Painite User Guide:

# list of methods present in the Action class:

1. Action()

Description: Initializes the WebDriver instance by assigning the value from the StartBrowser class.

Example: Action action = new Action();

1. navigateToApplication(String url)

* Purpose: Navigates the WebDriver to the specified URL.
* Usage: navigateToApplication("https://example.com");

1. getCurrentURL(WebDriver driver)

Purpose: Retrieves the current URL of the web page.

Usage: String currentURL = action.getCurrentURL(driver);

1. getTitle(WebDriver driver)

Purpose: Retrieves the title of the web page.

Usage: String pageTitle = action.getTitle(driver);

1. type(WebElement ele, String text)

* Purpose: Types the specified text into the input field identified by the provided WebElement.
* Usage: type(someWebElement, "Hello, World!");

1. WebElement type(By locator, String testData, String eleName)

* Purpose: Types the specified test data into the input field identified by the provided locator. It logs the success or failure of the type action along with relevant details.
* Usage: WebElement inputField = type(By.id("username"), "exampleUser", "Username Field");

1. click(By locator, String eleName)

* Purpose: Performs a click action on the web element identified by the provided By locator.
* Usage: click(By.id("buttonId"), "Submit Button");

1. handleException(String eleName, Exception e)

* Purpose: Handles exceptions by logging error messages, stack traces, and capturing a screenshot.
* Usage: Automatically called within the click method.

1. slider(WebDriver driver, WebElement ele, int x, int y)

* Purpose: Performs a slider action on the specified WebElement by dragging and dropping it by the given x and y offset values. It provides console feedback based on the success or failure of the slider action.
* Usage: boolean result = slider(driver, someSliderElement, 150, 0);

1. scrollByVisibilityOfElement(WebDriver driver, WebElement ele)

* Purpose: Scrolls the page to make the specified element visible.
* Usage: scrollByVisibilityOfElement(driver, someWebElement);

1. click(WebDriver driver, WebElement ele)

* Purpose: Performs a click action on the provided WebElement using the Actions class.
* Usage: click(driver, someWebElement);

1. draggable(WebDriver driver, WebElement source, int x, int y)

* Purpose: Performs a draggable action on the specified source WebElement by dragging and dropping it by the given x and y offset values. It provides console feedback based on the success or failure of the draggable action.
* Usage: boolean result = draggable(driver, someDraggableElement, 150, 0);

1. dragAndDrop(WebDriver driver, WebElement source, WebElement target)

* Purpose: Performs a drag-and-drop action by dragging the source WebElement and dropping it onto the target WebElement. It provides console feedback based on the success or failure of the drag-and-drop action.
* Usage: boolean result = dragAndDrop(driver, sourceElement, targetElement);

1. rightClick(WebDriver driver, WebElement ele)

* Purpose: Performs a right-click action on the specified WebElement using the Selenium Actions class. It provides console feedback based on the success or failure of the right-click action.
* Usage: boolean result = rightClick(driver, someElement);

1. findElement(WebDriver driver, WebElement ele)

* Purpose: Checks whether the provided WebElement is found and displayed on the web page.
* Usage: boolean elementFound = findElement(driver, someElement);

1. isDisplayed(WebDriver driver, WebElement ele)

* Purpose: Verifies whether the provided WebElement is both found and displayed on the web page.
* Usage: isDisplayed(driver, someWebElement);

1. isSelected(WebDriver driver, WebElement ele)

* Purpose: Checks if the element is selected.
* Usage: isSelected(driver, someWebElement);

1. isEnabled(WebDriver driver, WebElement ele)

* Purpose: Checks if the element is enabled.
* Usage: isEnabled(driver, someWebElement);

1. selectBySendkeys(String value, WebElement ele)

* Purpose: Selects a dropdown option by sending keys to the specified WebElement, typically used when interacting with a dropdown that allows text input.
* Usage: selectBySendkeys("OptionValue", dropdownElement);

1. selectByIndex(WebElement element, int index)

* Purpose: Selects an option from a dropdown by index.
* Usage: selectByIndex(someDropdownElement, 2);

1. selectByValue(WebElement element, String value)

* Purpose: Selects an option from a dropdown by value.
* Usage: selectByValue(someDropdownElement, "optionValue");

1. selectFromDropDownByVisibleText(WebElement element, String visibleText, String eleName)

* Purpose: Selects an option from a dropdown by visible text and logs the action status.
* Usage: selectFromDropDownByVisibleText(someDropdownElement, "OptionText", "DropdownName");

1. mouseHoverByJavaScript(WebElement ele)

* Purpose: Performs mouse hover using JavaScript on the specified WebElement.
* Usage: mouseHoverByJavaScript(someWebElement);

1. mouseover(WebDriver driver, WebElement ele)

* Purpose: Simulates a mouseover action on the specified WebElement using the Selenium Actions class.
* Usage: mouseover(driver, someElement);

1. mouseOverElement(WebDriver driver, WebElement element)

* Purpose: Simulates a mouseover action on the specified WebElement using the Selenium Actions class. This method does not return a value and is primarily used for logging or reporting purposes.
* Usage: mouseOverElement(driver, someElement);

1. moveToElement(WebDriver driver, WebElement ele)

* Purpose: Moves the mouse to the specified WebElement on a web page after scrolling it into view using JavaScript.
* Usage: moveToElement(driver, someElement);

1. JSClick(WebDriver driver, WebElement ele)

* Purpose: Performs a click action using JavaScript on the specified WebElement.
* Usage: JSClick(driver, someWebElement);

