FREE eBook

LEARNING selenium

Free unaffiliated eBook created from
Stack Overflow contributors.

#selenium

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: selenium

It is an unofficial and free selenium ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official selenium.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with selenium

Remarks

Selenium is a powerful library of commands in multiple languages (C#, Haskell, Java, JavaScript, Objective-C, Perl, PHP, Python, R, and Ruby) that allow a programmer to automate browser interaction. This is incredibly useful for developers testing applications.

Selenium offers methods to:

- Find an Element in a webpage
- Click on an Element
- Send a String to an Element
- Navigate to a web page
- Change to a different tab in the same browser window
- Take a screenshot of a webpage

Using these methods, a developer can have automatic tests checking:

- If an Element is in a page, and visible to a user
- A search or login form
- Buttons or interactive Elements
- Check the values or attributes of an Element

Selenium runs in webdrivers, which are similar to a normal web browser but allow Selenium to interact with them. A Selenium test typically opens up a new driver instance of whatever browser the developer is testing in, which is always a clean slate. This way, when running a Selenium test, the developer does not have to worry about previous cookies, or a browser cache affecting the results of their application.

Selenium also works when running a webdriver in headless mode.

Versions

Version	Release date
3.4.0	2017-04-11
3.3	2017-04-07
3.2	2017-02-27
3.1	2017-02-13
3.0.1	2016-11-19
3.0	2016-10-11

Examples

Simple Selenium test in Java

Below code is simple java program using selenium. The journey of the below code is

1. Open Firefox browser
2. Open google page
3. Print title of Google page
4. Find the search box location
5. Pass the value as Selenium in the search box
6. Submit the form
7. Shutdown the browser

```
package org.openqa.selenium.example;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import java.util.concurrent.TimeUnit;

public class Selenium2Example  {
    public static void main(String[] args) {
        // Create a new instance of the Firefox driver
        WebDriver driver = new FirefoxDriver();

        // An implicit wait is to tell WebDriver to poll the DOM for a certain amount of time
        // when trying to find an element or elements if they are not immediately available.
        // The default setting is 0. Once set, the implicit wait is set for the life of the
WebDriver object instance.
        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

        // Maximize the browser window to fit into screen
        driver.manage().window().maximize();

        // Visit Google
        driver.get("http://www.google.com");

        // Check the title of the page
        System.out.println("Page title is: " + driver.getTitle());

        // Find the text input element by its name
        WebElement element = driver.findElement(By.name("q"));

        // Enter something to search for
        element.sendKeys("Selenium!");

        // Now submit the form. WebDriver will find the form for us from the element
        element.submit();

        //Close the browser
        driver.quit();
    }
}
```

Simple selenium test in python

```
from selenium import webdriver

# Create a new chromedriver
driver = webdriver.Chrome()

# Go to www.google.com
driver.get("https://www.google.com")

# Get the webelement of the text input box
search_box = driver.find_element_by_name("q")

# Send the string "Selenium!" to the input box
seach_box.send_keys("Selenium!")

# Submit the input, which starts a search
search_box.submit()

# Wait to see the results of the search
time.sleep(5)

# Close the driver
driver.quit()
```

Setting up python Selenium via terminal (BASH)

The easiest way is to use pip and VirtualEnv. Selenium also requires python 3.*.

Install virtualenv using:

```
$: pip install virtualenv
```

Create/enter a directory for your Selenium files:

```
$: cd my_selenium_project
```

Create a new VirtualEnv in the directory for your Selenium files:

```
$: virtualenv -p /usr/bin/python3.0 venv
```

Activate the VirtualEnv:

```
$: source venv/bin/active
```

You should see now see (venv) at the beginning of each bash line. Install Selenium using pip:

```
$: pip install selenium
```

Selenium comes with the FireFox driver by default.
If you want to run Selenium in google chrome, also do this:

```
$: pip install chromedriver
```

You now have a version-controlled VirtualEnv. To make sure everything is set up correctly:

Start python:

```
$: python
```

Prints out:

```
Python 2.7.10 (default, Jul 14 2015, 19:46:27)
[GCC 4.2.1 Compatible Apple LLVM 6.0 (clang-600.0.39)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
```

Create a new webdriver (in this case, a chromedriver), and go to www.google.com:

```
>>> from selenium import webdriver
>>> driver = webdriver.Chrome()
>>> driver.get("https://www.google.com")
```

Close the driver and python interpreter:

```
>>> driver.quit()
>>> quit()
```

Deactivate the VirtualEnv:

```
$: deactivate
```

If the line `driver = webdriver.Chrome()` is throwing errors:

- Make sure you also have the chrome browser installed. If you don't, the Selenium chromedriver can not access the Chrome binary.
- webdriver.Chrome() can also take a parameter for your chromedriver location. If you installed it using pip, try (on mac) `driver = webdriver.Chrome("./venv/selenium/webdriver/chromedriver")`.

