INFT 1207

Week 8: Lecture 1

Introduction and Selenium IDE

Dr. Sukhwant Kaur Sagar

CONTENTS

Automated testing

- Automation testing uses the specialized tools to automate the execution of manually designed test cases without any human intervention.
- Automation testing tools can access the test data, controls the execution of tests and compares the actual result against the expected result.
- Automation testing covers both functional and performance test on an application.
  - Functional automation is used for automation of functional test cases. For example, regression tests, which are repetitive in nature, are automated.
  - Performance automation is used for automation of non-functional performance test cases. For example, measuring the response time of the application under considerable (say 100 users) load.

Automated Testing Tools

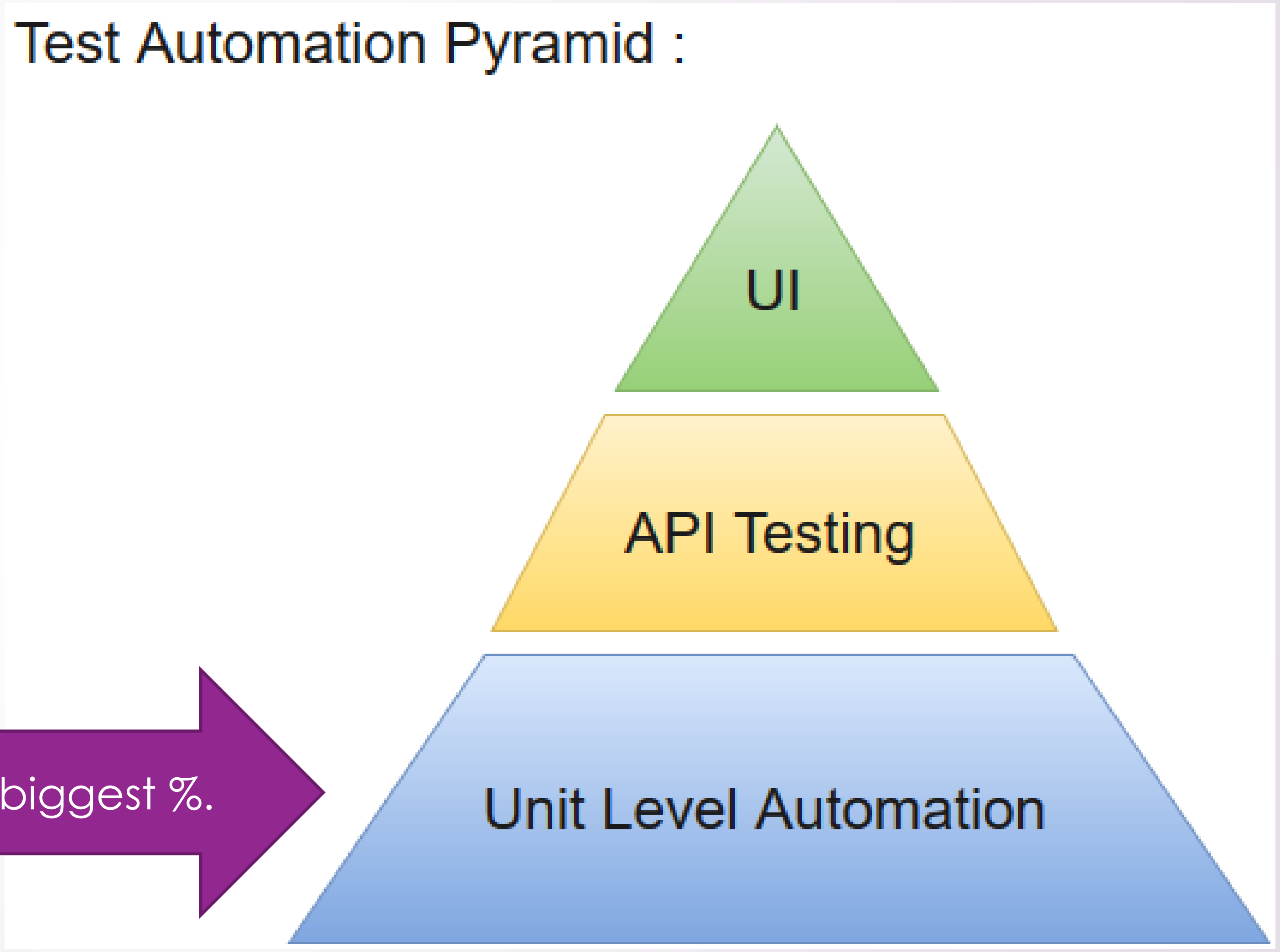
Functional automation Tools

- Quick Test Professional, provided by HP.
- Rational Robot, provided by IBM.
- Coded UI, provided by Microsoft.
- Selenium, open source.
- Auto It, open Source.

Non-functional Automation tools

- Load Runner, provided by HP.
- JMeter, provided by Apache.
- Burp Suite, provided by PortSwigger.
- Acunetix, provided by Acunetix.

Test Automation for Web Applications

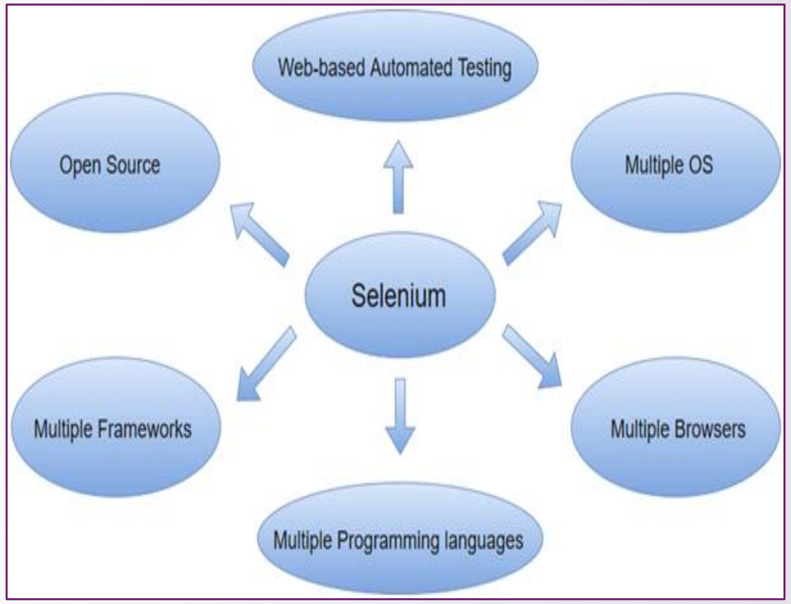


Unit testing represents the biggest %.

https://static.javatpoint.com/tutorial/selenium/images/selenium-basic-terminology2.png

What is Selenium?

