Sreenidhi Rajakrishnan
Automation Architect
@sreenidhirajakrishnan

Different Type of Exceptions In Selenium

Name of the Exception	Why It Occured	How to Handle IT	Code to Handle the Exception

• NoSuchElementException	Attempting to interact with a non-existent web element or using an incorrect locator.	Use try-catch blocks, implement Implicit Waits, or use Explicit Waits to wait for element visibility.	`try { WebElement searchBox = driver.findElement(By.id("non-existent-id")); }`
			
			`catch (WebDriverException e) { System.out.println("Element not found: " + e.getMessage()); }`

• StaleElementReferenceException	Attempting to interact with an element that has been invalidated, often due to a page refresh.	Re-locate the element after the page change or use retry logic.	`driver.navigate().refresh();`
			
			`try { searchBox.sendKeys("Selenium WebDriver"); }`
			
			`catch (WebDriverException e) { System.out.println("Stale element reference: " + e.getMessage()); }`

• ElementNotInteractableException	Attempting to interact with a web element that is hidden or disabled.	Use a try-catch block to log the error and ensure the element is in an interactable state.	`try { WebElement hiddenElement = driver.findElement(By.cssSelector("input[aria-hidden='true']")); hiddenElement.click(); }`
			
			`catch (WebDriverException e) { System.out.println("Element not interactable: " + e.getMessage()); }`

• NoAlertPresentException	Attempting to switch to an alert that is not present on the web page.	Handle with a try-catch block to catch WebDriverException when switching to the alert.	`try { Alert alert = driver.switchTo().alert(); alert.accept(); }`
			
			`catch (WebDriverException e) { System.out.println("No alert present: " + e.getMessage()); }`

• InvalidSelectorException	Using an invalid CSS or XPath selector to locate an element.	Verify the syntax of the selector and handle the error using a try-catch block.	`try { WebElement searchBox = driver.findElement(By.cssSelector("#$invalid-selector")); }`
			
			`catch (WebDriverException e) { System.out.println("Invalid selector: " + e.getMessage()); }`

• NoSuchSessionException	Attempting to interact with an element after the WebDriver session has been closed (e.g., after driver.quit()).	Ensure all driver interactions occur before the session is terminated.	`driver.quit();`
			
			`try { WebElement searchBox = driver.findElement(By.name("q")); }`
			
			`catch (WebDriverException e) { System.out.println("No session available: " + e.getMessage()); }`