Automated testing tool similarities and differences

IT814: Software Quality Assurance

Workshop presentation

Aparna Dasgupta, Niels Hansen, Annika Kuhnke, Mark Province

07/21/2021

ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
SCHOOL OF ENGINEERING

Organization

- Introduction
- JUnit
- Selenium
- Jenkins
- Similarities and differences
- Conclusions

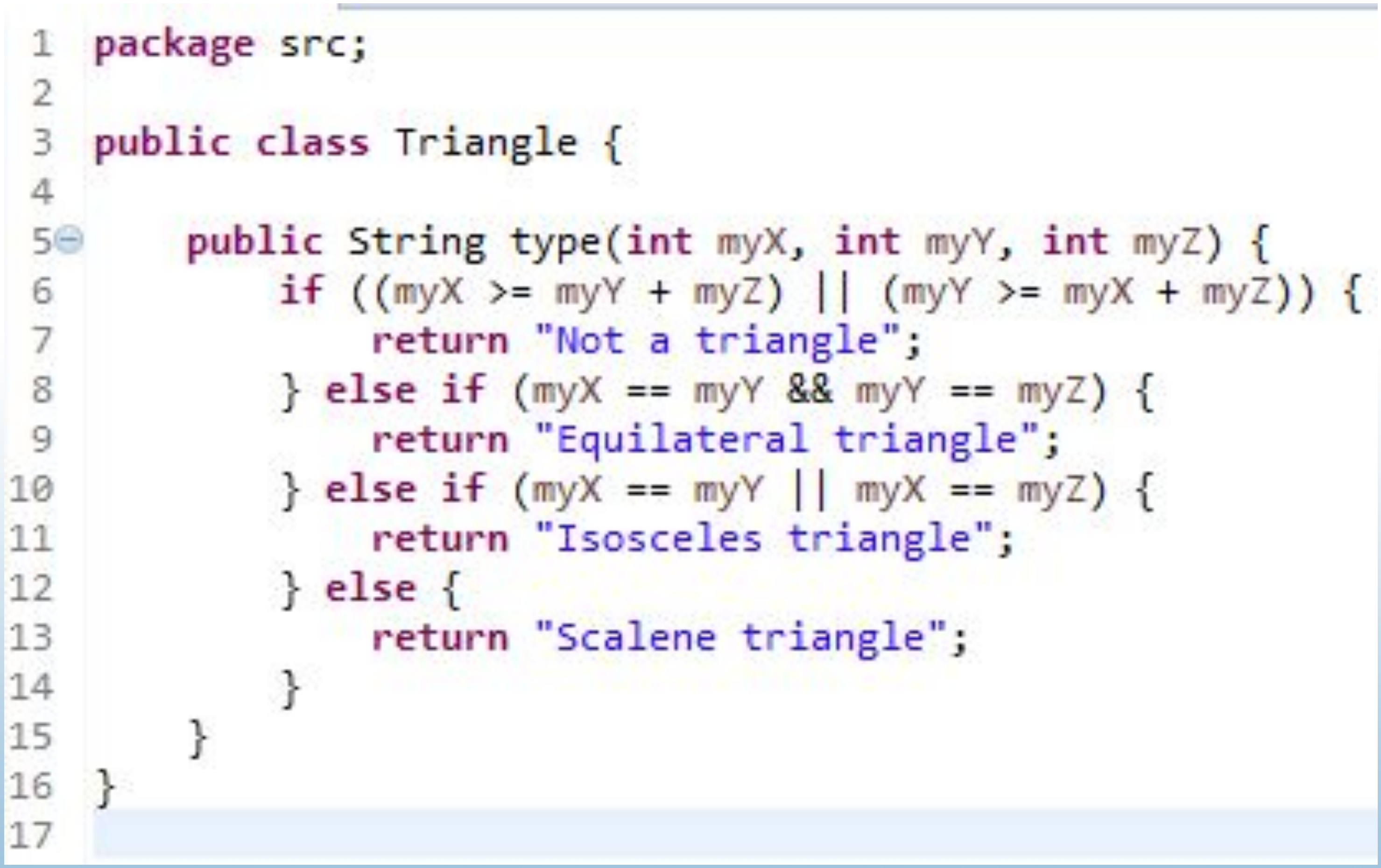
Resources used

- Bechtold, S., Brannen, S., Link, J., Merdes, M., Philipp, M., de Rancourt, J., Stein, C. (12 April 2021). JUnit 5 User Guide. JUnit.org. https://junit.org/junit5/docs/current/user-guide/
- JUnit - Quick Guide. TutorialsPoint. https://www.tutorialspoint.com/junit/junit_quick_guide.htm
- JUnit - Overview. TutorialsPoint. https://www.tutorialspoint.com/junit/junit_overview.htm
- The Selenium Browser Automation Project. Selenium.dev. https://www.selenium.dev/documentation/en/
- Edureka. (22 March 2020). Selenium Full Course - Learn Selenium in 12 Hours | Selenium Tutorial For Beginners | Edureka. Youtube. https://www.youtube.com/watch?v=FRn5J31eAMw
- Selenium - IDE. TutorialsPoint. https://www.tutorialspoint.com/selenium/selenium_ide.htm
- Jenkins User Documentation. Jenkins.io. https://www.jenkins.io/doc/
- Shrikanth, B. (17 December 2020). Jenkins for Test Automation. BrowserStack. https://www.browserstack.com/guide/jenkins-for-test-automation

Introduction - Automated vs manual testing

Automated	Manual
• Done using tools and scripts • More testing in less time: greater efficiency • Most tasks are automatable, including real user simulations • Easy to ensure greater test coverage	• Done by hand • Time-consuming and less efficient • Entirely manual tasks • Difficult to ensure sufficient test coverage

Introduction - Our case study

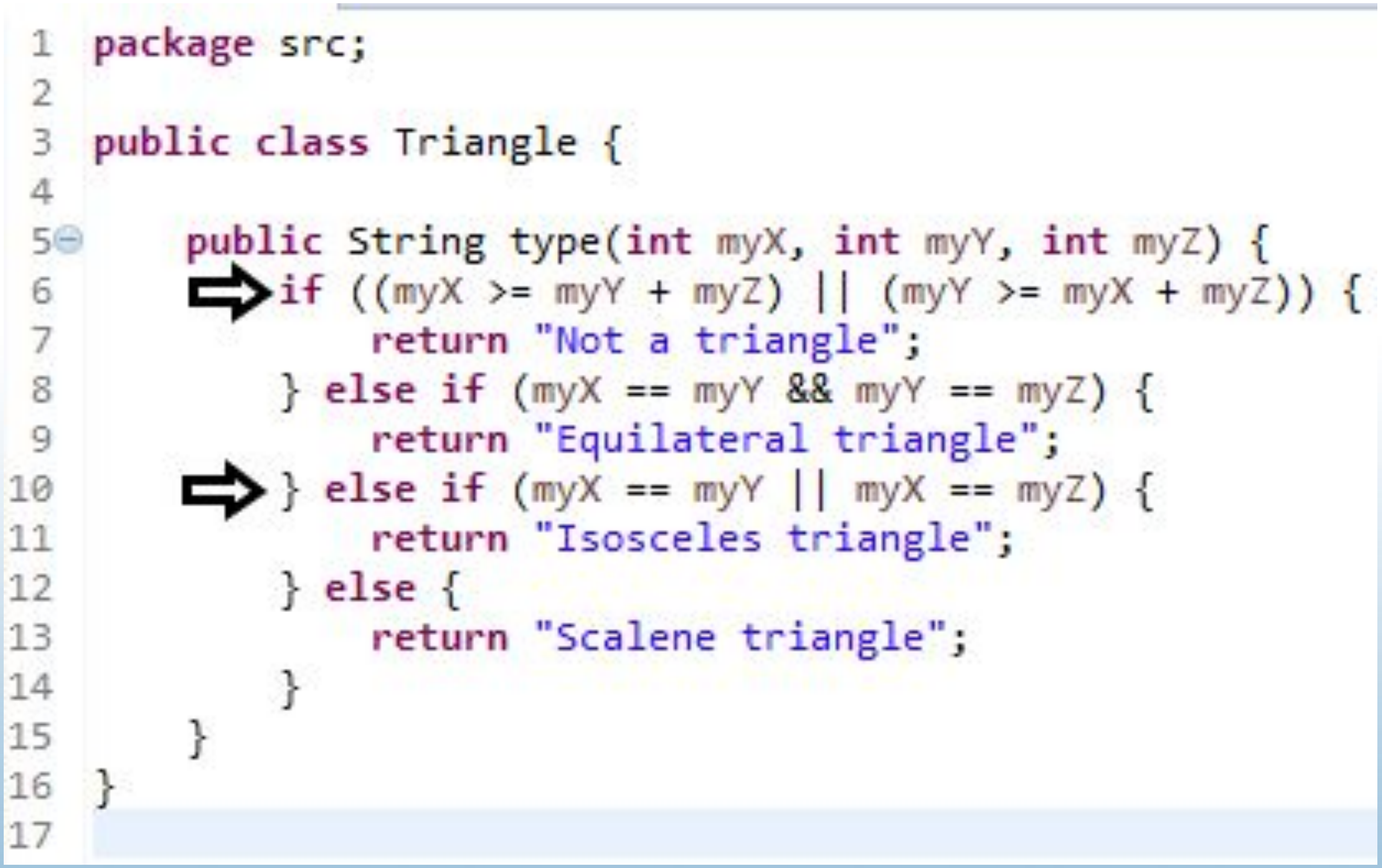


```
package src;

public class Triangle {

    public String type(int myX, int myY, int myZ) {
        if ((myX >= myY + myZ) || (myY >= myX + myZ)) {
            return "Not a triangle";
        } else if (myX == myY && myY == myZ) {
            return "Equilateral triangle";
        } else if (myX == myY || myX == myZ) {
            return "Isosceles triangle";
        } else {
            return "Scalene triangle";
        }
    }
}
```

