Конфигурирование и запуск Selenium-тестов JepRia-приложений с помощью TestNG

Оглавление

[Конфигурирование и запуск Selenium-тестов JepRia-приложений с помощью TestNG 1](#_Toc450737215)

[1. Используемое ПО 2](#_Toc450737216)

[2. Конфигурирование тестов для TestNG 2](#_Toc450737217)

[2.1 Определение (фильтрация) запускаемых тестовых методов 2](#_Toc450737218)

[2.1.1 Группировка тестов в java-классах 2](#_Toc450737219)

[2.1.2 Фильтрация запускаемых тестов в test-suite-файлах 2](#_Toc450737220)

[3. Определение цели для Selenium-тестирования в Ant 3](#_Toc450737221)

[4. Запуск тестов с помощью Ant 4](#_Toc450737222)

[4.1.1 Определение параметров запуска тестов в командной строке 4](#_Toc450737223)

[5. Дистрибутивы 5](#_Toc450737224)

[5.1 Selenium Standalone Server 5](#_Toc450737225)

[5.2 Браузер Firefox 5](#_Toc450737226)

[6. Пример тестирования модуля RequestFeature приложения JepRiaShowcase, разрабатываемого в стиле TDD 5](#_Toc450737227)

[6.1 Подготовка модуля к тестированию (на примере RequestFeature) 5](#_Toc450737228)

[6.2 Написание тестов 5](#_Toc450737229)

[6.3 Запуск тестов до реализации функционала 6](#_Toc450737230)

[6.4 Реализация функционала 6](#_Toc450737231)

# Используемое ПО

Для тестирования JepRia-приложений на уровне пользовательского интерфейса используется:

* *Selenium Standalone Server*, который содержит реализацию WebDriver для наиболее распространённых браузеров, а также все необходимые ему библиотеки и тестовый framework TestNG (ссылки на дистрибутив здесь: 5.1).
* Браузер Firefox (ссылки на дистрибутив здесь: 5.2)

# Конфигурирование тестов для TestNG

ТеstNg отличается возможностью очень гибкого конфигурирования в xml-описателях конфигурации тестирования (далее по тексту - в test-suite-файлах) наборов используемых тестовых методов: при помощи указания:

* используемых тестовых Java-классов
* используемых тестовых методов тестовых классов
* произвольно создаваемых групп тестовых методов

В дополнение к этому стоит отметить возможность запуска из ant и командной строки произвольного «набора наборов» TestSuites.

## Определение (фильтрация) запускаемых тестовых методов

### Группировка тестов в java-классах

Ниже приведен пример размещения тестового метода *find* в группах *find* и *mandatory,* осуществляемого при помощи параметра аннотации *@Test groups*:

/\*\*

\* Тест операции поиска по пустому шаблону

\*/

@Test(**groups = "find, mandatory")**

**public** **void** find() {

. . .

}

В конфигурационных test-suite-файлах указание имён таких групп используется для фильтрации запускаемых тестовых методов.

### Фильтрация запускаемых тестов в test-suite-файлах

Ниже приводится пример конфигурационного test-suite-файла:

*<?xml version="1.0" encoding="UTF-8"?>*

*<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">*

*<suite name="GoodsAutoTestSuite" parallel="none">*

*<test name="GoodsAutoTest">*

*<parameter name="baseUrl" value="http://${host}/JepRiaShowcase/JepRiaShowcase.jsp?em=Goods"/>*

