять и запустить тесты на Selenoid.

Краткое содержание:

поднятие Selenoid/ Selenoid UI;
знакомство с основными capabilities;
Browser.json (подключим android).

Цели занятия:

ответы на вопросы и подготовится к ДЗ.

Краткое содержание:

подготовка к ДЗ;
написание тестов.

Домашние задания

1 ДЗ #4: Разворачивание и подключение Selenoid

Цель: Необходимо подключить Selenoid и перенести существующие тесты на использование Selenoid.

Добавить возможность запускать тесты на mobile chrome.

Необходимо подключить Selenoid и перенести существующие тесты на использование Selenoid

Добавить возможность запускать тесты на mobile chrome

**1 Паттерн MVC.
Spring****Цели занятия:**

узнать паттерн MVC и его реализацию через Spring framewok.

Краткое содержание:

MVC паттерн:
причины его возникновения, зачем он нужен, как он реализуется в Spring.

**2 Краткий обзор
WireMock,
mockito****Цели занятия:**

кратко познакомиться с Wiremock, Mockito;
узнать отличия от Spring, как реализовать на данных фреймворках stub сервер.

Краткое содержание:

Wiremock;
Mockito;
отличия от Spring;
реализация stub сервер.

**3 Практика, Api-
Helper's****Цели занятия:**

изучить примеры реализации Api-Helper's;
написать тесты через Spring/rest.

Краткое содержание:

Api-Helper's (для чего нужны, какую проблематику решают, пример реализации);
написание тестов через Spring/rest;
ответы на вопросы;
разбор ДЗ.

Домашние задания

Цель: 1.Реализовать Stub сервер со следующими методами-заглушками

- получение списка пользователей
- получение списка курсов
- оценка пользователя

2. Создать "хелперы" (API-helpers) для работы с API.

Это можно сделать в отдельном проекте. API-helpers должны тестировать реальные сервисы в интернете или сервисы развернутые на вашей локальной машине.

Нужно поработать с:

- http-helper
- soap-helper

дополнительное, более сложное задание (по желанию) :

- создать рабочий sql-helper (для получения данных из БД)
- создать рабочий mq-helper (для отправки данных в менеджер сообщений)

Вышеуказанные задачи можно реализовать с помощью Citrus Framework.

По http запросам можно применить RestAssured, Retrofit или Citrus (на ваше усмотрение).

Для заглушек можно использовать Wiremock или Citrus Framework (на ваше усмотрение).

1. Необходимо реализовать Stub сервер на Wiremock со следующими endpoint'ами:

- /user/get/{id} для получение оценки пользователя
- /course/get/all для получения списка курсов
- /user/get/all - для получения списка всех пользователей

Контракты

Для user

```
{  
  "name":"Test user",  
  "course":"QA",  
  "email":"test@test.test"  
  "age": 23  
}
```

```
Для оценки:  
{  
  "name": "Test user",  
  "score": 78  
}
```

```
Для курсов:  
[  
  {  
    "name": "QA java",  
    "price": 15000  
  },  
  {  
    "name": "Java",  
    "price": 12000  
  }  
]
```

Подключить stub frontend, написать тесты для проверки json схем.

2.

- создать тест с http-helper
- создать тест с soap-helper

дополнительно (по желанию):

- создать тест с sql-helper
- создать тест с mq-helper

1 Charles Proxy**Цели занятия:**

познакомиться с наиболее популярным инструментом для sniffing траффика;
посмотреть на основные фишки, активно используемые при тестировании

Краткое содержание:

установка Charles, BurpSuite;
установка сертификатов для парсинга запросов.

**2 Appium.
Capabilities****Цели занятия:**

познакомиться с инструментом Appium;
написать и запустить тесты на мобильном устройстве.

Краткое содержание:

что такое Appium;
Capabilities;
написание тестов:
- поиск и выбор случайного курса;
- поиск и выбор определенного курса.

3 Selenide**Цели занятия:**

познакомиться с Selenide, узнать какую проблематику он решает;
познакомиться с основными командами и методами.

Краткое содержание:

знакомство с Selenide;
настройка через Configuration;
написание тестов с использованием Selenide;
подключение Appium к Selenide.

Цели занятия:

реализовать полученные знания, установить сниффер (Charles, BurpSuite) на предложенном приложении.

Краткое содержание:

скачивание и запуск тестового приложения;
написание под него тестов с использованием Selenide,
запуск их на Selenium;
ответы на вопросы, подготовка к ДЗ.

Домашние задания

1 ДЗ #5: Мобильное тестирование

Цель: Необходимо установить предложенное приложение, узнать какой метод выдает ошибку. Написать тесты на Appium. Ученику будет предоставлено мобильное приложение.

