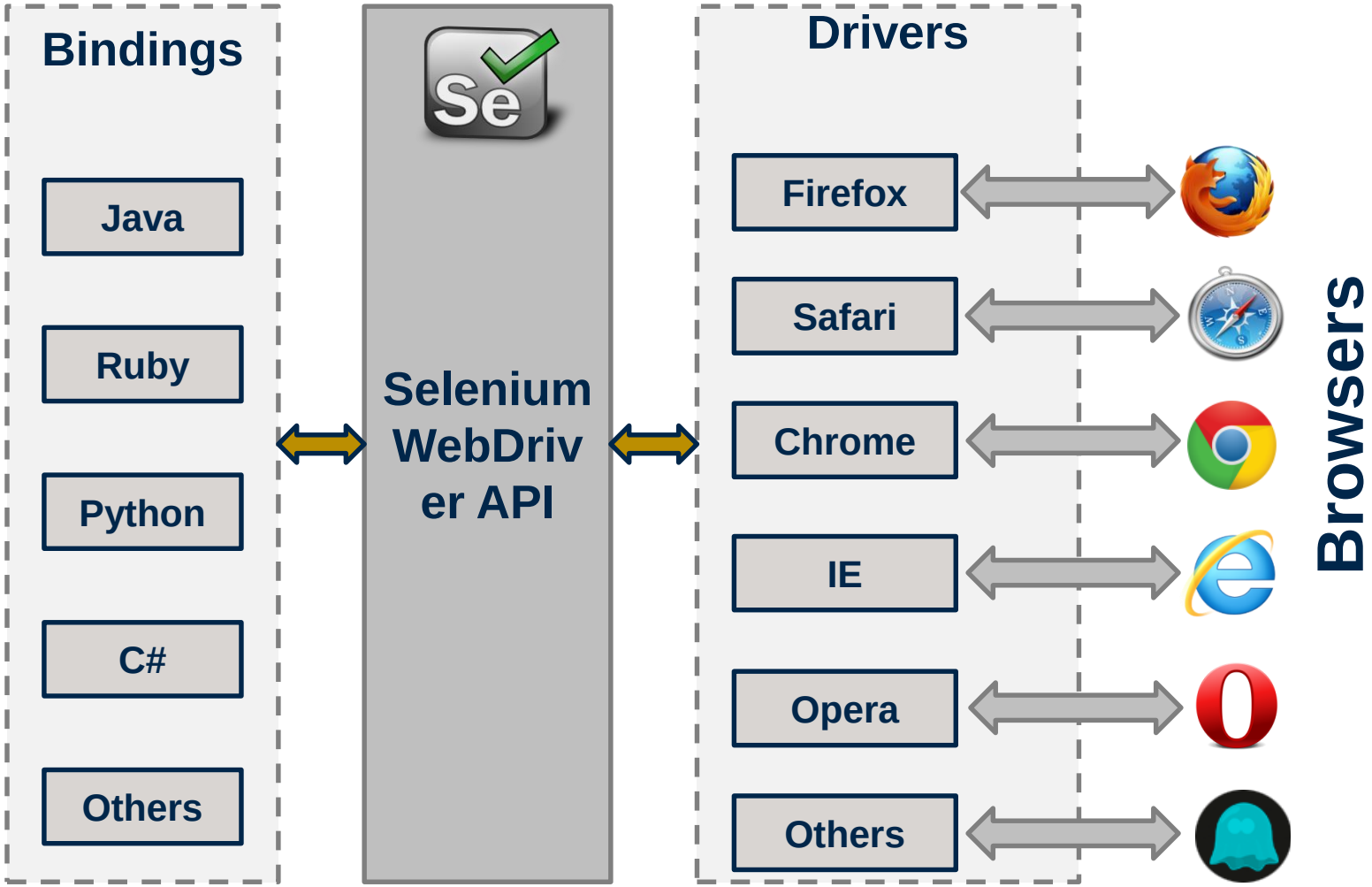
Test Automation & Advanced Selenium

Lesson 7: Selenium Web Driver – Advance

Lesson Objectives

- Selenium: How it works
- Different drivers
  - Chrome
  - Firefox
  - Internet Explorer
  - Headless Browser
    - Ghost Driver and Phantom JS
  - Mobile Browsers
    - Selendriod & Appium
  - Remote Web Driver
  - Capabilities
  - Profile setting
  - Selenium Grid

Selenium - How it works?



Different Drivers

Firefox:

WebDriver driver = new FirefoxDriver();

IE:

WebDriver driver = new InternetExplorerDriver();

Chrome:

WebDriver driver = new ChromeDriver();

Safari:

WebDriver driver = new SafariDriver();

Opera:

WebDriver driver = new OperaDriver()

GhostDriver and PhantomJs:

WebDriver driver = new PhantomJSDriver()

Different Drivers (Cont.)

