Selenium IDE

* Selenium IDE (Integrated Development Environment) is an open-source web automation testing tool under the Selenium Suite.
* Unlike Selenium WebDriver, it does not require any programming logic to write its test scripts rather you can simply record your interactions with the browser to create test cases. Subsequently, you can use the playback option to re-run the test cases.
* Selenium IDE is based on **JavaScript**.
* It is available for Chrome and Firefox only

# Adding Selenium IDE extension to chrome

* Open Chrom browser
* Click on more options (…)
* Select extensions
* Search for Selenium IDE extension
* Click on Add to Chrome
* To open IDE, click on the extension and select Selenium IDE

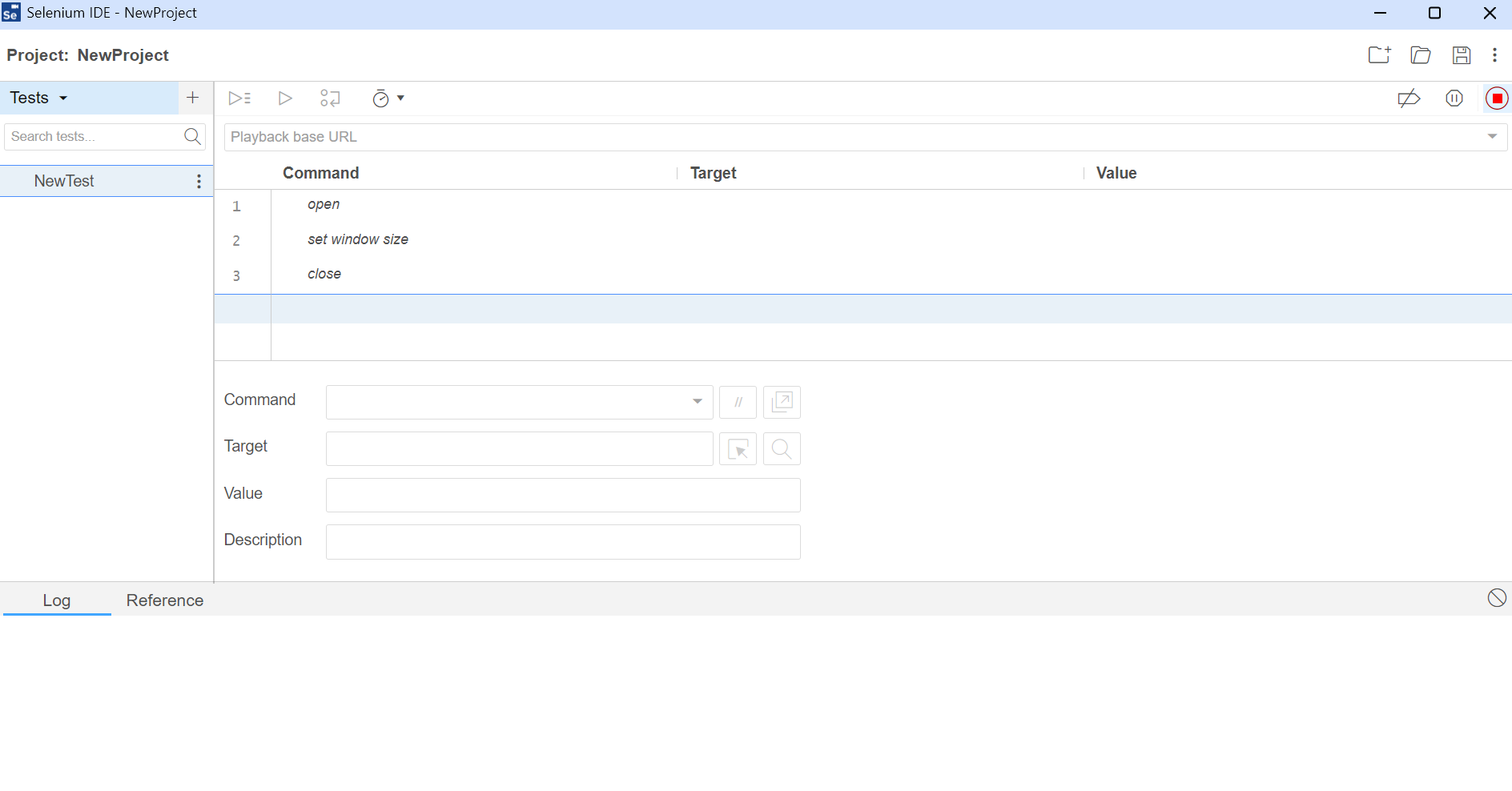
# Selenium IDE features

Selenium IDE has the following features:

