To find xpath easily\_ and to create manual test case steps\_\_\_\_\_\_\_ naveen

selectorshub url : [https://selectorshub.com/](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbHhJOVVhMEZMRlFKVTByQU1CNDU5ci1sMjc1UXxBQ3Jtc0tsTmtCUG5ERWdtU3VIY013LXIzS1pMT2hPZWtONURuTlo5ZVdhSmQxN0lsVHk0Q1g4THMxTE0zb254UTZPVGRYMVhlRmNYQlJEZjhMLVlSc1JwQlBlTVk3VHotYWpMbjhyQmlGMWsxZnJDNFFhZ3I4TQ&q=https%3A%2F%2Fselectorshub.com%2F&v=HLcKIl5lhq4) test case studio: WHAT IS TESTCASE STUDIO? : [https://selectorshub.com/testcase-stu...](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbGRMaDN1WHkzM3RoS3BlSzlwa2VyLV9CY3lnUXxBQ3Jtc0ttVGdiSVc2Sm5IZWk3TEhYY1FJVi1wcDhxNVB5T0FOWDRHN0hxVnlfVTZjRFdlQVNKOXdOR09VZzVSNU8xMEtyR1FCV0tkTWxSbm92czlWTzduWHFRcXRGZW1sRi0xbURxaU51QnJmMzdRZnNuM3JIYw&q=https%3A%2F%2Fselectorshub.com%2Ftestcase-studio%2F&v=HLcKIl5lhq4)

**Webdriver commands:**

* driver.get("[http://www.facebook.com](http://www.facebook.com/)");
* driver.getCurrentUrl();
* String CurrentUrl = driver.getCurrentUrl();
* driver.getPageSource();
* String PageSource = driver.getPageSource();
* driver.close(); //close current window.
* driver.quit(); //close all windows.
* driver.navigate().to( "[http://www.facebook.com](http://www.facebook.com/)" );
* driver.navigate().forward();
* driver.navigte().back();
* driver.navigate().refresh();
* WebElement element = driver.findElement(By.id(“value”));
* element.click();
* element.clear();
* driver.findElement(By.id(“value”)).clear();
* element.sendKeys(“uid”);
* driver.findElement(By.id(“value”)).sendKeys(“uid”);
* driver.manage().window().maximize();
* driver.manage().window().setPosition(new point(0, -1000)); // minimize window.
* int width = 600;

int height = 400;

Dimension dimension = new Dimension(width, height);

driver.manage().window().setSize(dimension); // resize window.

* Dimension dimension = element.getSize();

System.out.println(“Height: ” + dimension.height + “width: ” + dimension.width);

* boolean status = element.isDisplayed();
* boolean status = driver.findElement(By.id(“value”)).isDisplayed();
* boolean status = element.isEnabled();
* boolean status = element.isSelected();
* element.submit();
* String text = element.getText();
* String tagName = element.getTagName();
* String cssValue = element.getCssValue();
* WebElement element = driver.findElement(By.id(“submitButton”));

String attributeValue = element.getAttribute(“id”); // will return “submitButton”.

* Point point = element.getLocation();

System.out.println(“x coordinate: ” + point.x + “ y coordinate ” + point.y);

* driver.findElement(By.id(“idValue”));
* driver.findElement(By.name(“nameValue”));
* driver.findElement(By.className(“classValue”));
* driver.findElement(By.tagName(“tagNameValue”));
* driver.findElement(By.linkText(“linkTextValue”));
* driver.findElement(By.partialLinkText(“partialLinkTextValue”));
* driver.findElement(By.xpath(“xpathValue”));
* driver.findElement(By.cssSelector(“cssSelectorValue”));
* checkBox/radioButton

WebElement radioButton = driver.findElement(By.id(“idValue”));

radioButton.click();

List <WebElement> oradioButton = driver.findElements(By.name(“nameValue”));

boolean status = oradioButton.get(0).isSelected();

if(status == false){

oradioButton.get(0).click();