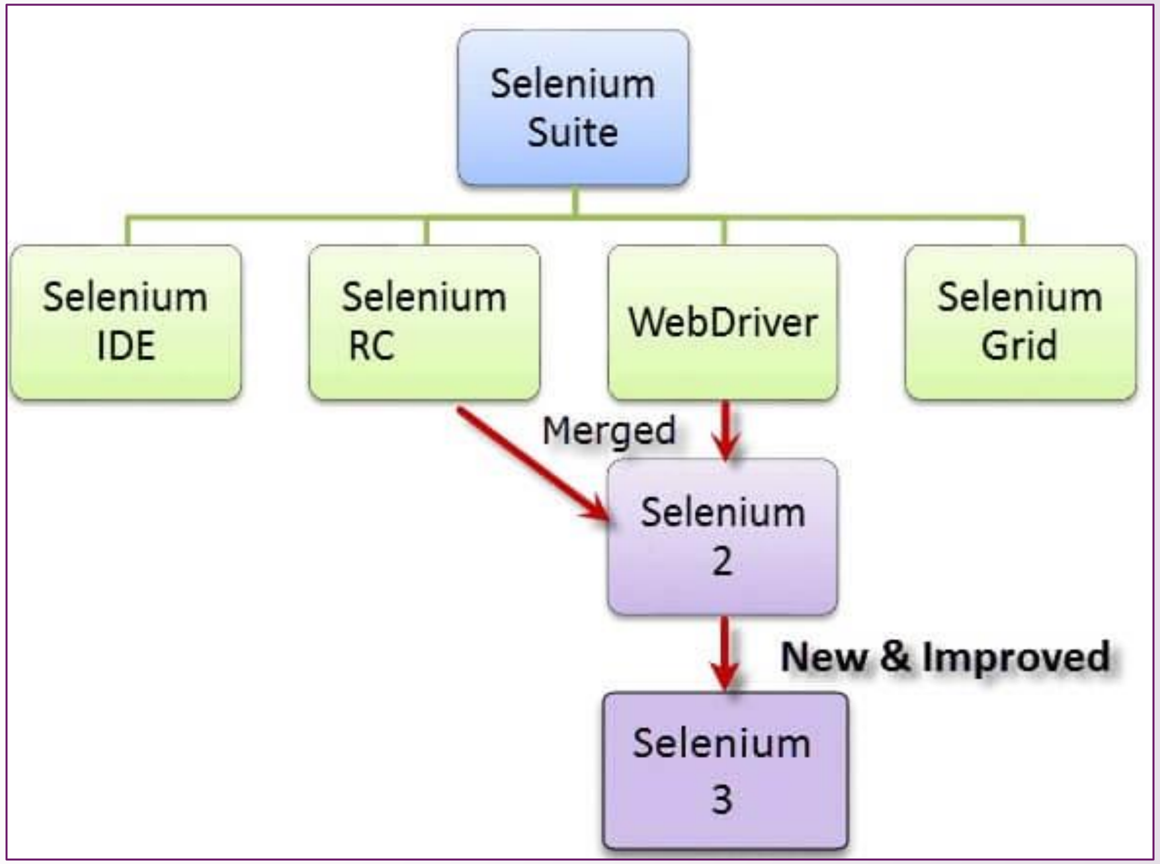
- Open source Web UI (User Interface) automation testing suite
- Jason Huggins in 2004 as an internal tool at Thought Works.
- Selenium supports automation across different browsers, platforms and programming languages.
- **Programming Languages:** C#, Java, Python, PHP, Ruby, Perl, and JavaScript
- **Operating Systems:** Android, iOS, Windows, Linux, Mac, Solaris.
- **Browsers:** Google Chrome, Mozilla Firefox, Internet Explorer, Edge, Opera, Safari, etc.



https://www.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Selenium-intro-1-new.jpg

Selenium Tool Suite

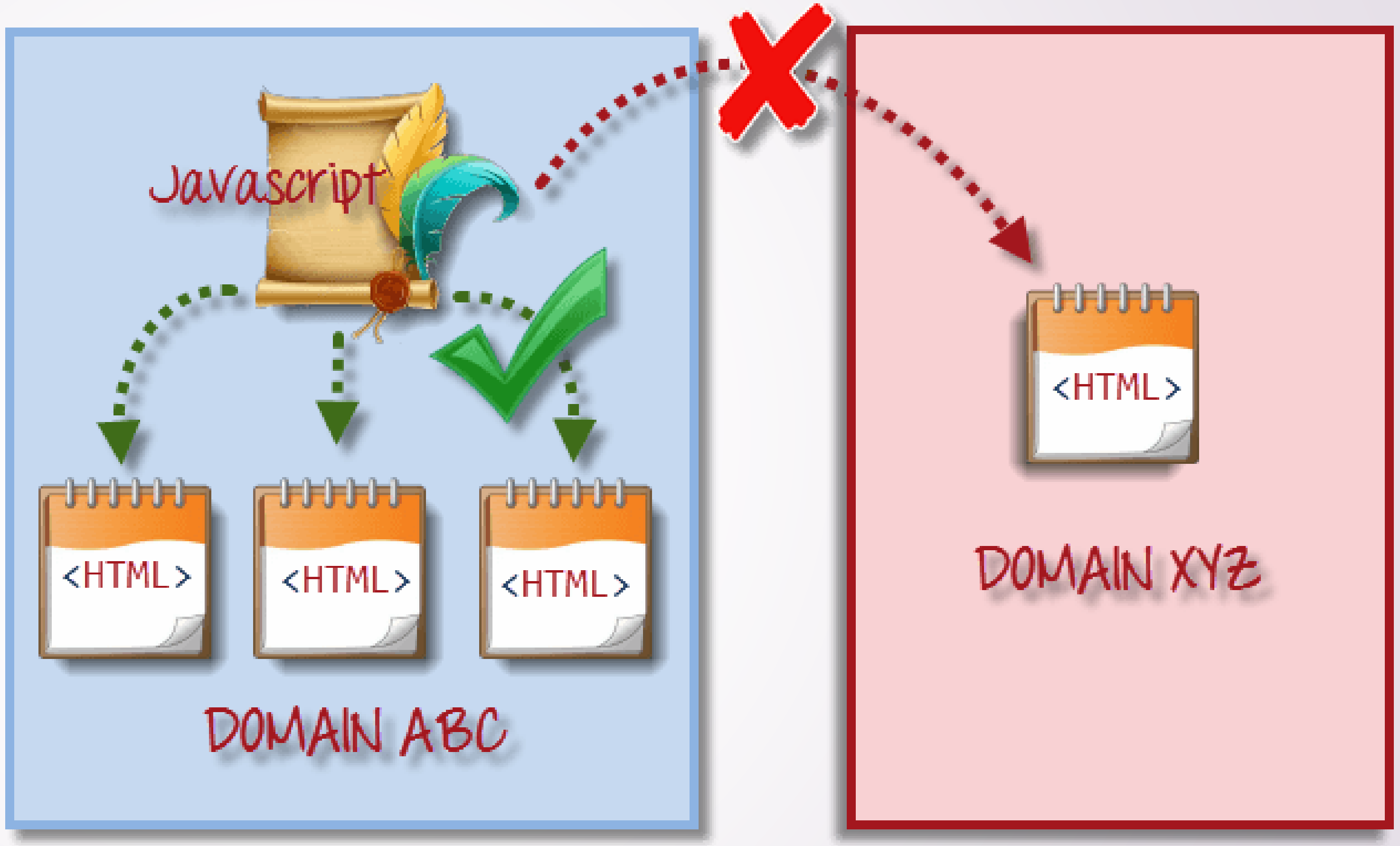
- Selenium Integrated Development Environment (IDE)
- Selenium Remote Control (RC)
- WebDriver
- Selenium Grid



https://cdn.guru99.com/images/SeleniumSuite.png

The Same Origin Policy Issue

This is the reason why prior to Selenium RC, testers needed to install local copies of both Selenium Core (a JavaScript program) and the web server containing the web application being tested so they would belong to the same domain



under Same Origin Policy, a Javascript program can only access pages on the same domain where it belongs. It cannot access pages from different domains

https://cdn.guru99.com/images/same_origin_policy.png

Birth of Selenium Remote Control (Selenium RC)

Unfortunately; testers using Selenium Core had to install the whole application under test and the web server on their own local computers because of the restrictions imposed by the **same origin policy.** So another ThoughtWork's engineer, **Paul Hammant**, decided to create a server that will act as an HTTP proxy to "trick" the browser into believing that Selenium Core and the web application being tested come from the same domain. This system became known as the **Selenium Remote Control** or **Selenium 1**.

