



Selenium

Selenium

Selenium is a free (open source) automated **testing** suite for web applications across different browsers and platforms

Registration Form

Sign up

Username

Password

Name

Address

Country

Email

Sex

☐ Male

☐ Female

Language

☐ English

☐ Non English

Submit

Testing web apps manually involves:

- Loading all transactions
- Downloading those transactions
- Creating pass/ fail reports for each
- Validating the form
- Taking screenshots for each validation



WEB APPLICATION

Automation Testing



Faster Execution



More Accurate



Lesser Investment In Human Resources



Features of Automation Testing



Supports Regression Testing



Frequent Executions



Supports Lights Out Execution

Selenium is a suite of software tools to automate web browsers.

It is open source and mainly used for functional testing and regression testing.



- Supports different PL → [Java](#), [Python](#), [C#](#), [PHP](#), [Ruby](#), [Perl](#), [JavaScript](#)
- Supports different OS → [Windows](#), [Mac](#), [Linux](#), [iOS](#), [Android](#)
- Supports different Browsers → [IE](#), [Firefox](#), [Chrome](#), [Safari](#), [Opera](#)

Features	QTP	IBM RFT	Selenium
License	Required	Required	Open Source
Cost	High	High	Less because it is open-source
Customer Support	Dedicated HP support	Dedicated IBM support	Open Source Community
Hardware resource consumption during script execution	High	High	Low
Coding experience	Not Much	Required	Should be very good along with technical capabilities of integrating the framework
Environment support	Only for Windows	Only for Windows	Windows, Linux, Solaris OS X (If browser & JVM or JavaScript support exists)
Language Support	VB Script	Java and C#	Java, C#, Ruby, Python, Perl, PHP, JavaScript