1. Menu bar
2. Tool bar
3. Address bar
4. Test Case pane
5. Test Script editor box
6. Start/Stop Recording Button
7. Logs

Testcase pane

Menu bar



Record

Test Script area

Log reference pane

Tool bar

Address bar

# Record and Playback

Selenium IDE consists of three important steps:

1. Record
2. Playback
3. Saving the test

# Selenium IDE commands

* Selenium commands, also known as "Selenese" are the set of commands used in Selenium IDE that run your tests.
* A sequence of Selenium commands (Selenese) together is known as test script.

Selenium commands are classified into three categories:

1. Actions
2. Accessors
3. Assertions

## Actions

Actions are commands that are done on the application. Example click, type, select etc. If any action fails or the application has a bug then the test execution will fail automatically.

Commonly used action commands:

|  |  |
| --- | --- |
| Command | Description |
| Open(url) | Launches application |
| type (locator,value) | Types in a text field |
| typeKeys (locator,value) | Simulates keystroke events on the specified element. |
| click (locator) | Performs click operation |
| doubleClick (locator) | Performs double click operation |
| close() | Closes the application |
| store (expression,variableName) | This command specifies the name of a variable in which the result is to be stored |

## Accessors

Accessors stores the result of an element in a variable.

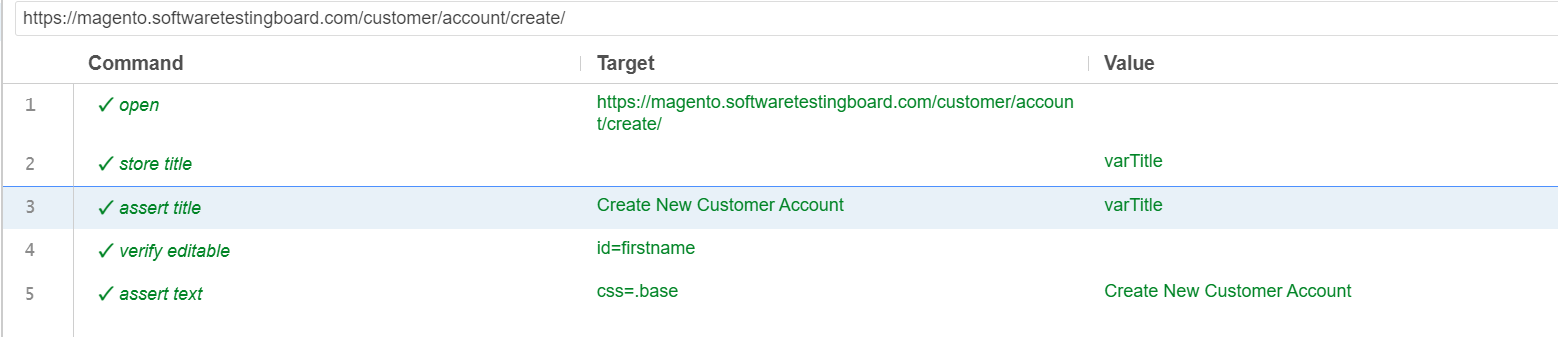
|  |  |
| --- | --- |
| Command | Description |
| storeTitle (variableName) | Gets the title of the current page. |
| storeText (locator, variableName) | Gets the text of an element. |
| storeValue (locator,variableName) | This command gets the (whitespace-trimmed) value of an input field. |
| storeTable (tableCellAddress, variableName) | This command gets the text from a cell of a table. |
| storeAllButtons (variableName) | It returns the IDs of all buttons on the page. |
| storeAllFields (variableName) | It returns the IDs of all input fields on the page. |
| storeAllLinks (variableName) | It returns the IDs of all links on the page. |
|  |  |