Simple Selenium Example in C#

```
//Create a new ChromeDriver
IWebDriver driver = new ChromeDriver();

//Navigate to www.google.com
driver.Navigate().GoToUrl("https://www.google.com");

//Find the WebElement of the search bar
IWebElement element = driver.FindElement(By.Name("q"));

//Type Hello World into search bar
element.SendKeys("Hello World");
```

```
//Submit the input
element.Submit();

//Close the browser
driver.Quit();
```

Read Getting started with selenium online: https://riptutorial.com/selenium/topic/1840/getting-started-with-selenium

Chapter 2: Accepting popup alerts with Selenium

Examples

Python example of Accepting alert

```
from selenium import webdriver

# Create a new webdriver
driver = webdriver.Chrome()
# Get a page that has a popup window (Use mouse to click "try it" button)
driver.get("http://www.w3schools.com/js/tryit.asp?filename=tryjs_alert")
# Accept the opened alert
driver.switch_to.alert.accept()
```

C# Extensions to WebDriver

```
public static IWebDriver dismissAlert(this IWebDriver driver)
{
    try
    {
        IAlert alert = driver.SwitchTo().Alert();
        alert.Dismiss();
    }
    catch {}
    return driver;
}
public static IWebDriver acceptAlert(this IWebDriver driver)
{
    try
    {
        IAlert alert = driver.SwitchTo().Alert();
        alert.Accept();
    }
    catch { }
    return driver;
}
```

How to use:

```
driver.acceptAlert();
driver.dismissAlert();
```

Java

For simple alert:

```
 Alert simpleAlert = driver.switchTo().alert();
```

```
String alertText = simpleAlert.getText();
System.out.println("Alert text is " + alertText);
simpleAlert.accept();
```

For Prompt alert:

```
Alert promptAlert  = driver.switchTo().alert();
String alertText = promptAlert.getText();
System.out.println("Alert text is " + alertText);
//Send some text to the alert
promptAlert.sendKeys("Accepting the alert");
Thread.sleep(4000); //This sleep is not necessary, just for demonstration
promptAlert.accept();
```

For Confirmation alert:

```
Alert confirmationAlert = driver.switchTo().alert();
String alertText = confirmationAlert.getText();
System.out.println("Alert text is " + alertText);
confirmationAlert.accept();
```

Another way you can do this, is wrap your code inside a try-catch:

```
try{
   // Your logic here.
} catch(UnhandledAlertException e){
  Alert alert = driver.switchTo().alert();
  alert.accept();
}

// Continue.
```

Read Accepting popup alerts with Selenium online:
https://riptutorial.com/selenium/topic/6787/accepting-popup-alerts-with-selenium

Chapter 3: First project in selenium with Java

Introduction

This is an introduction to Selenium, using Java. While we don't expect you to know anything regarding Selenium, you have to have prior Java knowledge to follow this course.

Download Links :

