Testing using Serenity and JUnit in Java

Overview

Serenity is a framework for BDD, promoting the separation between steps implementation and the actual specification in each test case.

In this tutorial, we will create some tests using Serenity along with JUnit in Java and we will use Xray just to have visibility of the test results.

Description

The following automated tests are taken from Serenity demo tutorials here.

> **Please note**
>
> The original example can be found in the public Github repository from Serenity.

The serenity configuration file should be updated in order to use the correct driver; in our case we'll use "chrome".

serenity.properties

```
webdriver.driver=chrome
#serenity.project.name = Demo Project using Serenity and Cucumber
#serenity.test.root=net.thucydides.showcase.junit.features
#serenity.dry.run=false
#serenity.requirement.types=feature, story
#serenity.use.unique.browser = true

serenity.browser,height = 1200
serenity.browser,width = 1200

#serenity.sourceDirectory=target/site/out
#serenity.outputDirectory=target/site/out
```

Steps are implemented in auxiliary classes such as the ones provided below.

src/test/java/net/thucydides/showcase/junit/steps/serenity/BuyerSteps.java

```
package net.thucydides.showcase.junit.steps.serenity;

import net.serenitybdd.core.Serenity;
import net.thucydides.core.annotations.Step;
import net.thucydides.core.steps.ScenarioSteps;
import net.thucydides.showcase.junit.model.ListingItem;
import net.thucydides.showcase.junit.model.OrderCostSummary;
import net.thucydides.showcase.junit.pages.CartPage;
import net.thucydides.showcase.junit.pages.HomePage;
import net.thucydides.showcase.junit.pages.ListingPage;
import net.thucydides.showcase.junit.pages.SearchResultsPage;

import static org.assertj.core.api.Assertions.assertThat;

public class BuyerSteps extends ScenarioSteps {

    HomePage homePage;
    SearchResultsPage searchResultsPage;
    ListingPage listingPage;
    CartPage cartPage;

    @Step
```

```
public void opens_home_page() {
    homePage.open();
}

@Step
public void searches_by_keyword(String keyword) {
    homePage.enterSearchTerms(keyword);
    homePage.search();
    homePage.dismissLocationMessage();
}

@Step
public void should_see_results_summary_containing(String keyword) {
    assertThat(searchResultsPage.getSearchHeader()).containsIgnoringCase(keyword);
}

@Step
public void searches_for_shop_called(String shopName) {
    homePage.searchForShopCalled(shopName);
}

@Step
public void should_see_shop_search_result_summary_of(String expectedMessage) {
    assertThat(searchResultsPage.getResultSummary()).isEqualToIgnoringCase(expectedMessage);
}

@Step
public ListingItem selects_listing(int articleNumber) {
    return searchResultsPage.selectListing(articleNumber);
}

@Step
public void should_see_product_details_for(ListingItem selectedListingItem) {
    ListingItem displayedListingItem = listingPage.getDisplayedListing();

}

@Step
public void adds_current_listing_to_cart() {
    listingPage.selectOptionIfPresent();
    listingPage.addToCart();
}

@Step
public void should_see_item_in_cart(ListingItem selectedItem) {
    assertThat(cartPage.getOrderCostSummaries()
            .stream().anyMatch(order -> order.getName().equals(selectedItem.getName()))).isTrue();
}

@Step
public void should_see_total_including_shipping_for(ListingItem selectedItem) {
    OrderCostSummary orderCostSummary = cartPage.getOrderCostSummaryFor(selectedItem).get();

    double itemTotal = orderCostSummary.getItemTotal();
    double shipping = orderCostSummary.getShipping();

    assertThat(itemTotal).isEqualTo(selectedItem.getPrice());
    assertThat(shipping).isGreaterThan(0.0);
}

@Step
public void should_see_product_rating() {
    assertThat(listingPage.getRating()).isGreaterThan(0);
}

@Step
public void should_see_twitter_link() {
    listingPage.twitterIcon().shouldBeVisible();
}

@Step
```

```
    public void should_see_tumblr_link() {
        listingPage.tumblrIcon().shouldBeVisible();
    }

    @Step
    public void should_see_facebook_link() {
        listingPage.facebookIcon().shouldBeVisible();
    }


    @Step
    public void should_see_nonexistant_field() {
        assertThat(listingPage.clickImaginaryButton()).isFalse();
    }


    @Step
    public void filters_by_local_region() {
        searchResultsPage.filterByLocalRegion();
    }
}
```