- Firefox Driver:

```
1  // Create a new instance of the Firefox driver
2  WebDriver driver=new FirefoxDriver();
3  // opening Google
4  driver.get("https://www.google.com/");
5  // Closing driver
6  driver.close();
```

- IE Driver:

```
1  //Setting system property for IE driver
2  System.setProperty("webdriver.ie.driver", "/path/to/IEDriver");
3  // Create a new instance of the IE driver
4  WebDriver driver=new InternetExplorerDriver();
5  // opening Google
6  driver.get("https://www.google.com/");
7  // Closing driver
8  driver.close();
```

- Chrome Driver:

```
1  //Setting system property for Chrome driver
2  System.setProperty("webdriver.chrome.driver", "/path/to/chromedriver");
3  // Create a new instance of the Chrome driver
4  WebDriver driver=new ChromeDriver();
5  // opening Google
6  driver.get("https://www.google.com/");
7  // Closing driver
8  driver.close();
```

Different Drivers (Cont.)

- Headless Browser

  - Web browser without a graphical user interface
  - Normally, interaction with a website are done with mouse and keyboard using a browser with a GUI
  - While most headless browser provides an API to manipulate the page/DOM, download resources etc.
  - So instead of, for example, actually clicking an element with the mouse, a headless browser allows you to click an element by code
  - Headers, Local storage and Cookies work the same way
  - List of Headless Browsers
    - PhanthomJS
    - HtmlUnit
    - TrifleJS
    - Splash

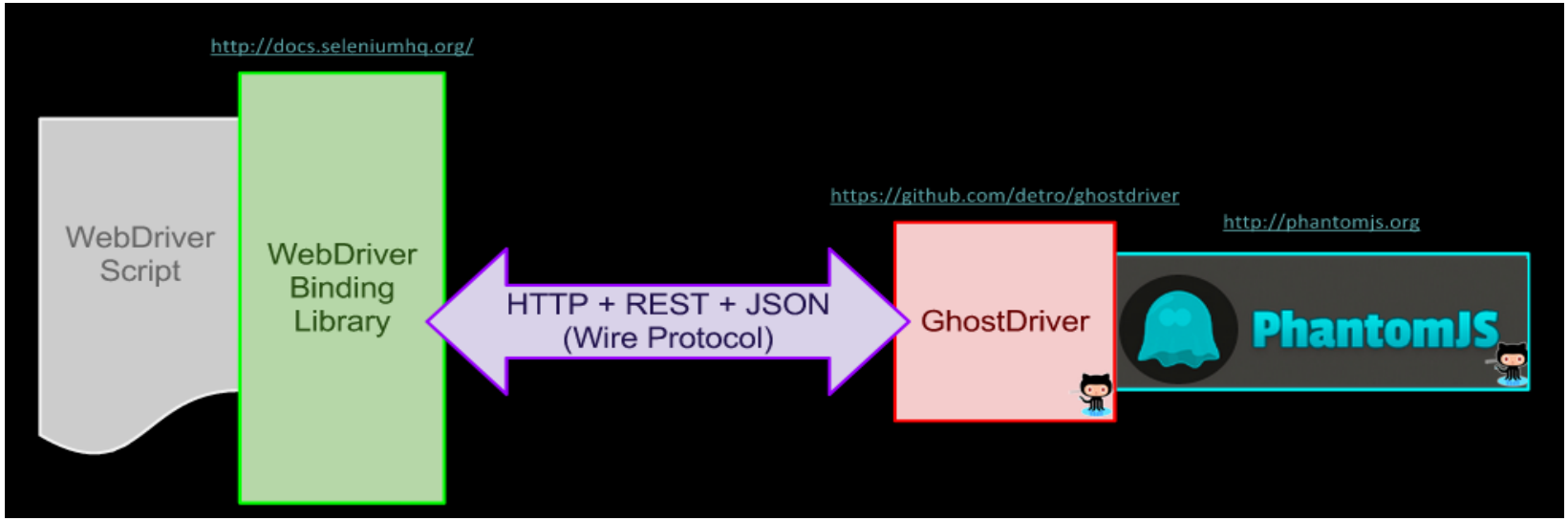
Different Drivers (Cont.)

- PhanthomJS (Headless Browser)
  - HEADLESS WEBSITE TESTING
    - Headless Web Kit with JavaScript API
  - SCREEN CAPTURE
    - Programmatically capture web contents, including SVG and Canvas
  - PAGE AUTOMATION
    - Access and manipulate webpages with the standard DOM API, or with usual libraries like jQuery
  - Example of interacting with a page using PhantomJS:

```
page.evaluate(function() {
        //Fill in form on page
        document.getElementById('Name').value = 'John Doe';
        document.getElementById('Email').value =
  'john.doe@john.doe';
        //Submit
        $('#SubmitButton').click();
    });
```

Different Drivers (Cont.)

- GhostDriver(Headless Browser)
  - Pure JavaScript implementation of the WebDriver Wire Protocol for PhantomJS Remote
  - WebDriver that uses PhantomJS as back-end



Different Drivers (Cont.)