Birth of Selenium Grid

Selenium Grid was developed by **Patrick Lightbody** to address the need of minimizing test execution times as much as possible. He initially called the system "**Hosted QA**." It was capable of capturing browser screenshots during significant stages, and also of **sending out Selenium commands to different machines simultaneously.**

Birth of Selenium IDE

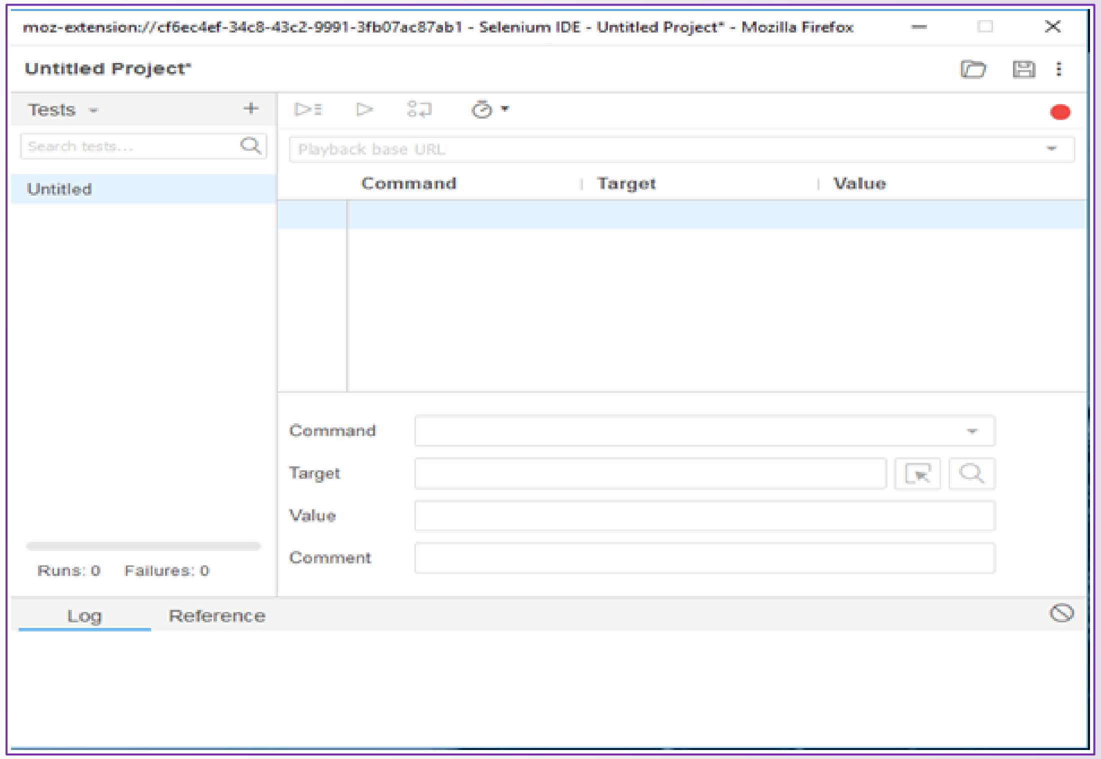
Shinya Kasatani of Japan created **Selenium IDE**, a Firefox extension that can automate the browser through a record-and-playback feature. He came up with this idea to further increase the speed in creating test cases. He donated Selenium IDE to the Selenium Project in **2006**.

Birth of WebDriver

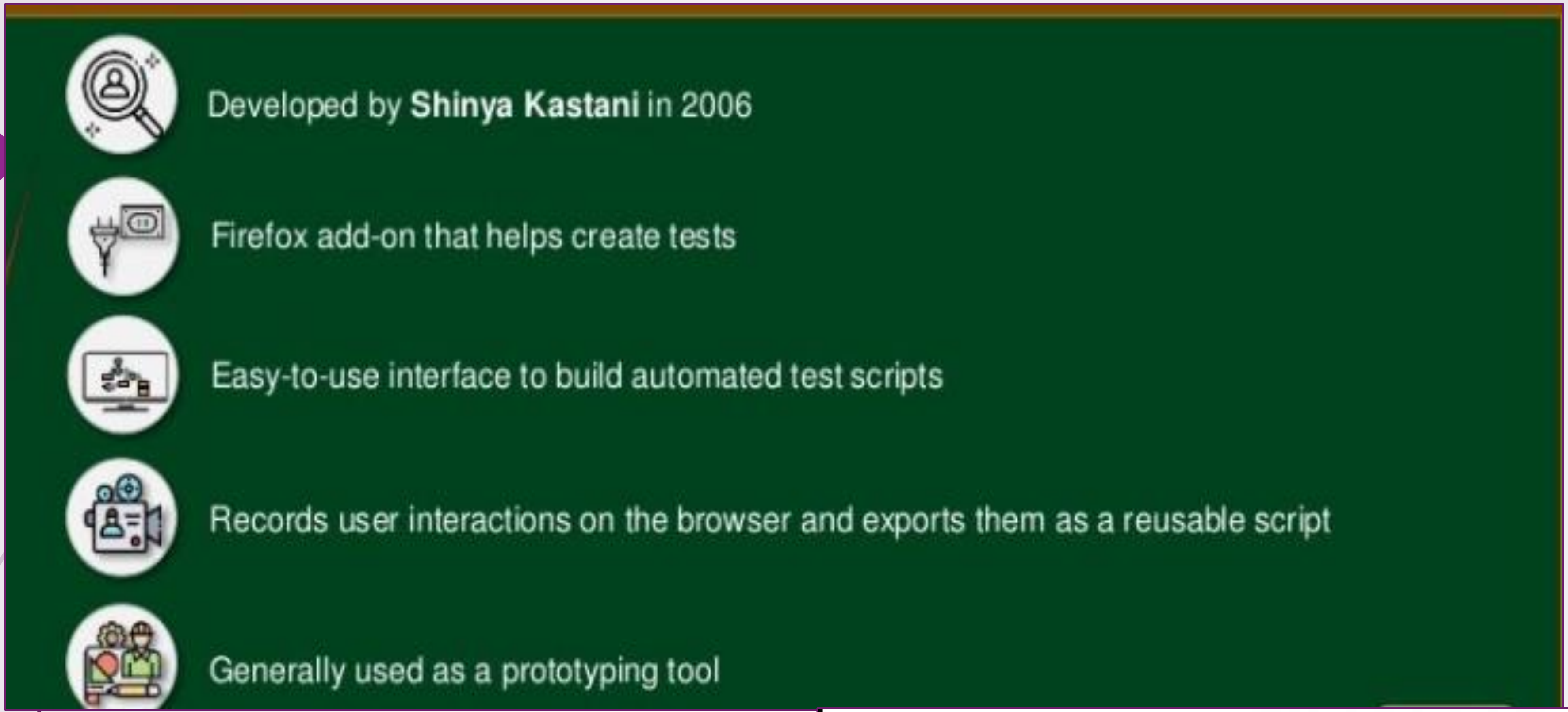
Simon Stewart created WebDriver circa **2006** when browsers and web applications were becoming more powerful and more restrictive with JavaScript programs like Selenium Core. **It was the first cross-platform testing framework that could control the browser from the OS level.**

Selenium Integrated Development Environment (IDE)

- Selenium IDE (Integrated Development Environment) is an open-source web automation testing tool under the Selenium Suite.
- Unlike Selenium WebDriver and RC, 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.



Note: Selenium IDE is available only as Mozilla Firefox and Chrome plug-in, which means you can't record your test cases on browsers other than Firefox and Chrome. The recorded test scripts can also be exported to programming languages like C#, Java, Ruby or Python.