src/test/java/net/thucydides/showcase/junit/features/cart/AddItemsToCartTest.java

```
package net.thucydides.showcase.junit.features.cart;

import net.serenitybdd.junit.runners.SerenityRunner;
import net.thucydides.core.annotations.Managed;
import net.thucydides.core.annotations.Steps;
import net.thucydides.core.annotations.WithTag;
import net.thucydides.core.annotations.WithTags;
import net.thucydides.showcase.junit.model.ListingItem;
import net.thucydides.showcase.junit.steps.serenity.BuyerSteps;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.openqa.selenium.WebDriver;

@RunWith(SerenityRunner.class)
public class AddItemsToCartTest {
    @Managed
    WebDriver driver;

    @Steps
    BuyerSteps buyer;

    @Test
    public void add_item_to_cart() {
        // GIVEN
        buyer.opens_home_page();
        buyer.searches_by_keyword("docking station");
        buyer.filters_by_local_region();

        // WHEN
        ListingItem selectedItem = buyer.selects_listing(2);
        buyer.adds_current_listing_to_cart();

        // THEN
        buyer.should_see_item_in_cart(selectedItem);
        buyer.should_see_total_including_shipping_for(selectedItem);
    }
}
```

The sample project contains several features specification, with the automated Tests. Below are two of them:

src/test/java/net/thucydides/showcase/junit/features/petstore/AddAPetToThePetStoreTest.java

```
package net.thucydides.showcase.junit.features.petstore;


import net.serenitybdd.junit.runners.SerenityRunner;
import net.thucydides.core.annotations.Steps;
import net.thucydides.showcase.junit.model.Pet;
import net.thucydides.showcase.junit.steps.serenity.PetStoreSteps;
import org.junit.Test;
import org.junit.runner.RunWith;

@RunWith(SerenityRunner.class)
public class AddAPetToThePetStoreTest {

    @Steps
    PetStoreSteps petStore;

    @Test
    public void shouldBeAbleToAddPetsToAStore() {

        Pet fido = new Pet("available","fido");

        petStore.when_i_add_the_pet_to_the_store(fido);

        petStore.the_pets_should_be_available();
    }

    @Test
    public void shouldBeAbleDeletePetsFromAStore() {

        Pet fido = new Pet("available","fido");

        petStore.when_i_add_the_pet_to_the_store(fido);

        petStore.when_i_delete_the_pet();

        petStore.the_pets_should_be_not_available();
    }

    @Test
    public void shouldBeAbleDeletePetsFromAStoreUsingJsonMap() {

        Pet fido = new Pet("available","fido");

        petStore.when_i_add_the_pet_to_the_store_with_json_map(fido);

        petStore.when_i_delete_the_pet();

        petStore.the_pets_should_be_not_available();
    }
}
```

Tests can be run using Maven.

```
mvn clean test
```

Since the previous command generates multiple JUnit XML files, we may need to merge them into a single XML file so it can be submitted into a Test Execution more easily. That can be achieved by using the junit-merge utility.

```
junit-merge  -o results.xml -d target/site/serenity/
```

After successfully running the Test cases and generating the aggregated JUnit XML report (e.g., results.xml), it can be imported to Xray (either by the REST API or through the **Import Execution Results** action within the Test Execution).

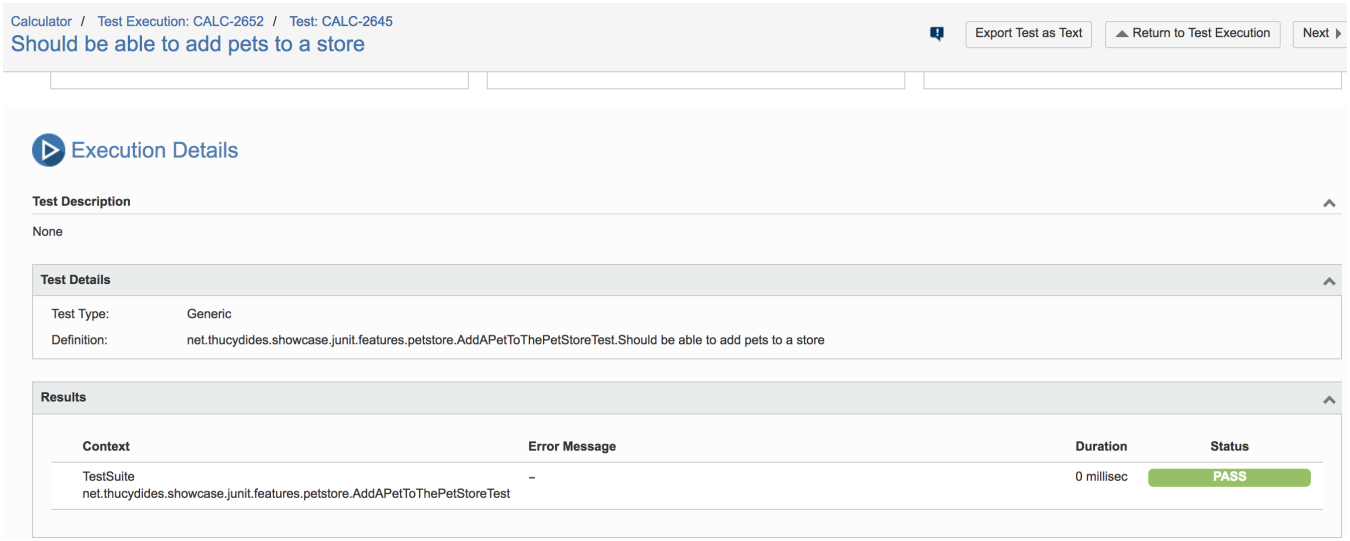
	Key	Summary	Test Type	#Req	#Def	Test Sets	Assignee	Steps Progress	Status
1	CALC-2645	Should be able to add pets to a store	Generic	0	0		Administrator	N/A	PASS
2	CALC-2647	Search for articles by shop name	Generic	0	0		Administrator	N/A	FAIL
3	CALC-2646	Add item to cart	Generic	0	0		Administrator	N/A	FAIL
4	CALC-2641	Search for articles by keyword	Generic	0	0		Administrator	N/A	FAIL
5	CALC-2643	Display product details from the search list	Generic	0	0		Administrator	N/A	FAIL
6	CALC-2642	Should be able delete pets from a store using json map	Generic	0	0		Administrator	N/A	PASS