Selenium

IntelliJ IDEA

ChromeDriver

JDK 8

Examples

Setting up IntelliJ Idea for Selenium

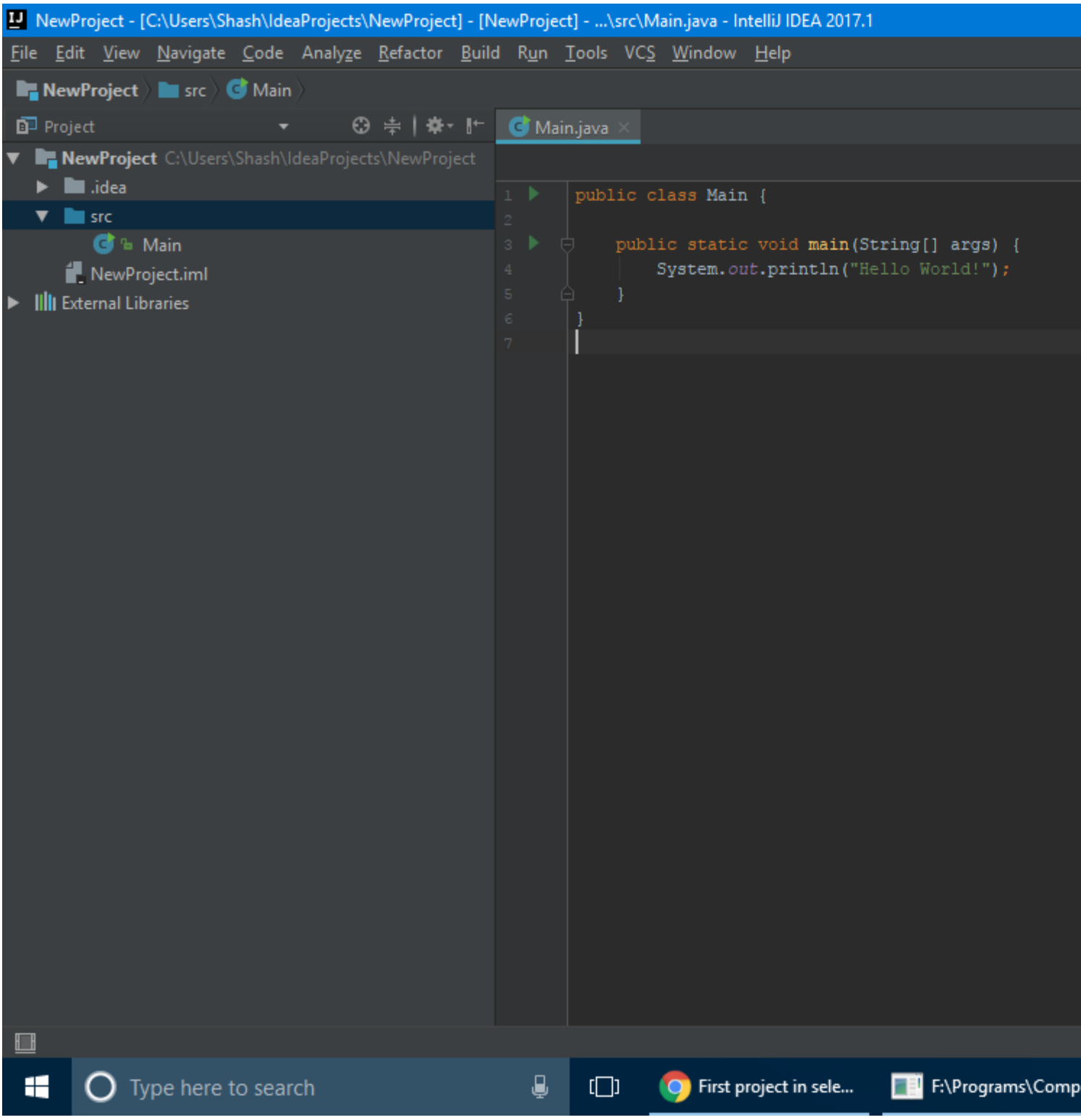
Prerequisites:

1. Java is installed
2. Selenium is extracted in a folder (Contains 2 files, and 1 folder)

Follow these steps to set up IntelliJ Idea for Selenium.

1. Click On **"New Project"**.
2. Choose Java < "Hello World" Application
3. Type the name of the Project, and create it.

