VM Options Browser specific scenario configuration Multiple Record Generation Implicit Wait Fluent Wait Browser Factory Browser Specific Assertion**VM Options - Browser Specific Scenario Specific ConfigurationMultiple Record Extraction and AssertionVM Options Browser Parameter and config file passing - Cucumber** -ea -DBROWSER=CHROME -ea -DCONFIGFILE=<path to config file> Configuration Loader **Multiple Record Extraction and Assertion//Get number of rows In table.** int Row\_count = driver.findElements(By.xpath("//\*[@id='post-body-6522850981930750493']/div[1]/table/tbody/tr")).size(); LOGGER.info("Number Of Rows = "+Row\_count); **//Get number of columns In table.** int Col\_count = driver.findElements(By.xpath("//\*[@id='post-body-6522850981930750493']/div[1]/table/tbody/tr[1]/td")).size(); LOGGER.info("Number Of Columns = "+Col\_count); **//divided xpath In three parts to pass Row\_count and Col\_count values.** String first\_part = "//\*[@id='post-body-6522850981930750493']/div[1]/table/tbody/tr["; String second\_part = "]/td["; String third\_part = "]"; for (int i=1; i<=Row\_count; i++){ for(int j=1; j<=Col\_count; j++){ String final\_xpath = first\_part+i+second\_part+j+third\_part; String Table\_data = driver.findElement(By.xpath(final\_xpath)).getText(); System.out.print(Table\_data +" "); } System.out.println(""); System.out.println(""); } }**Implicit Explicit and Fluent Wait**WebDriver driver = new FirefoxDriver();driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);Wait wait = new FluentWait(driver) .withTimeout(30, SECONDS) .pollingEvery(5, SECONDS) .ignoring(NoSuchElementException.class); WebElement foo = wait.until(new Function() { public WebElement apply(WebDriver driver) { return driver.findElement(By.id("foo")); }});

|  |
| --- |
| new WebDriverWait(driver, 10).until(ExpectedConditions.elementToBeClickable(By.id("someid")));**Browser specific Assertion** Capabilities cap = ((RemoteWebDriver) driver).getCapabilities(); String browserName = cap.getBrowserName().toLowerCase(); System.out.println(browserName); String os = cap.getPlatform().toString(); LOGGER.info(os); String v = cap.getVersion().toString(); System.out.println(v);**BROWSER FACTORY** public class BrowserFactory { public enum Browser { CHROME, FIREFOX, IEXPLORER; } final WebDriver webDriver; DesiredCapabilities capabilities; switch (browserType) { case FIREFOX: FirefoxProfile profile = new FirefoxProfile(); profile.setAssumeUntrustedCertificateIssuer(false); webDriver = new FirefoxDriver(profile); break; case IEXPLORER: webDriver = new InternetExplorerDriver(); break; case OPERA: webDriver = new OperaDriver(); break; case SAFARI: webDriver = new SafariDriver(); break; case CHROME: webDriver = new ChromeDriver(); break; case PHANTOMJS: capabilities = DesiredCapabilities.phantomjs(); webDriver = new PhantomJSDriver(capabilities); break; case RWD\_CHROME: capabilities = DesiredCapabilities.chrome(); capabilities.setCapability("webdriver.chrome.driver", "~/Documents/chromedriver"); webDriver = new RemoteWebDriver(new URL("http://127.0.0.1:4444/wd/hub"), capabilities); break; case RWD\_PHANTOMJS: capabilities = DesiredCapabilities.phantomjs(); capabilities.setCapability("phantomjs.binary.path", "/usr/bin/phantomjs"); webDriver = new RemoteWebDriver(new URL("http://127.0.0.1:4444/wd/hub"), capabilities); break; case RWD\_FIREFOX: capabilities = DesiredCapabilities.firefox(); webDriver = new RemoteWebDriver(new URL("http://127.0.0.1:4444/wd/hub"), capabilities); break; case RWD\_IE8\_VM: capabilities = DesiredCapabilities.firefox(); //capabilities.setCapability("webdriver.ie.driver", "C:\\selenium\\IEDriverServer.exe"); webDriver = new RemoteWebDriver(new URL("http://10.10.22.113:4444/wd/hub"), capabilities); break; case SAUCELABS: capabilities = DesiredCapabilities.firefox(); capabilities.setCapability("name", "Lexus Selenium"); webDriver = new RemoteWebDriver( new URL("http://" + saucelabsUsername + ":" + saucelabsKey + "[@ondemand.saucelabs.com:80/wd/hub](mailto:@ondemand.saucelabs.com:80/wd/hub)"), capabilities); break; default: throw new IllegalArgumentException("Invalid browser type set in class injection " + browserType.getBrowserTypeString()); } initWebDriver(webDriver); return webDriver; } |