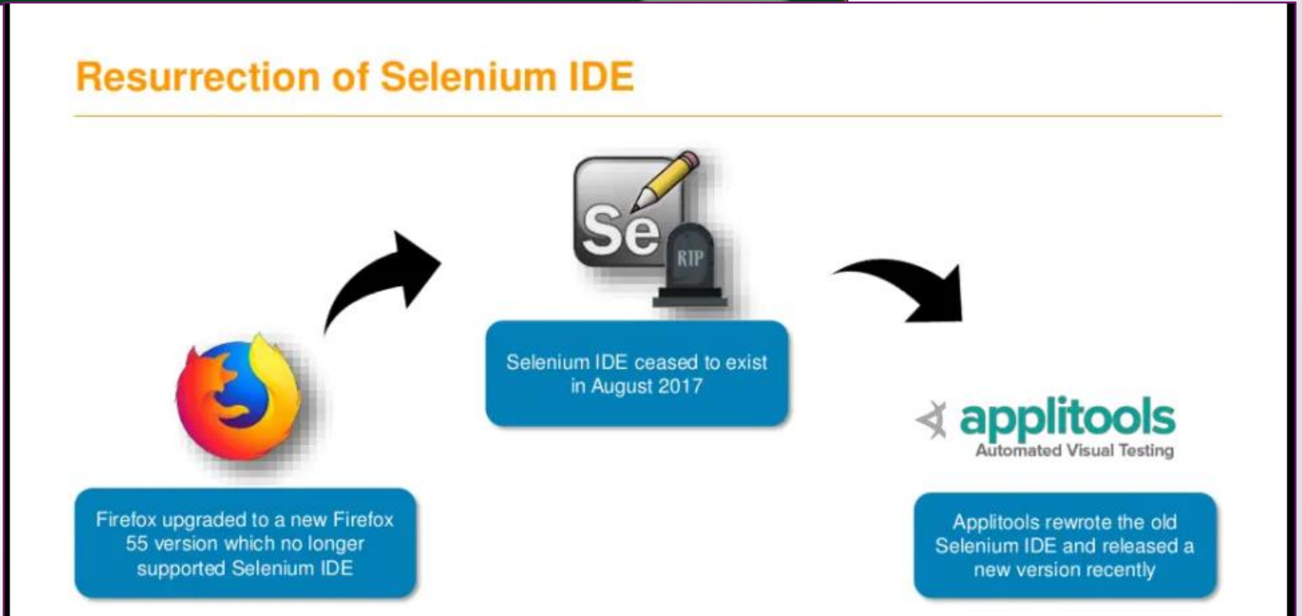
Developed by **Shinya Kastani** in 2006

Firefox add-on that helps create tests

Easy-to-use interface to build automated test scripts

Records user interactions on the browser and exports them as a reusable script

Generally used as a prototyping tool



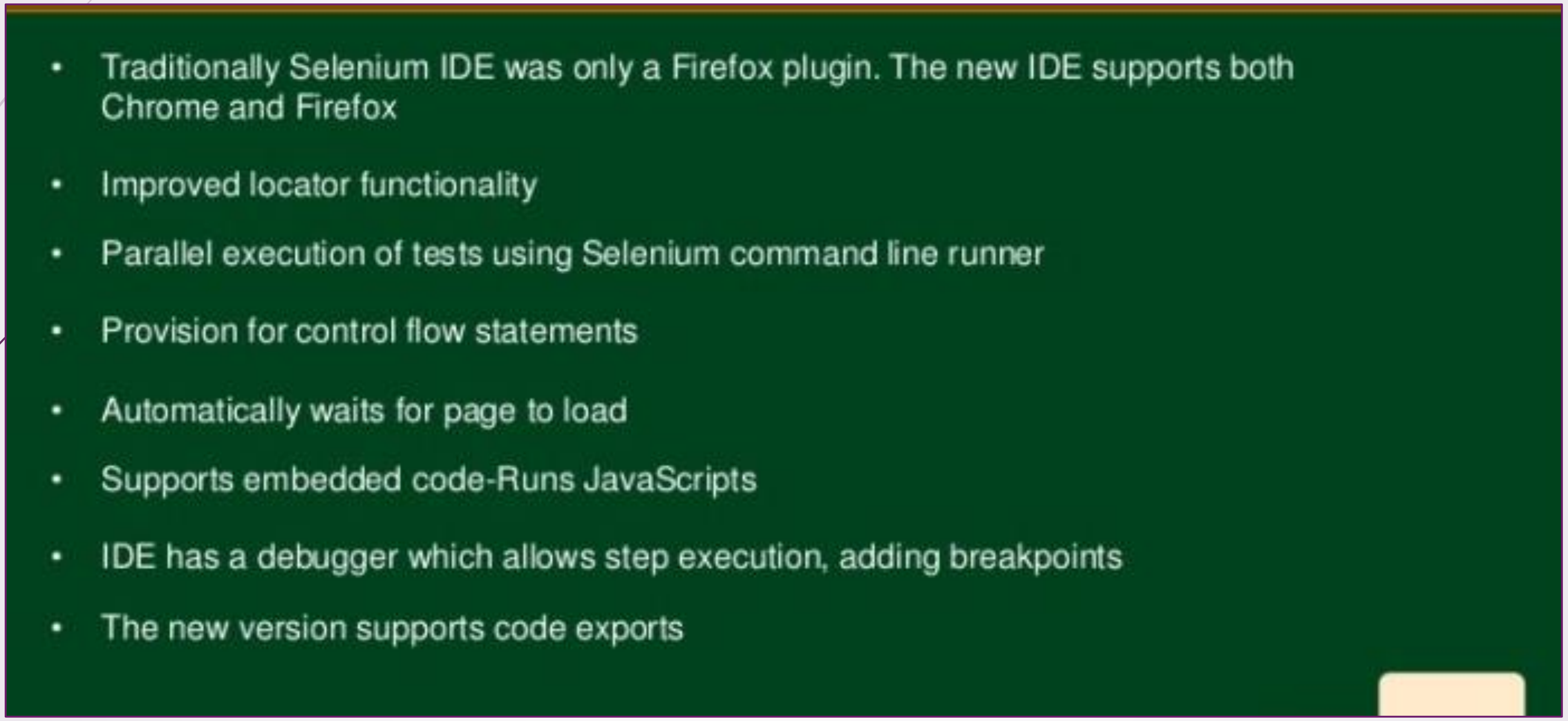
Resurrection of Selenium IDE

Firefox upgraded to a new Firefox 55 version which no longer supported Selenium IDE

Selenium IDE ceased to exist in August 2017

Applitools rewrote the old Selenium IDE and released a new version recently

Advancements with new Selenium IDE

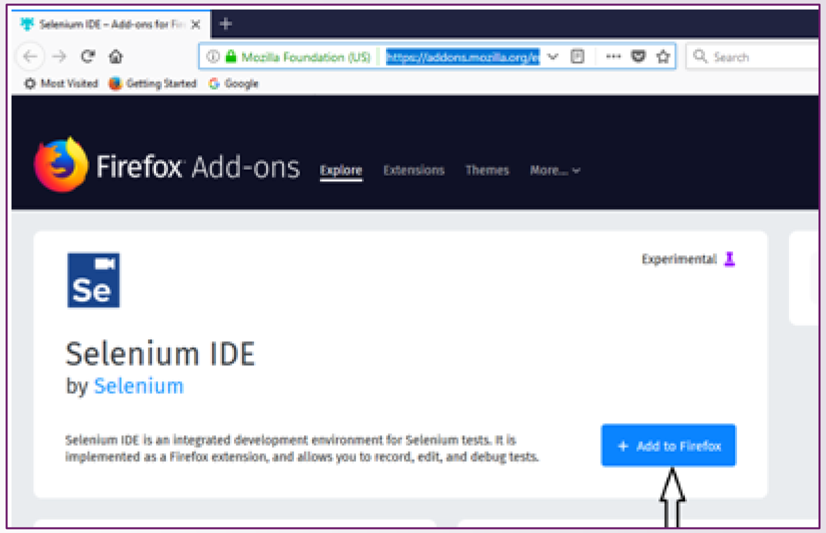


- Traditionally Selenium IDE was only a Firefox plugin. The new IDE supports both Chrome and Firefox
- Improved locator functionality
- Parallel execution of tests using Selenium command line runner
- Provision for control flow statements
- Automatically waits for page to load
- Supports embedded code-Runs JavaScripts
- IDE has a debugger which allows step execution, adding breakpoints
- The new version supports code exports

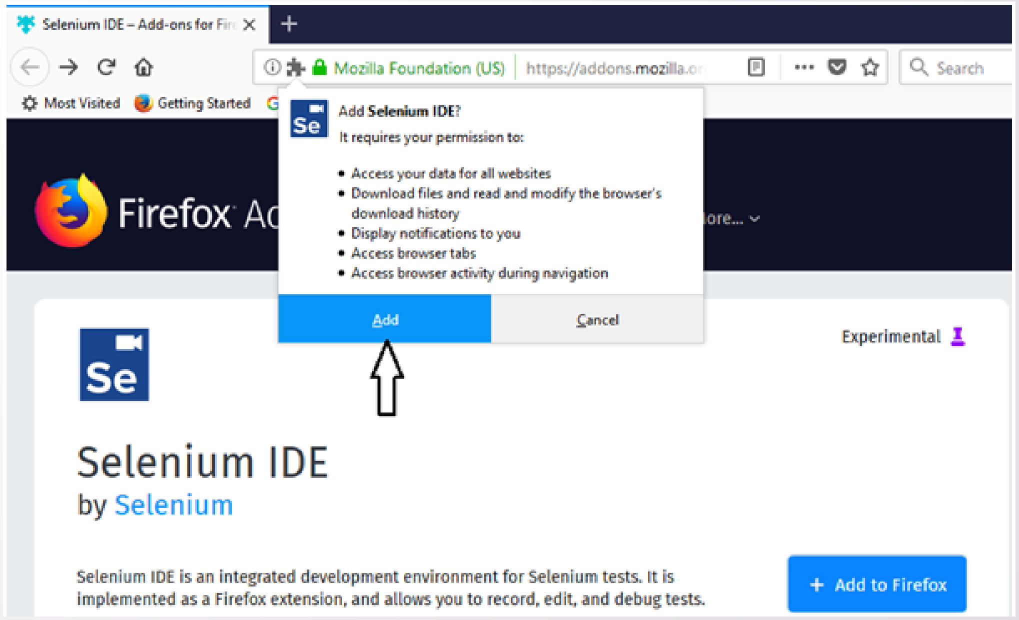
Selenium IDE-Installation - Firefox

- Launch Mozilla Firefox browser.
- Open URL: https://addons.mozilla.org/en-us/firefox/addon/selenium-ide/It will redirect you to the official add-on page of Firefox.
- Click on "Add to Firefox" button.
- A pop-up dialog box will be appeared asking you to add Selenium IDE as extension to your Firefox browser.
- Click on "Add" button.
- Restart you Firefox browser.
- Go to the top right corner on your Firefox browser and look for the Selenium IDE icon.

