# Selenium IDE-Installation

Selenium IDE is an open-source tool that allows you to easily record and playback tests in your browser so that you can catch bugs easily and also ensure that your web application is running smoothly. With features like debugging and test case management, Selenium IDE simplifies your testing process and helps you get more done in less time.

* **Selenium IDE** is a **Firefox** plugin that enables you to record, edit, and debug tests for web applications.
* It provides a simple interface for recording user interactions with a web application and then playing those interactions back to test for errors and bugs.
* **Selenium IDE** is a popular choice for those new to test automation because it requires no programming knowledge and provides a simple way to create automated tests.
* **Selenium IDE** is a powerful tool for automating web application testing. With its easy-to-use interface and record and playback functionality, Selenium IDE is a popular choice for those new to test automation.

## Installation in Google Chrome

To install Selenium IDE on Google Chrome add the extensions of Selenium IDE in the web browser. Follow the steps below:

**Step 1:**Open the Google Chromebrowser on your system and then go to

the [Link](https://chrome.google.com/webstore/detail/selenium-ide/mooikfkahbdckldjjndioackbalphokd). This will direct you to the Chrome web store where you are required to add the extension of Selenium IDE on your Chrome browser. Click on **Add to Chrome** button.

**Step 2:** As you click on**Add to Chrome** button, a dialog box of warning will appear on the screen. Simply click on the**Add extension** button to finally add the extension to your Chrome browser.

**Step 3:**Now you need to click on**the extension** icon on the Google Chrome browser which is at the top right corner. A list of extensions will appear on the screen, pin the Selenium IDE extension.

**Step 4:**Now click on **Selenium IDE** to start working with Selenium IDE. You can now record a new test in a new project, open an existing project, and also you can create a new project.  
Installation in Mozilla Firefox

Selenium IDE has now been successfully added to your Google Chrome web browser.

## *Installation in Mozilla Firefox*

To install Selenium IDE on Mozilla Firefox you need to follow the given steps:

**Step 1:**Open the **Mozilla Firefox** browser on your system and then go to the [Link](https://www.selenium.dev/selenium-ide/).

**Step 2:** As you click on the link you will be redirected to the Selenium IDE website where you need to choose the **Firefox download** button to download the extension of Selenium IDE on the Firefox browser.

**Step 3:** After clicking on the Firefox download button you need to click on the **Add to Firefox** button to add the extension of Selenium IDE on your browser. Grant all the permissions and click on the option.

**Step 4:**Now you need to go to the **extension** icon which is placed at the top right corner on your browser’s screen. A list of extensions will appear on the screen. Choose the **Selenium IDE** extension.

**Step 5:**Now click on **Selenium IDE** to start working with Selenium IDE. You can now record a new test in a new project, open an existing project, and also you can create a new project.

Selenium IDE has now been successfully added to your Mozilla Firefox web browser.

## Conclusion

Selenium IDE is a powerful tool for automating web application testing. With its easy-to-use interface and record and playback functionality, **Selenium IDE** is a popular choice for those new to test automation. Whether you’re a beginner or an experienced developer, **Selenium IDE** can help you create automated tests for your web applications and ensure that they’re running smoothly.

# Selenium IDE- Commands (Selenese)

Selenium commands, also known as "Selenese" are the set of commands used in Selenium IDE that run your tests. Using selenese, one can perform activities like:

* Testing the existence of UI elements based on their HTML tags.
* Test for specific content.
* Test for broken links.
* Testing input fields, selection list options, submitting forms and table data among other things.
* Testing of window size, mouse options, alerts, Ajax functionality, pop-up windows, event handling and many other web application features.

A sequence of Selenium commands (Selenese) together is known as test script.