Your Screen should look something like this

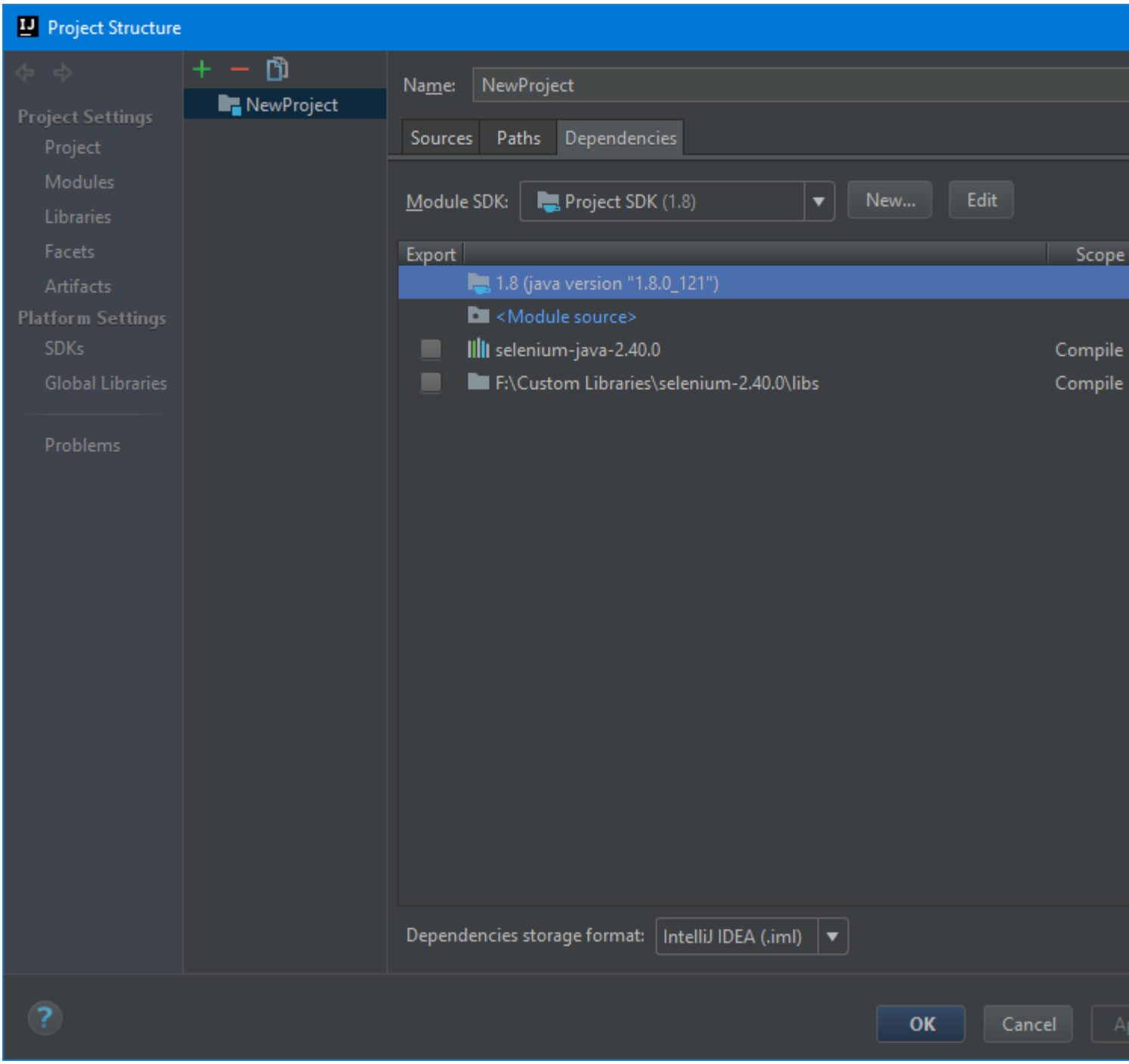


Now, go to

```
File < Project Structure < Modules < Dependencies
```

There, click on the green plus(+) icon, and choose Library. Then navigate to the extracted Selenium folder, and add "*selenium-java 2.4.0.jar*". After adding this, click on the green plus(+) icon again, and now choose *"Directory".* This time, locate the *libs* folder of Selenium, and click on ok, while selecting it.

At the end, your Project Structure should look like this



Now, click on OK, and you're all set.

Setting up ChromeDriver

Prerequisites : ChromeDriver is downloaded

Copy the following code into your class.

```
public static void main(String[] args) {
    System.setProperty("webdriver.chrome.driver", "path of the exe file\\chromedriver.exe");
}
```

If you're using linux, give the path to the ChromeDriver Binary.

Opening up a Website using Selenium

We use the `get` method to go to a website. For Example, this would open google

```
public static void main(String[] args) throws InterruptedException {
    System.setProperty("webdriver.chrome.driver", "path of the exe file\\chromedriver.exe");
    WebDriver driver = new ChromeDriver();
    driver.get("https:www.google.com");
    Thread.sleep(3000); //wait for 3 seconds
    driver.quit();      //close Chrome
}
```

`driver.quit()` closes the Browser. To create a delay, we use `Thread.sleep(3000)`.

Getting Elements in Selenium

Every Html-Element in Selenium is called a `WebElement`. For example, a `p` tag would be a `WebElement`, an `a` tag would be a `WebElement`, etc. Consider the following html Structure:

```
<a id="link1" href="https://www.google.com">google</a>
<p class="p1">
This is a paragraph
</p>
```

Now, if we wanted to get the `a` tag, we could do

```
WebElement link = driver.findElement(By.id("link1"));
```

Now, we can click on this, by

```
link.click();
```

Lets take another example. If we wanted the text of the `p` tag, *ie*, "*This is a paragraph*", we can do

```
WebElement p = driver.findElement(By.className("p1"));
System.out.println(p.getText());
```

We can also get Elements by tags, like

```
WebElement tag = driver.findElement(By.tagName("a"));
```

Working Example in Selenium

Now that we know the basics of Selenium, we can make our own project. For this example, we'll be making a program, which finds the Newest questions on stack-overflow.

