Validate Downloaded file after clicking on downloaded button/ link

It is very important to verify if the file is downloaded successful or not. Most of the cases we just concentrate on clicking the downloaded button. But at the same time it is also very important to confirm that file is downloaded successfully without any errors or if some other file is getting downloaded.

In most of the cases we know which file is getting downloaded after clicking on download button / link. Now when we know the file name, we can verify using java for the 'File Exists' in a downloaded folder location which we specify.

Even there are cases where file name is not unique. File name may be generated dynamically. In such cases we can also check for the file exists with the file extension.

We will see all the above cases and verify for the file which is being downloaded. We will see examples using Java File IO and WatchService API

```
package com.pack;

import java.io.File;
import java.io.FileFilter;
import java.nio.file.FileSystems;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.StandardWatchEventKinds;
import java.nio.file.WatchEvent;
import java.nio.file.WatchKey;
import java.nio.file.WatchService;
import java.util.Arrays;
import java.util.concurrent.TimeUnit;

import org.apache.commons.io.comparator.LastModifiedFileComparator;
import org.apache.commons.io.filefilter.WildcardFileFilter;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.firefox.FirefoxProfile;
import org.testng.Assert;
```

```
import org.testng.annotations.AfterClass;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Test;

public class FileDownloadVerify {

        private WebDriver driver;

        private static String downloadPath = "D:\\seleniumdownloads";
        private String URL="http://spreadsheetpage.com/index.php/file/C35/P10/";

        @BeforeClass
        public void testSetup() throws Exception{
                driver = new FirefoxDriver(firefoxProfile());
                driver.manage().window().maximize();
        }

        @Test
        public void example_VerifyDownloadWithFileName()  {
                driver.get(URL);
            driver.findElement(By.linkText("mailmerge.xls")).click();
            Assert.assertTrue(isFileDownloaded(downloadPath, "mailmerge.xls"), "Faile
d to download Expected document");
        }

        @Test
        public void example_VerifyDownloadWithFileExtension()  {
                driver.get(URL);
            driver.findElement(By.linkText("mailmerge.xls")).click();
            Assert.assertTrue(isFileDownloaded_Ext(downloadPath, ".xls"), "Failed to
download document which has extension .xls");
        }

        @Test
        public void example_VerifyExpectedFileName() {
                driver.get(URL);
            driver.findElement(By.linkText("mailmerge.xls")).click();
            File getLatestFile = getLatestFilefromDir(downloadPath);
            String fileName = getLatestFile.getName();
            Assert.assertTrue(fileName.equals("mailmerge.xls"), "Downloaded file name
is not matching with expected file name");
        }


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

Here we are defining the preferences to the Firefox browser and we will pass this to the Webdriver. We can set different preferences based on the requirement. In this tutorial there are many other preferences which are used when downloading a file using Firefox Preferences.

```
public static FirefoxProfile firefoxProfile() throws Exception {

                FirefoxProfile firefoxProfile = new FirefoxProfile();
                firefoxProfile.setPreference("browser.download.folderList",2);
                firefoxProfile.setPreference("browser.download.manager.showWhenStarti
ng",false);
                firefoxProfile.setPreference("browser.download.dir",downloadPath);
                firefoxProfile.setPreference("browser.helperApps.neverAsk.saveToDis
k",
"text/csv,application/x-msexcel,application/excel,application/x-excel,application/vn
d.ms-excel,image/png,image/jpeg,text/html,text/plain,application/msword,application/x
ml");

                return firefoxProfile;
        }
```

Below method takes the download directory and the file name, which will check for the file name mention in the directory and will return 'True' if the document is available in the folder else 'false'. When we are sure of the file name, we can make use of this method to verify.

```
public boolean isFileDownloaded(String downloadPath, String fileName) {
                boolean flag = false;
            File dir = new File(downloadPath);
            File[] dir_contents = dir.listFiles();

            for (int i = 0; i < dir_contents.length; i++) {
                if (dir_contents[i].getName().equals(fileName))
                    return flag=true;
                    }

            return flag;
        }
```

The below method takes two parameters, first takes the folder path and the file extension / mime type. it will return true if the file with the specific extension is available in the specified folder.

```
/* Check the file from a specific directory with extension */
        private boolean isFileDownloaded_Ext(String dirPath, String ext){
                boolean flag=false;
            File dir = new File(dirPath);
            File[] files = dir.listFiles();
            if (files == null || files.length == 0) {
                flag = false;
            }

            for (int i = 1; i < files.length; i++) {
                if(files[i].getName().contains(ext)) {
                        flag=true;
                }
            }
            return flag;
        }
```

The below method is used to get the document name based on the last action performed in the specified folder. This is done by using java LastModified which returns a long value representing the time the file was last modified.

```
/* Get the latest file from a specific directory*/
        private File getLatestFilefromDir(String dirPath){
            File dir = new File(dirPath);
            File[] files = dir.listFiles();
            if (files == null || files.length == 0) {
                return null;
            }

            File lastModifiedFile = files[0];
            for (int i = 1; i < files.length; i++) {
               if (lastModifiedFile.lastModified() < files[i].lastModified()) {
                   lastModifiedFile = files[i];
               }
            }
            return lastModifiedFile;
        }
```

When ever we click on download, based on the file size and network we need to wait for specific to complete download operation. If not we may encounter issues as the file is not downloaded.

We can also make use of 'Java Watch Service API' which monitors the changes in the directory. Note: This is compatible with Java 7 version. Below is the example program using Watch Service API. And here we will user only for 'ENTRY_CREATE' event.