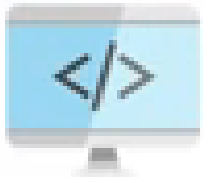
Selenium IDE drawbacks



Supports only Mozilla Firefox browser



Not suitable for dynamic web applications



No support for programming logic



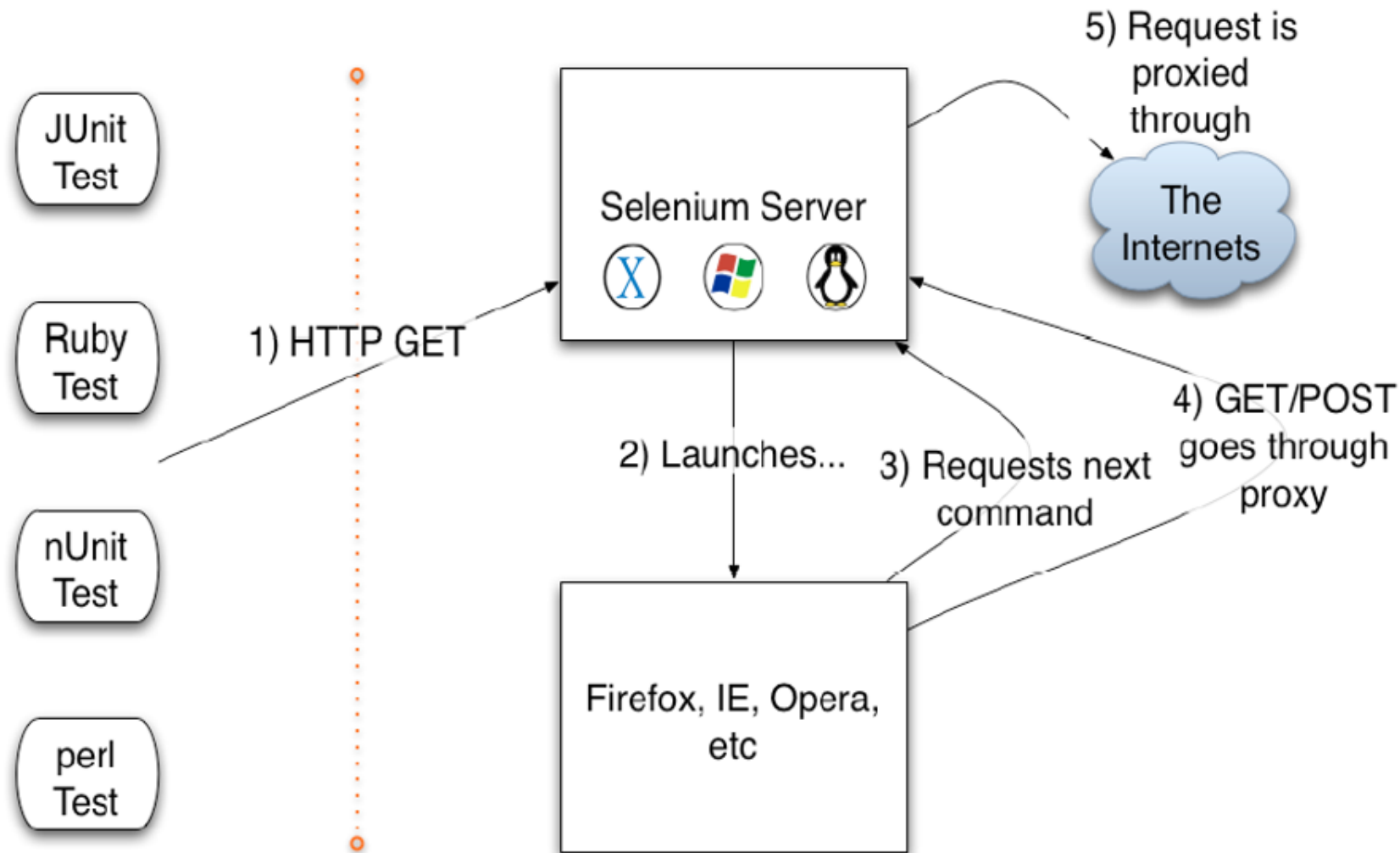
Data driven testing not possible



No centralized maintenance of objects/ elements

- **Selenium Remote Control (RC)** is used to write web application tests in different PL
- It interacts with browsers with the help of Selenium RC Server
- RC Server communicates using simple HTTP GET/POST requests
- Drawback is that every communication with RC server is timing consuming and hence RC is slow







Selenium WebDriver is a programming interface to create and execute test cases



Test cases are created and executed using Elements/ Object locators/ WebDriver methods



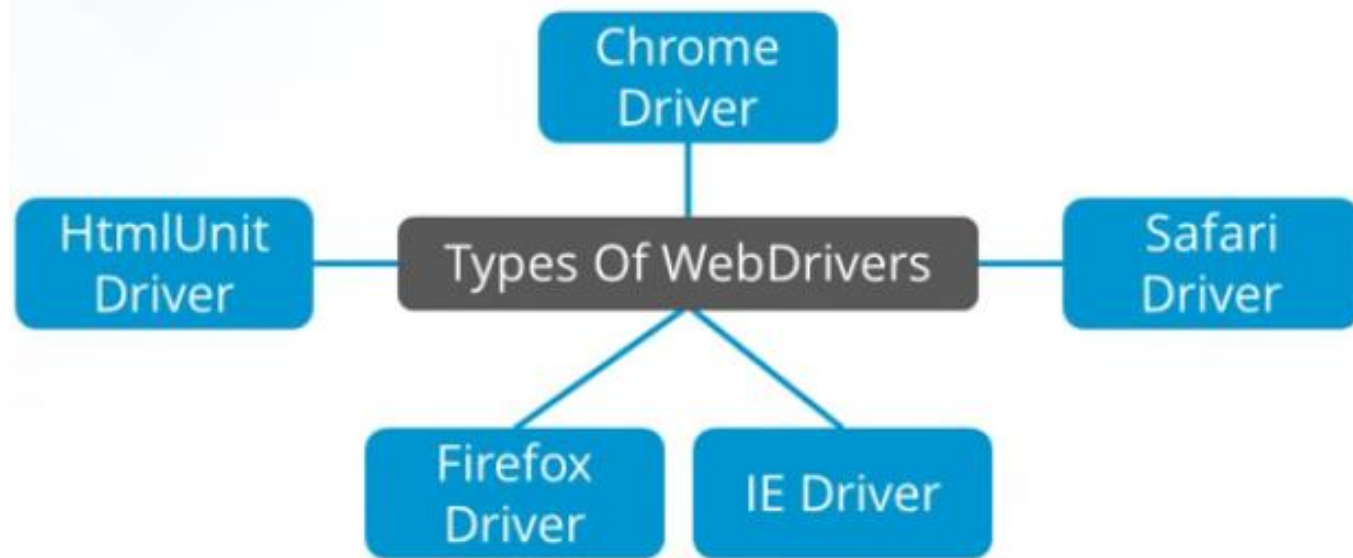
Selenium WebDriver has programming interface; not IDE