We start easy, lets open stack-overflow.

```
public static void main(String[] args) throws InterruptedException {
```

```
    System.setProperty("webdriver.chrome.driver", "path of the exe file\\chromedriver.exe");
    WebDriver driver = new ChromeDriver();
    driver.get("https:stackoverflow.com");
    Thread.sleep(3000);
    driver.quit();
}
```

Now, if you look at the source of the page, you find that all questions, are `a` tags, with an className of `question-hyperlink`. However, since there are multiple questions, we use a `List` of `WebElement`, instead of `WebElement`. Thus, we can do

```
public static void main(String[] args) throws InterruptedException {
    System.setProperty("webdriver.chrome.driver", "path to chromedriver\\chromedriver.exe");
    WebDriver driver = new ChromeDriver();
    driver.get("https:stackoverflow.com");
    List<WebElement> list = driver.findElements(By.className("question-hyperlink"));
}
```

Now, we need to get the `href` attribute of the `a` tag, which has the link of the question. To do this, we can use `getAttribute("href")` on the each `WebElement`, like

```
public static void main(String[] args) throws InterruptedException {
    System.setProperty("webdriver.chrome.driver", "path to chromedriver\\chromedriver.exe");
    WebDriver driver = new ChromeDriver();
    driver.get("https:stackoverflow.com");
    List<WebElement> list = driver.findElements(By.className("question-hyperlink"));
    System.out.println(list.size());
    list.forEach(e->System.out.println(e.getAttribute("href")));
    driver.quit();
}
```

This prints out the links of the top-questions on Stack-overflow.

Getting Attributes of WebElements in Selenium

To get the attribute of a `WebElement`, we use `getAttribute` on that `WebElement`. For example, consider the following html tag

```
<a id="click" href="https://www.google.com">
```

We can find the Element's `href` attribute by

```
WebElement e = driver.findElement(By.id("click"));
System.out.println(e.getAttribute("href")); //prints https://www.google.com
```

Read First project in selenium with Java online: https://riptutorial.com/selenium/topic/10012/first-project-in-selenium-with-java

Chapter 4: Getting started with Selenium in python

Remarks

What is Selenium?

Selenium is a library of commands to help a programmer interface with a browser like a real user.

Things that Selenium does:

Finding element(s) in a webpage's html

- Finds a single element:
    - `driver.find_element_by_css_selector("css.selector.of.element")` CSS Selector help
    - `driver.find_element_by_xpath("//xpath//of//element")` XPATH help
    - `driver.find_element_by_name("name_of_element")`
    - `driver.find_element_by_id("id_of_element")`
    - `driver.find_element_by_partial_link_text("element_link_text")`
    - `driver.find_element_by_class_name("class_name_of_element")`
    - `driver.find_element_by_tag_name("tag_name_of_element")`
- Finds a list of elements:
    - `driver.find_elements_by_css_selector("css.selector.of.elements")`
    - `driver.find_elements_by_xpath("//xpath//of//elements")`
    - `driver.find_elements_by_name("name_of_elements")`
    - `driver.find_elements_by_partial_link_text("elements_link_text")`
    - `driver.find_elements_by_class_name("class_name_of_elements")`
    - `driver.find_elements_by_tag_name("tag_name_of_elements")`
- Official documentation: selenium-python read the docs

Interact with elements:

"method" represents any of the above methods to find an element or list of elements.

- click function:
    - `driver.find_element_by_method.click()`
- send_keys function:
    - `driver.find_element_by_method.send_keys("text")` sends the String "text" to the element found.
    - `driver.find_element_by_method.send_keys(KeyCode.UP)` sends the KeyCode for the up arrow key to the element found.

Examples

Basic python Selenium

```
from selenium import webdriver

driver = webdriver.Chrome()  # Creates a new chromedriver instance
driver.get("https://www.python.org/")  # Go to https://www.python.org/
# Sends the text "python" to the text search box
driver.find_element_by_id("id-search-field").send_keys("python")
# Click on the search button
driver.find_element_by_css_selector("button[type=\"submit\"]").click()
```

Basic Selenium testcase

This is a basic example of a Selenium testcase using the python Unittest library

```
from selenium import webdriver
import unittest

class SeleniumTest(Unittest.testcase):

    def setUp(self):
        self.driver = webdriver.Chrome()
        self.driver.implicitly_wait(30)

    def test(self):
        self.driver.get("https//www.google.com")
        self.driver.find_element_by_id("lst-ib").send_keys("python")
        self.driver.find_element_by_css_selector("span[class=\"sbico\"]").click()

    def tearDown(self):
        self.driver.quit()
```

Read Getting started with Selenium in python online:
https://riptutorial.com/selenium/topic/6786/getting-started-with-selenium-in-python

