




General Tasks

This list of tasks will have the primary actions inside a web browser. These tasks include the following:

- [Open URL](#)
- [Click](#)
- [Set Value](#)
- [Assert Text](#)
- [Assert Value](#)
- [Wait](#)


 **Note:** These tasks are used for web test cases.


Open URL

The *openURL* task navigates to the specified URL of the website during the test case execution.

Syntax

`openURL(<URL>, <window_type>);`

Parameter	Data Type	Description
<URL>	STRING	<p>URL of the website that will be opened.</p> <div> Note:<ul style="list-style-type: none">• All URLs must contain the appropriate format (Example: <code>https://www.zoho.com</code>)• The open URL task will fail if the URL doesn't open within three seconds, leading to the termination of the test case.</div>

<window_type>	STRING	<p>The type of window in which the URL will be opened.</p> <p>Applicable values:</p> <ul style="list-style-type: none"> • same window • new tab • new window <div>  Note: When the test case is recorded the default window type selected would be "same window". </div>
---------------	--------	--

Example 1: Assume you are testing a website's login functionality. Use the openURL task to navigate to the URL of the website's login page to perform the login actions. The following script navigates to the URL.

```
1. openURL("https://www.zwatch.com" , "same window");
```

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.

Example 2: In the previous example, say, the URL was stored inside a <variable> named URL. Use the following script to open the stored URL.

```
1. openURL($URL , "new window");
```

where:


\$URL	The variable that stores the value of the URL to be opened.
"new window"	The URL needs to be opened in the new window.

Click

The task performs the action of clicking on any web element.

Syntax

click(<locator>);

Parameter	Data Type	Description
<locator>	STRING	Address to locate the element you need to click on  Note: You can inspect from the webpage to find the locators of the elements or record and store the elements. Learn more about Elements .

Example 1: You are testing an ecommerce website and want to verify the functionality of the "Add to Cart" button on a product page. The following script checks whether the button is clickable or not.

1. `openURL("https://www.zwatch.com" , "same window");`
2. `click("//span[text()='Add to Cart']");`

where:

"https://www.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.
"//span[text()='Add to Cart']"	This indicates the xpath of the element "Add to Cart" button. Here, the "//" is used to select the current node of the xpath, which is 'span' in this case. The '[']' is used to enclose the attribute name and its value.

Example 2: For the same example above, if the element's locators are recorded. <Learn more about recorded elements>.

1. `openURL("https://www.zwatch.com" , "same window");`
2. `click(ui.guest_shopping.add_to_cart_button);`

where:


"https://www.zwatch.com"	URL of the website that needs to be opened.
ui.guest_shopping.add_to_cart_button	The recorded locator of the element "Add to Cart" button.

Set Value

The task allows you to input values to any user interface element on a webpage. For example, to input values to fields in a login form.

Syntax

setValue(<locator>, <value>);

Parameter	Data Type	Description
<locator>	STRING	Address to locate the input element. <div>  Note: You can inspect from the webpage to find the locators of the elements or record and store the elements. Learn more about Elements. </div>
<value>	STRING	Value to be set in the input element.

Example: Let's assume the user is testing the login functionality of a website. The following script checks whether the user can input values to the username field of the login form.

1. `openURL("https://www.zwatch.com" , "same window");` // Navigates to the website. [Understand about openURL task](#).
2. `setValue("#login_id",$Username);`

where:

"https://www.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.
"#login_id"	The locator of the element "username field" of the login form. Here, the '#' is used to refer to the ID of the element.
\$Username	The variable where the username value is

Assert Text

The task verifies whether a specific text is present within the web element of the webpage.

Syntax


```
assertText(<locator>,<text>,<errorHandling>);
```

(or)

```
<variable> = assertText(<locator>, <text>, <errorHandling>);
```

Return Type

Boolean

Parameter	Data Type	Description
<locator>	STRING	Address to locate the element that should contain the expected text.  Note: You can inspect from the webpage to find the locators of the elements or record and store the elements. Learn more about Elements.
<text>	STRING	Text you expect to find in the element.
<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: <ul style="list-style-type: none">• 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 1: Consider an online shopping website's welcome page. When customers navigate to the URL, you want to ensure that the welcome message is correctly indicated. Here, the following script uses the `assertText` task to check if the welcome message is present on the page.