Note: Similar steps for chrome selenium extension: https://www.selenium.dev/selenium-ide/



Selenium IDE is an integrated development environment for Selenium tests. It is implemented as a Firefox extension, and allows you to record, edit, and debug tests.



Add Selenium IDE?

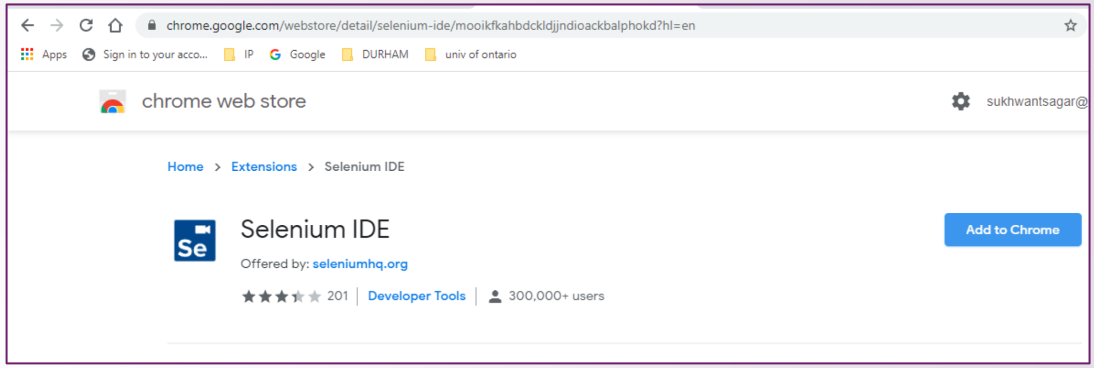
It requires your permission to:

- Access your data for all websites
- Download files and read and modify the browser's download history
- Display notifications to you
- Access browser tabs
- Access browser activity during navigation

Selenium IDE is an integrated development environment for Selenium tests. It is implemented as a Firefox extension, and allows you to record, edit, and debug tests.

Selenium IDE-Installation - Chrome

Note: Similar steps for chrome selenium extension: https://www.selenium.dev/selenium-ide/


chrome.google.com/webstore/detail/selenium-ide/mooikfkahbdckldjjndioackbalphokd?hl=en

Apps · Sign in to your acco... · IP · Google · DURHAM · univ of ontario

chrome web store · sukhwantsagar@

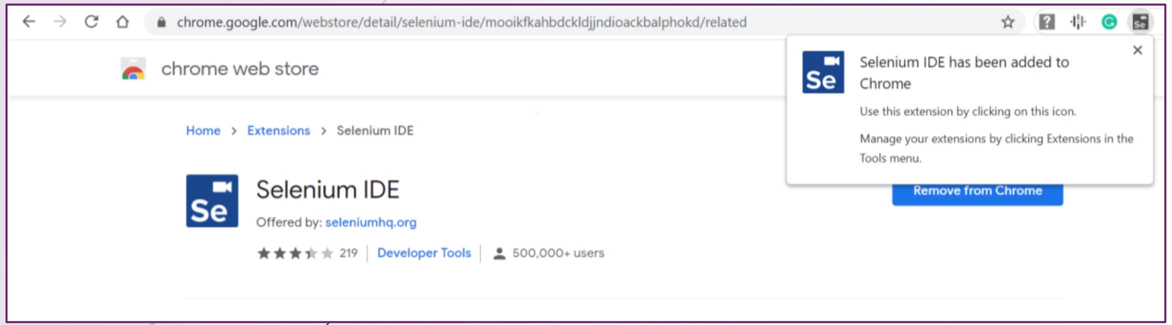
Home > Extensions > Selenium IDE

Selenium IDE

Offered by: seleniumhq.org

★★★★☆ 201 | Developer Tools | 300,000+ users

Add to Chrome


chrome.google.com/webstore/detail/selenium-ide/mooikfkahbdckldjjndioackbalphokd/related

chrome web store

Home > Extensions > Selenium IDE

Selenium IDE

Offered by: seleniumhq.org

★★★★☆ 219 | Developer Tools | 500,000+ users

Remove from Chrome

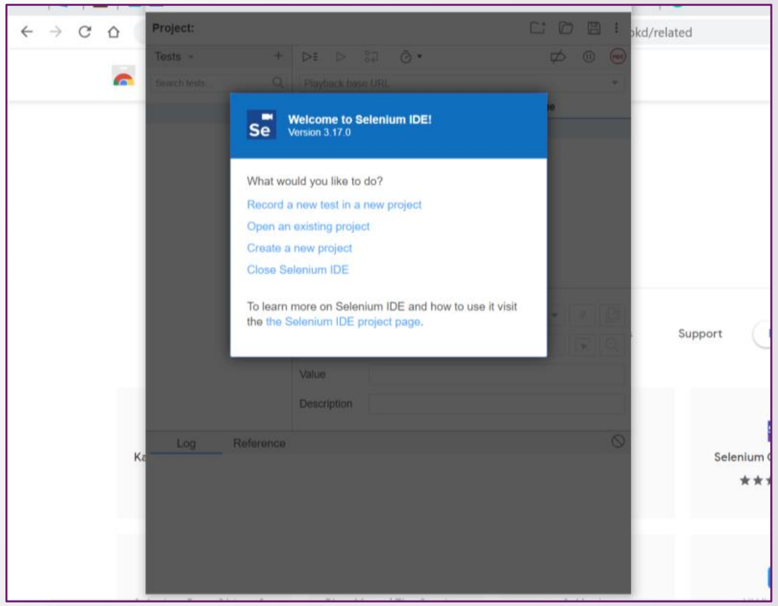
Selenium IDE has been added to Chrome

Use this extension by clicking on this icon.

Manage your extensions by clicking Extensions in the Tools menu.

Pin the selenium extension in the chrome toolbar by clicking on extensions icon

When you click on Selenium extension, it shows


Welcome to Selenium IDE!
Version 3.17.0

What would you like to do?

Record a new test in a new project

Open an existing project

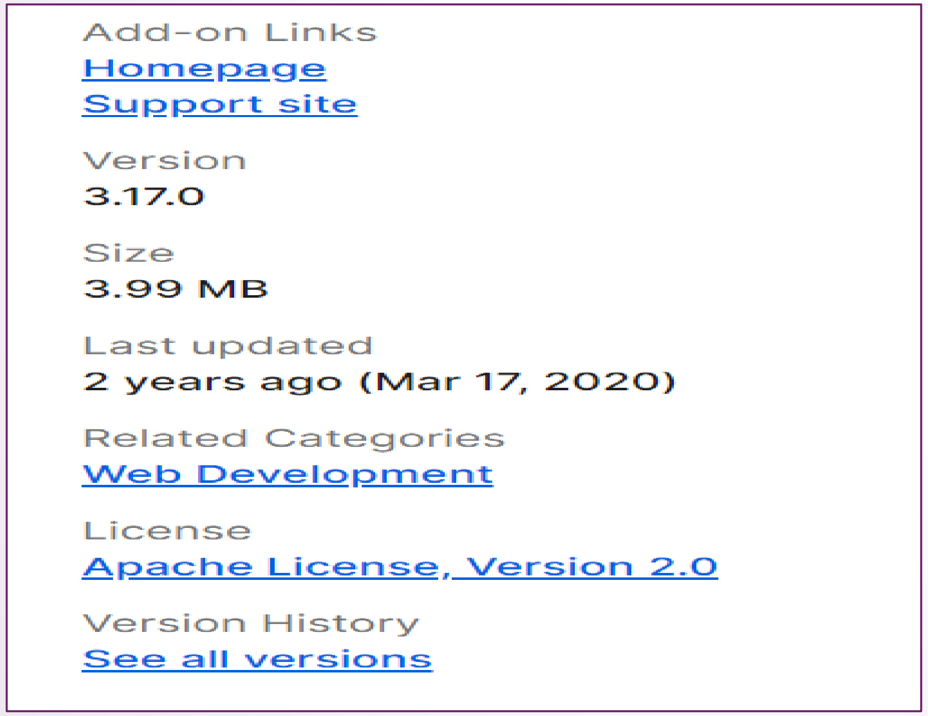
Create a new project

Close Selenium IDE

To learn more on Selenium IDE and how to use it visit the the Selenium IDE project page.