*<parameter name="browserName" value="${browserName}"/>*

*<parameter name="browserPath" value="${browserPath}"/>*

*<parameter name="username" value="${testUsername}"/>*

*<parameter name="password" value="${testPassword}"/>*

*<parameter name="jepriaVersion" value="${jepriaVersion}"/>*

*<parameter name="forceNewBrowser" value="No"/>*

*<parameter name="forceLogin" value="No"/>*

*<groups>*

*<define name="all">*

*<include name="find" />*

*<include name="create" />*

*<include name="delete" />*

*<include name="edit" />*

*<include name="goto" />*

*<include name="list" />*

*</define>*

*<run>*

*<!-- <include name="all" /> -->*

*<!-- <include name="edit" /> -->*

*<include name="setAndGetTextField" />*

*</run>*

*</groups>*

*<classes>*

*<class name="com.technology.jep.jepriashowcase.goods.auto.GoodsAutoTest">*

*<methods>*

*<!-- <include name="fillSearchForm" /> -->*

*…*

*</methods>*

*</class>*

*</classes>*

*</test> <!-- Test -->*

*</suite> <!-- Suite -->,*

где в теге *<test>* определяются параметры и набор запускаемых тестов.

Параметры запуска задаются в тегах *<parameter>*.

В test-suite-файлах множество запускаемых тестов определяется тегами *<groups>* и *<classes>*.

#### Теги <*parameter>*

В тегах *<parameter>* определяются *параметры запуска тестов* (не путать с параметрами тестов).

Параметры запуска тестов описывают основные условия запуска: тип браузера (chrome, firefox, …), на котором будут выполняться тесты, пользователя, из-под которого будет выполняться тестовый вход и т.п.

#### Тег <*groups>*

##### Тег *<define>*

В теге *<define>* можно определять дополнительные – составные запускаемые группы тестов.

##### Тег *<run>*

В теге *<run>* определяется набор запускаемых групп тестов.

#### Тег *<classes>*

В теге *<classes>* определяется набор java-классов, содержащих тестовые методы.

##### Тег *<сlass>/<methods>*

В подтеге *<class>/<methods>* с помощью тега *<include>* определяется набор используемых методов java-классов.

Быстрое изменение запускаемого набора тестов (фильтрацию тестов) удобно выполнять комментированием/раскомментированием строк файла (с тегами *<include>*).

# Определение цели для Selenium-тестирования в Ant

Цель (target) ***test-by-selenium*** для Selenium-тестирования JepRia-приложения определяется следующим образом:

*<target name="test-by-selenium" depends="testng-compile">*

*<taskdef classname="org.testng.TestNGAntTask" name="testng">*

*<classpath>*

*<pathelement location="${SELENIUM\_HOME}/libs/testng-6.8.5.jar" />*

*</classpath>*

*</taskdef>*

*<testng classpathref="run-testng-libs"*

*outputDir="${testng.report.dir}"*

*haltOnFailure="true"*

*verbose="2"*

*delegateCommandSystemProperties="true">*

*<classpath location="${test.src.dir}" />*

*<classpath location="${JEP\_RIA}/lib/jepria-test.jar" />*

*<classpath refid="selenium-libs" />*

*<sysproperty key="host" value="${host}"/>*

*<sysproperty key="browserName" value="${browserName}"/>*

*<sysproperty key="browserPath" value="${browserPath}"/>*

*<sysproperty key="jepriaVersion" value="${jepriaVersion}"/>*

*<sysproperty key="testUsername" value="${testUsername}"/>*

*<sysproperty key="testPassword" value="${testPassword}"/>*

*<xmlfileset dir="${test.conf.dir}" includes="\*\*/auto/\*\*/GoodsAutoTest.xml"/>*

*</testng>*

*</target>*

Группа тегов *<sysproperty>* используется для определения внешних переменных, передаваемых в test-suite (см. соответствующую группу тегов *<parameter>* в описателе test-suite: 2.1.2). Значения этих переменных задаются в файле свойств ant *test.properties*, доступном из *build.xml*, например:

*host=srvt16.d.t*

*testUsername=NagornyyS*

*testPassword=123*

*browserName=firefox*

*browserPath=D:/Firefox-33/firefox.exe*

*jepriaVersion=8.0.0*

# Запуск тестов с помощью Ant

Запуск selenium-тестов приложений JepRia из командной строки с помощью Ant выполняется командой

*ant test-by-selenium*

### Определение параметров запуска тестов в командной строке

Иногда бывает нужно выполнить быструю проверку работы тестов в конфигурациях, отличных от той, которая определена в используемых test-suite. Для этого не обязательно редактировать test-suite.

