|  |  |  |  |
| --- | --- | --- | --- |
| WebDriver initialization | | | |
| WebDriver driver; | | | WebDriver is an interface |
| driver = new ChromeDriver(); | | | Declaration for chrome browser |
| driver = new FirefoxDriver(); | | | Firefox browser |
| driver = new InternetExplorerDriver(); | | | Internet explorer browser |
| driver = new SafariDriver(); | | | Safari browser |
|  | | |  |
| ChromeOptions | | | |
| Chrome options class is used to manipulate various properties of Chrome driver. Chrome options class is generally used in conjunction with Desired Capabilities.  Below example shows a way to open Chrome browser in maximized mode using ChromeOptions class. We need to pass an instance of ChromeOptions class to the web driver initialization.  ChromeOptions options = new ChromeOptions();  options.addArgument("start-maximized");  ChromeDriver driver = new ChromeDriver(options);  Likewise we have  firefoxOptions  InternetExplorerOptions etc  More details:  <https://dzone.com/articles/desired-capabilities-in-selenium-testing-with-exam> | | | |  |  | | --- | --- | | List of ChromeOption arguments | | | start-maximized: | Opens Chrome in maximize mode | | incognito | Opens Chrome in incognito mode | | headless | Opens Chrome in headless mode | | disable-extensions: | Disables existing extensions on Chrome browser | | disable-popup-blocking: | Disables pop-ups displayed on Chrome browser | | make-default-browser | Makes Chrome default browser | | version | **Prints chrome browser version** | | disable-infobars | Prevents Chrome from displaying the notification 'Chrome is being controlled by automated software | |
| Desired Capabilities | | | |
| Desired capabilities class is used to modify multiple properties of web driver. Desired Capabilities class provides a set of key-value pairs to change individual properties of web driver such as browser name, browser platform, etc. Most commonly used method of Desired Capabilities class is setCapability method. Desired Capabilities is most frequently used with Selenium Grid where the same test case needs to be executed on different browsers.  More details:  <https://github.com/SeleniumHQ/selenium/wiki/DesiredCapabilities> | | | |  |  | | --- | --- | | List of DesiredCapabilities methods | | | getBrowserName() |  | | setBrowserName() |  | | getVersion() |  | | setVersion() |  | | getPlatform() |  | | setPlatform() |  | |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
| Selenium Navigations | | | |
| Navigate to url | driver.get(“http://newexample.com”)  driver.navigate().to(“http://newexample.com”) | | |
| Refresh page | driver.navigate().refresh() | | |
| Navigate forwards in browser history | driver.navigate().forward() | | |
| Navigate backwards in browser history | driver.navigate().back() | | |
|  |  | | |
| Selenium Locators | | | |
| By ID  <input id=”q” type=”text”>…</input> | | WebElement element = driver.findElement(By.id(“q”)) | |
| By Name  <input id=”q” name=”search” type=”text” /> | | WebElement element = driver.findElement(By.name(“search”)); | |
| By Class Name  <div class=”username” style=”display: block;”>…</div> | | WebElement element = driver.findElement(By.className(“username”)); | |
| By Tag Name  <div class=”username” style=”display: block;”>…</div> | | WebElement element = driver.findElement(By.tagName(“div”)); | |
| By Link Text  <a href=”#”>Refresh</a> | | WebElement element = driver.findElement(By.linkText(“Refresh”)); | |
| By Partial Link Text  <a href=”#”>Refresh Here</a> | | WebElement element = driver.findElement(By.partialLinkText(“Refresh”)); | |
| By XPath  <form id=”testform” action=”submit” method=”get”>  Username: <input type=”text” />  Password: <input type=”password” />  </form> | | WebElement element = driver.findElement(By.xpath(“//form[@id=’testform’]/input[1]”)); | |
| By CSS Selector  <form id=”testform” action=”submit” method=”get”>  <input class=”username” type=”text” />  <input class=”password” type=”password” />  </form> | | WebElement element = driver.findElement(By.cssSelector(“form#testform>input.username”)); | |
|  | |  | |
| Java Selenium commands for operation on Elements | | | |
| button/link/image | | click()  getAttribute()  isDisplayed()  isEnabled() | |
| Text field: | | sendKeys()  clear() | |
| Checkbox/Radio: | | isSelected()  click() | |
| Dropdown/Combobox/Select | | Select select = new Select(WebElement);  select.selectByIndex();  select.selectByValue();  select.selectByVisibleText();  select.deselectAll();  select.deselectByIndex();  select.deselectByValue();  select.deselectByVisibleText(); | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
| Element properties: | | isDisplayed()  isSelected()  isEnabled() | |
| Read Attribute | | getAttribute(“”) | |
| Get attribute from a disabled text box | | driver.findElement(By).getAttribute(“value”); | |
| Screenshot:  TakesScreenshot takeScreenshot = (TakesScreenshot) driver;  File file = takeScreenshot.getScreenshotAs(OutputType.FILE);  FileInputStream fis = new FileInputStream(file);  FileOutputStream fos = new FileOutputStream(new File(System.getProperty("user.dir")+"/test.jpg"));  int cursor;  while((cursor=fis.read())!=-1) {  fos.write(cursor);  }  fos.close();  fis.close(); | | | |
|  | |  | |
| Frames | | | |
| Using Frame Index | | driver.switchTo().frame(1); | |
| Using Name of Frame | | driver.switchTo().frame(“name”) | |
| Using web Element Object | | driver.switchTo().frame(element); | |
| Get back to main document | | driver.switchTo().defaultContent(); | |
|  | |  | |
| Alerts | | | |
| Fetch the text message from the alert window | | driver.switchTo().alert.getText(); | |
| Click on the Ok/Accept button | | driver.switchTo().alert.accept(); | |
| Click on Cancel button | | driver.switchTo().alert.dismiss(); | |
| Send text to a textbox in the alert window | | driver.switchTo().alert.sendKeys(“Text”); | |
|  | |  | |
| Window handler | | | |
| String handle=driver.getWindowHandle();  Set<String> handles = getWindowHandles();  driver.switchTo().window(handle);   |  |  | | --- | --- | | How to switch to newly created window | String curWindow=driver.getWindowHandle(); | | Get all window handles | Set<String> handles = getWindowHandles();  for(string handle: handles)  {  if (!handle.equals(curWindow))  {  driver.switchTo().window(handle);  }  } | | | | |
|  | |  | |