Assertion

This list of tasks performs validation checks that allow you to verify if a certain condition is true or false.

Assertion tasks include the following:

- Assert Selected
- Assert Disabled
- Assert Visible
- Assert Focused
- Assert Hidden
- Assert ImageExists
- Assert File Download
- Assert Not Present
- Throw Error

Assert Selected

This task checks whether a required element is selected from an option that's part of a field that has a predefined set of options. For example, to ensure selection of items from fields such as drop-downs and radio buttons in a form.

Syntax

assertSelected(<locator>,<errorHandling>);

(or)

<variable> = assertSelected(<locator>,<errorHandling>);

Return Type
Boolean

Parameter	Data Type	Description
<locator>	STRING	Address to locate the target element of the dropdown or select element containing the list of options. **Note**: You can inspect from the webpage to find the locators of the elements or record and store the elements. Learn more about *Elements*.
<errorHandling>	STRING	If the verification fails, it returns a false response, as the element mismatches with the actual element. For such scenarios, choose how the consequent test scripts should behave: • **errorhandling.STOP_ON_ERROR** : This stops the execution of a test case if there is a mismatch. • **errorhandling.CONTINUE_ON_ERROR** : This gives the current response (true or false) for the test case and continues executing the test case.

Example: You are automating the testing of a product filter feature on an ecommerce website. Shoppers can filter products based on their color. The following script ensures that the color red is selected.

```
1. assertSelected(ui.zwatch.watchcolor.RED,errorhandling.STOP_ON_ERROR);
```

where:

ui.zwatch.watchcolor.RED	The recorded element to find the red color option in the dropdown.
errorhandling.STOP_ON_ERROR	This error handling behaviour stops the execution of the test case, if the option isn't selected.

Assert Disabled

This task verifies whether an element is disabled in a webpage.

Syntax

assertDisabled(<locator>,<errorHandling>);

(or)

<varaiable> = assertDisabled(<locator>,<errorHandling>);

Return Type
Boolean

Parameter	Data Type	Description
<locator>	STRING	Address to locate the element that has to be checked for its disable state. **Note**: You can inspect from the webpage to find the locators of the elements or record and store the elements. Learn more about *Elements*.
<errorHandling>	STRING	If the verification fails, it returns a false response, as the element mismatches with the actual element. For such scenarios, choose how the consequent test scripts should behave: • **errorhandling.STOP_ON_ERROR** : This stops the execution of a test case if there is a mismatch. • **errorhandling.CONTINUE_ON_ERROR** : This gives the current response (true or false) for the test case and continues executing the test case.

Example: You are automating the testing to see if the "Add to Cart" button is disabled for the products that are out of stock. The following script ensures that the "Add to Cart" button is disabled for the selected product.

```
1. assertDisabled(ui.zwatch.addtocart.GMT,errorhandling.STOP_ON_ERROR);
```

where:

ui.zwatch.addtocart.GMT	The recorded element of the add to cart button of the product that needs to be disabled.
errorhandling.STOP_ON_ERROR	This error handling behaviour stops the execution of the test case, if the button isn't disabled.

Assert Visible

This task is used to check whether the specified element is visible in the DOM or in the visible area of the screen.

Syntax

assertVisible(<locator>,<errorHandling>");

(or)

<variable> = assertVisible(<locator>,<errorHandling>");

Return Type
Boolean

Parameter	Data Type	Description
<locator>	STRING	Address to locate the element that has to be checked for visibility. **Note**: You can inspect from the webpage to find the locators of the elements or record and store the elements. Learn more about *Elements*.
<errorHandling>	STRING	If the verification fails, it returns a false response, as the element mismatches with the actual element. For such scenarios, choose how the consequent test scripts should behave: • **errorhandling.STOP_ON_ERROR** : This stops the execution of a test case if there is a mismatch.

		• **errorhandling.CONTINUE_ON_ERROR** : This gives the current response (true or false) for the test case and continues executing the test case.

