

## BLOG

[Home](#) / [Blog](#) / [How To...Selenium: How to Use Expected Conditions](#)

## How To...Selenium: How To Use Expected Conditions

[GO BACK](#)Posted by [Ronan](#) | 04/01/2017

Each month on the first Tuesday of the month, we will post a new blog post to take you through a step-by-step guide on how to address a particular aspect of using Selenium as part of our How To series. Our guest blogger Alex will showcase and demonstrate how to solve a new problem in detail.

Find the previous posts here:

### [How To...Select Elements In Selenium WebDriver Scripts](#)

### [How To...Use Explicit Waits In Selenium WebDriver](#)

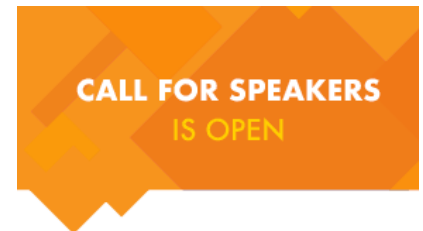
### [How To...Manage Exceptions In Selenium WebDriver](#)

### [How To...Execute Javascript In Selenium](#)

### [How To...Take Screenshots In Selenium](#)

### [How To...Interact With Sliders In Selenium](#)

### [How To...Interact With Modal Windows](#)



## Latest eBooks

## Latest Webinars



Optimizing Test Automation through Android & iOS Property Files

#Mobile Testing



A Pattern Language for Agile Roadmapping

#Agile Testing



Beautiful Testing is Efficient Testing

#Agile Testing



Make It Fail; Debugging

#Other



Structured Conversations

#BDD



Observing and Reasoning About Errors

#People/Skills



Automation & Management: Conversations To Create Trust.

#Upcoming! #Test Automation



Neuroscience; The Secret to Becoming a Better Agile Coach

#Upcoming! #Agile Testing

Want some help with Selenium & Expected conditions? You need custom expected conditions when the built-in Selenium WebDriver expected conditions are not sufficient for creating complex conditions.

## SHORT REVIEW OF EXPECTED CONDITIONS

Explicit **waits** and expected conditions are a great help for Selenium scripts.

Selenium WebDriver comes with a lot of expected conditions such as

1. ExpectedCondition < WebElement > **elementToBeClickable**(By locator)
2. ExpectedCondition < Boolean > **elementToBeSelected**(By locator)
3. ExpectedCondition < WebElement > **presenceOfElementLocated**(By locator)
4. ExpectedCondition < Boolean > **titleContains**(String title)
5. ExpectedCondition < Boolean > **titleIs**(String title)
6. ExpectedCondition < Boolean > **urlContains**(String fraction)
7. ExpectedCondition < Boolean > **urlToBe**(String url)
8. ExpectedCondition < WebElement > **visibilityOfElementLocated**(By locator)

There are 2 main expected condition types that can be used with explicit **waits**:

### ExpectedCondition < WebElement >

This type of condition has a web element locator as parameter.

The **wait** applies the condition which tries finding the web element, depending on its status.

If the condition can find the element, it returns the element as result.

If it cannot find the element, the **wait** tries the condition again after a short delay.

### ExpectedCondition < Boolean >

This type of condition has a string parameter.

The **wait** applies the condition to the parameter.

If the result of applying the condition is true, true is returned as result.

If the result is false, the **wait** tries the condition again after a short delay.

While the explicit **wait** applies the expected condition, the condition code may generate various exceptions.

**All these exceptions are ignored by the wait.**

**Enough theory, lets see some code.**

The following short example uses explicit **waits** and expected conditions to

- verify if a page title is correct
- verify if a page url is correct
- find web elements

```
public class TestClass {
    WebDriver driver;
    WebDriverWait wait;

    By searchFieldXpath = By.id("globalQuery");
    By searchButtonXpath = By.className("search_button");

    By resultLinkLocator = By.xpath("//a[@testid='bib_link']")[1];

    String homeUrl = "http://www.vpl.ca";
    String homeTitle = "Vancouver Public Library - Home";

    String resultsTitle = "Search | Vancouver Public Library | BiblioCommons";
```