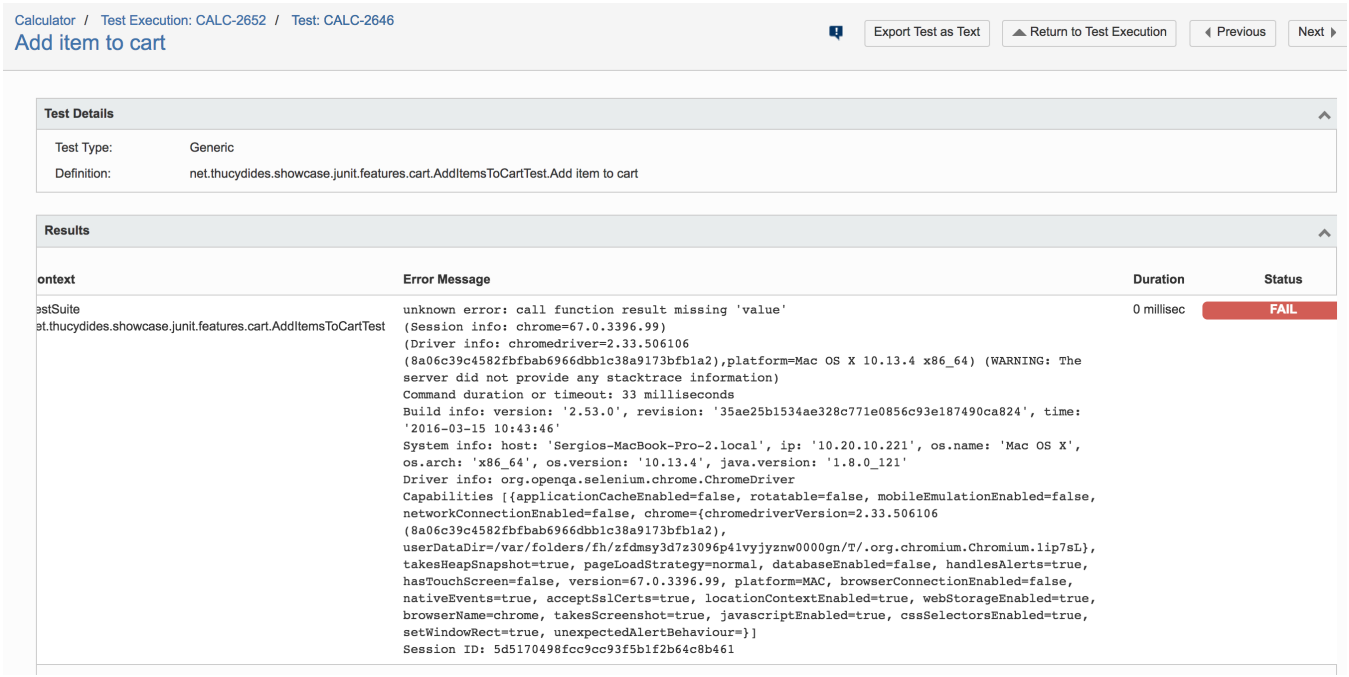
Showing 1 to 6 of 6 entries

First Previous 1 Next Last

Each JUnit's Test Case is mapped to a Generic Test in Jira, and the **Generic Test Definition** field contains the name of the package, the class and the method name that implements the Test Case. The summary of each Test issue is filled out with the name of the method corresponding to the JUnit Test.

The Execution Details of the Generic Test contains information about the Test Suite, which in this case corresponds to the Test Case class, including its namespace.





References

- https://github.com/serenity-bdd/serenity-demos/tree/master/junit-webtests