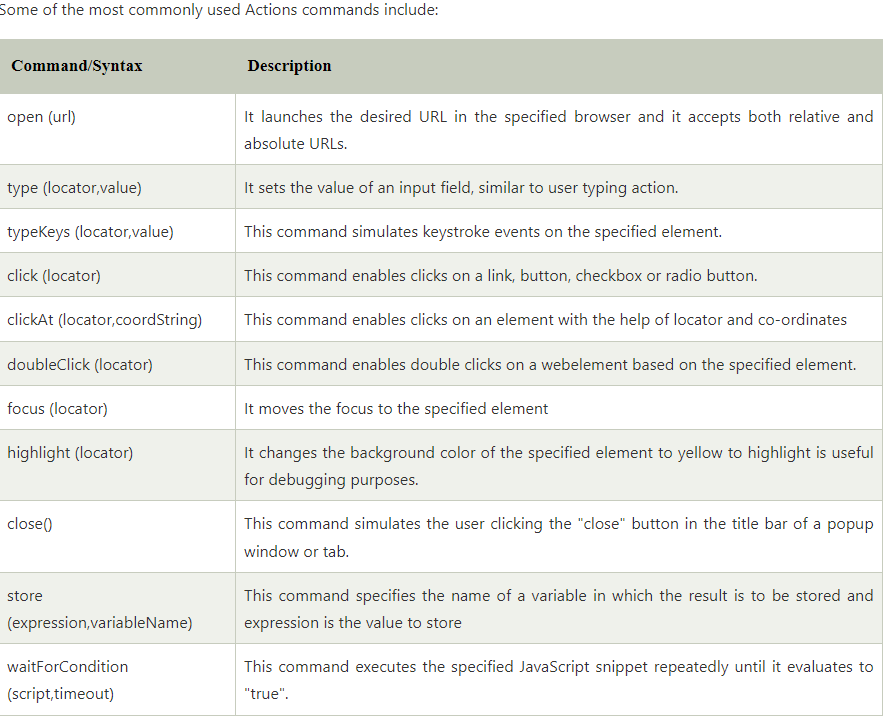
## Types of Selenium Commands

Selenium commands are basically classified in three categories:

1. Actions
2. Accessors
3. Assertions

## 1. Actions

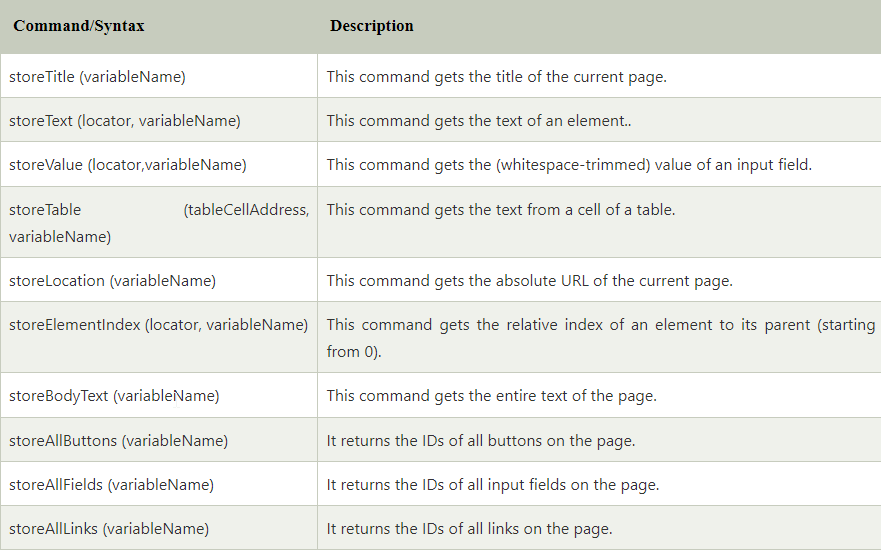
Actions are the selenium commands that generally manipulate the state of the application. Execution of Actions generates events like click this link, select that option, type this box, etc. If an Action fails, or has a bug, the execution of current test is stopped.



## 2. Accessors

Accessors are the selenium commands that examine the state of the application and store the results in variables. They are also used to automatically generate Assertions.

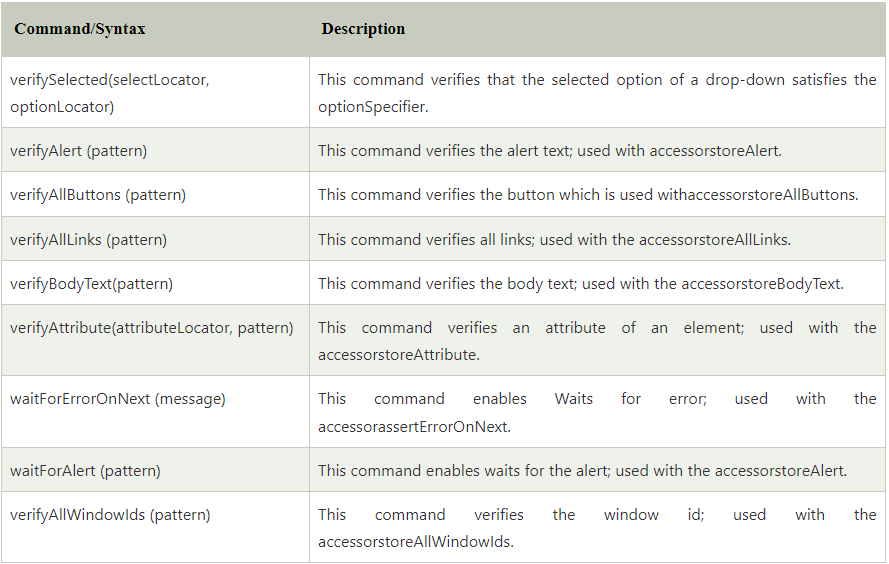
Some of the most commonly used Accessors commands include:



3. **Assertions**

Assertions are the commands that enable testers to verify the state of the application. Assertions are generally used in three modes assert, verify and waitfor.