1. `openURL("https://www.zwatch.com" , "same window");` // Navigates to the website. [Understand about openURL task.](#)
2. `assertText(ui.welcome_message.h3,"Welcome to Zylker Watches!",errorhandling.STOP_ON_ERROR);`

where:

"https://www.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.
ui.welcome_message.h3	The stored locator of the element that displays the welcome message.
"Welcome to Zylker Watches!"	The welcome message text that needs to be displayed.
errorhandling.STOP_ON_ERROR	This error handling behaviour stops the execution of the test case, if the message isn't displayed.

Assert Value


The task verifies the input value against the expected user input value in the element within a webpage. For example, it can be used to validate login form inputs.

Syntax

```
assertValue(<locator>, <value>, <errorHandling>);
```

Return Type

Boolean

Parameter	Data Type	Description
<locator>	STRING	Address to locate the element that should contain the value.  Note: You can inspect from the webpage to find the locators of the elements or record and store the elements. Learn more about Elements.
<value>	STRING	Value you expect to find in the element.

<errorHandling>	STRING	<p>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:</p> <ul style="list-style-type: none"> • 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: Let's assume a user wants to test the login functionality of a website. In this case, the following script can be used to check whether the user name entered is the same as required.

1. `openURL("www.zwatch.com", "same window");`
2. `assertValue("#userName","Chris",errorhandling.STOP_ON_ERROR);`

where:

"https://www.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.
#userName	The locator of the element
Chris	The name that has to be verified
errorhandling.STOP_ON_ERROR	This error handling behaviour stops the execution of the test case, on not finding the expected value.

Wait


The task waits until a specified time before executing the next actions in a test case workflow. The task can be used to wait for a defined time duration or also be configured to wait for dynamic time durations which depends on specific events, such as element clickability, visibility, and more.

Syntax

`wait(<time>);`

Parameter	Data Type	Description
<time>	LONG	Duration to wait in seconds.

wait(<time>,<locator>,<event>);

Parameter	Data Type	Description
<time>	LONG	Duration to wait for the element in seconds. The maximum time you can specify in a wait task is 150s.
<locator>	STRING	<p>Address to locate the element based on which the wait condition should be applied.</p> <div>  Note: You can inspect from the webpage to find the locators of the elements or record and store the elements. Learn more about Elements. </div>
<event>	STRING	<p>An event defines the specific waiting condition for the element. The 'Wait Task' will pause its execution until the specified event occurs. The supported events for the element:</p> <ul style="list-style-type: none"> • event.HIDE : event that is triggered to detect when an element in a webpage is currently hidden or not displayed. • event.PRESENT : event that is triggered when an element becomes present or visible on a webpage • event.SHOW : event that is triggered when an element becomes visible or is displayed on a webpage • event.CLICKABLE : event that is triggered when an element becomes clickable, indicating it's ready for user interaction.

Example 1: The user wants to test a web application with a multi-step form that users need to complete. After the user fills out each section of the form, there's a requirement to introduce a delay before proceeding to the next step to simulate a user thinking or taking an action. In this case, you can use the wait task to pause the automation script for a specific duration after each form section is completed.

1. `wait(5); // Pause for 5 seconds`

where:

5	Specifying the time in seconds, for the element to wait.
---	--

Example 2: Suppose you're testing a webpage where a loading spinner appears while a form is being submitted. The following script can be used to configure a wait task to wait for the spinner to become visible after clicking the "Submit" button.

1. `openURL("https://www.zwatch.com" , "same window");` // Navigates to the website. [Understand about openURL task.](#)
2. `click("//lyte-search[@id='submit-button']//input");` // Locate the "Submit" button and click it. [Learn more about Click task.](#)
3. `wait(10,"//lyte-search[@id='submit-button']//input",event.SHOW);`

where:

"https://www.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.
10	Specifying the time in seconds, for the element to wait.
"//lyte-search[@id='elementGroupSearch']//input"	This determines the locator of the element you want to wait for. Here, the '/' is used to select the current node of the xpath, which is 'lyte-search' in this case. The '[']' is used to enclose the attribute name and its value.
event.SHOW	This event waits for the element which is visible in the DOM.