Selenium IDE Extensions

Firefox



Add-on Links
Homepage
Support site

Version
3.17.0

Size
3.99 MB

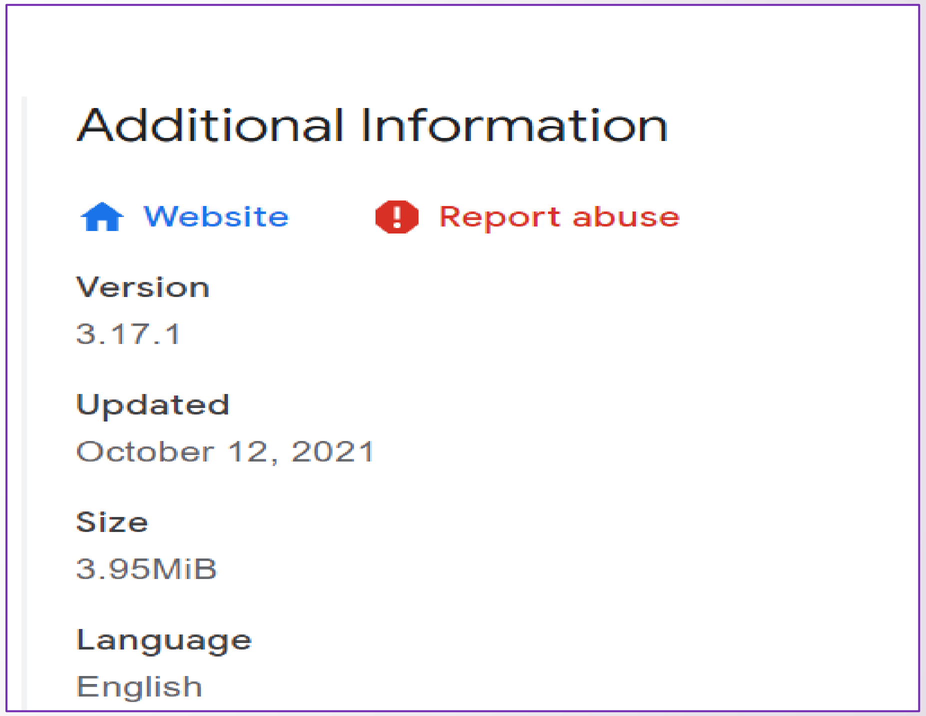
Last updated
2 years ago (Mar 17, 2020)