Selenium IDE supports only IDE; doesn't have programming interface



Fast as it interacts with browser directly; RC needs RC server to interact with browser



Each browser has its own driver on which the application runs. Selenium WebDriver makes direct calls to the browser

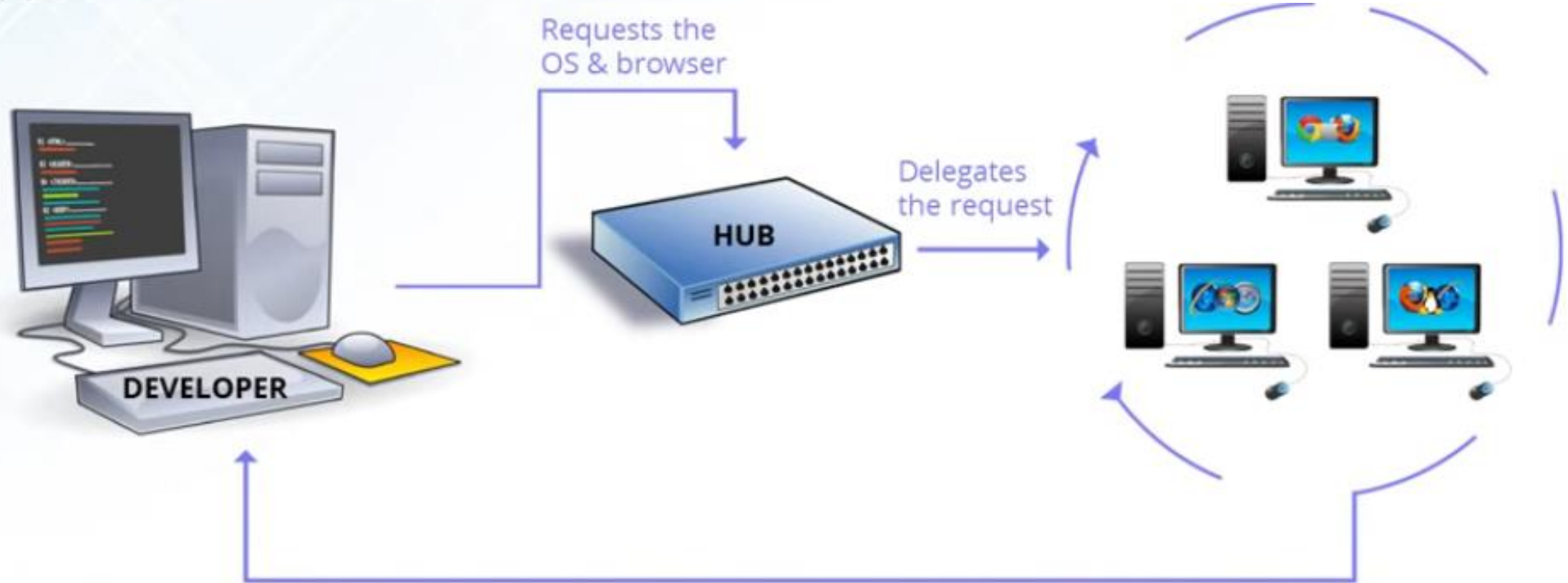


For testing on a local machine, you need to use WebDriver. But for testing on a remote machine, you need Selenium RC Server to be running before testing