Introduction - Our case study



```
package src;

public class Triangle {

    public String type(int myX, int myY, int myZ) {
        if ((myX >= myY + myZ) || (myY >= myX + myZ)) {
            return "Not a triangle";
        } else if (myX == myY && myY == myZ) {
            return "Equilateral triangle";
        } else if (myX == myY || myX == myZ) {
            return "Isosceles triangle";
        } else {
            return "Scalene triangle";
        }
    }
}
```

JUnit

- What is JUnit?
- JUnit history
- Regression-testing framework
- Features
- How JUnit works
- Annotations
- Display names
- Assertions
- Assumptions
- Conditional test execution
- Dynamic tests
- Test suites
- Best practices
- Demonstration

What is JUnit?

- Unit testing framework for the Java programming language
- Originally written by Erich Gamma and Kent Beck
- Part of the xUnit family

JUnit history

- Beck developed the first xUnit, SUnit, in the mid-90’s
- Beck and Gamma developed JUnit on a flight
- JUnit has become the standard tool for test-driven development in Java
- xUnit tools have since been developed for many other languages

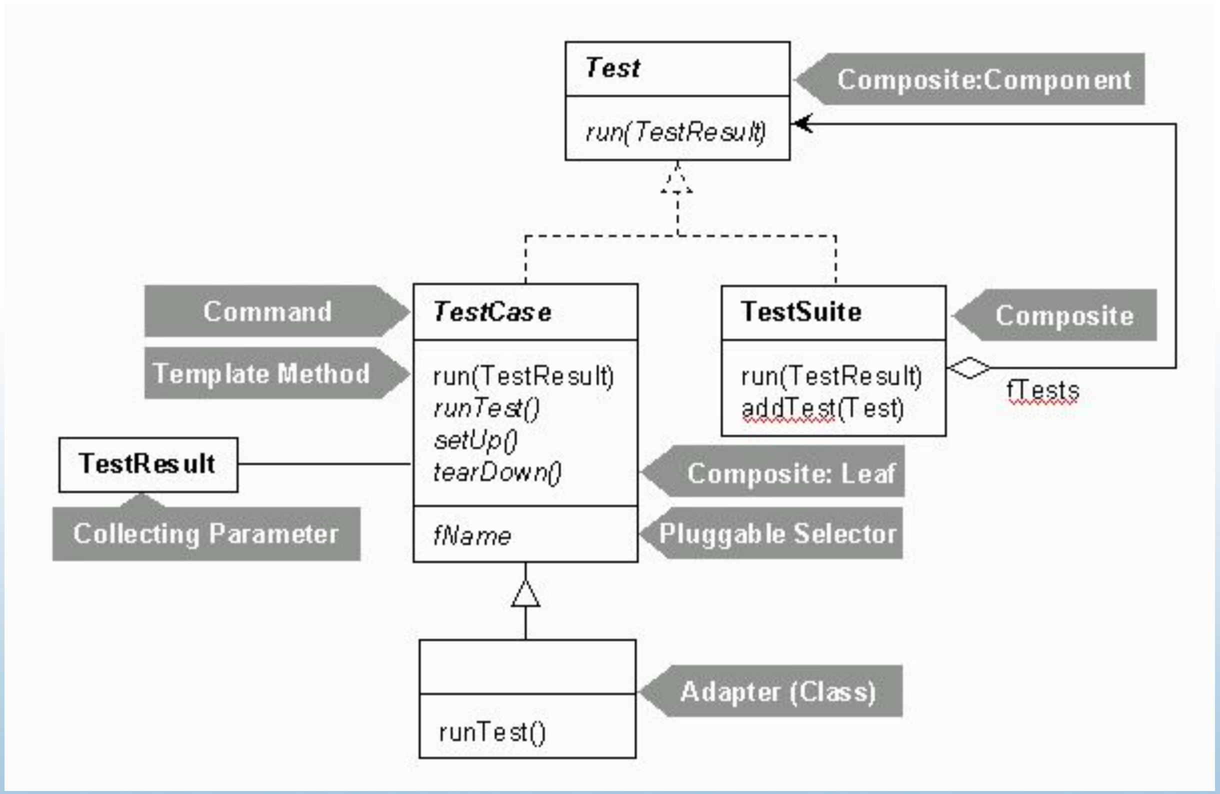
Regression-testing framework

- JUnit is a regression-testing framework.
- It creates a relationship between development and testing.
- Modifications in the code will not break your system without your knowledge.

Features

- Asserts
- Test setup and teardown
- Exception testing
- Test suites
- Parameterized testing
- Rules
- Integration with popular build systems

How JUnit works



Annotations

- Tags applied to methods or classes
- Tell JUnit when to run a test method
- Predefined and can be implemented directly
- Include Test, BeforeEach, AfterEach, BeforeAll, AfterAll, and Disabled

Display names

- Declare custom display names with special characters
- Displayed in test reports
- Default generators:
  - Standard
  - Simple
  - ReplaceUnderscores
  - IndicativeSentences

Assertions

- Contains methods or statements used to write tests
- Important methods include
  - assertEquals(boolean expected, boolean actual)
  - assertFalse(boolean condition)
  - assertTrue(boolean condition)
  - assertNotNull(Object object)
  - assertNull(Object object)
  - assertNotSame(boolean expected, boolean actual)
  - assertSame(boolean expected, boolean actual)
  - fail()
  - fail(String message)
  - assertArrayEquals(array expected, array actual)
  - assertArrayEquals(String message, array expected, array actual)

Assumptions

- Support conditional test execution
- Important methods include:
  - assumeFalse(boolean assumption)
  - assumeTrue(boolean assumption)
  - assumingThat(boolean assumption, executable)

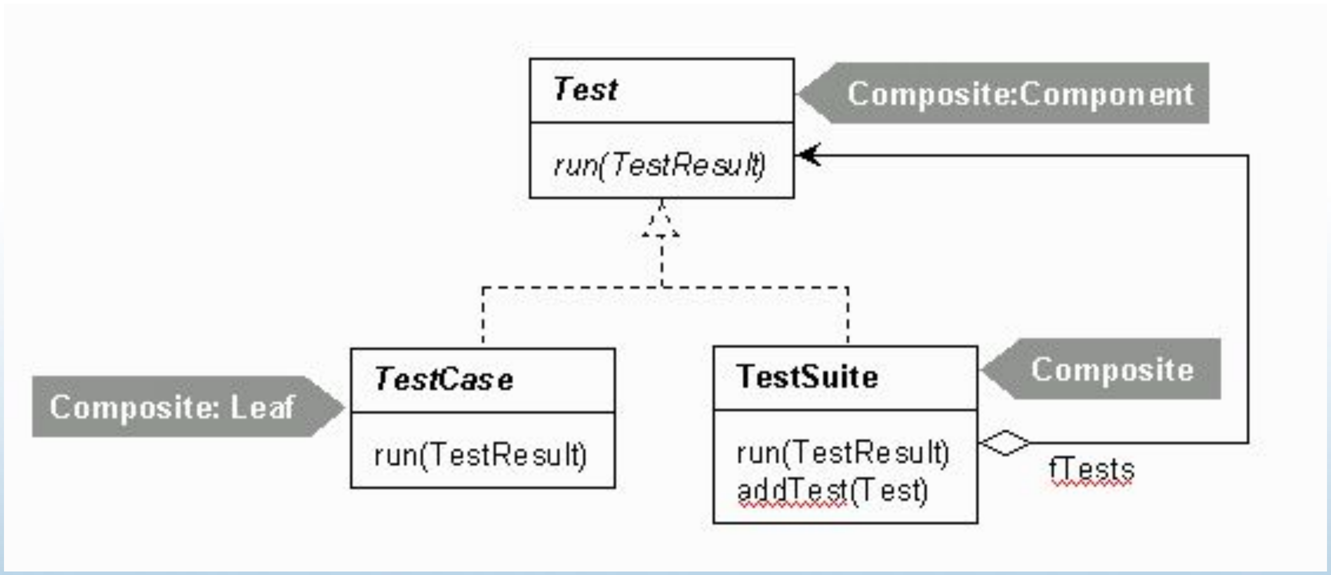
Conditional test execution

- Enable or disable containers and tests declaratively
- Types
  - Operating system conditions
  - Java runtime environment conditions
  - System property conditions
  - Environment variable conditions
  - Custom conditions