Related Categories
Web Development

License
Apache License, Version 2.0

Version History
See all versions

Google Chrome



Additional Information

Website   Report abuse

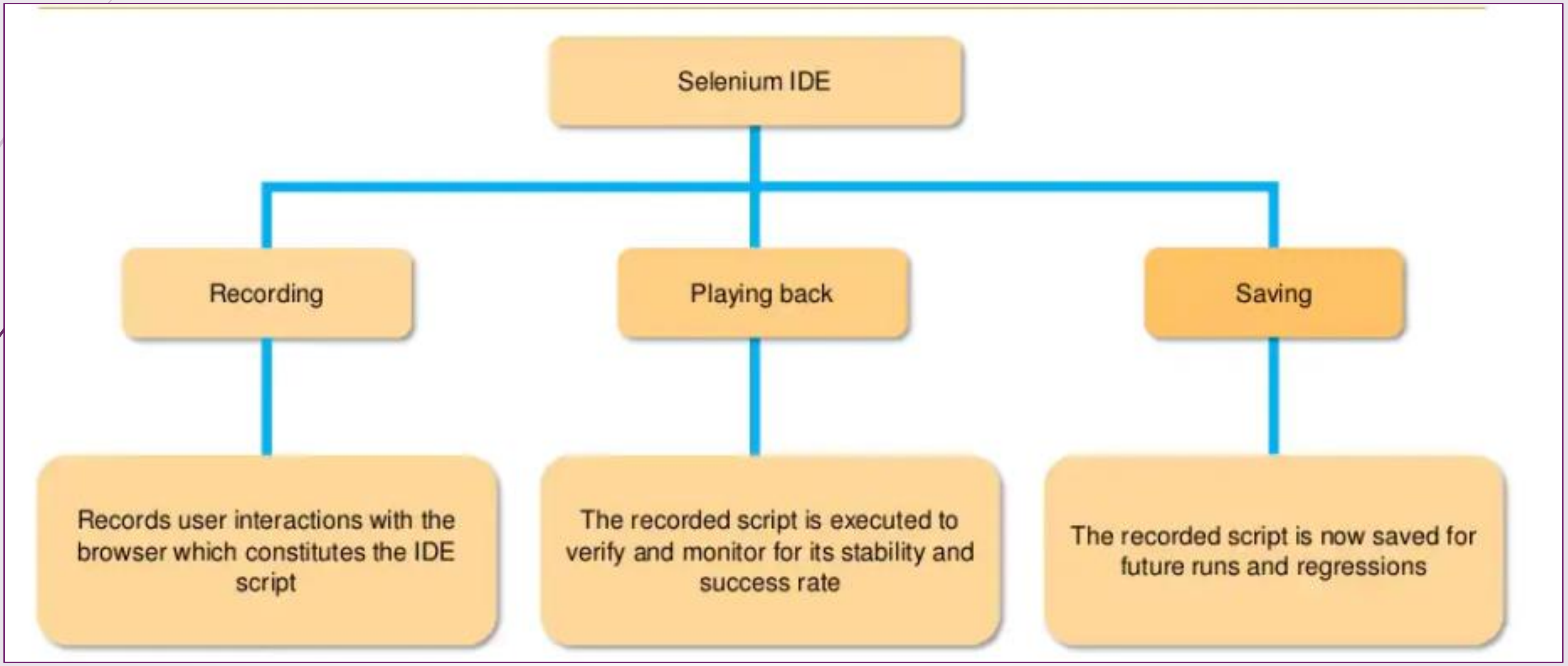
Version
3.17.1

Updated
October 12, 2021

Size
3.95MiB

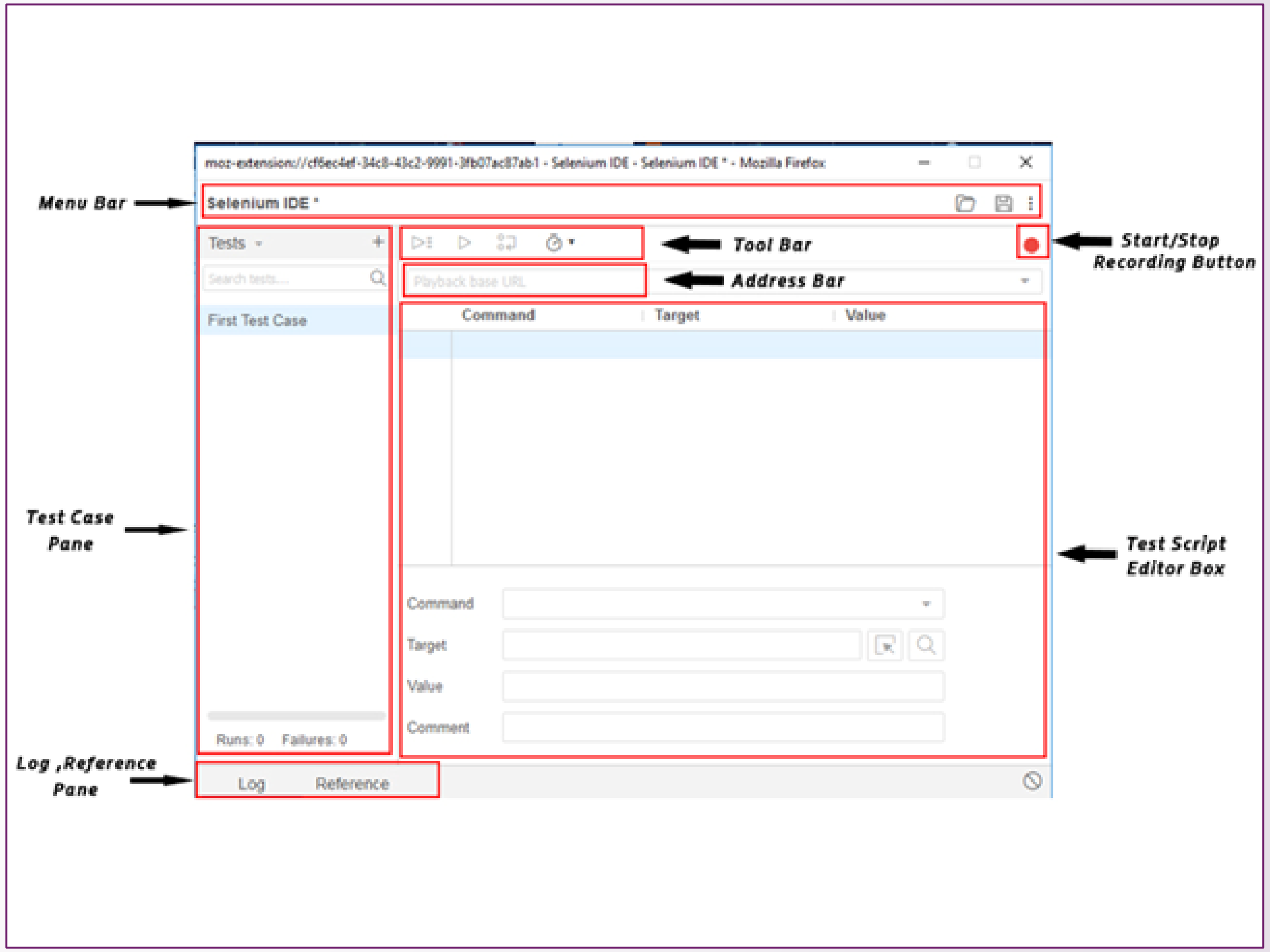
Language
English

Selenium IDE Working Principle

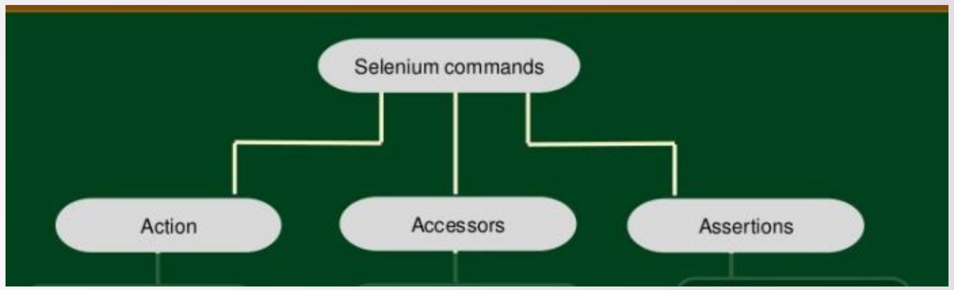

Selenium IDE
Recording
Playing back
Saving
Records user interactions with the browser which constitutes the IDE script
The recorded script is executed to verify and monitor for its stability and success rate
The recorded script is now saved for future runs and regressions

Selenium IDE-Features

- Menu Bar
- Tool Bar
- Address Bar
- Test Case Pane
- Test Script Editor Box
- Start/Stop Recording Button
- Log, Reference Pane



Selenium Commands

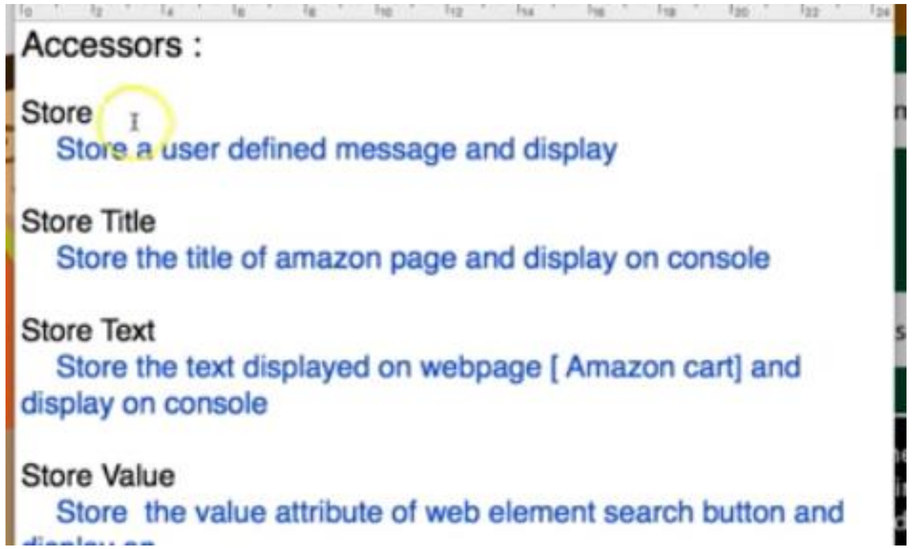


- **Actions** are those commands which interact directly with the application by either altering its state or by pouring some test data.
- **Accessors** are those commands which allow the user to store certain values to a user-defined variable. These stored values can be later on used to create assertions and verifications.
- **Assertions** are very similar to Accessors as they do not interact with the application directly. Assertions are used to verify the current state of the application with an expected state.
- **Forms of Assertions:**
  - **#1. assert** – the "assert" command makes sure that the test execution is terminated in case of failure.
  - **#2. verify** – the "verify" command lets the Selenium IDE carry on with the test script execution even if the verification is failed.
  - **#3. wait for** – the "waitFor" command waits for a certain condition to be met before executing the next test step. The conditions are like the page to be loaded, element to be present. It allows the test execution to proceed even if the condition is not met within the stipulated waiting period.

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.

Command/Syntax	Description
open (url)	It launches the desired URL in the specified browser and it accepts both relative and absolute URLs.
type (locator,value)	It sets the value of an input field, similar to user typing action.
typeKeys (locator,value)	This command simulates keystroke events on the specified element.
click (locator)	This command enables clicks on a link, button, checkbox or radio button.
clickAt (locator,coordString)	This command enables clicks on an element with the help of locator and co-ordinates
doubleClick (locator)	This command enables double clicks on a webelement based on the specified element.
focus (locator)	It moves the focus to the specified element
highlight (locator)	It changes the background color of the specified element to yellow to highlight is useful for debugging purposes.
close()	This command simulates the user clicking the "close" button in the title bar of a popup window or tab.
store (expression,variableName)	This command specifies the name of a variable in which the result is to be stored and expression is the value to store
waitForCondition (script,timeout)	This command executes the specified JavaScript snippet repeatedly until it evaluates to "true".



Accessors :

Store
Store a user defined message and display

Store Title
Store the title of amazon page and display on console

Store Text
Store the text displayed on webpage [Amazon cart] and display on console

Store Value
Store the value attribute of web element search button and

Command/Syntax	Description
storeTitle (variableName)	This command gets the title of the current page.
storeText (locator, variableName)	This command 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.
storeLocation (variableName)	This command gets the absolute URL of the current page.
storeElementIndex (locator, variableName)	This command gets the relative index of an element to its parent (starting from 0).
storeBodyText (variableName)	This command gets the entire text of the page.
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.

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.

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.