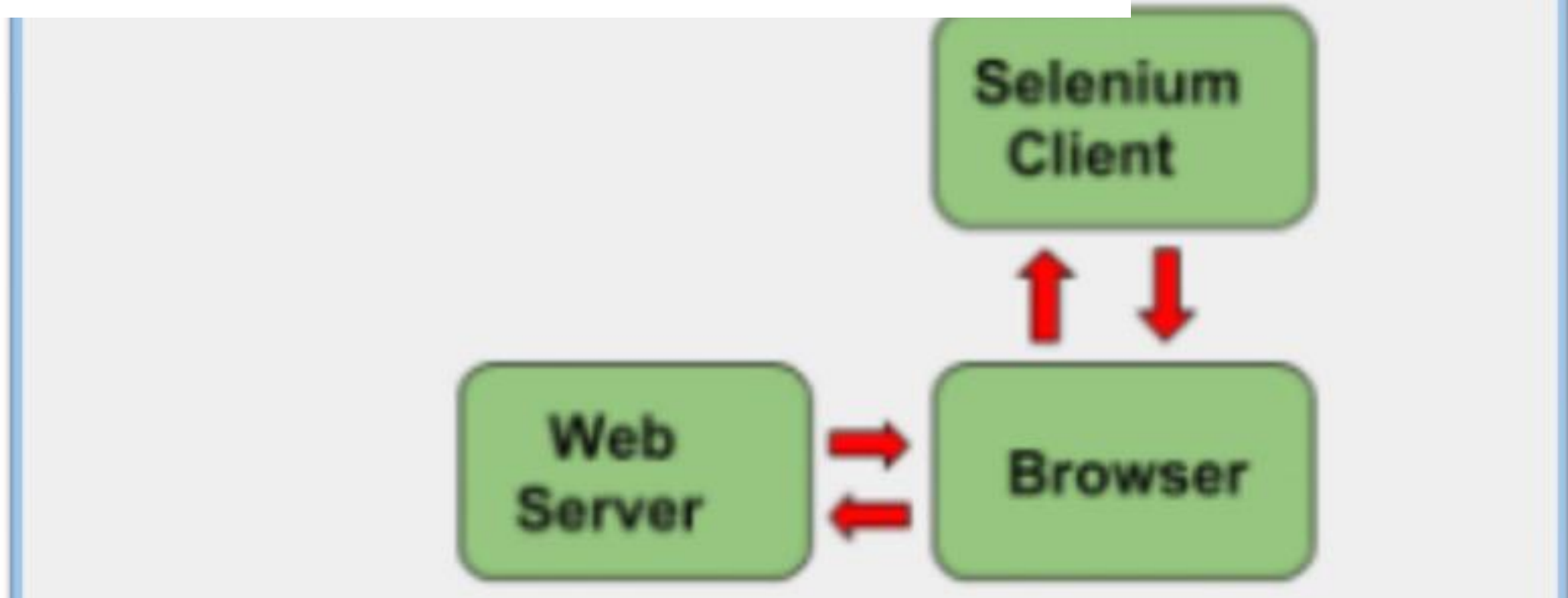
While testing on a remote machine, commands from WebDriver go to Selenium RC Server which is then interpreted on remote machine to automate the browser

- **Selenium Grid** is used to run multiple test scripts at the same time on multiple machines
- Parallel execution is achieved with the help of Hub-Node architecture
- Hub can control different test scripts on various browsers, OS and PL in various nodes
- Hub and nodes are started using **jar** files
- Supports RC test as well as WebDriver test