Example: You are automating the testing of a product availability feature on an e-commerce website.The following script checks whether the specified product is visible or not.

```
1. assertVisible(ui.zwatch.GMT,errorhandling.STOP_ON_ERROR);
```

where:

ui.zwatch.GMT	The recorded element of the product that is selected.
errorhandling.STOP_ON_ERROR	This error handling behaviour stops the execution of the test case, if the product isn't visible on the webpage.

Assert Focused

This task checks whether the specific element in a webpage is currently active and can receive user interactions, such as keyboard input or mouse clicks.

Syntax

assertFocused(<locator>,<errorHandling>);

(or)

<variable> = assertFocused(<locator>,<errorHandling>);

Return Type
Boolean

Parameter	Data Type	Description
<locator>	STRING	Address to locate the element whose focus state needs to be checked.

		Note: You can inspect from the webpage to find the locators of the elements or record and store the elements. Learn more about *Elements*.
<errorHandling>	STRING	If the verification fails, it returns a false response, as the element mismatches with the actual element. For such scenarios, choose how the consequent test scripts should behave: • **errorhandling.STOP_ON_ERROR** : This stops the execution of a test case if there is a mismatch. • **errorhandling.CONTINUE_ON_ERROR** : This gives the current response (true or false) for the test case and continues executing the test case.

Example: You are automating the testing of a search functionality on an ecommerce website. The following script ensures that the search input field is correctly focused.

```
1. assertFocused(ui.zwatch.product-search,errorhandling.STOP_ON_ERROR);
```

where:

ui.zwatch.product-search	The recorded element of the search input field.
errorhandling.STOP_ON_ERROR	This error handling behaviour stops the execution of the test case, if the search field isn't focused.

Assert Hidden

This task checks whether the specified element is hidden in the DOM, or in the visible area of the screen.

Syntax

assertHidden(<locator>,<errorHandling>);

(or)

<variable> = assertHidden(<locator>,<errorHandling>);

Return Type
Boolean

Parameter	Data Type	Description
<locator>	STRING	Address to locate the element to check whether it is currently hidden. **Note**: You can inspect from the webpage to find the locators of the elements or record and store the elements. Learn more about *Elements*.
<errorHandling>	STRING	If the verification fails, it returns a false response, as the element mismatches with the actual element. For such scenarios, choose how the consequent test scripts should behave: • **errorhandling.STOP_ON_ERROR** : This stops the execution of a test case if there is a mismatch. • **errorhandling.CONTINUE_ON_ERROR** : This gives the current response (true or false) for the test case and continues executing the test case.

Example: You are automating the testing of a user registration functionality on an ecommerce website. The following script ensures that the password input field is hidden during user registration.

```
1. assertHidden(ui.zwatch.password-field,errorhandling.STOP_ON_ERROR);
```

where:

ui.zwatch.confirmation-message	The recorded element of the password input field.
errorhandling.STOP_ON_ERROR	This error handling behaviour stops the execution of the test case, if the password input

	field isn't hidden from the user's visibility on the webpage.

Assert ImageExists

This task is used to check whether the image is rendered in the element within the DOM or in the designated area of the webpage.

Syntax

assertImageExists(<locator>,<errorHandling>);

(or)

<variable> = assertImageExists(<locator>,<errorHandling>);

Return Type
Boolean

Parameter	Data Type	Description
<locator>	STRING	Address to locate the element to check whether the image exists or not. **Note**: You can inspect from the webpage to find the locators of the elements or record and store the elements. Learn more about *Elements*.
<errorHandling>	STRING	If the verification fails, it returns a false response, as the element mismatches with the actual element. For such scenarios, choose how the consequent test scripts should behave: • **errorhandling.STOP_ON_ERROR** : This stops the execution of a test case if there is a mismatch. • **errorhandling.CONTINUE_ON_ERROR** : This gives the current response (true or false) for the test case and continues executing the test case.

Example: You are automating the testing of product images on an ecommerce website. The following script ensures that the image of a product is correctly displayed and accessible to users.

```
1. assertImageExists(ui.zwatch.GMT-img,errorhandling.STOP_ON_ERROR);
```

where:

ui.zwatch.GMT-img	The recorded element of the image of product that has to be checked.
errorhandling.STOP_ON_ERROR	This error handling behaviour stops the execution of the test case, if the image doesn't exist.

