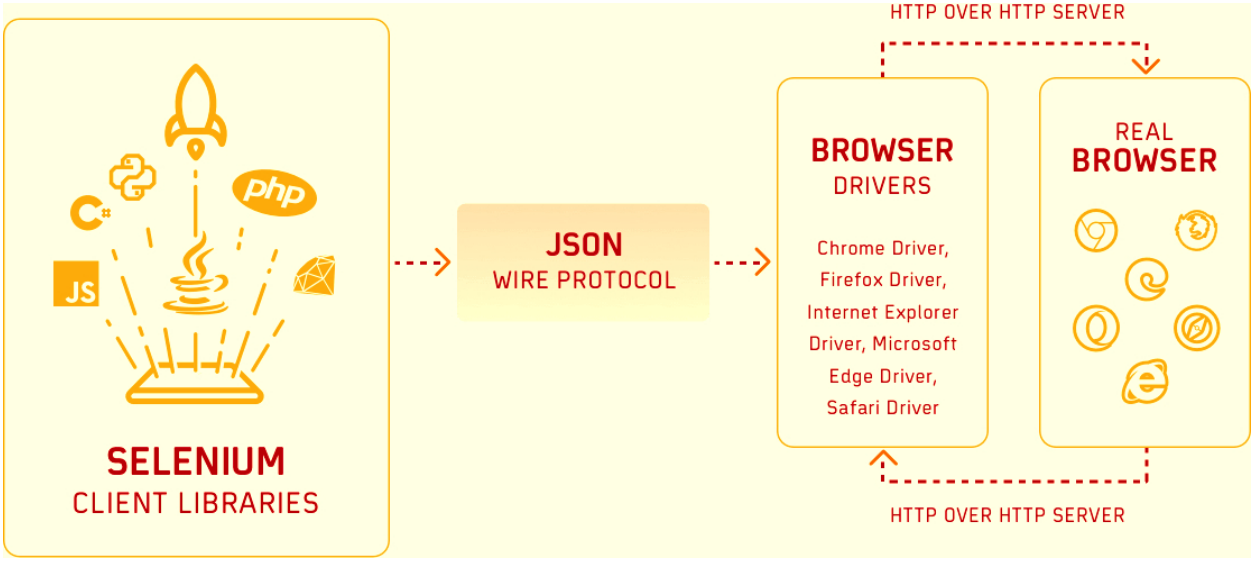
Selenium WebDriver 4

The JSON Wire Protocol



Selenium WebDriver provides a programming interface for driving the browser for automation testing. Selenium WebDriver is a client which is giving an interface to write test in programming languages like Java or Python or many other languages in the market. Server doesn't care or know about what language you are using for writing your tests because it only cares about the correct protocol which is JSON Wire protocol. The JSON wire protocol (JSONWP) is a transport mechanism created by WebDriver developers. This wire protocol is a specific set of predefined, standardized endpoints exposed via a RESTful API. The purpose of WebDriver and JSONWP is the automated testing of websites via a browser such as Firefox driver, IE driver, and Chrome driver. For each Selenium command, an HTTP request is created and sent to the browser driver.

The browser driver uses an HTTP server for getting the HTTP requests The implementation steps are executed on the browser.

HTTP server sends the status back to the automation script

The HTTP server determines the steps needed for implementing the Selenium command