Selenium WebDriver Architecture

1. It is a well-designed object-oriented API developed to automate web and mobile software applications testing process
2. It is faster than Selenium RC
3. Selenium WebDriver directly talks to the browser
4. It's API is more concise than Selenium RC



Selenium Client Library

 Java

 python

 Ruby

 C#

 JS

 JSON WireProtocol
Over HTTP

Browser Drivers

 Se











Real Browsers











Installation of Selenium

- Download Chrome driver
- Download Seleniumserver standard JAR File
- Go to Netbeans/ Eclipse
- Add the Selenium JAR File into the Project
- Create a Java Class and Write the program using Selenium Lib

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class OpenGoogle {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        //1. Open Chrome
        //2. Google home page

        System.setProperty("webdriver.chrome.driver", "D:\\se

        WebDriver driver = new ChromeDriver();
        driver.get("http://www.google.co.in");
    }
}
```

```
public static void main(String[] args) {  
    // TODO Auto-generated method stub  
  
    System.setProperty("webdriver.gecko.driver", "C:\\\\Users\\Desi  
    WebDriver driver= new FirefoxDriver();  
    driver.get("http://www.google.co.in");  
  
    //enter a search term  
    //click the wikipedia link  
    driver.findElement(By.name("q")).sendKeys("Agni"+Keys.ENTER);  
}
```

Demo: Verify page title

```
public static void main( String[] args )
{
    // Create a new instance of the Firefox driver
    WebDriver driver = new FirefoxDriver();
    // (1) Go to a page
    driver.get("http://www.google.com");
    // (2) Locate an element
    WebElement element = driver.findElement(By.name("q"));
    // (3-1) Enter something to search for
    element.sendKeys("Purdue Univeristy");
    // (3-2) Now submit the form. WebDriver will find the form for us from the element
    element.submit();
    // (3-3) Wait up to 10 seconds for a condition
    WebDriverWait waiting = new WebDriverWait(driver, 10);
    waiting.until( ExpectedConditions.presenceOfElementLocated( By.id("pnnext") ) );
    // (4) Check the title of the page
    if( driver.getTitle().equals("purdue univeristy - Google Search") )
        System.out.println("PASS");
    else
        System.err.println("FAIL");

    //Close the browser
    driver.quit();
}
```



```
public class WebDriverClass {  
    public static void main(String[] args) {  
  
        System.setProperty("webdriver.gecko.driver", "/geckodriver.exe");  
        WebDriver driver = new FirefoxDriver();  
        driver.get("https://facebook.com");  
        driver.manage().window().maximize();  
        driver.getTitle();  
        driver.navigate().to("https://edureka.co/testing-with-selenium-webdriver");  
  
        driver.navigate().back();  
        driver.navigate().refresh();  
        driver.wait(5000);  
  
        driver.findElement(By.name("email")).sendKeys("xxxxx@gmail.com");  
        driver.findElement(By.pass()).sendKeys("xxxxxx");  
        driver.quit();  
    }  
}
```


```
public static void main(String[] args) {  
    // TODO Auto-generated method stub  
    System.setProperty("webdriver.chrome.driver", "D:\\selenium_tr  
    WebDriver driver = new ChromeDriver();  
    driver.get("http://www.leafground.com/pages/Link.html");  
    |  
    driver.findElement(By.LinkText("Go to Home Page")).click();  
}  
  
}
```

```
System.setProperty("webdriver.chrome.driver",  
"D:\\selenium_training\\chromedriver.exe");  
  
WebDriver driver= new ChromeDriver();  
  
driver.get("http://www.leafground.com/pages/Button.html");  
  
WebElement getPositionButton=driver.findElement(By.id("position"));  
  
Point xypoint=getPositionButton.getLocation();  
int xValue=xypoint.getX();  
int yValue=xypoint.getY();  
System.out.println("X value is :"+ xValue+"Y value is :"+ yValue);
```