1. switchToDefaultFrame(WebDriver driver)

* Purpose: Switches the WebDriver focus back to the default content or main frame of the web page.
* Usage: switchToDefaultFrame(driver);

1. switchToFrameByIndex(WebDriver driver, int index)

* Purpose: Switches to a frame using its index.
* Usage: switchToFrameByIndex(driver, 0);

1. switchToFrameByWebElement(WebDriver driver, WebElement frameElement)

* Purpose: Switches to a frame using a WebElement.
* Usage: switchToFrameByWebElement(driver, someFrameWebElement);

1. switchToFrameByNameOrId(WebDriver driver, String nameOrId)

* Purpose: Switches to a frame using its name or ID.
* Usage: switchToFrameByNameOrId(driver, "frameName");

1. switchToDefaultContent(WebDriver driver)

* Purpose: Switches back to the default content.
* Usage: switchToDefaultContent(driver);

1. handleAlert(WebDriver driver, String action)

* Purpose: Handles alerts by accepting or dismissing them.
* Usage: handleAlert(driver, "accept");

1. waitForElementToBeClickable(WebDriver driver, By locator, int timeoutInSeconds)

* Purpose: Waits for an element to be clickable within the specified timeout.
* Usage: waitForElementToBeClickable(driver, By.id("someId"), 10);

1. waitForElementToBePresent(WebDriver driver, By locator, int timeoutInSeconds)

* Purpose: Waits for an element to be present within the specified timeout.
* Usage: waitForElementToBePresent(driver, By.name("someName"), 15);

1. waitForElementToBeVisible(WebDriver driver, By locator, int timeoutInSeconds)

* Purpose: Waits for an element to be visible within the specified timeout.
* Usage: waitForElementToBeVisible(driver, By.className("someClass"), 20);

1. waitForElementToBeInvisible(WebDriver driver, By locator, int timeoutInSeconds)

* Purpose: Waits for an element to be invisible within the specified timeout.
* Usage: waitForElementToBeInvisible(driver, By.xpath("//someXpath"), 10);

1. fluentWait(WebDriver driver, final By locator)

* Purpose: Performs a fluent wait for an element to be present.
* Usage: fluentWait(driver, By.cssSelector("someSelector"));

1. captureScreenshot(String screenshotName)

* Purpose: Captures a screenshot and saves it to a specified location.
* Usage: captureScreenshot("HomePage");

1. switchToNewWindow(WebDriver driver)

* Purpose: Switches the WebDriver focus to a new window, typically used when dealing with multiple browser windows.
* Usage: switchToNewWindow(driver);

1. switchToOldWindow(WebDriver driver)

* Purpose: Switches the WebDriver focus back to the original window, typically used when dealing with multiple browser windows.
* Usage: switchToOldWindow(driver);

1. switchWindowByTitle(WebDriver driver, String windowTitle, int count)

* Purpose: Switches the WebDriver focus to a window with a specified title, using the window index.
* Usage: switchWindowByTitle(driver, "TargetWindowTitle", 2);

1. implicitWait(WebDriver driver, int timeOut)

* Purpose: Sets the implicit wait timeout for the WebDriver instance. Implicit wait instructs the WebDriver to wait for a certain amount of time before throwing a NoSuchElementException when attempting to find an element.
* Usage: implicitWait(driver, 10);

1. explicitWait(WebDriver driver, WebElement element, int timeOut)

* Purpose: Applies explicit wait for a specific WebElement to be present and visible in the DOM before further interaction. Explicit wait allows waiting for a certain condition to be met before proceeding with the execution.
* Usage: explicitWait(driver, someElement, 20);

1. fluentWait(WebDriver driver, WebElement element, int timeOut)

* Purpose: Applies fluent wait for a specific WebElement to be present and visible in the DOM before further interaction. Fluent wait provides more flexibility by allowing the definition of polling intervals and ignoring specific exceptions.
* Usage: fluentWait(driver, someElement, 20);

1. pageLoadTimeOut(WebDriver driver, int timeOut)

* Purpose: Sets the maximum time to wait for a page to load completely. This method configures the page load timeout for the WebDriver instance.
* Usage: pageLoadTimeOut(driver, 30);

1. logElementStatus(boolean flag)

* Purpose: Logs the status of whether an element is displayed or not.
* Usage: Automatically called within other methods.

1. logOptionSelectionStatus(boolean flag)

* Purpose: Logs the status of option selection in a dropdown.
* Usage: Automatically called within other methods.

1. logActionStatus(String status)

* Purpose: Logs the status of a general action.
* Usage: Automatically called within other methods.

1. logFrameSwitchStatus(int index, boolean flag)

* Purpose: Logs the status of switching to a frame by index.
* Usage: Automatically called within other methods.

1. logFrameSwitchStatus(WebElement frameElement, boolean flag)

* Purpose: Logs the status of switching to a frame by a WebElement.
* Usage: Automatically called within other methods.

1. logFrameSwitchStatus(String nameOrId, boolean flag)

* Purpose: Logs the status of switching to a frame by name or ID.
* Usage: Automatically called within other methods.

1. logAlertActionStatus(String action, boolean flag)

* Purpose: Logs the status of an alert action.
* Usage: Automatically called within other methods.

1. logElementInvisibilityStatus(By locator, boolean flag)

* Purpose: Logs the status of element invisibility.
* Usage: Automatically called within other methods.