```
        public static String getDownloadedDocumentName(String downloadDir, String
fileExtension)
        {
                String downloadedFileName = null;
                boolean valid = true;
                boolean found = false;

                //default timeout in seconds
                long timeOut = 20;
                try
                {
                        Path downloadFolderPath = Paths.get(downloadDir);
                        WatchService watchService = FileSystems.getDefault().newW
atchService();
                        downloadFolderPath.register(watchService, StandardWatchEv
entKinds.ENTRY_CREATE);
                        long startTime = System.currentTimeMillis();
                        do
                        {
                                WatchKey watchKey;
                                watchKey = watchService.poll(timeOut,TimeUnit.SEC
ONDS);
                                long currentTime = (System.currentTimeMillis()-st
artTime)/1000;
                                if(currentTime>timeOut)
                                {
                                        System.out.println("Download operation ti
```

‹ Resizing a web element using movebyoffset How to verify entries in Exported CSV file ›

Tags:

File Download

Selenium Tutorials:

Selenium Tutorials

Selenium Tutorials

- Working with Frames in Selenium Webdriver
- How to handle javascript alerts, confirmation and prompts?
- Defining Firefox Profile preferences using Selenium Webdriver
- Why Automation Testing is required?
- Importance of Automation Testing
- Mouse Hover Actions in Selenium Webdriver
- StaleElementReferenceExceptions in Selenium Webdriver
- Difference between Webdriver get() and Webdriver navigate()
- Navigation Methods in Webdriver with Examples
- Handling Cookies in Webdriver
- How to addcookie with Selenium Webdriver
- How to delete Cookies in Selenium Webdriver
- WebDriver Methods
- Automation Test Frameworks
- Taking Screenshot using Webdriver
- Webdriver SELECT Methods to work with Dropdowns
- Webdriver Select with Multiple Attribute
- Handling keyboard events and mouse hover events using Webdriver
- Set browser width and height in Selenium Webdriver
- Selenium Automation Framework Example
- UI Map or Object Repository using Properties File
- Test Data in automation framework
- Reporting Structure in Selenium Automation Framework
- Test Configuration File / Executable File
- Uploading a file with Selenium Webdriver
- Upload file using AutoIT
- Uploading a file with sendKeys method
- Read data from Properties file using Java Selenium
- Scrolling web page with Selenium Webdriver using java
- Keyword Driven Framework Example
- Install AutoIt in Windows
- Examples with xpath and Css (ID, Name, Text and Links)
- Testing in multiple browsers using selenium and testng
- Working with Checkbox using Resuable Methods
- Working with Select examples

- File upload using Robots
- Download file using selenium webdriver
- Perform operations on new window
- Introduction to Page Object Model Framework
- Mouse hover using javaScript Executor
- Working with Multiple Checkboxes with Safecheck
- Click element using JavaScriptExecutor
- Drag and Drop using Webdriver Action Class
- WebDriver Waits Examples
- Simple Page Object Model example
- Double Click on element using Webdriver
- Right Click Context Menu example
- Working with AutoComplete Text box
- Working with Date Picker Calendar
- Working with Ajax controls using Webdriver
- Get Attribute values using Webdriver
- Get Css values using Webdriver
- Resizing a web element using movebyoffset
- Validate Downloaded file after clicking on downloaded button/ link
- How to verify entries in Exported CSV file
- Element is not clickable at point SeleniumWebdriverException
- Extract PDF text And Verify Text Present in PDF using WebDriver
- Handling Authentication Window with WebDriver (In Firefox, Chrome and IE)

Recent Post

- Handling Authentication Window with WebDriver (In Firefox, Chrome and IE)
- Selenium Webdriver Tutorials
- Locators for Selenium
- Optional annotation in TestNG
- Parameterization in TestNG using testng.xml

Automation Framework

- Framework Structure
- Object Repository
- Test Data
- Test Scripts
- Reports
- Configuration and Execution

Tags Cloud

Asserts (1) AutoComplete (1) AutoIt (2) Automation (3) Build.xml (1) Checkboxes (2) Cookies (3) Css (2) Css Examples (2) DataProvider (1) Date Picker (1) Drag and Drop (1) Exceptions (4) File Download (3)

Firefox Profile (2) Framework Frameworks (5) iFrames (1) Interview Questions (1) JavascriptExecutor (3) JUnit (2) Keyword Driven (1) Listeners (3) Manual Testing (1) Maven (5) Mouse Hover (2) Page Object Model (2) Page Scroll (1) Parallel Test (3) properties file (1)

Home

- Home
- Tutorials
- About
- Contact
- Privacy Policy
- Site Map

Tutorial Menu

- Selenium Tutorials
- TestNG Tutorial
- JUnit Tutorial
- JXL Tutorial
- Apache POI
- ANT Tutorial
- Maven Tutorial

Selenium Tutorials

- Basic Webdriver Tutorials
- Advanced Webdriver Tutorials
- Selenium Framework Examples
- JavascriptExecutor Examples

Interview Questions

- Testing Concepts
- Quality Standards
- General Questions
- Roles & Responsibilities

- Technology & Environment

We welcome your questions and requests