|  |
| --- |
| **Graphics video Recording of Test Cases** <https://unmesh.me/2012/01/13/recording-screencast-of-selenium-tests-in-java/>import org.openqa.selenium.By; import org.openqa.selenium.WebDriver; import org.openqa.selenium.WebElement; import org.openqa.selenium.firefox.FirefoxDriver; import org.openqa.selenium.support.ui.ExpectedCondition; import org.openqa.selenium.support.ui.WebDriverWait; import org.monte.media.math.Rational; import org.monte.media.Format; import org.monte.screenrecorder.ScreenRecorder; import static org.monte.media.AudioFormatKeys.\*; import static org.monte.media.VideoFormatKeys.\*; import java.awt.\*; public class SeScreenCastDemo { public static void main(String[] args) throws Exception { //Create a instance of GraphicsConfiguration to get the Graphics configuration //of the Screen. This is needed for ScreenRecorder class. GraphicsConfiguration gc = GraphicsEnvironment// .getLocalGraphicsEnvironment()// .getDefaultScreenDevice()// .getDefaultConfiguration(); //Create a instance of ScreenRecorder with the required configurations ScreenRecorder screenRecorder = new ScreenRecorder(gc, new Format(MediaTypeKey, MediaType.FILE, MimeTypeKey, MIME\_AVI), new Format(MediaTypeKey, MediaType.VIDEO, EncodingKey, ENCODING\_AVI\_TECHSMITH\_SCREEN\_CAPTURE, CompressorNameKey, ENCODING\_AVI\_TECHSMITH\_SCREEN\_CAPTURE, DepthKey, (int)24, FrameRateKey, Rational.valueOf(15), QualityKey, 1.0f, KeyFrameIntervalKey, (int) (15 \* 60)), new Format(MediaTypeKey, MediaType.VIDEO, EncodingKey,"black", FrameRateKey, Rational.valueOf(30)), null); //Create a new instance of the Firefox driver WebDriver driver = new FirefoxDriver(); //Call the start method of ScreenRecorder to begin recording screenRecorder.start(); //And now use this to visit Google driver.get("[http://www.google.com](http://www.google.com/)"); //Find the text input element by its name WebElement element = driver.findElement(By.name("q")); //Enter something to search for element.sendKeys("Cheese!"); //Now submit the form. WebDriver will find the form for us from the element element.submit(); //Check the title of the page System.out.println("Page title is: " + driver.getTitle()); //Google's search is rendered dynamically with JavaScript. //Wait for the page to load, timeout after 10 seconds (new WebDriverWait(driver, 10)).until(new ExpectedCondition<Boolean>() public Boolean apply(WebDriver d) { return d.getTitle().toLowerCase().startsWith("cheese!"); }}); //Should see: "cheese! - Google Search" System.out.println("Page title is: " + driver.getTitle()); //Close the browser driver.quit(); //Call the stop method of ScreenRecorder to end the recording screenRecorder.stop(); } } |

**Screenshot Selenium - Place it in after method - Listeners** WebDriver driver = new FirefoxDriver();driver.get("http://www.google.com/");File scrFile = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);// Now you can do whatever you need to do with it, for example copy somewhereFileUtils.copyFile(scrFile, new File("c:\\tmp\\screenshot.png"));@After("@UI" )public void embedScreenshotOnFail(Scenario s) { if (s.isFailed()) { try { byte[] screenshot = ((TakesScreenshot) getDefaultDriver()).getScreenshotAs(OutputType.BYTES); s.embed(screenshot, "image/png" ); s.write("URL at failure: " + getDefaultDriver().getCurrentUrl()); } catch (WebDriverException wde) { s.write("Embed Failed " + wde.getMessage()); } catch (ClassCastException cce) { cce.printStackTrace(); } }}