The execution status is sent back to the HTTP server

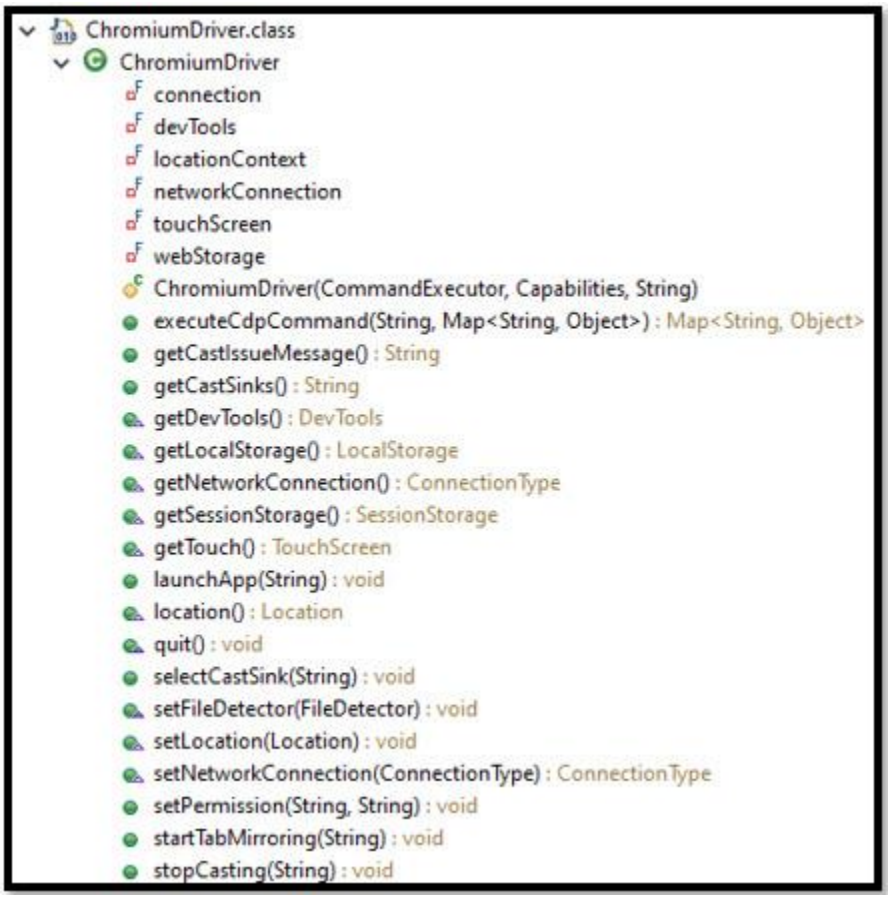
JSON Wire Protocol Over HTTP has been removed from W3C WebDriver Protocol. That means information is not transferred over HTTP by sending HTTP Requests and receiving HTTP Responses. With Selenium 4, information is transferred directly back and forth from the client to the server without the JSON Wire Protocol.

An advantage involves testing applications that will execute more consistently between browsers. Kudos to W3C, an acronym for World Wide Web Consortium for developing web standards. A standardization of W3C promotes compatibility beyond WebDriver API implementations.

ChromiumDriver

Selenium WebDriver uses a driver to manage each browser. ChromeDriver is an executable that Selenium WebDriver uses to control Google Chrome and EdgeDriver controls Microsoft Edge. Both drivers extend ChromiumDriver as an update in Selenium 4. The ChromiumDriver class has methods to create a connection with DevTools.

```
public class ChromeDriver extends ChromiumDriver

public class EdgeDriver extends ChromiumDriver
```



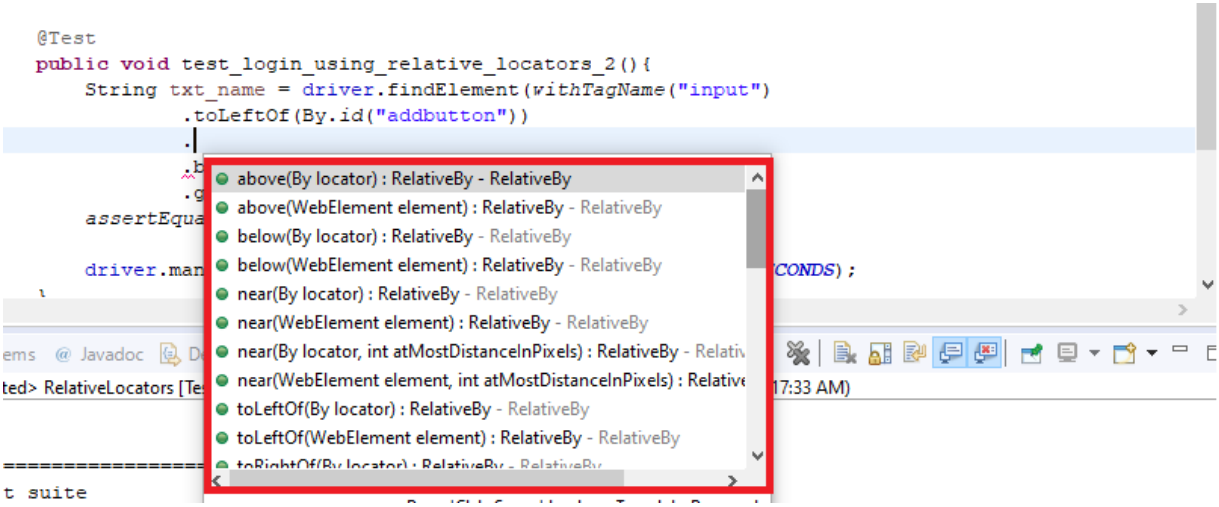

```
        WebDriver driver = new ChromeDriver();
        driver.navigate().to("https://www.google.com");
        driver.manage().window().fullscreen();

        driver.switchTo().;

    }

}
```