Dynamic tests

- Generated at runtime by a factory method
- Executed differently than those annotated with Test annotation
- Do not support lifecycle callbacks

Test suites


Test
run(TestResult)
Composite:Component
TestCase
run(TestResult)
Composite: Leaf
TestSuite
run(TestResult)
addTest(Test)
Composite
fTests

Best practices

- Independent tests
- Strongest assertion possible
- Separate test and production code
- Use timeouts
- Naming conventions

JUnit - Demonstration

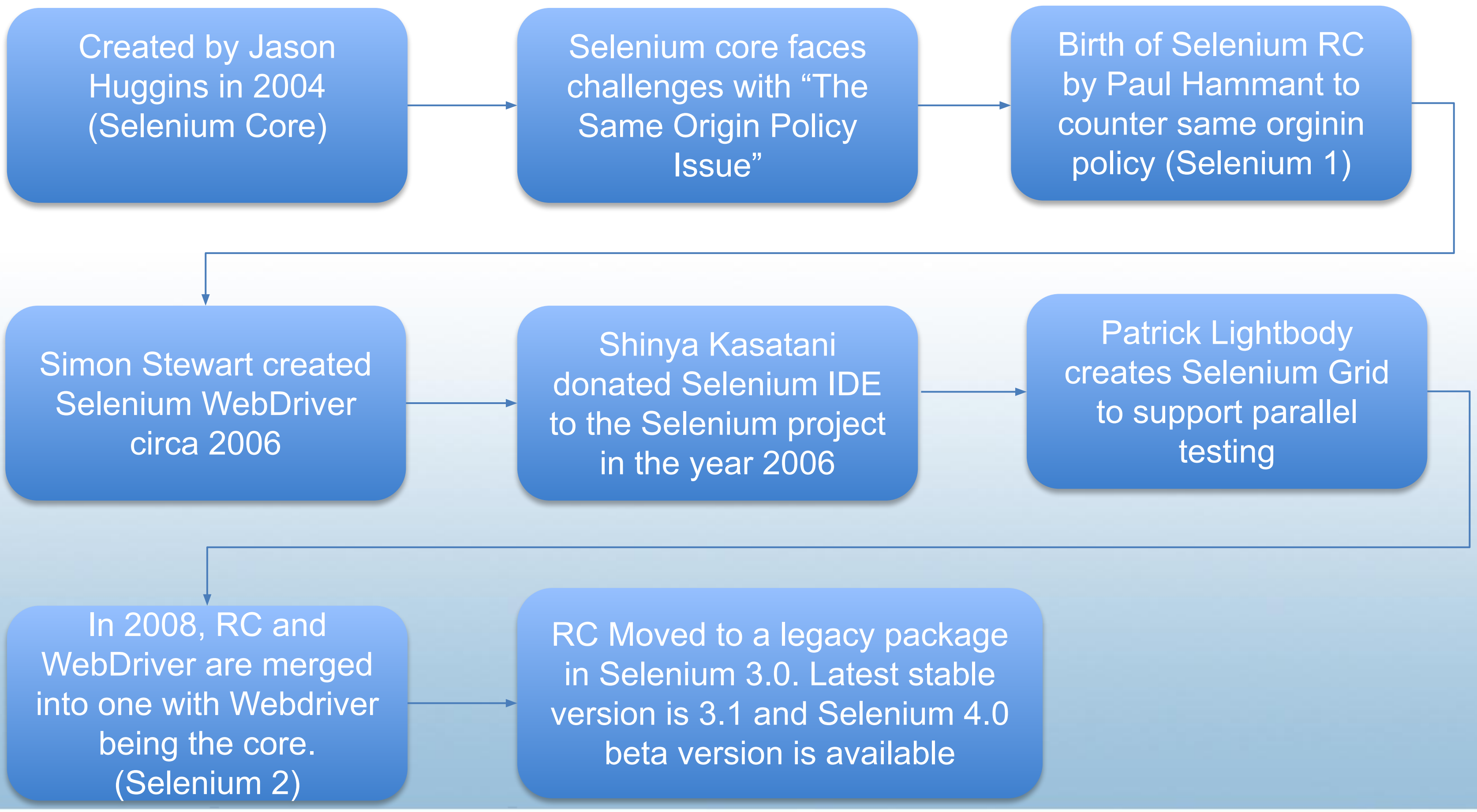
Selenium

- What is Selenium?
- Brief history
- Introduction to Selenium suite
- Features and benefits
- Selenium IDE
- Selenium RC
- Selenium WebDriver
- Selenium Grid
- Demonstration

What is Selenium?

- Selenium is a free (open source) automation testing framework
- Supports multiple programming languages
- Supports multiple web browsers
- Supports multiple operating systems

History



Created by Jason Huggins in 2004 (Selenium Core)

Selenium core faces challenges with “The Same Origin Policy Issue”

Birth of Selenium RC by Paul Hammant to counter same orginin policy (Selenium 1)

Simon Stewart created Selenium WebDriver circa 2006

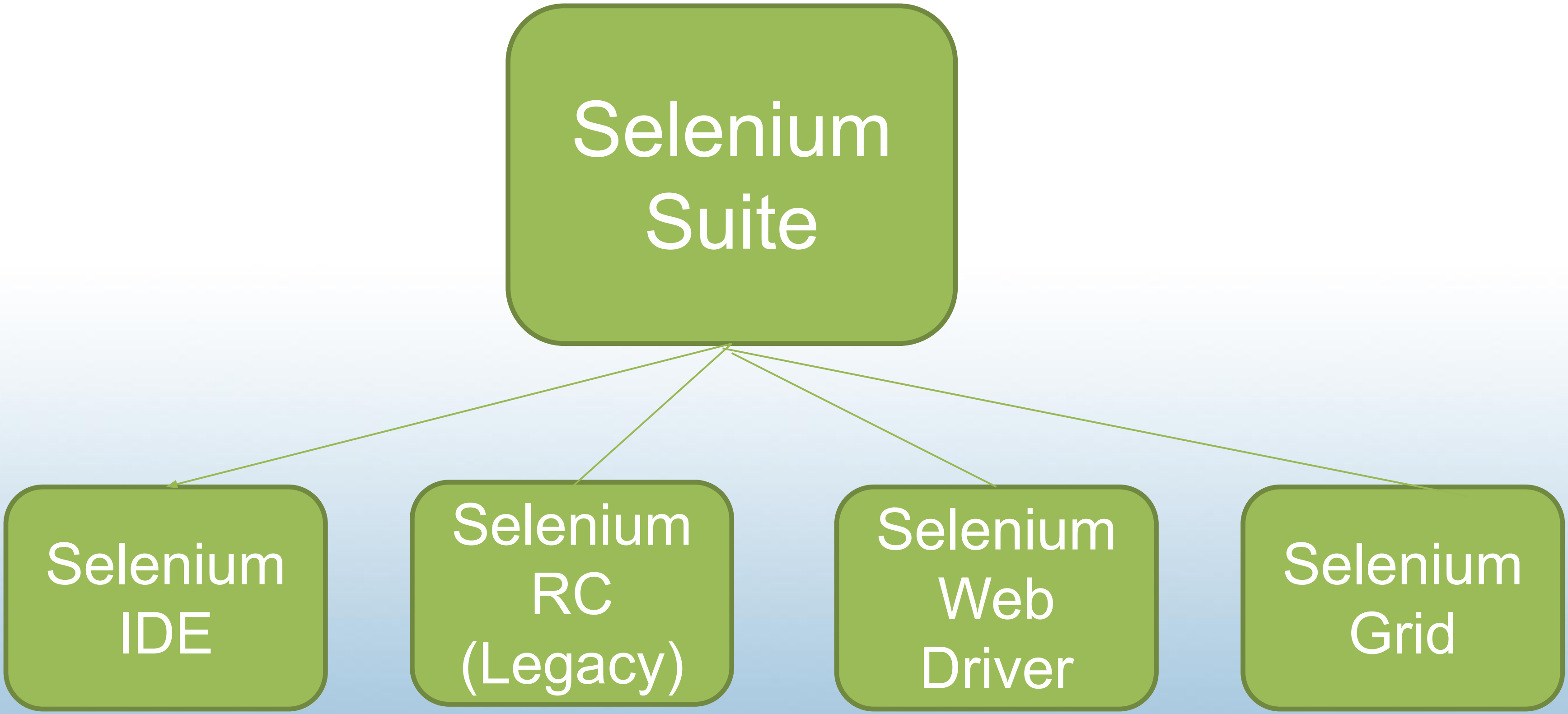
Shinya Kasatani donated Selenium IDE to the Selenium project in the year 2006