Перекрыть параметры запуска тестов (см. 3) текущих test-suite можно параметрами команды запуска ant – эти параметры имеют больший приоритет, чем соответствующие параметры test-suite.

Для того, чтобы это сделать, при запуске ant нужно задать нужные внешние переменные с использованием ключа *-D*, например, значение параметра *browserName* можно изменить командой

*ant* ***–DbrowserName=”chrome”*** *test-by-selenium*

# Дистрибутивы

## Selenium Standalone Server

Используемый дистрибутив 2.43.1 находится здесь:

[X:\!IT DEPARTMENT\Divisions\Applications\Distrib\Selenium\selenium-2.43.1](file:///X:\!IT%20DEPARTMENT\Divisions\Applications\Distrib\Selenium\selenium-2.43.1)

Последние версии скачиваются с официальной страницы отсюда: <http://www.seleniumhq.org/download/>.

## Браузер Firefox

Не все версии Firefox нормально работают с Selenium.

Firefox 33 работает нормально, лежит здесь:

[X:\!IT DEPARTMENT\Divisions\Applications\Distrib\FireFox\Firefox Setup 33.0.exe](file:///X:\!IT%20DEPARTMENT\Divisions\Applications\Distrib\FireFox\Firefox%20Setup%2033.0.exe)

# Пример тестирования модуля RequestFeature приложения JepRiaShowcase, разрабатываемого в стиле TDD

Тесты для приложений, разрабатываемых в стиле TDD, должны быть написаны *до* реализации функционала, основываясь на поставленных требованиях к приложению. Ввиду отсутствия самого тестируемого функционала, до начала разработки приложение не должно успешно проходить эти тесты, однако сам процесс тестирования должен успешно запускаться. Для этого необходимо иметь базовую реализацию модуля, не содержащую функциональных деталей.

## Подготовка модуля к тестированию (на примере RequestFeature)

Базовая реализация модуля заключается в следующем:

* URL модуля должен существовать и быть доступен (без указания URL в качестве параметра тестового сценария, сценарий не будет успешно запускаться):   
  <http://srvt16.d.t/JepRiaShowcase/JepRiaShowcase.jsp?em=RequestFeature>
* На данном этапе модуль только *корректно* *загружается по URL* и не содержит никаких функциональных элементов. В клиентской фабрике модуля формируются стандартные представления [List|Detail]FormViewImpl, под управлением стандартных презентеров [List|Detail]FormPresenter; весь серверный код (в том числе логика взаимодействия с базой данных) отсутствует.

## Написание тестов

В качестве самого примитивного теста рассмотрен тест заполнения полей на форме добавления новой записи. Тестовые методы заносятся в файл RequestFeatureAutoTest.xml, а их реализация – в RequestFeatureAutoTest.java. Идентификация несуществующих пока что полей происходит по специальным id, заданным в файле RequestFeatureAutomationConstant.java.

Примитивный тест включает в себя:

* Загрузку модуля по URL
* Регистрация в системе под заданным логином и паролем
* Переход в режим создания записи
* Заполнение полей формы значениями из data-файлов
* Проверка корректности заполнения полей.

## Запуск тестов до реализации функционала

Тестирование (конфигурируемое в файле test.properties) запускается командой:  
ant test-by-selenium  
(при этом может потребоваться перекомпиляция измененных тестовых файлов) . На данном этапе в процессе тестирования происходит успешная регистрация, переход в режим создания записи; далее все тесты на заполнение полей завершаются ошибками-исключениями org.openqa.selenium.NoSuchElementException, что говорит об отсутствии функциональных элементов на форме.

## Реализация функционала

TOWRITE: Реализация заключается в наполнении формы полями с используемыми при тестировании id……..