## Assertions

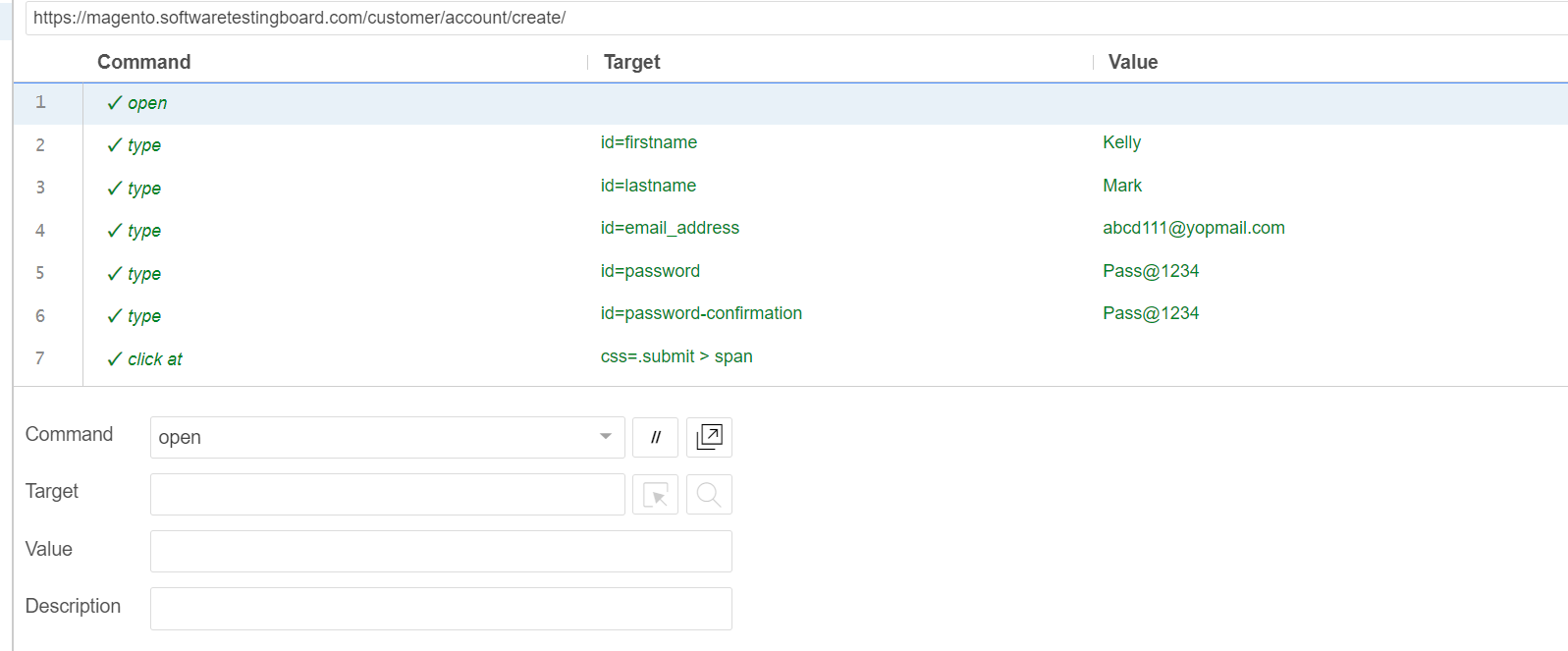
Assertions are the commands that enable testers to verify the state of the application.

