Class acro/num:

Class name

Instructions to use selenium for Automation

Learning objectives:

* Learning what selenium is.
* Learning why selenium is a very useful tool.
* Learning the basic set up of selenium basic imports.
* Learn how to use selenium drivers.

**What Exactly is Selenium?**

Selenium is a very powerful tool for controlling web browsers through programs and performing browser automation. It is functional for all browsers, works on all major OS and its scripts are written in many different languages this includes: [Python](https://www.geeksforgeeks.org/python-programming-language/), [Java](https://www.geeksforgeeks.org/java/), [C#](https://www.geeksforgeeks.org/csharp-programming-language/), as well as many more, we will be working in our example with Python. This Selenium Tutorial covers topics some of the basics. Let’s get started!

**Prerequisite**:

* Install selenium type the following command into CMD: **(Skip this step if already installed)**

pip install -U selenium

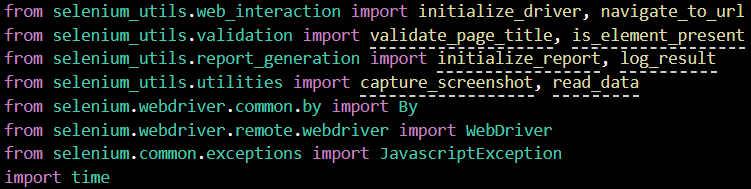
* To check if you have selenium installed type the following command into the command line:

python -c "import selenium; print(selenium.\_\_version\_\_)"

* Output example: 

**Setting Selenium project:**

Some important imports:



**Selenium Was to find elements in a Web Page:**

There are three ways to find elements is selenium the different types are the following.

1. By ID:
2. By CSS SELECTOR
3. XPATH

For more information please visit:

[Locator strategies | Selenium](https://www.selenium.dev/documentation/webdriver/elements/locators/)

You can also find these in and webpage by opening inspect and hovering over the element in the inspect screen. Then right click and go to copy and it give all the path ways to access the desired element.

**It should look like the following:**

A screenshot of a computer

Description automatically generated

**Creating a basic selenium script for Textbox:**

**Step 1: Creating drivers.**

**Selenium Web driver** is an open-source collection of APIs, which is used for testing web applications. The Web driver tool is used for automating web application testing to verify that a specific part of an application works as expected or not. It mainly supports browsers like Firefox, Chrome, Safari and Internet Explorer.

The first step is to instantiate driver onto browser for chrome:

The way to do this is create a variable:



**Step 2: Navigate to the chrome webpage to do this set your drivers get variable to the following.**

A black background with white text

Description automatically generated

The get method for the web driver allows it to navigate to your desired page in this example we used: <https://formy-project.herokuapp.com/keypress> this will open the webpage in the desired driver which is chrome for this example.

To have the page stay open for a while and have the driver function properly make sure you put time.sleep(4) and driver.close at the bottom of all code in this example:



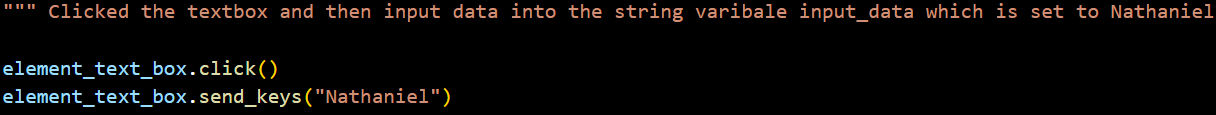
**Step 3:** **Find the text box element for this example we will be assigning the element by ID**

To manipulate an element on the webpage we must be able to access them one of the ways to access an element is by Id for us to do that we have to initialize a variable to the element we want to grab from the web page in this case a text box with the id of name to do this we write the following code:



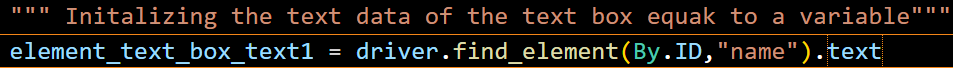
**Step 4: we will be performing Actions.**

Now we can manipulate this textbox next we have to activate the text box as if it was clicked by a user and then adding some input data as if we were a user, to do this we have activate the textbox variable with the click() method and then use the send\_keys() method to what we want the text box to display. Let’s use our name to do this. Write the following code.