}

* List <WebElement> ocheckBox = driver.findElements(By.name(“nameValue”));

int checkBoxSize = ocheckBox.size();

//store check box name using value attribute.

for(int i = 0 ; i < ocheckBoxSize; i++ ){

String sValue = ocheckBox.get(i).getAttribute(“value” / “id”/ “locator name”);

if(sValue.equalsIgnoreCase(“toolsqa”)){

ocheckBox.get(i).click();

break;

}

}

* driver.findElement(By.id(“checkBox”)).click();
* WebElement element = driver.findElement(By.id(“country”));

Select oselect = new Select(element);

oselect.selectAll();

oselect.selectByVisibleText(“value”);

oselect.selectByIndex(4);

oselect.selectByValue(“value”);

boolean status = oselect.isMultiple();

oselect.deselectAll();

oselect.deselectByVisibleText(“value”);

oselect.deselectByIndex(4);

oselect.deselectByValue(“value”);

List <WebElement> optionList = oselect.getOptions();

int totalOptions = optionList.size();

for(int i=0 ; i<totalOptions ; i++){

String sValue = optionList.get(i).getText();

System.out.println(sValue);

}

List <WebElement> selected\_options = oselect.getAllSelectedOptions();

boolean status = oselect.isMultiple();

String fisrtSelectedOption = oselect.getFirstSelectedOption().getText();

# **WebDriverWait wait = new** **WebDriverWait(driver, Duration.ofSeconds(30));** for selenium 4.

# **WebDriverWait wait = new** **WebDriverWait(driver, 30);** for selenium 3.

// Wait till the element is not visible

WebElement element=wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("value")));

-------------------------------------------------------------------------------------------------------------

**We can handle dropdowns in 3 ways ….**

i. if it has Select tag then using Select class

ii. if it has no Select tag then simply click on the menu and then click the option.

iii. if it has no Select tag then simply click on the menu and then get list of options then using for loop get the text of options and then click the option using if else condition.

=>public class BootstrapDropdown {

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

FirefoxDriver driver = new FirefoxDriver();

driver.get("[https://www.edureka.co](https://www.edureka.co/)");

driver.get("http://seleniumpractise.blogspot.in/2016/08/bootstrap-dropdown-example-for-selenium.html");

driver.findElement(By.className("icon-Home-page-humburger-menu-01")).click();

Thread.sleep(2000);

//Dropdown items are coming in <a> tag so below xpath will get all elements and findElements will return list of WebElements.

List<WebElement> list = driver.findElements(By.xpath("//ul[@id='hiddencat']//li/a"));

for (WebElement element : list)

{

System.out.println("Values " + element.getAttribute("innerHTML"));

}

if (element.getAttribute("innerHTML").contains("JavaScript")) {

element.click();

breake;

}

### Approach to handle calendar in Selenium Webdriver

Step 1- Click on calendar

Step 2- Get all td (column element) of tables using findElements method

Step 3- using for loop get text of all column elements

Step 4- using if else condition we will check specific date

Step 5- If date is matched then click and break the loop.

driver.get("https://demoqa.com/");

**public** **class** ConfirmDateOnCalendar {

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

*confirmDate*();

}

**public** **static** **void** confirmDate(**String** day) {

//store the date in String format

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("please enter a date");

String day = String.*valueOf*(sc.nextInt());

//go to url

WebDriver driver=**new** ChromeDriver();

driver.get("http://seleniumpractise.blogspot.in/2016/08/how-to-handle-calendar-in-selenium.html");

//click on calendar

driver.findElement(By.*id*("datepicker")).click();

//click on specific date

List<WebElement> columns=driver.findElements(By.*xpath*("//table[@class='ui-datepicker-calendar']//td"));

**for**(WebElement column : columns){

String date=column.getText();

**if**(date.equalsIgnoreCase(day)){

column.click();

**break**; //break the loop

}

}

System.***out***.println("Date is confirmed successfully");