Assert File Download

The task verifies whether the file is downloaded within the specified time.

Syntax

assertFileDownload(<filename>,<time>,<errorHandling>);

(or)

<variable> = assertFileDownload(<filename>,<time>,<errorHandling>);

Return Type
Boolean

Parameter	**Data Type**	**Description**
<filename>	STRING	Expected name of the file which has to be verified on the webpage
<time>	LONG	The maximum time the task should wait for the file to be downloaded in seconds.
<errorHandling>	STRING	If the verification fails, it returns a false response, as the element mismatches with the actual element. For such scenarios, choose how the consequent test scripts should behave:

		• **errorhandling.STOP_ON_ERROR** : This stops the execution of a test case if there is a mismatch. • **errorhandling.CONTINUE_ON_ERROR** : This gives the current response (true or false) for the test case and continues executing the test case.

Example: You are automating the testing of the downloadable product manuals on an e-commerce website. Customers should be able to download product manuals after purchase. The following script ensures that the file is downloaded from the product webpage.

```
1. hoverAndClick(ui.zwatch.GMT-manual);
2. assertFileDownload("zwatch_GMT-manual",600,errorhandling.STOP_ON_ERROR);
```

where:

ui.zwatch.GMT-manual	The recorded element of the manual is present on the website.
"zwatch_GMT-manual"	The file name of the file that has to be downloaded
600	The set download timeout in seconds
errorhandling.STOP_ON_ERROR	This error handling behaviour stops the execution of the test case, if the file is not downloaded.

Assert Not Present

The task verifies the absence or non-existence of a specific web element on a webpage.

Syntax

assertNotPresent(<locator>,<errorHandling>);

Return Type
Boolean

Parameter	Data Type	Description
<locator>	STRING	Address to locate the element to check whether it is currently present.

		Note: You can inspect from the webpage to find the locators of the elements or record and store the elements. Learn more about *Elements*.
<errorHandling>	STRING	If the verification fails, it returns a false response, as the element mismatches with the actual element. For such scenarios, choose how the consequent test scripts should behave: • **errorhandling.STOP_ON_ERROR** : This stops the execution of a test case if there is a mismatch. • **errorhandling.CONTINUE_ON_ERROR** : This gives the current response (true or false) for the test case and continues executing the test case.

Example: You are responsible for automating the testing of a product listing page on an ecommerce website that offers premium products. The website has a feature where premium products are only visible to registered users, and guest users cannot access or view them. The following script ensures that the specified product is restricted correctly.

```
1. assertNotPresent(ui.zwatch.product-Blue-Bay-GMT,errorhandling.STOP_ON_ERROR);
```

where:

ui.zwatch.product-Blue-Bay-GMT	The recorded element for the product that has to be checked for its presence on the website.
errorhandling.STOP_ON_ERROR	This error handling behaviour stops the execution of the test case, if the product is listed.

Throw Error

The task generates an exception along with a message. This validates custom error logic or functions and generates corresponding custom error messages.

Syntax

throwError(<message>);

where:

Parameter	Data Type	Description
<message>	STRING	Error message that needs to be printed.

Example: A Quality Assurance tester, is conducting testing on an e-commerce website to ensure its visual elements meet design specifications. Specifically, they are tasked with verifying the width of the website's logo. The tester creates the following custom function scripts to test the same. Here, the "throw error" task generates an exception, if the width of the logo is not 110px.

```
1. openURL("https://zwatch.com/", "same window"); // Navigates to the website. Understand about openURL task.
2. logoWidthCheck = getWidth("img[data-zs-logo]"); // Assigning the width of the logo to a variable. Learn more about getWidth task.
3. if(logoWidthCheck != 110)
4. {
5. throwError("Logo width incorrect");
6. }
```

where:

"https://zwatch.com"	URL of the website that needs to be opened.
"same window"	This represents that the URL needs to be opened in the same window.
logoWidthCheck	Variable that stores the width of the logo
"img[data-zs-logo]"	Locator element to find the logo on the website
"Logo width incorrect"	Error message that needs to be printed if the width is incorrect