```
public class Day1 {  
  
    WebDriver driver;  
  
    public void invokeBrowser(){  
  
        try {  
            System.setProperty("webdriver.chrome.driver", "D:\\Akanksha\\Selenium\\chromedriver_win32_new\\chromedriver.exe");  
            driver = new ChromeDriver();  
            driver.manage().deleteAllCookies();  
            driver.manage().window().maximize();  
            driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);  
            driver.manage().timeouts().pageLoadTimeout(30, TimeUnit.SECONDS);  
  
            driver.get("http://www.edureka.co");  
            searchCourse();  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
  
    }  
  
    public void searchCourse(){  
        try {  
            driver.findElement(By.id("search-inp1")).sendKeys("Java");  
            Thread.sleep(3000);  
            driver.findElement(By.id("search-button-bottom")).click();  
  
        } catch (InterruptedException e) {  
            e.printStackTrace();  
        }  
    }  
}
```



```
public class DemoAutomation {  
  
    public static void main(String[] args) {  
  
        System.setProperty("webdriver.chrome.driver", "C:\\\\browserdrivers\\chromedriver.exe")  
  
        ChromeDriver driver = new ChromeDriver();  
        //FirefoxDriver driver = new FirefoxDriver();  
  
        //EdgeDriver driver = new EdgeDriver();  
        driver.get("http://www.ebay.com");  
        driver.manage().window().maximize();  
        driver.findElement(By.xpath("//*[@id=\"gh-ac\"]")).sendKeys("mobile");  
        driver.findElement(By.xpath("//*[@id=\"gh-btn\"]")).click();  
        driver.close();  
        //driver.quit();  
    }  
}
```


Pre Conditions				
1	Browser is open and https://www.saucedemo.com/ website is launched			
Step No	Test Step	Expected Result	Actual Result	Test Data
1	User enters correct username and password	User should be able to type the username and password successfully		Username: "standard_user" Password: "secret_sauce"
2	User click on "Login" button	User is logged into portal successfully		
Post Condition				
1	Browser is closed			

```
public static void main(String[] args) {
```

```
    WebDriverManager.chromedriver().setup();
```

```
    ChromeDriver driver = new ChromeDriver();
```

```
    driver.get("https://www.saucedemo.com/");
```

```
    driver.findElement(By.id("user-name")).sendKeys("standard_user");
```

```
    driver.findElement(By.id("password")).sendKeys("secret_sauce");
```

```
    driver.findElement(By.xpath("/html/body/div[2]/div[1]/div[1]/div/form/input[3]")).click();
```

```
    driver.close();
```

```
}
```



```
public static void main(String[] args) {
```

```
    if(browser.equals("Firefox"))
```

```
    {
```

```
        WebDriverManager.firefoxdriver().setup();
```

```
        FirefoxDriver driver = new FirefoxDriver();
```

```
    }
```

```
    else if(browser.equals("chrome"))
```

```
    {
```

```
        WebDriverManager.chromedriver().setup();
```

```
        ChromeDriver driver = new ChromeDriver();
```

```
    }
```

```
    WebDriverManager.chromedriver().setup();
```

```
    //ChromeDriver driver = new ChromeDriver();
```

```
    FirefoxDriver driver = new FirefoxDriver();
```

```
    driver.get("https://www.saucedemo.com/");
```

```
    driver.findElement(By.id("user-name")).sendKeys("standard_user");
```


```
    driver.findElement(By.id("password")).sendKeys("secret_sauce");
```

```
    driver.findElement(By.xpath("/html/body/div[2]/div[1]/div[1]/div/form/input[3]")).click
```