}

}

**public class Calandar\_custom** {

static WebDriver driver;

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

departureDate("July 2022", "16");

}

**public static void departureDate(String month\_year, String day)** {

WebDriverManager.chromedriver.setup();

driver = new ChromeDriver();

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

driver.manage().window().maximize();

driver.findElement(By.xpath("//input[@id='departureCalendar']")).click();

new WebDriverWait(driver, *Duration.ofSeconds(20)*).until(ExpectedConditions.visibilityOfAllElementsLocatedBy(By.xpath("//div[@class='DayPicker-Months']")));

String monthYearValue = driver.findElement(By.xpath("//div[@class='DayPicker-Caption']")).getText();

System.out.println(monthYearValue);

while(!(monthYearValue.equalsIgnoreCase(month\_year))) {

driver.findElement(By.xpath("//span[@class='DayPicker-NavButton DayPicker-NavButton--next']")).click();

monthYearValue = driver.findElement(By.xpath("//div[@class='DayPicker-Caption']")).getText();

System.out.println(monthYearValue);

}

driver.findElement(By.xpath("//div[text()=' " +day+ " ']")).click();

System.out.println("departure date selected");

}

}

String month\_year = driver.findElement(By.*xpath*("//\*[@class='DayPicker-Caption']")).getText();

System.***out***.println("month year: "+ month\_year);

String month = month\_year.split(" ")[0].trim();

String year = month\_year.split(" ")[1].trim();

System.***out***.println("month : "+ month+ " year : "+year);

=> Robot Class.

* Using Robot Class we can handle KeyBoard and mouse events in Selenium.
* Robot Class can help in upload / download files using selenium web driver.
* Methods in Robot Class can be effectively used to do the interaction with popups in Web

. Applications.

* Robot Class can easily integrated with current automation framework (keyword, data-driven or . hybrid).

# <https://demoqa.com/keyboard-events/>

# <https://docs.oracle.com/javase/10/docs/api/java/awt/Robot.html>

# <http://demo.guru99.com/selenium/upload/>

import java.awt.Robot;

import java.awt.event.InputEvent;

import java.awt.event.KeyEvent;

public class Robot\_Class {

public static void main(String[] args) {

System.setProperty("webdriver.gecko.driver","D:\\geckodriver.exe");

WebDriver driver = new FirefoxDriver();

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

// to open note pad.

String path = “c:\\windows\\notepad.exe”;

RunTime run = RunTime.getRunTime();

run.exec(path);

// ALT+F to open the file menu.

// ALT+F+X will close Notepad.

// ALT+F4 will also close a window.

How do I find the path of an application in Windows?

**Right-click on the shortcut of the program - Properties - Shortcut - file location or path.**

// Create object of Robot class

Robot robot = new Robot();

robot.setAutoDelay(3000);

**// to copy file path for uploading a file.**

StringSelection stringSelection = new StringSelection(“file\_path with double backSlash”);

Toolkit.getDefaultToolkit().getSystemClipBoard().setContents(stringSelection, null);

**// to paste file path.**

robot.keyPress(KeyEvent.VK\_control);

robot.keyPress(KeyEvent.VK\_V);

robot.keyRelease(KeyEvent.VK\_control);

robot.keyRelease(KeyEvent.VK\_V);

robot.setAutoDelay(3000);

robot.keyPress(KeyEvent.VK\_ENTER);

robot.keyRelease(KeyEvent.VK\_ENTER);

// This will press Escape key on keyboard.

robot.keyPress(KeyEvent.VK\_ESCAPE);

// This will release the CAPS\_LOCK key.

robot.keyRelease(KeyEvent.VK\_CAPS\_LOCK);

To press down arrow key on the Keyboard we use (robot.keyPress(KeyEvent.VK\_DOWN));

To press the TAB key of keyboard we use (robot.keyPress(KeyEvent.VK\_TAB));

To press Enter key we use (robot.keyPress(KeyEvent.VK\_ENTER));