Chapter 5: Mobile app automation

Examples

Android + Chrome + Python

To be able to run tests `Chrome` broser should be pre-installed on `Android` device,

Python + Chrome + Android

To be able to work with web-application on `Android` device using `Selenium` below pre-conditions should be met:

- `Android SDK` installed on computer
- `Chrome` browser installed on `Android` device
- Debugging mode enabled on `Android` device

Start `adb` and `chromedriver` server with below commands from `cmd`/`Terminal`:

```
adb start-server
chromedriver
```

Note down `chromedriver` server port number from log that looks like

> Starting ChromeDriver 2.15.322448 (52179c1b310fec1797c81ea9a20326839860b7d3) on port **9515**

Connect `Android` device to computer with `USB` cable

Below is simple `Python` code to get `Google` page:

```
from selenium import webdriver

capabilities = {
 'chromeOptions': {
   'androidPackage': 'com.android.chrome',
                   }
                }
driver = webdriver.Remote('http://localhost:9515', capabilities) # Specify your port number
value
driver.get('http://google.com')
driver.quit()
```

Read Mobile app automation online: https://riptutorial.com/selenium/topic/9449/mobile-app-automation

Chapter 6: Selenium IDE

Examples

Try a simple Selenium script: Search Wikipedia on Google

Prerequisites:

- Install Firefox
- Install Selenium IDE addon (https://addons.mozilla.org/fr/firefox/addon/selenium-ide/)

Open the plugin. A button displaying a red circle must be shown. If it's pressed, it means you can start your scenario. The plugin is recording everything you do **within this Firefox instance**.
Do whatever you want to be recorded.
In the end, save your scenario; you will notice that Selenium IDE's scenarios are html files.
You can also open files from other users. For instance, copy and paste the code below in a new html file and import it via your plugin. You will be able to run the -very simple- scenario.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head profile="http://selenium-ide.openqa.org/profiles/test-case">
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<link rel="selenium.base" href="https://www.google.com/" />
<title>sample-test</title>
</head>
<body>
<table cellpadding="1" cellspacing="1" border="1">
<thead>
<tr><td rowspan="1" colspan="3">sample-test</td></tr>
</thead><tbody>
<tr>
    <td>open</td>
    <td>/</td>
    <td></td>
</tr>
<tr>
    <td>type</td>
    <td>id=lst-ib</td>
    <td>Wikipedia</td>
</tr>
</tbody></table>
</body>
</html>
```

This DSL (domain specific language) is commonly named "selenese".
Its most common functions are listed here.

Read Selenium IDE online: https://riptutorial.com/selenium/topic/7694/selenium-ide

Chapter 7: Selenium simple example C# and Nunit

Remarks

This is a very basic example or starting Selenium, accessing and using a page and then shutting down Selenium within NUnit.

Examples

Simple Selenium-NUnit

Prereqs:

- Selenium and required Browser Drivers are installed (Available in Nuget)
- NUnit has been installed in VS and added to the project

```
using NUnit.Framework;
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;
using OpenQA.Selenium.Firefox;
using OpenQA.Selenium.IE;
using System;
[TestFixture]
public class GoToGoogle
{
    //The WebDriver object
    IWebDriver driver;
    //Ran before test cases
    [TestFixtureSetUp]
    public void setup()
    {
        //Initialize the webdriver
        //An example of IE
        driver = new InternetExplorerDriver();
        //Firefox Example
        //driver = new FirefoxDriver();
        //An example of Chrome
        //driver = new ChromeDriver();

        //Wait x seconds to find the element and then fail, x = 5 here
        driver.Manage().Timeouts().ImplicitlyWait(TimeSpan.FromSeconds(5));
    }
    //Ran after the test case has completed
    [TestFixtureTearDown]
    public void tearDown()
    {
        driver.Quit();
    }
    [Test]
    public void gotoGoogle()
    {
```

```
        //going to google.com
        driver.Navigate().GoToUrl("www.google.com");
        //Assert we are on google.com
        Assert.AreEqual(driver.Title, "Google");
        //Getting the search field
        IWebElement searchField = driver.FindElement(By.Name("q"));
        //Typing in the search field
        searchField.SendKeys("Selenium Tutorial");
        //Submitting
        searchField.Submit();

    }


}
```

Read Selenium simple example C# and Nunit online:
https://riptutorial.com/selenium/topic/6537/selenium-simple-example-csharp-and-nunit

Chapter 8: Selenium simple example C# and Nunit

Examples

Simple page load test and make sure the title of the page is correct

```
public class HomepageTests
{
    private IWebDriver _driver;