Some of the most commonly used Assertions commands are:



**Developing Test Cases & Test Suites With Selenium-IDE**

Certainly! Here's a simple example of how you can develop login test cases and test suites using Selenium IDE:

**Test Case 1: Verify Successful Login**

* **Open Browser**:
  + Command: open
  + Target: URL of the login page
* **Enter Username**:
  + Command: type
  + Target: Locator of the username input field
  + Value: Your username
* **Enter Password**:
  + Command: type
  + Target: Locator of the password input field
  + Value: Your password
* **Click Login Button**:
  + Command: click
  + Target: Locator of the login button
* **Verify Successful Login**:
  + Command: assertTextPresent
  + Target: Text displayed after successful login (e.g., "Welcome, [Username]")

**Test Case 2: Verify Error Message for Invalid Credentials**

* **Open Browser**:
  + Command: open
  + Target: URL of the login page
* **Enter Invalid Username**:
  + Command: type
  + Target: Locator of the username input field
  + Value: Invalid username
* **Enter Invalid Password**:
  + Command: type
  + Target: Locator of the password input field
  + Value: Invalid password
* **Click Login Button**:
  + Command: click
  + Target: Locator of the login button
* **Verify Error Message**:
  + Command: assertTextPresent
  + Target: Text of the error message displayed for invalid credentials

**Test Suite: Login Tests**

* **Login Test 1 - Verify Successful Login**: Include Test Case 1 (Verify Successful Login).
* **Login Test 2 - Verify Error Message for Invalid Credentials**: Include Test Case 2 (Verify Error Message for Invalid Credentials).

**Using Selenium IDE:**

* Open Selenium IDE.
* Create a new test case for each of the test cases described above.
* Name each test case accordingly (e.g., "Verify Successful Login").
* Organize test cases into a test suite named "Login Tests".
* Run the test suite to execute all test cases sequentially.
* Review test results for any failures or errors.
* That's it! You've now developed simple login test cases and organized them into a test suite using Selenium IDE. You can expand upon these test cases by adding additional validations, such as checking for invalid input handling or testing different scenarios.
* In Selenium IDE, you typically create test cases using the graphical user interface, but you can export these test cases to various programming languages, including Java, for further customization and execution. Here's how you can develop simple login test cases and test suites using Selenium IDE and Java:
* **Step 1: Create Test Cases in Selenium IDE**
* Open Selenium IDE and record test cases for login functionality.
* Include steps to navigate to the login page, enter valid/invalid credentials, and submit the login form.
* Add assertions to verify successful login or display of error messages.
* **Step 2: Export Test Cases to Java**
* After recording and verifying test cases in Selenium IDE, go to "File" > "Export Test Case As...".
* Choose "Java / JUnit / WebDriver" as the export format.
* Save the exported Java file to your desired location.
* **Step 3: Set Up Java Project**
* Open your preferred Java IDE (e.g., Eclipse, IntelliJ IDEA).
* Create a new Java project or use an existing one.
* Add the Selenium WebDriver library to your project's build path. You can download it from the Selenium website or use Maven/Gradle for dependency management.
* **Step 4: Copy and Customize Java Code**
* Open the exported Java file containing your test cases.
* Copy the Java code from the exported file and paste it into your Java project.
* Customize the Java code as needed, such as adding package declarations, importing necessary classes, and organizing test methods.
* Update WebDriver initialization and other configurations as per your project setup.
* **Step 5: Execute Test Cases**
* Run the Java test class containing your login test cases using your Java IDE's test runner or command-line tools (e.g., JUnit, TestNG).
* Monitor the execution results to ensure that the login functionality behaves as expected.
* Debug any issues and make necessary adjustments to your test cases or application code.

**import org.openqa.selenium.WebDriver;**

**import org.openqa.selenium.chrome.ChromeDriver;**

**import org.testng.annotations.Test;**

**public class LoginTest {**

**@Test**

**public void testSuccessfulLogin() {**

**WebDriver driver = new ChromeDriver();**

**driver.get("https://example.com/login");**

**// Your Selenium WebDriver code for login test steps**

**// Example: Find elements, enter credentials, click login button, assert success message**

**driver.quit();**

**}**

**}**