Your Automated Execution Does Not  
Have to be Flaky

#Upcoming! #Test Automation



Succeeding with Rapid and  
Continuous Testing

#Agile Testing



Augmenting the Agile Team – A  
Testing Success Story

#Agile Testing

## BLOG CATEGORIES

- 📍 People/Skills (92)
- 📍 Test Automation (86)
- 📍 Test Management (36)
- 📍 Agile Testing (33)
- 📍 Mobile Testing (28)
- 📍 Other (28)
- 📍 Start-Up Series (14)
- 📍 Non-Functional Testing (11)
- 📍 On This Day (326)
- 📍 Poll (38)

```

@Before
public void setUp() {
    driver = new FirefoxDriver();
    wait = new WebDriverWait(driver, 10);
}

@After
public void tearDown() {
    driver.quit();
}

@Test
public void test1() {

    driver.get(homeUrl);

    if (!wait.until(titleContains(homeTitle)) || !wait.until(urlContains(homeUrl)))
        throw new RuntimeException("home page is not displayed");

    WebElement searchField = wait.until(elementToBeClickable(searchFieldXpath));
    searchField.click();
    searchField.sendKeys(keyword);

    WebElement searchButton = wait.until(elementToBeClickable(searchButtonXpath));
    searchButton.click();

    if (!wait.until(titleContains(resultsTitle)) || !wait.until(urlContains(resultsUrl)))
        throw new RuntimeException("results page is not displayed");
}
}

```

We use 2 expected conditions for verifying if a page is displayed.

Selenium does not include by default an expected condition that checks that both the page title and url are correct.

This is where we can create custom expected conditions.

## HOW TO CREATE A CUSTOM EXPECTED CONDITION SELENIUM

A custom expected condition is a class that

- implements the **ExpectedCondition** interface
- has a constructor with the parameters of the expected condition
- overrides the **apply** method

Lets rewrite the previous exercise with a custom expected condition that verifies if a page is displayed:

```

public class TestClass {

    WebDriver driver;
    WebDriverWait wait;

    By searchFieldXpath = By.id("globalQuery");
    By searchButtonXpath = By.className("search_button");

    By resultLinkLocator = By.xpath("//a[@testid='bib_link']")[1]);

    String homeUrl = "http://www.vpl.ca";
    String homeTitle = "Vancouver Public Library - Home";

    String resultsTitle = "Search | Vancouver Public Library | BiblioCommons";
    String resultsUrl = "https://vpl.bibliocommons.com/search";

    @Before
    public void setUp() {
        driver = new FirefoxDriver();
        wait = new WebDriverWait(driver, 10);
    }

    @After
    public void tearDown() {
        driver.quit();
    }

    @Test
    public void test1() {

        driver.get(siteUrl);

        if (!wait.until(new PageLoaded(homeTitle, homeUrl)))
            throw new RuntimeException("home page is not displayed");

        WebElement searchField = wait.until(elementToBeClickable(searchFieldXpath));
        searchField.click();
    }
}

```

---

By using this website you consent to our use of cookies. For more information on cookies see our [Read More](#)

```

WebElement searchButton = wait.until(elementToBeClickable(searchButtonXpath));
searchButton.click();

if (!wait.until(new PageLoaded(resultsTitle, resultsUrl)))
    throw new RuntimeException("results page is not displayed");
}

}

public class PageLoaded implements ExpectedCondition {
    String expectedTitle;
    String expectedUrl;

    public PageLoaded(String expectedTitle, String expectedUrl) {
        this.expectedTitle = expectedTitle;
        this.expectedUrl = expectedUrl;
    }

    @Override
    public Boolean apply(WebDriver driver) {
        Boolean isTitleCorrect = driver.getTitle().contains(expectedTitle);
        Boolean isUrlCorrect = driver.getCurrentUrl().contains(expectedUrl);

        return isTitleCorrect && isUrlCorrect;
    }
}

```

The PageLoaded custom expected condition is used for verifying if

- HomePage is displayed
- ResultsPage is displayed

For ResultsPage, we would like to verify not only that the page title and url are correct but also that