    [SetUp]
    public void LoadDriver()
    {
       _driver = new ChromeDriver();

    }

    [Test]
    public void Valid_Home_Page_Title()
    {
        _driver.Navigate().GoToUrl("Your homeoage url / local or remote");
        StringAssert.Contains("Expected title of your page", _driver.Title);

    }

    [TearDown]
    public void UnloadDriver()
    {
        _driver.Quit();
    }
}
```

Read Selenium simple example C# and Nunit online:
https://riptutorial.com/selenium/topic/6554/selenium-simple-example-csharp-and-nunit

Chapter 9: Take a screenshot of a webpage

Examples

Python Selenium take/save screenshot of webpage

```
from selenium import webdriver

# Create a new cromedriver
driver = webdriver.Chrome()
# Go to www.google.com
driver.get("https://www.google.com")
# Saves a .png file with name my_screenshot_name to the directory that
# you are running the program from.
screenshot_name = "my_screenshot_name.png"
driver.save_screenshot(screenshot_name)
```

driver.save_screenshot will return 'true' if the screenshot was taken, and 'false' if it was not. Saving screenshots also works with headless browsers. If you want to save a screenshot in a different directory, just add the filepath (relative to where you are running the code from). For example:

```
screenshot_name = "screenshots/my_screenshot_name.png"
```

Will save the screenshot in directory "screenshots" inside the directory that python is being run from.

C# TakeScreenshot extension

```
public static Screenshot TakeScreenshot(this IWebDriver _driver)
{
    return ((ITakesScreenshot)_driver).GetScreenshot();
}
```

Usage example:

```
driver.TakeScreenshot().SaveAsFile(@"/Test/Test.png",ImageFormat.Png);
```

Java Selenium take/save screenshot of webpage and add on report

```
public void Screenshot() throws Throwable{
    final byte[] screenshot = ((TakesScreenshot) driver).getScreenshotAs(OutputType.BYTES);
    scenario.embed(screenshot, "image/png"); // ... and embed it in the report.
    Thread.sleep(1000);
}
```

Alternately

```
 public static void captureScreenShot(WebDriver ldriver){

  // Take screenshot and store as a file format
  File src= ((TakesScreenshot)ldriver).getScreenshotAs(OutputType.FILE);
try {
  // now copy the  screenshot to desired location using copyFile method

 FileUtils.copyFile(src, new File("C:/selenium/"+System.currentTimeMillis()+".png"));
   }

catch (IOException e)

{
System.out.println(e.getMessage());
  }
}
```

Read Take a screenshot of a webpage online: https://riptutorial.com/selenium/topic/6789/take-a-screenshot-of-a-webpage

Chapter 10: Waiting in Selenium

Introduction

One of the most common stumbling blocks for newer Selenium users is waiting until a page is fully loaded. Human users can easily tell if a page has fully loaded or if it is still loading. Selenium, however, just waits for a set amount of time. Therefore, it is often convenient to have a good way to wait for elements in a page. While it is *possible* to do this with a loop and sleep() functions, there are much cleaner ways already built into Selenium.

Examples

Explicit Wait in Python

When browser navigates to a dynamic page (most commonly AJAX-based web application), the elements on the page can take different time to load, and furthermore: some elements will only load in response to some user actions. In such cases direct retrieval of an element may fail:

```
# Don't do this: may fail on dynamic page with ElementNotVisibleException
element = driver.find_element_by_name('q')
```

The most obvious solution it seems to introduce the wait before retrieving elements:

```
# Don't do this: inefficient solution for ElementNotVisibleException
time.sleep(5) # delays for 5 seconds
element = driver.find_element_by_name('q')
```

But such solution is inefficient, since it causes the test to always wait for 5 seconds, even when the element in most cases appears after 1 second (and only sometimes requires up to 5 seconds). It doesn't look much if it's one place, but usually each test deals with multiple elements, and there are multiple tests, which adds up to overall test duration.

A better solution is to wait for element to appear for up to 5 seconds, but return from the wait as soon as element is found. `WebDriverWait` allows you to do just that.

The following example navigates to www.google.com, waits (for up to 5 seconds) for the search bar to load, and then searches for "selenium".

```
from selenium import webdriver
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By

# Create a new chromedriver instance
driver = webdriver.Chrome()
```

```
# Go to www.google.com
driver.get("https://www.google.com")

try:
    # Wait as long as required, or maximum of 5 sec for element to appear
    # If successful, retrieves the element
    element = WebDriverWait(driver,5).until(
         EC.presence_of_element_located((By.NAME, "q")))

    # Type "selenium"
    element.send_keys("selenium")

    #Type Enter
    element.send_keys(Keys.ENTER)

except TimeoutException:
    print("Failed to load search bar at www.google.com")
finally:
    driver.quit()
```

Wait in Java with selenium

Explicit wait : Wait for a certain condition to occur before proceeding further in the code.