- Mobile Browsers
  - Mobile web browsers differ greatly in terms of features offered an operating systems supported
  - Best can display most websites and offer page zoom and keyboard shortcuts, while others can only display websites optimizes for mobile devices
  - Appium and Selendroid are the two cross browser mobile automation tool

- Appium(Mobile Browser)
  - Open-source tool for automating native, mobile web, and hybrid applications on iOS and Android platforms, which is handled by a Appium node.js server.
  - Cross-platform which allows to write tests against multiple platforms (iOS, Android), using the same API
  - Enables code reuse between iOS and Android test suites

Different Drivers (Cont.)

Selendroid

Test automation framework which is used Android native & hybrid applications (apps) and mobile web

- Full compatibility with the JSON Wire Protocol
- No modification of app under test required in order to automate it
- Testing the mobile web using built in Android driver webview app
- UI elements can be found by different locator types
- Selendroid can interact with multiple Android devices (emulators or hardware devices) at the same time
- Existing emulators are started automatically
- Supports hot plugging of hardware devices
- Full integration as a node into Selenium Grid for scaling and parallel testing

Remote Web Driver

- Implementation class of the WebDriver interface that a test script developer can use to execute their test scripts via the Remote WebDriver server on a remote machine
- Needs to be configured so that it can run your tests on a separate machine
- If **driver is not Remote WebDriver**, communication to the web browser is local. So driver will be used as below:

  Webdriver driver = new FirefoxDriver();

  using driver will access Firefox on the local machine, directly
- If **Remote WebDriver**, needs location Selenium Server (Grid) web browser
- For example,

  WebDriver driver = new RemoteWebDriver(new URL("http://localhost:4444/wd/hub"), DesiredCapabilities.firefox());

Capabilities and Profile Setting

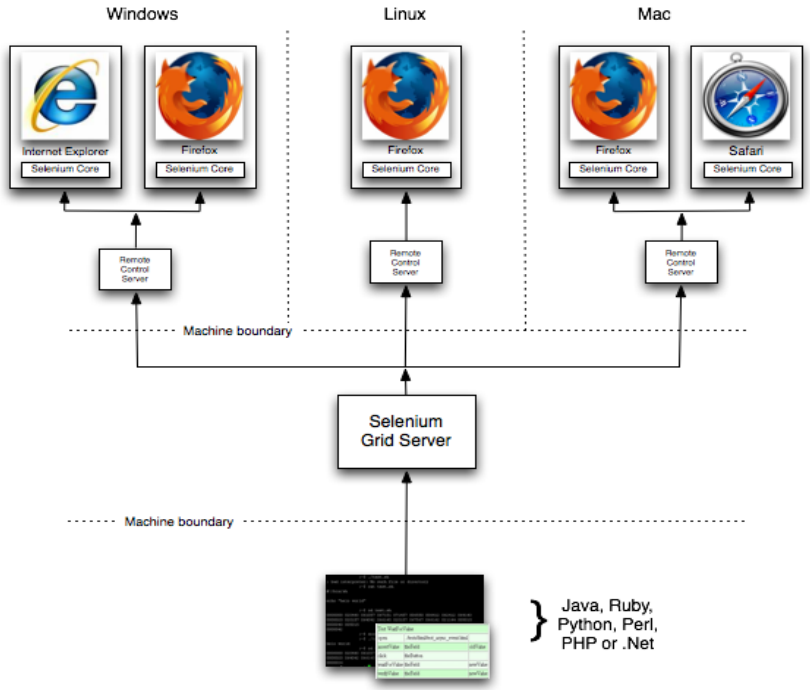
- Capabilities describes a series of key/value pairs that encapsulate aspects of a browser
- DesiredCapabilities set properties for the WebDriver
- Profile used to create custom Firefox profile and use it with desired capabilities

- Example to use Firefox profile with desired capabilities:

```
DesiredCapabilities dc=DesiredCapabilities.firefox();
FirefoxProfile profile = new FirefoxProfile();
dc.setCapability(FirefoxDriver.PROFILE, profile);
Webdriver driver =  new FirefoxDriver(dc);
```

Selenium Grid

Allows you run your tests on different machines against different browsers in parallel. That is, running multiple tests at the same time against different machines running different browsers and operating systems. Essentially, Selenium-Grid support distributed test execution. It allows for running your tests in a distributed test execution environment.



Summary

- In this lesson, you have learnt
- In this lesson, you have understood that how the selenium works and interacts with client server.
- Different drivers that are available for selenium-Chrome , IE ,Firefox ,headless browsers and mobile browsers.
- In remote webdriver the server will always run on the machine with the browser you want to test. And ,there are two ways to user the server: command line or configured in code.
- Capabilities gives facility to set the properties of browser. Such as to set BrowserName, Platform, Version of Browser.
- There are two reasons why you might want to use Selenium-Grid.
  - To run your tests against multiple browsers, multiple versions of browser, and browsers running on different operating systems.
  - To reduce the time it takes for the test suite to complete a test pass.

Review Question

- Question 1:
  - PhanthomJS is
  - Chrome Driver
  - Mobile Browser
  - Headless Browser
  - Firefox Driver
- Question 2: True/False
  - Firefox saves your information such as cookies and browser history in a file called your profile.
- Question 3: Fill in the Blanks
  - Client driver will send the program that is written in eclipse IDE as _____ and send it to Selenium server