* **assert(variable, expected value)** − This command is used to verify if the variable is matching the expected value. The variable is converted to string for comparison. If assert did not pass, then the execution will stop at the point of mismatch.
* **assert alert(alert text)** − This command is used to verify if an alert has the text matching the alert text. If assert did not pass, then the execution will stop at the point of mismatch.
* **assert checked(locator)** − This command is used to verify if the target element with the locator value is checked. If assert did not pass, then the execution will stop at the point of mismatch.
* **assert editable(locator)** − This command is used to verify if the target element with the locator value is editable. If assert did not pass, then the execution will stop at the point of mismatch.
* **assert element present(locator)** − This command is used to verify if the target element with the locator value is available on the web page. If assert did not pass, then the execution will stop at the point of mismatch.
* **assert element not present(locator)** − This command is used to verify if the target element with the locator value is not available on the web page. If assert did not pass, then the execution will stop at the point of mismatch.
* **assert not checked(locator)** − This command is used to verify if the target element with the locator value is not checked. If assert did not pass, then the execution will stop at the point of mismatch.
* **assert not editable(locator)** − This command is used to verify if the target element with the locator value is not editable. If assert did not pass, then the execution will stop at the point of mismatch.
* **assert not selected value(locator, text)** − .This command is used to verify if the value attribute of the selected option of the target dropdown element does not have the given text. If assert did not pass, then the execution will stop at the point of mismatch.
* **assert not text(locator, text)** − This command is used to verify if the target element with the locator value does not have the given text.
* **assert selected value(locator, text)** − This command is used to verify if the value attribute of the selected option of the target dropdown element has the given text. If assert did not pass, then the execution will stop at the point of mismatch.
* **assert selected label(locator, text)** − This command is used to verify if the label of the selected option of the target dropdown element has the given text. If assert did not pass, then the execution will stop at the point of mismatch.
* **assert text(locator, text)** − This command is used to verify if the target element with the locator value has the given text. If assert did not pass, then the execution will stop at the point of mismatch.
* **assert value(locator, text)** − This command is used to verify the value of an edit box. For a radio button/checkbox, the value will be set to on/off provided it is selected or not. If assert did not pass, then the execution will stop at the point of mismatch.
* **verify(variable, expected value)** − This command is used to verify if the variable is matching the expected value. The variable is converted to string for comparison. It is a soft assert and execution will not stop in case of a mismatch.
* **verify checked(locator)** − This command is used to verify if the target element with the locator value is checked. It is a soft assert and execution will not stop in case of a mismatch.
* **verify editable(locator)** − This command is used to verify if the target element with the locator value is editable. It is a soft assert and execution will not stop in case of a mismatch.
* **verify element present(locator)** − This command is used to verify if the target element with the locator value is available on the web page. It is a soft assert and execution will not stop in case of a mismatch.
* **verify element not present(locator)** − This command is used to verify if the target element with the locator value is not available on the web page. It is a soft assert and execution will not stop in case of a mismatch.
* **verify not editable(locator)** − This command is used to verify if the target element with the locator value is not editable. It is a soft assert and execution will not stop in case of a mismatch.
* **verify not selected value(locator, text)** − This command is used to verify if the value attribute of the selected option of the target dropdown element does not have the given text. It is a soft assert and execution will not stop in case of a mismatch.
* **verify not text(locator, text)** − This command is used to verify if the target element with the locator value does not have the given text. It is a soft assert and execution will not stop in case of a mismatch.
* **verify selected label(locator, text)** − This command is used to verify if the label of the selected option of the target dropdown element has the given text. It is a soft assert and execution will not stop in case of a mismatch.
* **verify selected value(locator, text)** − This command is used to verify if the value attribute of the selected option of the target dropdown element has the given text. It is a soft assert and execution will not stop in case of a mismatch.
* **verify text(locator, text)** − This command is used to verify if the target element with the locator value has the given text. It is a soft assert and execution will not stop in case of a mismatch.
* **assert title(text)** − This command is used to verify if the alert generated has the given alert text. If assert did not pass, then the execution will stop at the point of mismatch.
* **verify value(locator, text)** − This command is used to verify the value of an edit box. For a radio button/checkbox, the value will be set to on/off provided it is selected or not. It is a soft assert and execution will not stop in case of a mismatch.
* **wait for element editable(locator, wait time)** − This command is used to wait for the given wait time(in milliseconds) for the target element with the locator value to be editable.
* **wait for element not editable(locator, wait time)** − This command is used to wait for the given wait time(in milliseconds) for the target element with the locator value not to be editable.
* **wait for element not present(locator, wait time)** − This command is used to wait for the given wait time(in milliseconds) for the target element with the locator value to be unavailable on the web page.
* **wait for element not visible(locator, wait time)** − This command is used to wait for the given wait time(in milliseconds) for the target element with the locator value to be invisible on the web page.
* **wait for element present(locator, wait time)** − This command is used to wait for the given wait time(in milliseconds) for the target element with the locator value to be available on the web page.
* **wait for element visible(locator, wait time)** − This command is used to wait for the given wait time(in milliseconds) for the target element with the locator value to be visible on the web page.