Patrick Lightbody creates Selenium Grid to support parallel testing

In 2008, RC and WebDriver are merged into one with Webdriver being the core. (Selenium 2)

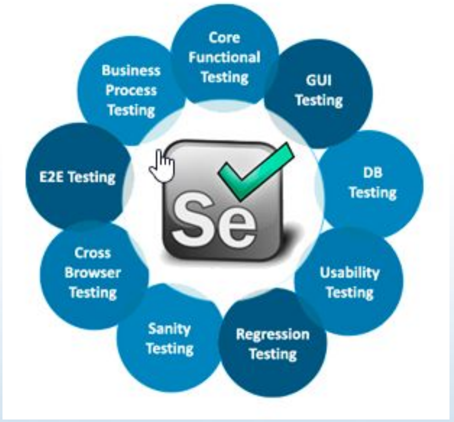
RC Moved to a legacy package in Selenium 3.0. Latest stable version is 3.1 and Selenium 4.0 beta version is available

Introduction to Selenium suite



Features

- Web-based automation testing tool
- Easy to implement
- Less hardware usage
- Open source
- Multi-browser support
- Supports most of the operating systems
- Test cases can be written in different languages



Selenium IDE

Selenium Integrated Development Environment is a browser extension that has a recording and playback features.

Selenium IDE features

- Automatically record and play back tests on Firefox and Chrome
- Organizing tests into suites for easy management
- In-built assertion functionality
- Uses Selenese commands
- Easy to find web bugs
- Supports Firefox and Chrome
- Faster execution
- Supports CI/CD pipeline

What’s new in Selenium 4 IDE

- Improved GUI for intuitive user experience
- The new IDE has a SIDE tool or Selenium IDE runner
- Improved control flow mechanism
- Enhanced element locator strategy
- The code for test cases recorded can be exported in desired language binding like java, C#, Python, .NET and javascript

Selenium IDE commands (Selenese)

A command refers to what Selenium has to do and commands in Selenium are of three types.

- Actions
- Accessors
- Assertions

Selenese actions commands

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".

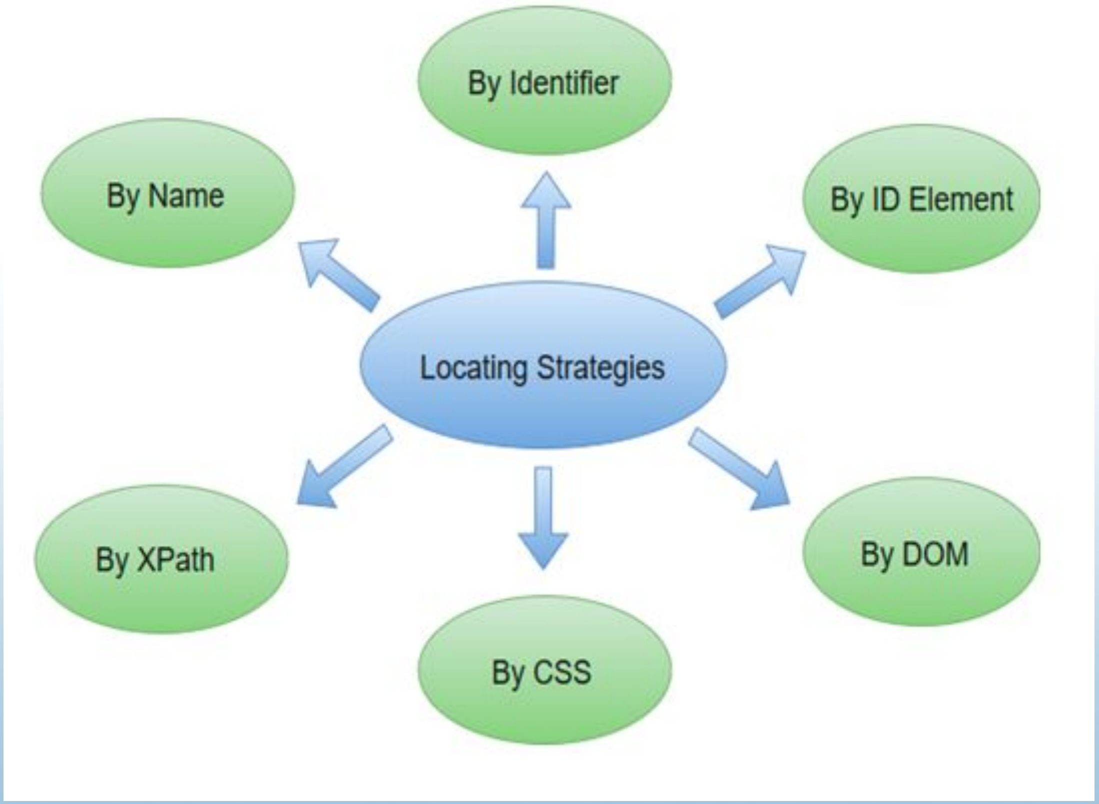
Selenese accessors commands

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.

Selenese assertion commands

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.

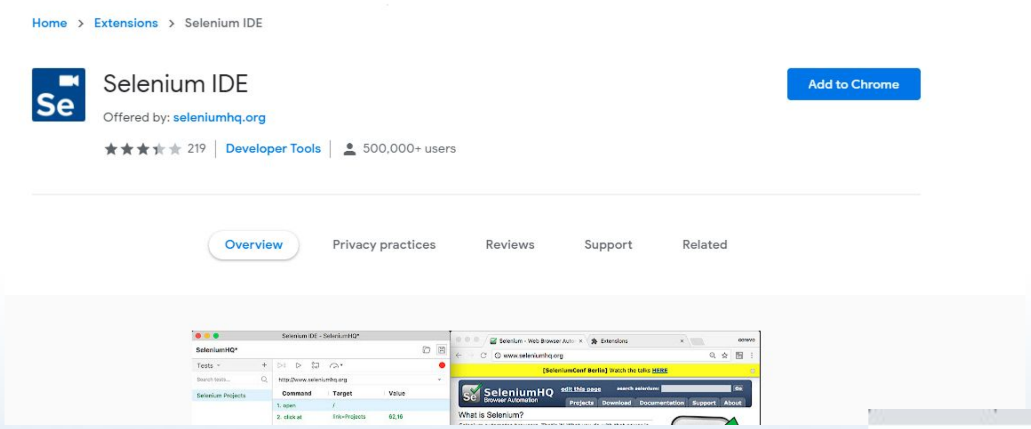
Element location strategies



Installing Selenium IDE

- Download the latest version from https://www.selenium.dev/downloads/
- For Chrome: https://chrome.google.com/webstore/detail/selenium-ide/mooikfkahbdckldjjndioackbalphokd
- For Firefox: https://addons.mozilla.org/en-US/firefox/addon/selenium-ide/