// This will press Left mouse button.

robot.mousePress(InputEvent.BUTTON1\_DOWN\_MASK);

// This will release Left mouse button.

robot.mouseRelease(InputEvent.BUTTON1\_DOWN\_MASK);

Point coordinates = driver.findElement(By.id("img\_location")).getLocation();

robot.mouseMove(coordinates.getX(), coordinates.getY());

robot.mouseMove(40, 130);

// Press Left button

robot.mousePress(InputEvent.BUTTON1\_MASK);

// Release Left button

robot.mouseRelease(InputEvent.BUTTON1\_MASK);

// Press Right button

robot.mousePress(InputEvent.BUTTON3\_MASK);

// Release Right button

robot.mouseRelease(InputEvent.BUTTON3\_MASK);

// Scroll MouseWheel

robot.mouseWheel(5);

robot.mouseWheel(-100);

// Get Screen Size

java.awt.Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();

// Capture ScreenShot

BufferedImage img = robot.createScreenCapture(new Rectangle(screenSize));

// Store image into file

File path = new File("D://profile.jpg");

// Write image path

ImageIO.write(img, "JPG", path);

**=> Handle windows:**

-- driver.get("https://naukri.com/");

* String mainWindow = driver.getWindowHandle()

Set<String> allwindows = driver.getWindowHandles();

Iterator<String> itr = allwindows.iterator();

while(itr.hasNext()){

String childWindow = itr.next();

if(!mainWindow.equals(childWindow)){

driver.switchTo().window(childWindow);

System.out.println(driver.switchTo().window(childWindow).getTitle());

driver.close();

}}

driver.switchTo().window(mainWindow);

// by naveen.

**public** **static** **boolean** switchToRightWindow(String titleWord) {

Set <String> windows = *driver*.getWindowHandles();

List <String> windowList = **new** ArrayList <String>(windows);

**for**(String window : windowList) {

String windowTitle = *driver*.switchTo().window(window).getTitle();

**if**(windowTitle.contains(titleWord)) {

Reporter.*log*("found right window" , **true**);

Reporter.*log*("window title : "+ *driver*.getTitle(), **true**);

**return** **true**;

}

}

**return** **false**;

}

Note: the "return" statement terminates not only the loop but also the entire function in which it is used.

**public** **static** **void** closeAllChildTabs() {

String mainWindow = *driver*.getWindowHandle();

Set <String> windows = *driver*.getWindowHandles();

List <String> windowList = **new** ArrayList <String>(windows);

**for**(String windo : windowList) {

**if**(!windo.equals(mainWindow)) {

*driver*.switchTo().window(windo).close();

Reporter.*log*("closed all tabs except main tab");

}

}

}

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

driver.switchTo().window(switchToRightWindow);

}

=======================\*\*\*\*\* IFrame \*\*\*\*\*=========================

=> right click on the page then we can see if there is "this frame" , it means it has frames.

and we can check iframe in 'view page source'. after right click.

# frames using 3 ways;

i.by name or id,

ii. by index.

iii. by webElement.

= we can identify the frames element in google chrome elemenet inspector typing //iframe .

Then we can see frame id, name or other locators.

then we can write xpath and use as WebElement in Eclipse.

= after switching frame we have to come back main frame, then switch to next frame if needed.

driver.get("http://demo.guru99.com/test/guru99home/");

driver.switchTo().frame(frame\_name);

int total\_frames = driver.findElements(By.tagName("iframe")).size();

System.out.println(total\_frames);

driver.switchTo().defaultContent();

WebElement selection = driver.findElement(By.id("selection"));

List <WebElement> options = selection.findElements(By.tagName("option"));

for(WebElement option : options){

if( option.getText().equls("Germany")){

option.click();

break;

}

or

WebElement dropDown = driver.findElement(By.id("value"));

dropDown.click();

WebElement dropDownOne = driver.findElement(By.id("value"));

dropDownOne.click();

Actions actions = new Actions(driver);

actions.contextClick(element).build().perform(); // right click.