```
    driver.close();
```

Element selection strategies

There are eight different built-in element location strategies in WebDriver:

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
tag name	Locates elements whose tag name matches the search value
xpath	Locates elements matching an XPath expression

Basic Methods in WebDriver Interface

- `get(java.lang.String url)`
 - Load a new web page in the current browser window.
- `manage()`
 - Gets the Option interface.
- `getCurrentUrl()`
 - Get a string representing the current URL that the browser is looking at.
- `getTitle()`
 - The title of the current page.
- `getPageSource()`
 - Get the source of the last loaded page.
- `navigate()`
 - An abstraction allowing the driver to access the browser's history and to navigate to a given URL.
- `quit()`
 - Quits this driver, closing every associated window.
- `close()`
 - Close the current window, quitting the browser if it's the last window currently open.

- **getWindowHandle()**
 - Return an opaque handle to this window that uniquely identifies it within this driver instance.
- **getWindowHandles()**
 - Return a set of window handles which can be used to iterate over all open windows of this WebDriver instance by passing them to `switchTo().WebDriver.Options.window()`
- **switchTo()**
 - Send future commands to a different frame or window.
- **findElement(By by)**
 - Find the first WebElement using the given method.
- **findElements(By by)**
 - Find all elements within the current page using the given mechanism.

Working with WebElements in Selenium WebDriver

- `sendKeys(java.lang.CharSequence... keysToSend)` - Use this method to simulate typing into an element, which may set its value.
- `clear()` - If this element is a text entry element, this will clear the value.
- `click()` - Click this element.
- `getAttribute(java.lang.String name)` - Get the value of the given attribute of the element.
- `getCssValue(java.lang.String propertyName)` - Get the value of a given CSS property.
- `getLocation()` - Where on the page is the top left-hand corner of the rendered element?
- `getSize()` - What is the width and height of the rendered element?
- `getTagName()` - Get the tag name of this element.
- `getText()` - Get the visible text
- `isDisplayed()` - Is this element displayed or not? This method avoids the problem of having to parse element's "style" attribute.
- `isEnabled()` - Is the element currently enabled or not? This will generally return true for everything disabled input elements.
- `isSelected()` - Determine whether or not this element is selected or not.


```
public class WorkingWithWebElements {  
  
    public static void main(String[] args) {  
  
        WebDriverManager.chromedriver().setup();  
        ChromeDriver driver = new ChromeDriver();  
        driver.get("https://www.sugarcrm.com/au/request-demo/");  
        driver.manage().window().maximize();  
        driver.findElement(By.name("firstname")).sendKeys("Testing");  
        driver.findElement(By.name("firstname")).clear();  
        System.out.println(driver.findElement(By.name("firstname")).getAttribute("class"));  
  
    }  
  
}
```

```
public static void main(String[] args) throws InterruptedException {
```

```
WebDriverManager.chromedriver().setup();  
ChromeDriver driver = new ChromeDriver();  
driver.get("https://www.sugarcrm.com/au/request-demo/");  
driver.manage().window().maximize();  
WebElement ddown = driver.findElement(By.name("employees_c"));  
Select select = new Select(ddown);
```

```
select.selectByValue("level1");  
Thread.sleep(2000);
```

```
select.selectByVisibleText("51 - 100 employees");  
Thread.sleep(2000);
```

```
select.selectByIndex(5);  
Thread.sleep(2000);
```

```
}
```

```
WebElement ddown = driver.findElement(By.id("multi-select"));
Select select = new Select(ddown);
select.selectByValue("California");
Thread.sleep(2000);
select.selectByIndex(5);
Thread.sleep(2000);
List<WebElement> allItems = select.getAllSelectedOptions();
System.out.println(allItems.size());
select.deselectAll();
Thread.sleep(2000);
select.selectByValue("California");
Thread.sleep(2000);
select.selectByIndex(5);
Thread.sleep(2000);

select.deselectByIndex(5);
List<WebElement> allItems1 = select.getAllSelectedOptions();
System.out.println(allItems1.size());
```



```

int rowcount = driver
    .findElements(By.xpath("html/body/div[1]/div[3]/div[2]/div/div/form/div/

int statuscount = 0;

for (int i = 1; i <= rowcount; i++) {

    String status = driver.findElement(By.xpath("//*[@id='resultTable']/tbody/tr

    if (status.equals("Enabled")) {
        statuscount = statuscount + 1;
    }

}

System.out.println("No of employess Enabled:" + statuscount);
driver.close();
}

```

Product	Article	Price
Product1	0001	10
Product2	0002	15
Product3	0003	12
Product4	0004	20