С использованием Appium (Selenide по желанию)

- 1) Установить приложение на эмулятор
- 2) Написать тесты:
 - а) На проверку раздела Posts
 - б) На проверку раздела Posts/{ID}
 - в) На проверку раздела Users/{ID}
- 3) Установить Burpsuite/Charles, понять, какой метод выдает ошибку

1 **Что такое CI/CD, обзор популярных серверов (Jenkins, Gudson, Travis, TeamCity)**

Цели занятия:

рассмотреть CI- и CD-процессы;
рассмотреть основные инструменты и ключевые понятия.

Краткое содержание:

CI- и CD-процессы;
основные инструменты и ключевые понятия;
Jenkins;
Gudson;
Travis;
TeamCity.

2 **Jenkins**

Цели занятия:

настроить и запустить сервис, интегрировать с git и Docker;
рассмотреть pipeline-подход.

Краткое содержание:

установка;
знакомство с Jenkins;
первичная настройка;
интеграция git и Docker;
подключение GitHub'a;
WebHooks;
Schedulling.

3 Allure, выгрузка отчетов

Цели занятия:

организовать понятную отчетность по автотестам,
внедрить инструменты для сбора логов тестов,
скриншотов приложения и записи видео.

Краткое содержание:

подключение Allure к тестам;
Allure-portal;
подключение к CI/CD;
что такое отчетность;
снятие скриншотов;
запись видео.

Цели занятия:

настраивать Jenkins'a с работой Grid/Selenoid;
запустить тесты на Selenoid и на Selenium Grid.

Краткое содержание:

запуск тестов на Selenoid;
запуск тестов на Selenium Grid;
запуск тестов API;
настройка Jenkins'a с работой Grid/Selenoid;
ответы на вопросы.

Домашние задания

- 1 ДЗ #6: Разворачивание Jenkins'a и подключение джоб

Цель: Необходимо настроить Jenkins, который будет запускать тесты по триггеру и выдавать читаемый отчет в Allure.

Сдача проекта будет происходить путем выдачи xml с настроенными job'ами

- 1) Джоба с запуском Selenium/Selenide тестов по тестированию Otus'a (можно из ранее сделанных) с возможностью выбора браузера
- 2) Джоба с запуском Apprium тестов по тестированию мобильного приложения (можно из ранее сделанных)

Обратите внимание, в п.2 должна быть реализована автоматическое скачивание приложения.

1 Консультация

Цели занятия:

получить ответы на вопросы по проекту, ДЗ и по курсу.

Краткое содержание:

вопросы по улучшению и оптимизации работы над проектом;
затруднения при выполнении ДЗ;
вопросы по программе.

Домашние задания

1 Проектная работа

Цель: В проектной работе студенту предлагается реализовать на практике все полученные знания. Необходимо будет развернуть CI/CD (jenkins), подключить к нему джобы для запуска автотестов (веб, мобайл), настроить отчеты Allure, реализовать запуск тестов на selenoid

- 1) Необходимо скачать предложенный фронт и мобильное приложение
- 2) Развернуть и настроить selenoid
- 3) Развернуть и настроить Jenkins
- 4) Настроить отчеты Allure
- 5) Написать автотесты на WEB с использованием Cucumber/Selenide
- 6) Написать автотесты на Mobile App (android) с использованием Appium
- 7) Написать Stub/Mock для Web & Mobile App
- 8) Написать API тесты на mock с использованием JsonSchemaValidation
- 9) Замокать падающий метод

Тесты должны запускаться либо по пушу в мастер и по расписанию (каждый день в полночь)

Контракты:

- 1) GET /users/get/all
Возвращает массив объектов
[
{'id': int,

```
'name': String,  
'grade': String,  
},  
{'id': int,  
'name': String,  
'grade': String,  
},  
]
```

2) GET /users/{id}
Возвращает объект

```
{  
'id': int,  
'name': String,  
'grade': String,  
'school_name': String,  
'city': String,  
},
```

Дополнительно:
Необходимо реализовать слушатель, который
будет подсвечивать элементы перед нажатием.
В отчет необходимо добавлять видео с прогоном
теста.

2 Защита проектных работ

Цели занятия:

реализовать проект автоматизации тестирования с
применением имеющихся знаний и навыков для
заданного приложения;
защитить проект и получить рекомендации экспертов.

Краткое содержание:

презентация проектов перед комиссией;
вопросы и комментарии по проектам.

1 **Карьерный
день с
рекрутером**

Цели занятия:

подготовить резюме в зависимости от выбранного направления (web, mobile, back) для предстоящего собеседования.

Краткое содержание:

как подготовить резюме;
что ожидать на собеседовании.

2 **Собеседование
у
преподавателей**

Цели занятия:

какие вакансии и позиции есть для выпускников (опционально - в России и за рубежом) и на какие компании стоит обратить внимание.
разобрать требования к qa automation на рынке труда в России;
обсудить основные вопросы, которые задают на собеседовании.

Краткое содержание:

рынок вакансий qa automation;
собеседование.