How to locate elements if the elements have the same name,id and attributes in same tags?

If you don't find even single attribute to be distinguished , then get its parent element , find all child elements and iterate all child elements using loop, find your desired child element by comparing to any attribute which indicate your item Or select your element by index.

**Execptions in selenium**

1. **ElementNotVisibleException**: In spite of the element being present in the DOM, it is not visible (can not be interactive). For example, elements defined in HTML with *type =”hidden”*
2. **ElementNotSelectableException**: An element is disabled (can not be clicked/selected) in spite of being present in the DOM
3. **NoSuchElementException**: Webdriver is not able to determine the elements during runtime, i.e., the *FindBy* method cannot find a particular component
4. **NoSuchFrameException**: Webdriver attempts to switch to an invalid frame, which is unavailable
5. **NoAlertPresentException**: Webdriver is trying to switch to an invalid alert, which is unavailable
6. **NoSuchWindowException**: Webdriver is trying to switch to an invalid window, which is unavailable
7. **StaleElementReferenceException**: The referenced element is no longer present on the DOM page (a reference to a component is now Stale). For example, the item belongs to a different frame than the current one or the user has navigated away to another page
8. **SessionNotFoundException**: Webdriver is acting immediately after ‘quitting’ the browser
9. **TimeoutException**: The command did not complete in the specified time. For example, the element didn’t display at the specified time. This is especially encountered when working with waits
10. **WebDriverException**: Webdriver is acting immediately after ‘closing’ the browser

JavascriptExecutor js = (JavascriptExecutor)driver;

Js.executeScript(script,arguments);

Js.executeScript(“arguments[0].click()”,button)

Js.executeScript(“alert(‘welcom’))

ScrollDown

Js.executeScript(“windows.scrollBy(0,600));

***@Factory***: *The use of the factory annotation is when the tester needs to execute the test methods multiple times, which are present in the same class.*

***@Dataprovider***: *The dataprovider annotation enables the tester to run a test method multiple times using a different set of data provided by the dataprovider*.

Disable a test @Test(enabled = false)

*@Test (expectedException = numberFormatException.class)*

The invocationCount is an attribute that defines the number of times a test method has to run in a single execution. So, if the invocationCount sets as 5, then the test method would run five times each time I execute the TestNG test case.

Syntax for invocationCount:

*@Test (invocationCount = 5)*

In TestNG, "***timeout***" is a parameter that defines the maximum time that a method can take for execution.

*@Test (timeout = 1000)*

@Test(dependsOnGroups = { "SignIn" })

@Test (priority = 1)

* *@BeforeSuite*
* *@BeforeTest*
* *@BeforeClass*
* *@BeforeMethod*
* *@Test*
* *@AfterMethod*
* *@AfterClass*
* *@AfterTest*
* *@AfterSuite*

Xpaths:

//input[contains(@id =”name”)]

//input[starts-with(@id=”na”)]

//tagname[text(),”interview’]

//label[text()="Full Name"]/ancestor::form