```

driver.findElement(By.xpath("html/body/table/tbody/tr["+i+"]/td["+j+"]")).getText())

```

```
public class NewTest {  
    public WebDriver driver;  
    @Test  
    public void openMyBlog() {  
        driver.get("https://www.softwaretestingmaterial.com/");  
    }  
    @BeforeClass  
    public void beforeClass() {  
        System.setProperty("webdriver.gecko.driver", "D:\\Selenium\\Drivers\\geckodriver.exe");  
        driver = new FirefoxDriver();  
    }  
    @AfterClass  
    public void afterClass() {  
        driver.quit();  
    }  
}
```

#Sample Feature Definition Template

@login

Feature: Login Feature

This feature is used to test login functionality

@positive

Scenario: Valid Username and Valid Password Test

Given I am on the login page

When I enter "dan@gmail.com" and "test1234"

Then I should be successfully logged in

@negative

Scenario Outline: Invalid Username and Invalid Password Test

Given I am on the login page

When I enter "<username>" and "<password>"

Then I should get login error message

Examples:

username	password
invalid-dan@gmail.com	test1234
dan@gmail.com	invalid-test1234

```
public class MouseOver extends Base{

    public void demo() {
        WebElement navIcon = driver.findElement(By.xpath("//span[text()='Account & Lists']/span[@class='nav-icon nav-arrow']"))
        // WebElement createWishList = driver.findElement(By.partialLinkText("Create a Wish List"));

        Actions ac = new Actions(driver);
        ac.moveToElement(navIcon).build().perform();

        // driver.findElement(By.partialLinkText("Create a Wish List")).click();

    }

    public static void main(String[] args) {
        MouseOver obj = new MouseOver();
        obj.setupBrowser("chrome", "https://www.amazon.in");
        obj.demo();
    }
}
```

```
@Test
```

```
public void testTitle() throws IOException {
```

```
    // Find elements by attribute lang="READ_MORE_BTN"
```

```
    List<WebElement> elements = driver
```

```
        .findElements(By.cssSelector("[lang=\"READ_MORE_BTN\"]"));
```

```
    //Click the selected button
```

```
    elements.get(0).click();
```

```
    assertTrue("The page title should be chagned as expected",
```

```
        (new WebDriverWait(driver, 5))
```

```
            .until(new ExpectedCondition<Boolean>() {
```

```
                public Boolean apply(WebDriver d) {
```

```
                    return d.getTitle().equals("我眼中软件工程师该有的常识");
```

```
                }
```

```
            })
```

```
    );
```

```
}
```



```
@ParameterizedTest
```

```
public void loginTest(String itemName, String price) {
```

```
    // The xPath for the element is a bit complex. Basically what we are looking for is the div that contains the product price
```

```
    // and has the same ancestor as the div that contains the itemName (so we know we are checking the price for our specific product)
```

```
    String xPath = String.format
```

```
        ("//div[contains(text(), '%s')]/ancestor::div[@class='inventory_item_description']//div[@class='inventory_item_price']",  
         itemName);
```

```
    WebElement priceElement = driver.findElement(By.xpath(xPath));
```

```
    System.out.println(priceElement.getText());
```

```
    Assert.assertEquals(priceElement.getText(), price);
```

```
}
```

```
# Max Base
# https://github.com/BaseMax
from selenium import webdriver
from selenium.webdriver.common.keys import Keys

browser = webdriver.Chrome()
browser.get('https://www.github.com/')

# document.querySelector("input.form-control.input-sm.header-search-input.jump-to-field.js-jump-to-field.js-site-search-focus")
search = browser.find_element_by_css_selector('input.form-control.input-sm.header-search-input.jump-to-field.js-jump-to-field.js-site-sea
search.send_keys('BaseMax' + Keys.RETURN)
print(browser.title)

browser.quit()
```