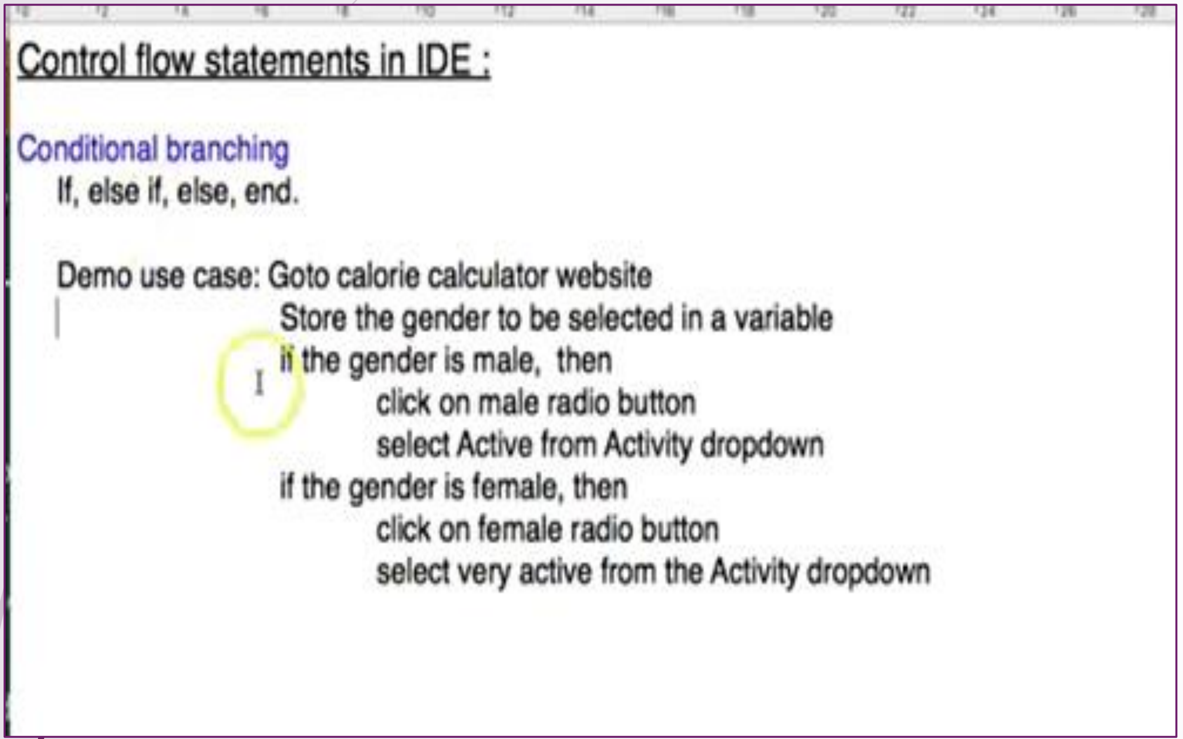
COMMAND/SYNTAX	DESCRIPTION
verifySelected(selectLocator, optionLocator)	This command verifies that the selected option of a drop-down satisfies the optionSpecifier.
verifyAlert (pattern)	This command verifies the alert text; used with accessorstoreAlert.
verifyAllButtons (pattern)	This command verifies the button which is used withaccessorstoreAllButtons.
verifyAllLinks (pattern)	This command verifies all links; used with the accessorstoreAllLinks.
verifyBodyText(pattern)	This command verifies the body text; used with the accessorstoreBodyText.
verifyAttribute(attributeLocator, pattern)	This command verifies an attribute of an element; used with the accessorstoreAttribute.
waitForErrorOnNext (message)	This command enables Waits for error; used with the accessorassertErrorOnNext.
waitForAlert (pattern)	This command enables waits for the alert; used with the accessorstoreAlert.
verifyAllWindowIds (pattern)	This command verifies the window id; used with the accessorstoreAllWindowIds.

Some of the commonly commands used

Command	**Description**	**#Arguments**
open	Opens a specified URL in the browser.	1
assertTitle, VerifyTitle	Returns the current page title and compares it with the specified title	1
assertElementPresent, verifyElementPresent	Verify / Asserts the presence of an element on a web page.	1
assertTextPresent, verifyTextPresent	Verify / Asserts the presence of a text within the web page.	1
type, typeKeys, sendKeys	Enters a value (String) in the specified web element.	2
Click, clickAt, clickAndWait	Clicks on a specified web element within a web page.	1
waitForPageToLoad	Sleeps the execution and waits until the page is loaded completely.	1
waitForElementPresent	Sleeps the execution and waits until the specified element is present	1
chooseOkOnNextConfirmation, chooseCancelOnNextConfirmation	Click on "OK" or "Cancel" button when next confirmation box appears.	0

DEMOS by Faculty

Control Flow statements

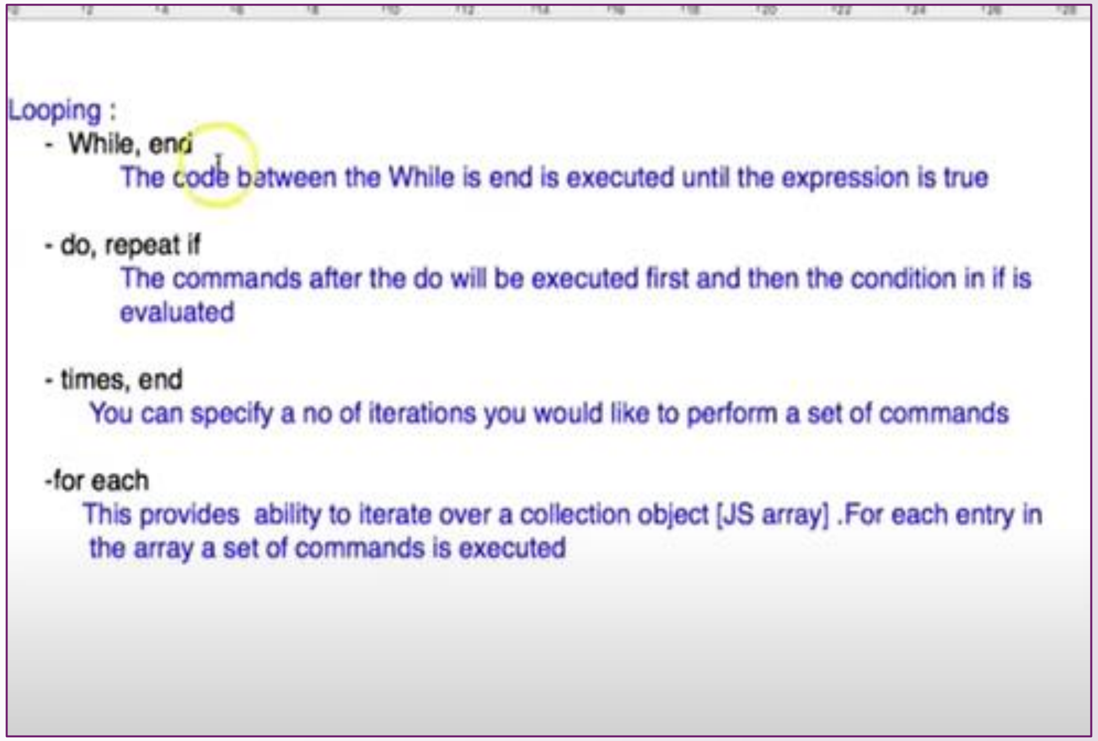


Control flow statements in IDE :

Conditional branching

If, else if, else, end.

```
Demo use case: Goto calorie calculator website
            Store the gender to be selected in a variable
            if the gender is male,  then
                click on male radio button
                select Active from Activity dropdown
            if the gender is female, then
                click on female radio button
                select very active from the Activity dropdown
```



Looping :

- While, end
  The code between the While is end is executed until the expression is true
- do, repeat if
  The commands after the do will be executed first and then the condition in if is evaluated
- times, end
  You can specify a no of iterations you would like to perform a set of commands
- for each
  This provides ability to iterate over a collection object [JS array] .For each entry in the array a set of commands is executed

Limitations of Selenium IDE

- Not suitable for testing extensive data
- Connections with the database can not be tested
- Cannot handle the dynamic part of web-based applications
- Does not support capturing of screenshots on test failures
- No feature available for generating result reports

References

- https://www.seleniumhq.org
- https://www.javatpoint.com/selenium-basic-terminology