****

**Step 5: Creating another variable for textbox data.**

In this step we will create another variable for setting the text data of the text box equal to an element to do this we write the following:

****

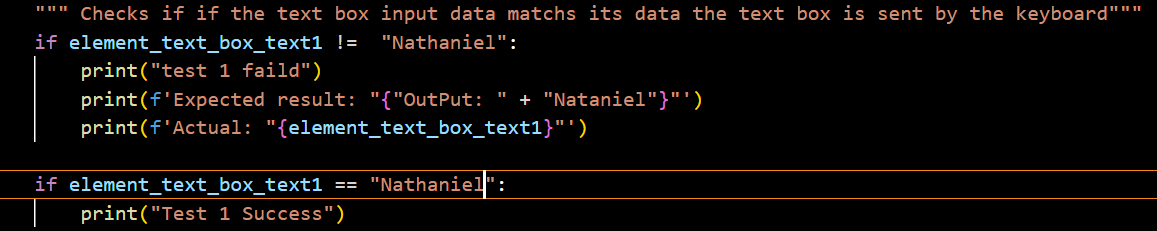
Now we can manipulate the data of text box as well as what the text box displays without typing into it.

**Step 6: Changing the textbox data to our Name.**

In this step we need to change the textbox text data using the .text method to do this write the following code.

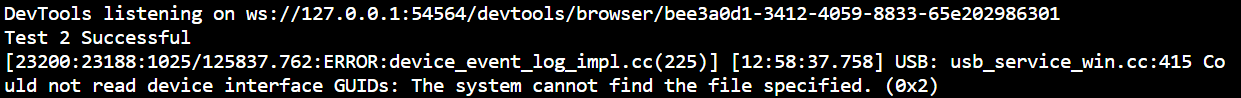


Now that we have done the initial set up we will be using, we can create our test.

****

**Step 7: Run Test**

Once everything has been written, you can run the program to see if your program test will run and output in the terminal success or failed. You should look something like the following:



Do not worry about the Error if it shows in execution, it’s just a **warning** will not stop the test from completing.

**Creating a basic selenium script for Auto complete:**

Have you ever wanted to add auto complete a test script in selenium on a specific input text type? Well fear not because you can, and the best part is it’s not all to different from getting access to your elements text let me explain with a tutorial.

**Step 1: Setting up your test script:**

First let’s get our test script set up and ready to be used for this exercise.

I will be using the same “Formy” site.

A screenshot of a computer program

Description automatically generated

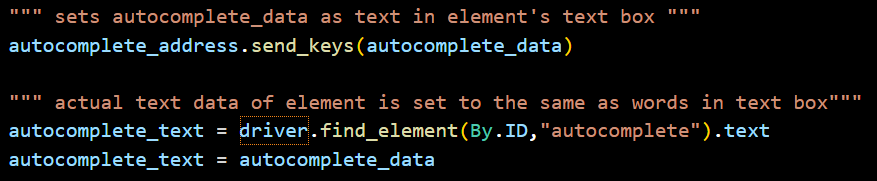
This is the beginning work we will need to start creating a driver and set the link to the website being used for the driver to link to the websites elements for set up.

Next, we need to set up a connection to the element from our web driver.

A computer screen with text

Description automatically generated

This enables us to connect to the element with the Id of name and display the text we want into the text box shown on the side, for example a neighborhood (1110, autumn spring dr) address. There is more work for us to manipulate the actual output of text from this element instead of what is displayed into the text box we will need to create a variable that can manipulate text with the following code.



This takes autocomplete\_data which equals (1110, autumn springs dr) and sets it to the text output of the variable autocomplete\_text where in this case connected to the element with the ID of “autocomplete”.text using are earlier created driver showed in code.

