**High level requirement analysis**

First of all, the purpose of creation spy tool is not clear.

If it's for a quick elements description on the page, than it is suitable to use selenium IDE http://www.seleniumhq.org/docs/02\_selenium\_ide.jsp. It allows to record user actions with further automatic identification of elements locators and creation of corresponding action. Then this action can be easily imported to Java code.

The view of spy is also incomprehensible. Perhaps it is meant something like ready-made solution QTP Spy(Quick Test Pro) which allows to find components on a Web page, has settings for recognizing only buttons and then use this element for scenario composition.

Headline of Functional Specifications

To speed up test development, you are creating a GUI wrapper to TestComplete or Selenium. This wrapper should enable faster development of automated tests by streamlining some common tasks such as finding and mapping objects, etc.

does not match further requirements (no mention about test case).

Сoncerning actions that should belong to button:

ClickMiddle, ClickRight - uncertain names, can imply click in the middle of the button or click with middle mouse key

Close - can not be applied to button

Detailed analysis for Test Case table is given with comments. For more convenient view please use link https://docs.google.com/document/d/1vgEutiPUF\_b5ZiMctOpeAi6bMnFZmid7V3KK1qyyg\_c/edit?usp=sharing

|  |  |  |
| --- | --- | --- |
| Sr. No. | Test Step | Expected Result |
| 1 | Go to https://www.crossover.com/ | Crossover home page should open. |
| 2 | Click on JOBS and verify it was navigated to jobs(available-jobs) page | Default available-jobs page should open. |
| 3 | Focus on the "Job title, keywords" field. | Focus should set on the "Job title, keywords" field. |
| 4 | Type the text "Chief". | The text "Chief" should get typed in the "Job title, keywords" field. |
| 5 | Click on the "SEARCH JOBS" button and verify that it shows the result accordingly. | The search should display results containing the text "Chief" only. |
| 6 | Click on the "RESET" button. | Default available-jobs page should open. |
| 7 | Click on "All Job Categories" drop-down box. | "All Job Categories" drop-down box should display various job categories. |
| 8 | Click on "Java" and verify that it gives the result as per search criteria. | The search should display results containing the text "Java" only. |
| 9 | Click on the "RESET" button. | Default available-jobs page should open. |
| 10 | Click on "COMPANIES" and verify that it was navigated to Main page. | Crossover home page should open. |
| 11 | Close the browser. | The browser under test execution should get closed. |

By definition test case should describe one test function. So this table can not be named as Test Case, maybe Test Scenario. But anyway it touches different functions such search and navigation menu and should be divided into independent scenarios. Without dividing consideration, for covering this table with automation test I assume representation of this table:

|  |  |  |
| --- | --- | --- |
| Sr. No. | Test Step | Expected Result |
| 1 | Go to https://www.crossover.com/ | Current page url should be <https://www.crossover.com/>, html body contains text “home page” |
| 2 | Click on JOBS link | Current page url should be <https://www.crossover.com/x/#/marketplace/available-jobs>, available job table present |
| 4 | Type the text "Chief” in job title, keywords field |  |
| 5 | Click on the "SEARCH JOBS" button | The search results titles should contain the text "Chief" |
| 6 | Click on the "RESET" button. | Current page url should be <https://www.crossover.com/x/#/marketplace/available-jobs> |
| 8 | Click on "Java" | The search results titles should contain the text "Java" |
| 9 | Click on the "RESET" button. | Current page url should be <https://www.crossover.com/x/#/marketplace/available-jobs> |
| 10 | Click on "COMPANIES" | Current page url should be <https://www.crossover.com/>, html body contains text “home page” |

Also requirement to use \Source\ ATC\ package is improper due to oracle naming convention:

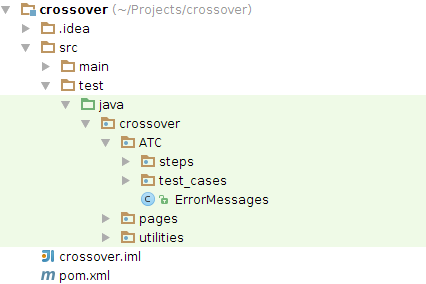
Package names are written in all lower case

<https://docs.oracle.com/javase/tutorial/java/package/namingpkgs.html>

Some of requirements presumably were copied from developer task and are not designed for Chief Architect - Test Automation position.