Example of assert and verify



Sample test created:

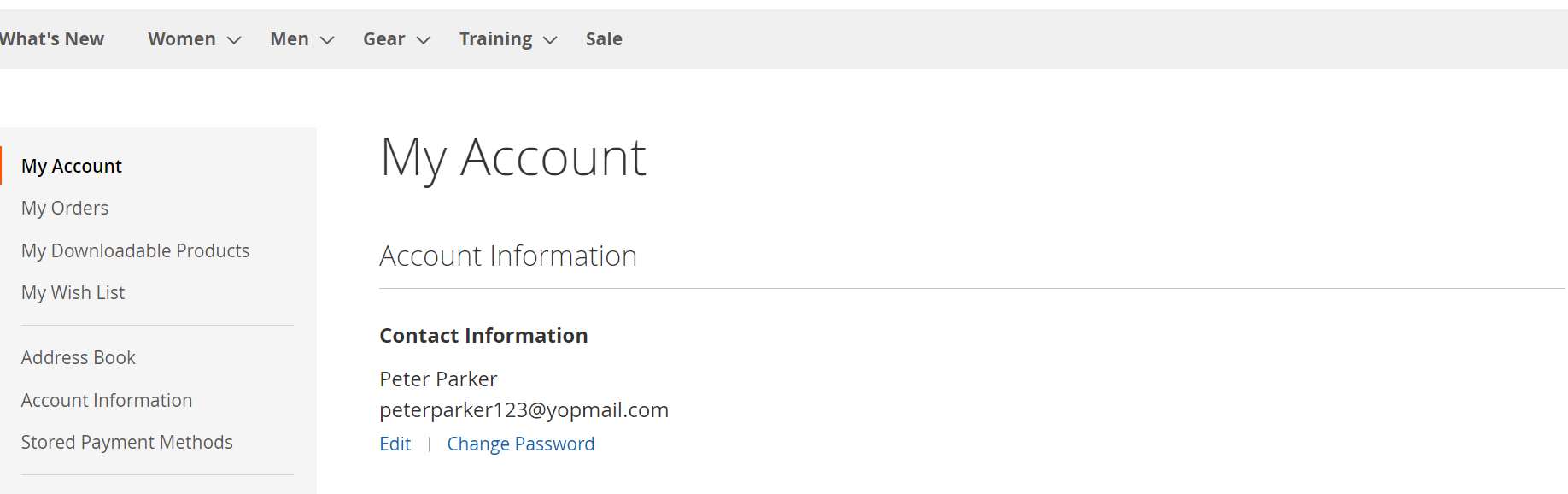


# Locators

* [Locating by Identifier](https://www.javatpoint.com/selenium-ide-locating-strategies-by-identifier)
* [Locating by ID Element](https://www.javatpoint.com/selenium-ide-locating-strategies-by-id)
* [Locating by Name](https://www.javatpoint.com/selenium-ide-locating-strategies-by-name)
* [Locating by XPath](https://www.javatpoint.com/selenium-ide-locating-strategies-by-xpath)
* [Locating by CSS](https://www.javatpoint.com/selenium-ide-locating-strategies-by-css)
* [Locating by DOM](https://www.javatpoint.com/selenium-ide-locating-strategies-by-dom)

Steps:

1. Go to https://magento.softwaretestingboard.com/
2. Click on Create account



1. After login, verify if the Name of the customer is correct
2. Verify if Edit and Change Password links are editable
3. Logout and close the browser