Installing Selenium IDE

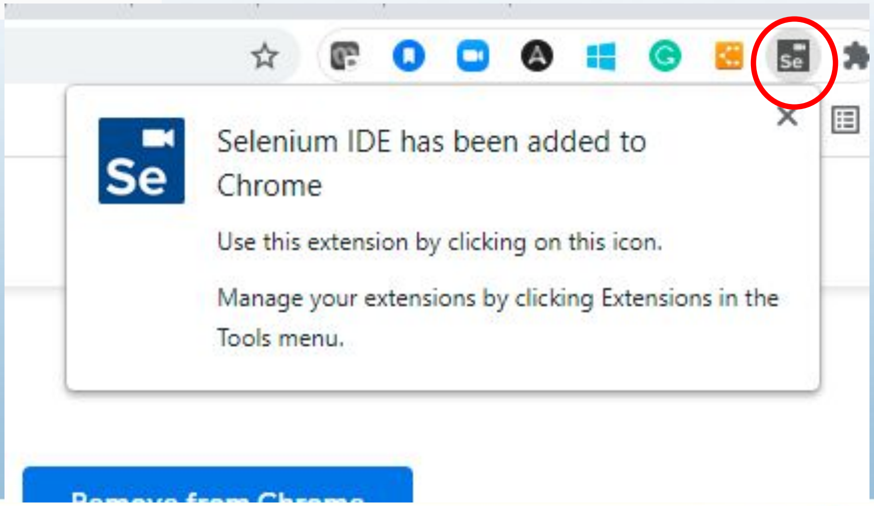


Home > Extensions > Selenium IDE

Selenium IDE

Offered by: seleniumhq.org

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

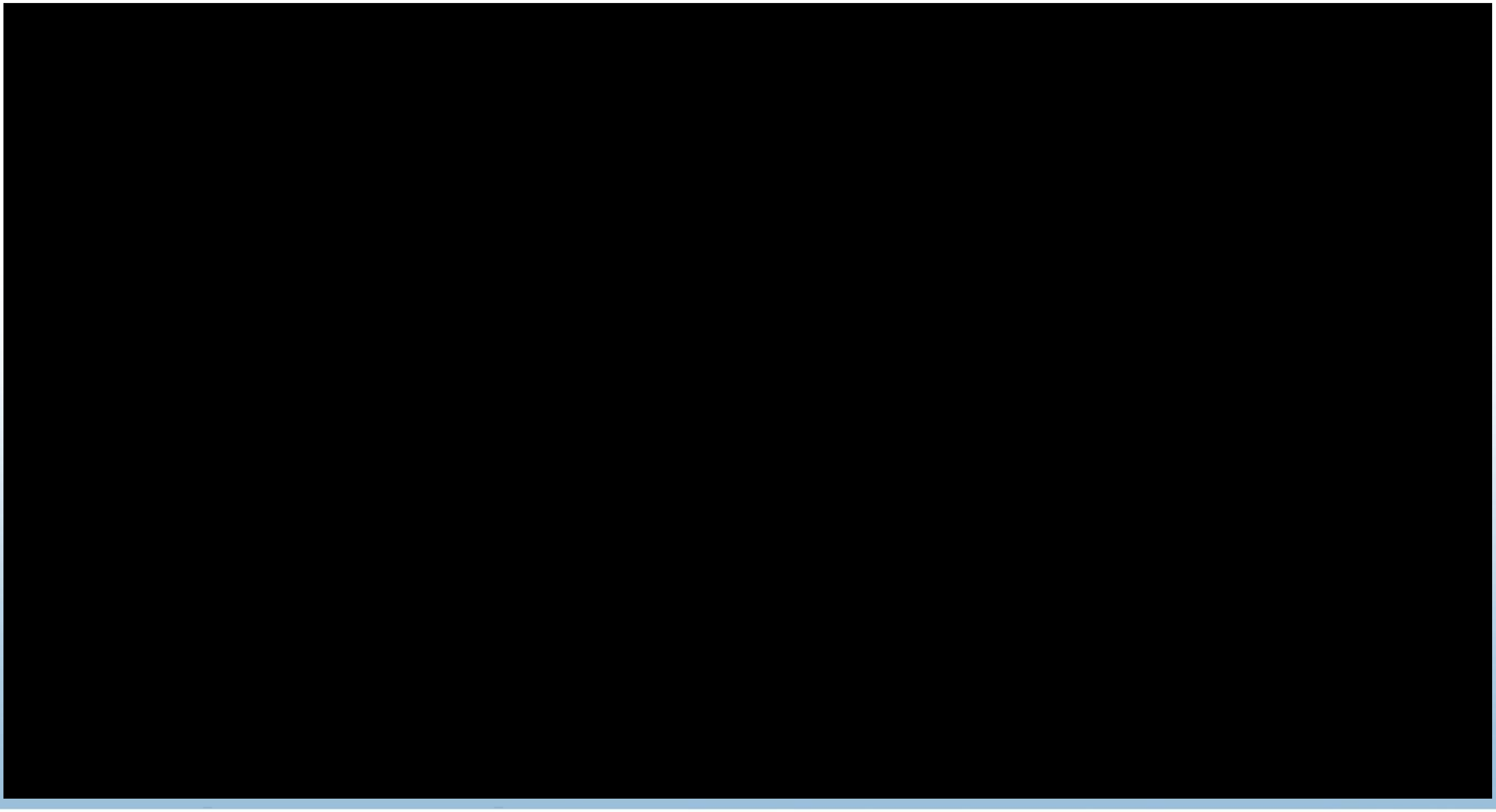


Selenium IDE has been added to Chrome

Use this extension by clicking on this icon.

Manage your extensions by clicking Extensions in the Tools menu.

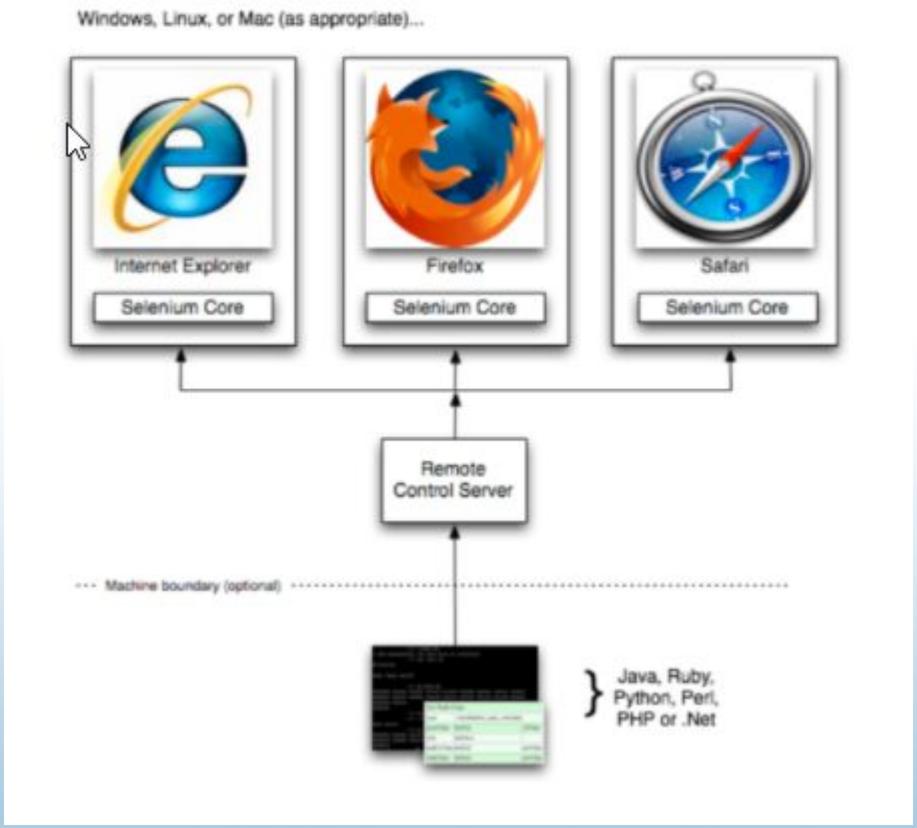
Running a test with Selenium IDE



Selenium RC

Selenium RC allows us to write automated web application UI tests with the help of full power of programming languages such as Java, C#, Perl, Python and PHP to create more complex tests such as reading and writing files, querying a database, and emailing test results.

RC architecture


Windows, Linux, or Mac (as appropriate)...
Internet Explorer
Selenium Core
Firefox
Selenium Core
Safari
Selenium Core
Remote Control Server
Machine boundary (optional)
Java, Ruby, Python, Perl, PHP or .Net

Limitation of RC

- Complicated architecture
- Execution of test scripts is time-consuming as Selenium RC uses JavaScript commands as instructions to the browser. This results in slow performance
- No support for Headless HTMLUnit browsers (invisible browser)

Selenium WebDriver

Selenium WebDriver is the most important component of the Selenium suite. Unlike Selenium RC, it does not involve any proxy server and it controls the browser directly from the OS (operating system) level. This entails a significant reduction in complexity. Starting from Selenium 3.0, RC is no longer used and WebDriver has become the core.