```
WebDriver driver = new FirefoxDriver();
driver.get("http://google.com");
WebElement myElement = (new WebDriverWait(driver, 10))
  .until(ExpectedConditions.presenceOfElementLocated(By.id("myElement")));
```

Implicit wait: Wait for a certain amount of time when trying to find an element or elements if they are not immediately available.

```
WebDriver driver = new FirefoxDriver();
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
driver.get("http://google.com");
WebElement myElement = driver.findElement(By.id("myElement"));
```

Read Waiting in Selenium online: https://riptutorial.com/selenium/topic/6901/waiting-in-selenium

Chapter 11: WebDriver Factory

Examples

WebDriver Factory C#

```
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;
using OpenQA.Selenium.Firefox;
using OpenQA.Selenium.IE;


/// <summary>
/// Factory for creating WebDriver for various browsers.
/// </summary>
public static class WebDriverFactory
{
    /// <summary>
    /// Initilizes IWebDriver base on the given WebBrowser name.
    /// </summary>
    /// <param name="name"></param>
    /// <returns></returns>
    public static IWebDriver CreateWebDriver(WebBrowser name)
    {
        switch (name)
        {
            case WebBrowser.Firefox:
                return new FirefoxDriver();
            case WebBrowser.IE:
            case WebBrowser.InternetExplorer:
                InternetExplorerOptions ieOption = new InternetExplorerOptions();
                ieOption.IntroduceInstabilityByIgnoringProtectedModeSettings = true;
                ieOption.EnsureCleanSession = true;
                ieOption.RequireWindowFocus = true;
                return new InternetExplorerDriver(@"./", ieOption);
            case "safari":
                return new RemoteWebDriver(new Uri("http://mac-ip-address:the-opened-port"),
DesiredCapabilities.Safari());
            case WebBrowser.Chrome:
            default:
                ChromeOptions chromeOption = new ChromeOptions();
                string location = @"./";
                chromeOption.AddArguments("--disable-extensions");
                return new ChromeDriver(location, chromeOption);
        }
    }
}

public enum WebBrowser
{
    IE,
    InternetExplorer,
    Firefox,
    Chrome
}
```

```
// Usage
var driver = WebDriverFactory.CreateWebDriver(WebBrowser.Chrome);
```

Read WebDriver Factory online: https://riptutorial.com/selenium/topic/7727/webdriver-factory

Chapter 12: WebDriverManager for Selenium - a very neat tool from Boni Garcia

Introduction

I switched to Selenium 3 and started using Chrome instead of Firefox. I discovered that for Chrome I need to download a binary for WebDriver to handle the browser. For that to work I need to set absolute path to this binary as Java variable. If binary gets updated, I need to update that binary manually in my test framework - which takes time and is really annoying. I discovered a very neat Java library that does it for me: https://github.com/bonigarcia/webdrivermanager

Examples

Below examples shows how easy it is to use

```
ChromeDriverManager.getInstance().setup();
FirefoxDriverManager.getInstance().setup();
OperaDriverManager.getInstance().setup();
PhantomJsDriverManager.getInstance().setup();
EdgeDriverManager.getInstance().setup();
InternetExplorerDriverManager.getInstance().setup();
```

Read WebDriverManager for Selenium - a very neat tool from Boni Garcia online: https://riptutorial.com/selenium/topic/10582/webdrivermanager-for-selenium---a-very-neat-tool-from-boni-garcia

Credits

S. No	Chapters	Contributors
1	Getting started with selenium	akhilsk, Alex.K., Brydenr, Community, Eugene S, Priya, Pseudo Sudo, Ruslan López Carro, Thomas, Venkatesh Achanta
2	Accepting popup alerts with Selenium	Brydenr, Paul Muir, Priya, slackmart
3	First project in selenium with Java	Frank Underwood
4	Getting started with Selenium in python	Brydenr
5	Mobile app automation	Andersson
6	Selenium IDE	Y-B Cause
7	Selenium simple example C# and Nunit	Paul Muir
8	Take a screenshot of a webpage	Brydenr, DuarteNGF, Paul Muir, Priya
9	Waiting in Selenium	Brydenr, John Fisher, Kiril S., Priya, Pseudo Sudo
10	WebDriver Factory	Buaban
11	WebDriverManager for Selenium - a very neat tool from Boni Garcia	sen4ik