**Technical implementation analysis**

Delivered test framework has next structure:



There are few levels of logic:

* Level of test scenarios
* Level of scenario steps
* Level of page logic (Page object pattern is used, for enhancing test maintenance and reducing code duplication)

ATC package contains automated test cases classes:

-Steps folder contains classes that describe steps for each page

-Test\_cases folder contains classes that describe logic of test cases

-Pages folder describe methods for iterating with each page

- Utilities contains button wrapper class and also auxiliary methods for selenium

For implementing button wrapper I used Html Elements Java library that provide easy-to-use way of interaction with web-page elements. Extending TypifiedElement abstract class allows to create own typified elements and then use this element as WebElement while describing Page Factory. It also has own HtmlElementLocatorFactory that is needed for initialization fields annotated with @FindBy.

ClickMiddle method was not implemented, because Interface org.openqa.selenium.interactions.Mouse does not have method for middle mouse button (There is text “// TODO: Scroll wheel support”). If this method was present, it would be possible to extend MouseAction like this:

public class MiddleClick extends MouseAction implements Action {

public MiddleClick(Mouse mouse, Locatable where) {

super(mouse, where);

}

public void perform() {

moveToLocation();

mouse.clickMiddle(getActionLocation());

}

public List<Object> asList() {

return Arrays.<Object>asList("click", getTargetId(), Button.MIDDLE, 1);

}

}

But due to TODO mark in selenium library it is impossible to use built-in methods like Actions for implementing this.

In practice, I would not implement this accessory method until it will be needed.

Ways for improving this assignment:

* Use additional library like Serenity(Thucydides). It is common practice to use addition framework that simplifies work with browser, extends selenium methods, generates detailed report. It allows to not carry about driver transferring between classes, use PageObject class that has common page method like openPage(), checkUrlPatterns() and many others. So using this framework is one of the way of speeding up test development.
* Extend all page objects from customPageObject and implement common methods like open page in parent class
* Divide generalTestCase into few test cases like this:

@Test

public void verifyHomePageOpen(){

homePageSteps.openHomePage();

homePageSteps.verifyPageOpen();

}

@Test

public void verifyHomePageNavigationMenu(){

homePageSteps.openHomePage();

homePageSteps.selectMenuTab(HomePage.MenuTab.JOBS);

availableJobsSteps.verifyPageOpen();

}

@Test

public void verifyKeywordSearch(){

final String searchKeyword = "Chief";

availableJobsSteps.searchForKeyword(searchKeyword);

availableJobsSteps.verifySearchFilterByKeywords(searchKeyword);

availableJobsSteps.verifyJobTitlesContainsKeywords(searchKeyword);

availableJobsSteps.resetSearchResult();

availableJobsSteps.verifyPageOpen();

}

@Test

public void verifyCategorySearch(){

availableJobsSteps.searchForJobCategory(JobCategory.JAVA);

availableJobsSteps.verifyCategoryIsSelected(JobCategory.JAVA.getCategoryName());

availableJobsSteps.verifyJobTitlesContainsKeywords(JobCategory.JAVA.getCategoryName());

availableJobsSteps.resetSearchResult();

availableJobsSteps.verifyPageOpen();

}

@Test

public void verifyAvailableJobsNavigationMenu(){

availableJobsSteps.openAvailableJobsPage();

availableJobsSteps.selectMenuTab(AvailableJobsMenuTab.COMPANIES);

homePageSteps.verifyPageOpen();

}

### 