- there are 25 results loaded in the page
- total result count is > 0

The following custom expected condition does it all:

```

public class ResultsPageLoaded implements ExpectedCondition {
    By resultLocator = By.xpath("//div[contains(@data-analytics, 'SubFeature')]");
    By resultCountLocator = By.xpath("//span[@class='items_showing_count']");

    String expectedTitle;
    String expectedUrl;

    public ResultsPageLoaded(String expectedTitle, String expectedUrl) {
        this.expectedTitle = expectedTitle;
        this.expectedUrl = expectedUrl;
    }

    @Override
    public Boolean apply(WebDriver driver) {
        Boolean isTitleCorrect = driver.getTitle().contains(expectedTitle);
        Boolean isUrlCorrect = driver.getCurrentUrl().contains(expectedUrl);

        int resultPerPageCount = driver.findElements(resultLocator).size();

        WebElement resultCountElement = driver.findElement(resultCountLocator);

        return
            isTitleCorrect &&
            isUrlCorrect &&
            resultPerPageCount == 25 &&
            count(resultCountElement) > 0;
    }

    private int count(WebElement element) {
        String resultCountText = element.getText();
        int index1 = resultCountText.indexOf("of") + 3;
        int index2 = resultCountText.indexOf("items") - 1;
        resultCountText = resultCountText.substring(index1, index2);
        return Integer.parseInt(resultCountText);
    }
}

```

Custom expected conditions can return not only a boolean value but also a WebElement.

Lets define a custom condition that

- finds an element from its parent element
- returns the element if it is displayed

```

public class ResultField implements ExpectedCondition {
    By resultLocator;
    By parentLocator;

    public ResultField(By resultLocator, By parentLocator) {
        this.resultLocator = resultLocator;
        this.parentLocator = parentLocator;
    }

    @Override
    public WebElement apply(WebDriver driver) {
        WebElement parent = driver.findElement(parentLocator);
        WebElement result = parent.findElement(resultLocator);

        return (result.isDisplayed() ? result : null);
    }
}

```

How do you use it?

```

WebElement titleLink = wait.until(new ResultField(resultLocator, titleLocator));
titleLink.click();

```

## ABOUT THE AUTHOR



Alex lives in Vancouver, Canada. He has worked as a software tester since 2005. Alex teaches manual testers [test automation with Selenium WebDriver and Java](#). Alex blogs on testing and automation at <http://test-able.blogspot.ca>.

Like 2

## See Also

How To Selenium: Page Objects- Partial Classes Bas...

How To Selenium: Code Reuse In Page Objects

How To Selenium: Page Objects – Page Object ...

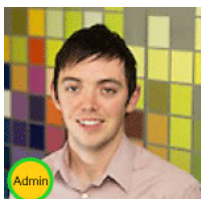
How To Selenium: Page Objects – Partial Clas...

How To Selenium: Page Objects – Partial Clas...

How To Selenium: More Readable Page Objects -...

## BLOG POST ADDED BY

GO BACK



RONAN

Join the discussion!

Share your thoughts on this article by commenting below.

ONE COMMENT TO HOW TO...SELENIUM: HOW TO USE EXPECTED CONDITIONS

Accept



dichotomy says:

January 20, 2018 at 8:37 am

I am regular visitor, how are you everybody? This article posted at this web page is genuinely pleasant.

Reply

Leave a Reply

Powered by [OneAll Social Login](#)  
Your email address will not be published. Required fields are marked \*

Comment

Name \*

Email \*

Website

POST COMMENT

ABOUT

- Welcome to Huddle
- How to Contribute
- How do I get Involved?
- Frequently Asked
- Questions
- Sitemap

RESOURCES

- Mobile Testing
- People/Skills
- Test Automation
- Test Management
- Cloud Testing
- Functional Testing
- Non-Functional Testing
- Upload Resource

CONTACT US

Huddle -  
EuroSTAR Software Testing  
IDA Business Park, Dangan  
H91 P2DK Galway, Ireland  
Tel: +353 (0)91 514 470  
E-mail: [hello@testhuddle.com](mailto:hello@testhuddle.com)

