Selenium Refresher

Crash Course on Selenium

Locators

- Locators are how you actually select elements
- Types of locators:
  - The “easy” locators
    - Locate by name
    - Locate by id
    - Locate by class name
    - Locate by link text (specific to a tags)
    - Locate by partial link text
    - Locate by tag name (a, body, div, h1)
  - The “hard” locators
    - Xpath selectors
    - CSS selectors

findElement v. findElements

- driver.findElement(By.className(“someClass”));
  - What happens if two or more elements have the same class ‘someClass’?
  - It will only select the very first element that appears in our HTML document
- driver.findElements(By.className(“someClass”));
  - This will return a list containing all elements with the class “someClass”
  - List<WebElement> elements = …

XPath: Absolute Path v. Relative Path

- Absolute path starts from the parentmost element in the DOM and traverses all the way to the element that you want to select
  - Ex. /html/body/div[1]/div[2]/div[1]/div/div/div/div[2]/div/div[1]/form/div[1]/div[1]/input
  - Fragile approach, any small change to the positioning of different elements will result in you not being able to locate the input element that you want to select
- Relative path allows us to dive straight to the elements we are concerned with
  - This is where you use // at the beginning
  - Ex. //div[@id=‘passContainer’]/input

CSS Selectors v. XPath Selectors

- Technically, CSS Selectors are faster than XPath selectors
    - Order of magnitude faster (10x)
    - However, pretty negligible because of how fast both are
    - So, in reality, it doesn’t really matter which one you use
- XPath selectors
    - Selecting by the text of the element isn’t really possible with CSS selectors
        - //*[text()=‘some text’]
        - Or
        - //*[contains(text(), ‘ello wor’)]
    - Traversing from a child element up to the parent
        - //*[text()='Top Tutorials']/parent::div
        - We can even traverse from the child to a parent, then back down to a different child
        - //*[text()='Top Tutorials']/parent::div/a[1]

Selenium Waits

- We have 3 types of waits:
  - Implicit waits: Implicit waits set a timeout for ALL searches of a WebElement after the point in which the implicit wait was initially configured. It will wait up to a maximum of the specified duration before throwing a NoSuchElementException
  - Explicit waits: Allow us to explicitly wait for a certain element up to the time specified in our WebDriverWait object that we construct. If we exceed this maximum time, it throws a TimeoutException rather than NoSuchElementException as in the case of an implicit wait
  - Fluent wait
- The ones we really want to focus the most attention on would be implicit waits v. explicit waits