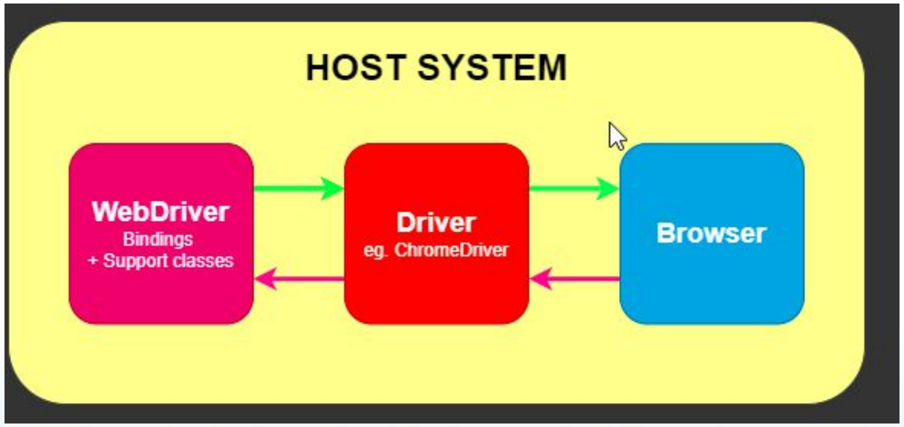
WebDriver features

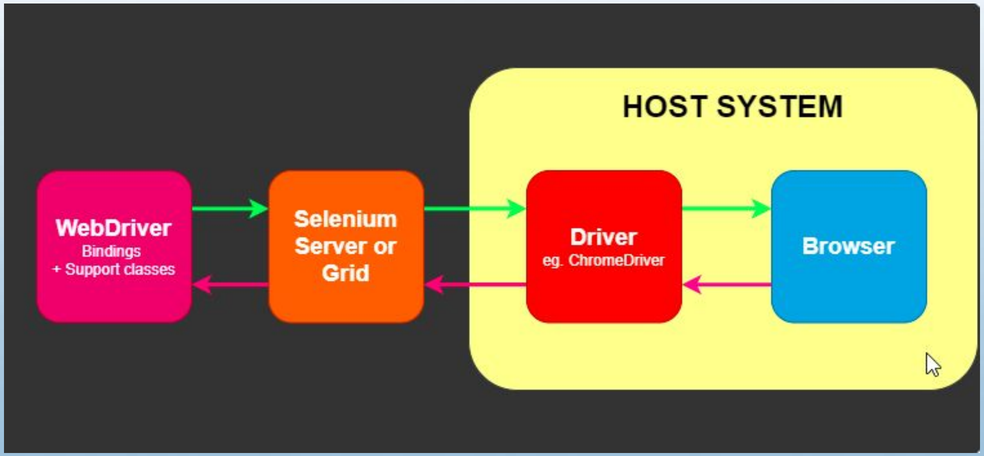
- Simpler architecture as compared to Selenium RC
- WebDriver has a more concise API and a set of easy to use commands
- Test script execution is faster than Selenium RC as it makes direct calls to the browser using browser drivers for a particular browser
- WebDriver also provides support for Headless HTMLUnit browser, IPhoneDriver and AndroidDriver along with support to all available browser and operating system
- It is one of the most preferred test automation framework in the industry

What’s new in Webdriver 4.0

- A significant change under the hood for WebDriver is the complete W3C compliance of the WebDriver APIs
- More reliable and efficient cross browser tests
- Deprecation of JSON Wire Protocol
- Relative locators like above, below, toLeftof etc were added

WebDriver architecture

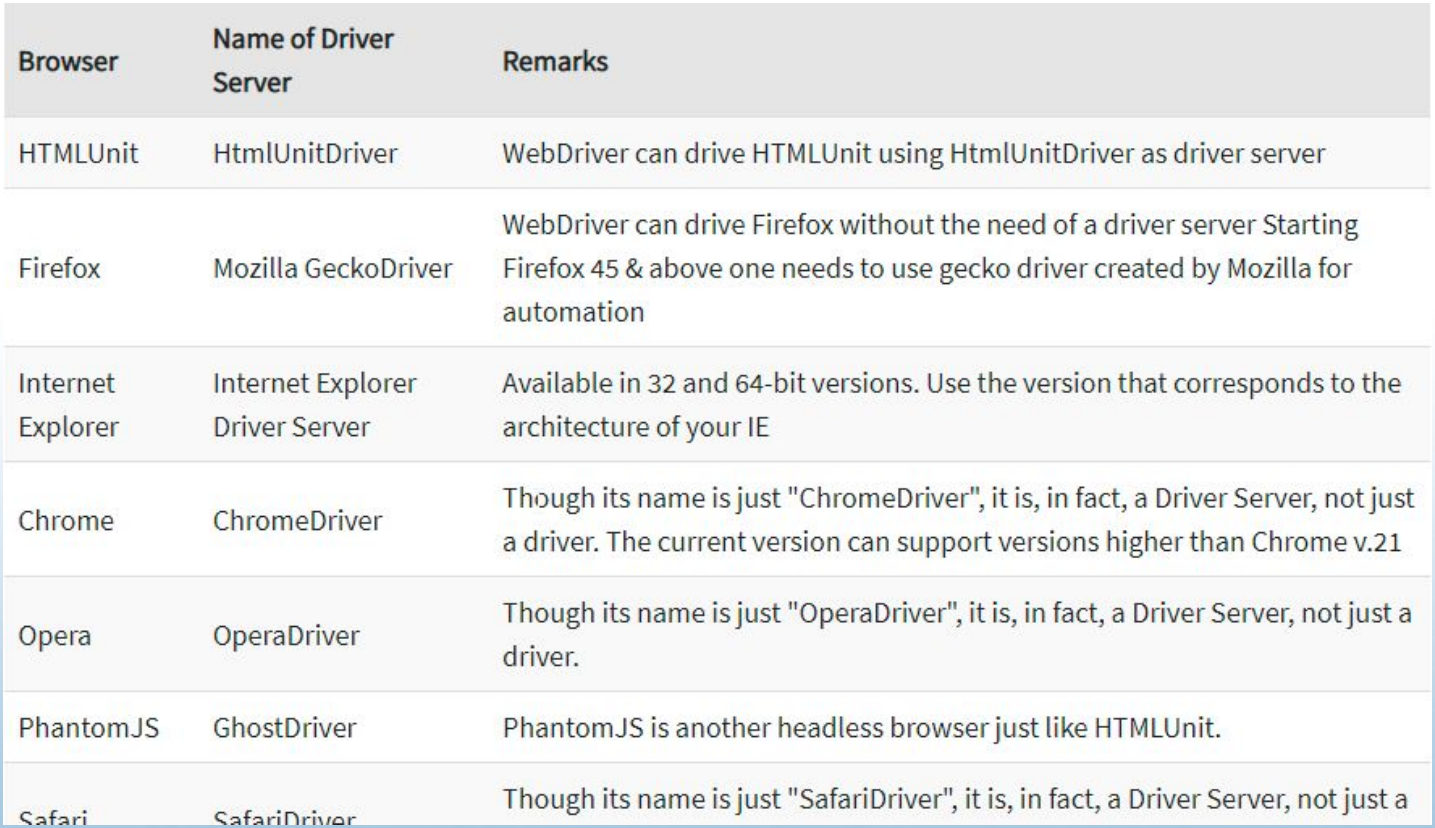




WebDriver installation

- Install Java SDK - https://www.oracle.com/java/technologies/javase-downloads.html
- Install Eclipse IDE - http://www.eclipse.org/downloads/
- Install Selenium WebDriver files - https://www.selenium.dev/downloads/