activeElement() : WebElement - TargetLocator
alert() : Alert - TargetLocator
defaultContent() : WebDriver - TargetLocator
equals(Object obj) : boolean - Object
frame(int index) : WebDriver - TargetLocator
frame(String nameOrId) : WebDriver - TargetLocator
frame(WebElement frameElement) : WebDriver - Targe
getClass() : Class<?> - Object
hashCode() : int - Object
newWindow(WindowType typeHint) : WebDriver - Targ
notify() : void - Object
notifyAll() : void - Object
parentFrame() : WebDriver - TargetLocator
toString() : String - Object
wait() : void - Object
wait(long timeout) : void - Object
wait(long timeout, int nanos) : void - Object

Selenium IDE: New and refreshed Selenium IDE

- Selenium IDE is a tool for record and playback. Additionally, it will have advanced and rich features in Selenium 4.0.
- Moreover, the limitation of a lack of ability for parallel execution will overcome in the new version of IDE. In other words, a new version is coming with improved browser support. Additionally, parallel execution will be possible along with the passed/failed status of test cases and execution time.
- New IDE is W3C compatible and completely dependent on WebDriver.
- Moreover, the new version will have a new Node JSbased CLI runner.Selenium 4 Relative Locator – Methods

above	Web element to be searched/located appears above the specified element
below	Web element to be searched/located appears below the specified element.
toLeftOf	Web element located appears to the left of the specified element
toRightOf	Web element located appears to the right of the specified element
near	Web element located is at most 50 pixels away from the specified element



Action class

```
Actions act1 = new Actions(driver);
WebElement toDoList1= driver.findElement(By.id("toDoListBtn"));
act1.
```

- build() : Action - Actions
- click() : Actions - Actions
- click(WebElement target) : Actions - Actions
- clickAndHold() : Actions - Actions
- clickAndHold(WebElement target) : Actions - Actions
- contextClick() : Actions - Actions
- contextClick(WebElement target) : Actions - Actions
- doubleClick() : Actions - Actions
- doubleClick(WebElement target) : Actions - Actions
- dragAndDrop(WebElement source, WebElement target) : Actions - Actions
- dragAndDropBy(WebElement source, int xOffset, int yOffset) : Actions - Actions
- equals(Object obj) : boolean - Object
- getClass() : Class<?> - Object
- hashCode() : int - Object
- keyDown(CharSequence key) : Actions - Actions
- keyDown(WebElement target, CharSequence key) : Actions - Actions
- keyUp(CharSequence key) : Actions - Actions
- keyUp(WebElement target, CharSequence key) : Actions - Actions
- moveByOffset(int xOffset, int yOffset) : Actions - Actions
- moveToElement(WebElement target) : Actions - Actions
- moveToElement(WebElement target, int xOffset, int yOffset) : Actions - Actions
- notify() : void - Object
- notifyAll() : void - Object
- pause(Duration duration) : Actions - Actions
- pause(long pause) : Actions - Actions
- perform() : void - Actions
- release() : Actions - Actions
- release(WebElement target) : Actions - Actions
- sendKeys(CharSequence... keys) : Actions - Actions
- sendKeys(WebElement target, CharSequence... keys) : Actions - Actions
- tick(Action action) : Actions - Actions
- tick(Interaction... actions) : Actions - Actions
- toString() : String - Object

Press 'Ctrl+Space' to show Template Proposals

Timeouts