To ensure that the change has taken hold I will use a test case to compare and test if the autocomplete data is equal to the autocomplete\_text in the following:

A screen shot of a computer program

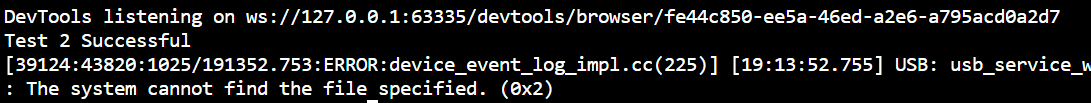
Description automatically generated

This validates that the change has taken place, if so we will see the test pass if not the test will fail the following is the output of this test as well as what it looks like on the website.

**Out Put:**

**A screenshot of a computer

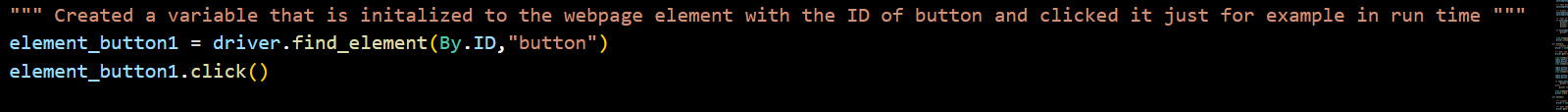
Description automatically generated**

****

Your output would look something like this do not worry about the error it’s just a warning.

**Creating a basic selenium script for Button:**

Toaccess a button button’s, click function is very similar to the textbox element for example. The follow is code that displays how to click a button:



**Creating a basic selenium script for Radio Button:**

Radio buttons are very common in web applications so being able to automate them in testing is very helpful. The way to access or select a radio button whenever you are on a web application is done by clicking it right. Well, that’s the same way to automate them using selenium.

**For example:**

Initial set up

**A black screen with white text

Description automatically generated**

**Code for getting the radio button element by XPATH and ID in the following:**

**A computer screen with text on it

Description automatically generated**

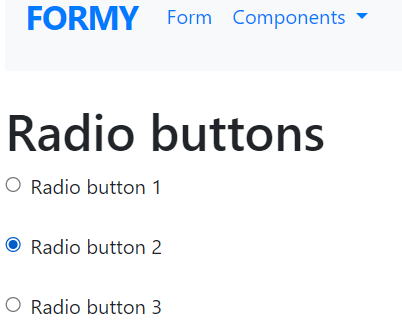
In this code you can see first there is a time.sleep(1) this makes the could pause for how long you want in seconds.

Next you can see the radio\_button\_2 is initialized to the location of the element on the page which is the second radio button on the page. Then it clicked be using the click() function.

The time.sleep(1) is halting the program (pausing the program at each on so you can see it run in execution.

radio\_ button1 and radio\_button3 are variables that have also been initialized to a radio button on the Formy website and the .click is used to perform the click action on the web page.

When ran one of its first iterations looks like the following:



A screenshot of a computer

Description automatically generated

**Creating a basic selenium script for checkbox:**

Although checkboxes seem different there actually not the same way you set up interaction with radio buttons are exactly the same!

For example:

A screenshot of a computer

Description automatically generated

In the function test4

We create a driver and initialize variable to manipulate the action of clicking just like we did with radio buttons.

Then we added a test to test whether or not everything functioned correctly with adding an end time.sleep(4) to hold the wind open after complete execution of the program before calling the webdriver’s close method with driver.close()

The following is the explained code:

A black background with colorful text

Description automatically generated

The following code example will be provided as the following file name: **formy tests.py**

**Conclusion:**

In conclusion selenium is a very powerful tool for web automation it can perform action on webpages which makes it a great tool for development on website automation.

For more information on selenium:

[2. Getting Started — Selenium Python Bindings 2 documentation (selenium-python.readthedocs.io)](https://selenium-python.readthedocs.io/getting-started.html)