WebDriver installation



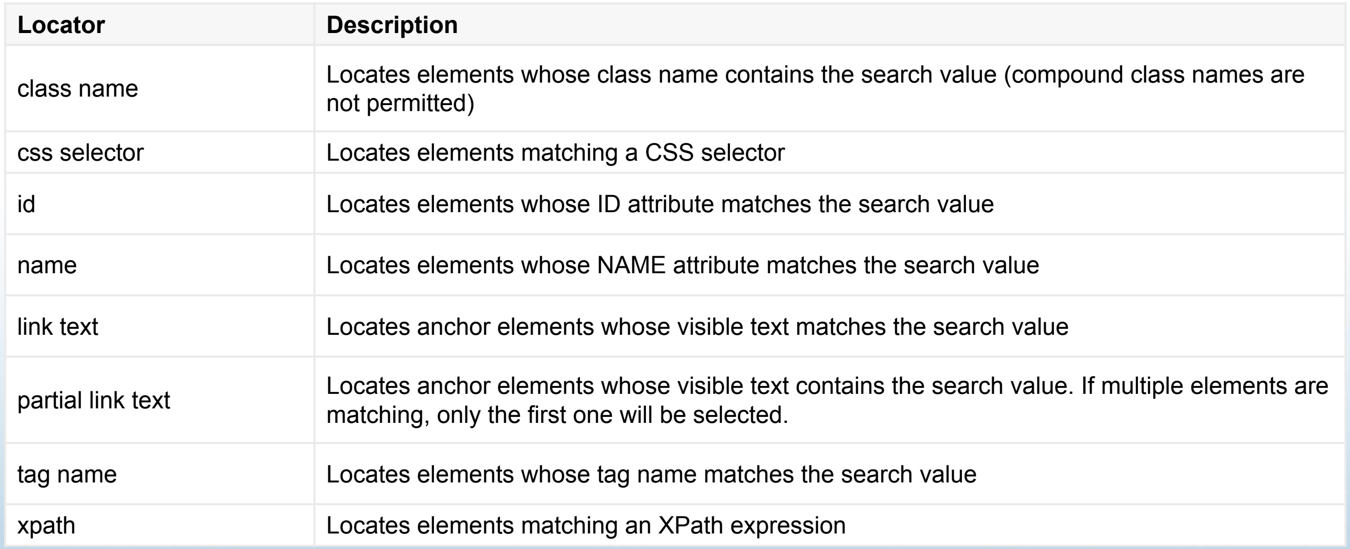
Browser	Name of Driver Server	Remarks
HTMLUnit	HtmlUnitDriver	WebDriver can drive HTMLUnit using HtmlUnitDriver as driver server
Firefox	Mozilla GeckoDriver	WebDriver can drive Firefox without the need of a driver server Starting Firefox 45 & above one needs to use gecko driver created by Mozilla for automation
Internet Explorer	Internet Explorer Driver Server	Available in 32 and 64-bit versions. Use the version that corresponds to the architecture of your IE
Chrome	ChromeDriver	Though its name is just "ChromeDriver", it is, in fact, a Driver Server, not just a driver. The current version can support versions higher than Chrome v.21
Opera	OperaDriver	Though its name is just "OperaDriver", it is, in fact, a Driver Server, not just a driver.
PhantomJS	GhostDriver	PhantomJS is another headless browser just like HTMLUnit.
Safari	SafariDriver	Though its name is just "SafariDriver", it is, in fact, a Driver Server, not just a

WebDriver methods

- Browser methods: Perform actions on a browser. Example: getcurrenturl(), gettitle(), etc.
- WebElements methods: Perform actions on WebElements. Example: Sendkeys(), getText(), etc.
- Navigation methods: Load a web page, refresh a web page, or move backwards and forwards in our browser's history. Example: to(), back(), forward (), etc.
- Wait methods: Pause between execution statements. Example: pageLoadTimeOut(), ImplicitWait(), etc.
- Switch methods: Switch to alerts, windows, and frames. An alert is also known as a pop-up. Example: Switchto()

WebDriver element locating strategies

- 8 built-in element locating strategies:



Locator	Description
class name	Locates elements whose class name contains the search value (compound class names are not permitted)
css selector	Locates elements matching a CSS selector
id	Locates elements whose ID attribute matches the search value
name	Locates elements whose NAME attribute matches the search value
link text	Locates anchor elements whose visible text matches the search value
partial link text	Locates anchor elements whose visible text contains the search value. If multiple elements are matching, only the first one will be selected.
tag name	Locates elements whose tag name matches the search value
xpath	Locates elements matching an XPath expression

- 5 relative locators (above, below, toLeftof, toRightof, near)

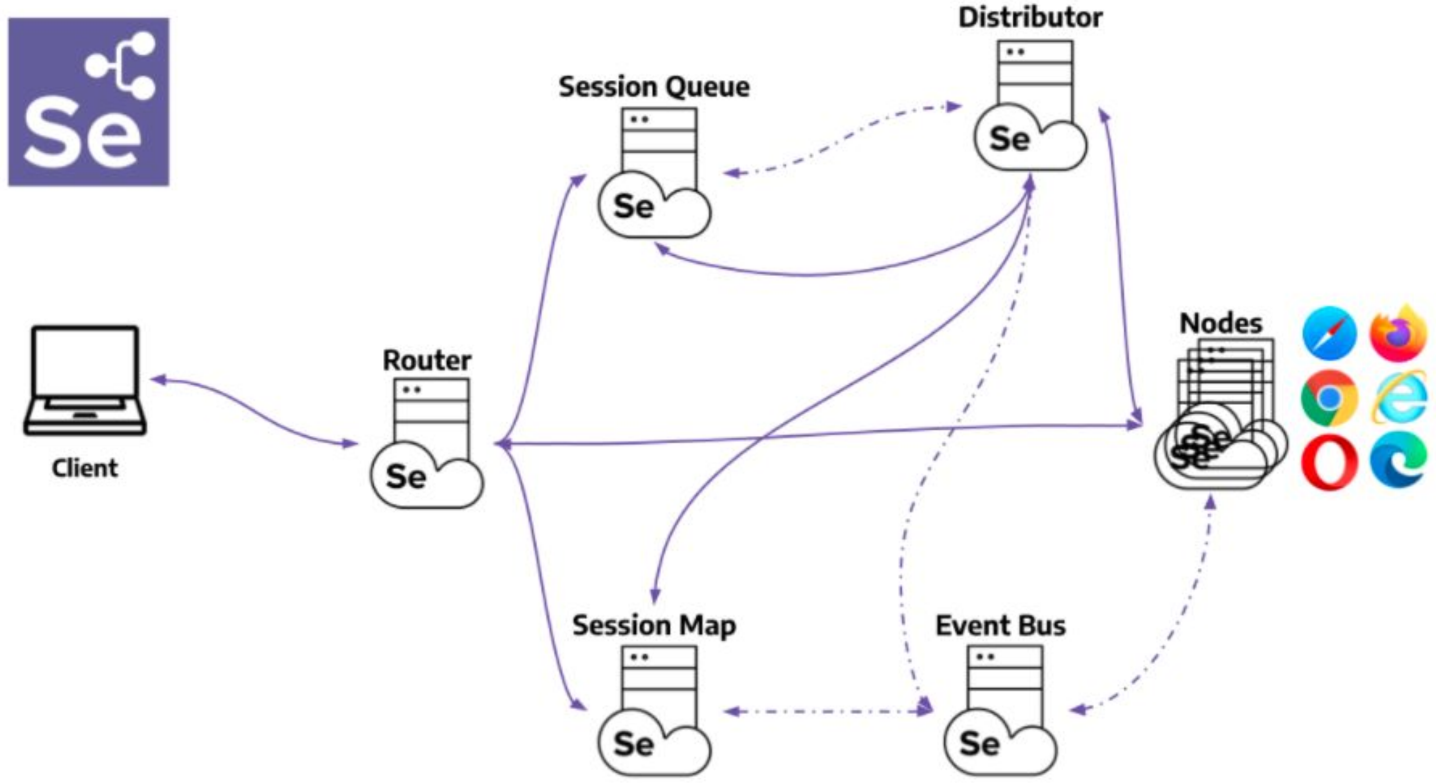
Selenium Grid

Selenium Grid allows the execution of WebDriver scripts on remote machines (virtual or real) by routing commands sent by the client to remote browser instances. It aims to provide an easy way to run tests in parallel on multiple machines.

Grid features

- Central entry point for all tests
- Management and control of the nodes / environment where the browsers run
- Scaling
- Running tests in parallel
- Cross platform testing
- Load balancing
- Reduce the time it takes for the test suite to complete a test pass
- Grid 4 offers scalability, observability

Grid architecture



Selenium Grid installation

- Different modes of installation:
  - Standalone
  - Hub and Node
  - Distributed
  - Docker
- Selenium server(Grid) can be downloaded from - https://www.selenium.dev/downloads/
- Complete setup and configuration instructions - https://www.selenium.dev/documentation/en/grid/grid_4/setting_up_your_own_grid/

Selenium - Demonstration

10 minute break

Jenkins

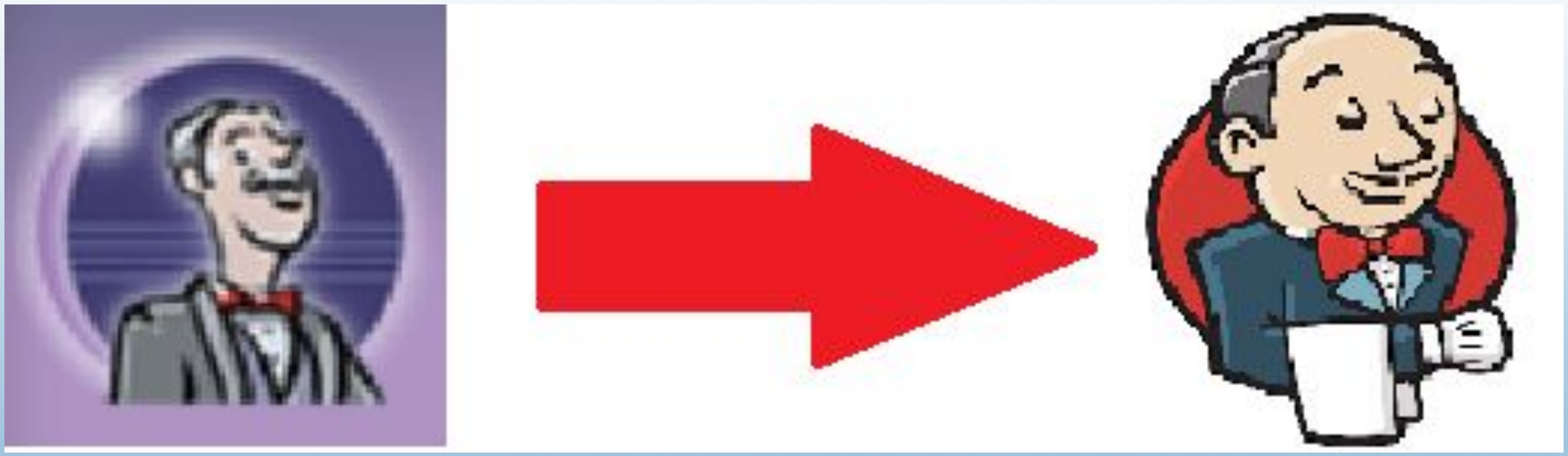
- What is Jenkins?
- Jenkins history
- How Jenkins was built
- How Jenkins works
- Various features of Jenkins
- Demonstration

What is Jenkins?

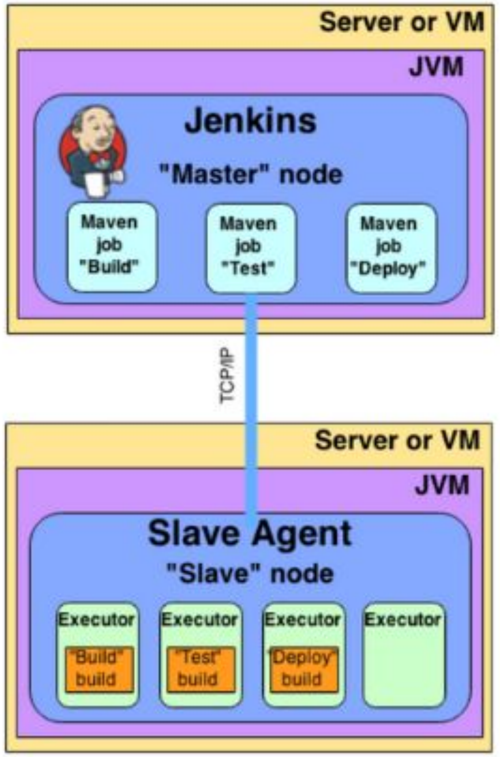
- Open source
- CI/CD
- Windows/Linux/macOS

Jenkins history

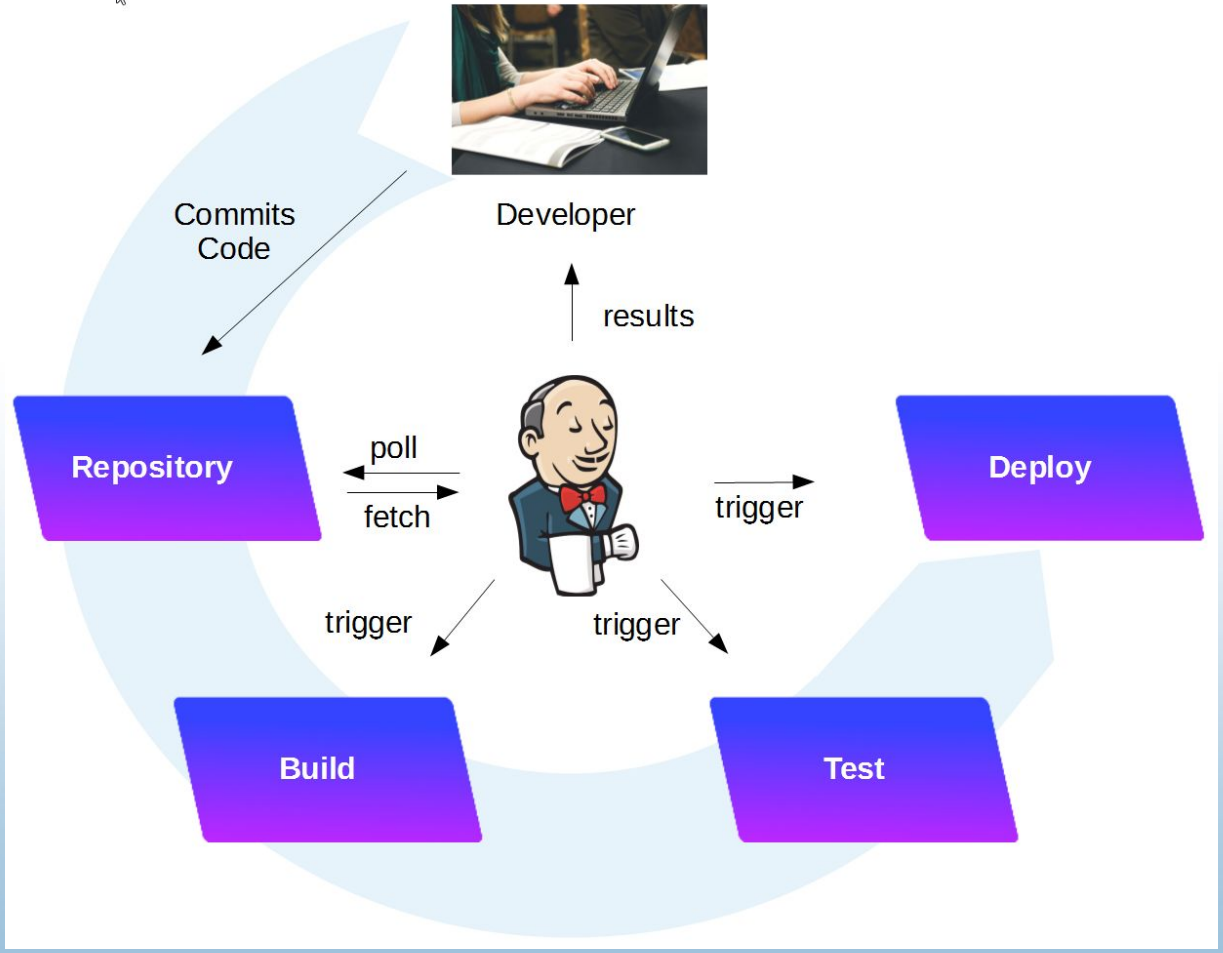
2004 (Hudson) ➡ 2011 (Jenkins) ➡ 2016 ➡ 2017 ➡ 2018 ➡ 2020 ➡ present



Jenkins – How it was built



Jenkins – How it works



Jenkins – Various features

- Easy installation
- Easy configuration
- Available plugins
- Extensible
- Easy distribution
- Jenkins is free

Jenkins - Demonstration

Similarities

All	JUnit and Selenium
• Free and open source • Client and server-side testing • Fixtures and group-fixtures • Integrate with each other • Grouping	• Are libraries/imports
Selenium and Jenkins	JUnit and Jenkins
• Support multiple languages • Integrates easily with multiple platforms	• Written in Java

Differences

- Primary usage
- Mocks
- Distributed tests
- Licensing

Summary

- Introduction
- JUnit
- Selenium
- Jenkins
- Similarities and differences
- **Conclusions**

Conclusions

- Fit your testing tool to your use case
- JUnit for unit testing Java programs
- Selenium for web testing
- Jenkins is an industry standard for CI/CD
- Use tools in combination