actions.doubleClick(element).perform(); // double click.

actions.moveToElement(element).click().perform();

actions.dragAndDrop(from\_element, target\_element).perform();

actions.clickAndHold(element).perform();

actions.release().perform();

Alert : driver.get("file:///F:/Work/Selenium/Testing-Presentation/DemoWebPopup.html");

Alert alert = driver.switchTo.alert();

alert.accept();

alert.dissmiss();

driver.switchTo().alert().getText(); // To capture alert message.

driver.switchTo().alert().sendKeys("Text"); //To send data to alert box.

**# how to know entire page height and width ?**

**=> right click > inspect > console > type document.body.scrollHeight;**

**=> right click > inspect > console > type document.body.scrollWidth;**

JavaScriptExecutor je = (JavaScriptExecutor) driver;

je.executeScript("arguments[0].scrollIntoView(true);", element);

je.executeScript("scroll(0, 400)");

je.executeScript("scrollBy(0, 2500)");

je.executeScript(“window.scrollBy(0, 1000)”);

je.executeScript(“window.scrollTo(0, 1000)”);

je.executeScript(“window.scrollTo(0, document.body.scrollHeight)”);

je.executeScript("window.scrollTo(arguments[0], arguments[1] )”, element.getLocation().x, element.getLocation().y);

// for highlight.

je.executeScript("arguments[0].setAttribute('style, 'border : solid 2px red; ' ');", element));

je.executeScript("arguments[0].setAttribute('style', 'border : solid 2px white' );", element));

je.executeScript("arguments[0]

.setAttribute('style', 'background : yellow; border : solid 2px red ' ');", element);

* String page\_title = driver.getPageTitle();

Assert.assertTrue(page\_title.contains("google"));

Assert.assertEquals(12,13, "number did not match");

SoftAssert assertion = new SoftAssert ();

assertion.assertTrue(page\_title.contains("google"));

assertion.assertEquals(12,13, "number did not match");

assertion.assertAll();

* DesiredCapabilities cap = DesiredCapabilities.chrome();

cap.setCapability(CapabilityType.ACCEPT\_SSL\_CERTS, true);

WebDriver driver = new ChromeDriver(cap);

// same for all browser.

CapabilityType.ACCEPT\_SSL\_CERTS // to accepr ssl certificate.

CapabilityType.SUPPORTS\_APPLICATION\_CACHE // to enable app cache.

CapabilityType.SUPPORTS\_FINDING\_BY\_CSS // to enable css selector.

CapabilityType.SUPPORTS\_JAVASCRIPT // to enable javascript.

CapabilityType.TAKES-SCREENSHOT // enable screenshot ability.

CapabilityType.SUPPORT\_WEB\_STORAGE // enable storage capability.

CapabilityType.SUPPORTS\_ALERTS // enable window pop up handle.

* String coockie = driver.manage().getCoockieNamed(coockie name).getValue();

Set < Coockie > coockis = driver.manage().getCoockies();

for(Coockie coockie : coockis){

String coockieName = coockie.getName();

String coockieValue = coockie.getValue();

System.out.println(" coockie value" + coockie

+ " coockieName "+ coockieName

+ " coockieValue "+ coockieValue);

}

Coockie coockie = new Coockie("coockieName", "coockieValue");

driver.manage().addCoockie(coockie);

driver.manage().deleteCoockieNamed(coockieName);

driver.manage().deleteCoockie(coockieName);

driver.manage().deleteAllCoockies();

* element.sendKeys(Keys.ENTER);
* element.sendKeys(Keys.RETURN);
* element.sendKeys(Keys.SHIFT);
* element.sendKeys(Keys.TAB);
* element.sendKeys(Keys.F5);
* driver.findElement(By.id(“value”)).sendKeys(Keys.5);
* driver.findElement(By.id(“value”)).sendKeys(Keys.control+"a"); //block the text.
* driver.findElement(By.id(“value”)).sendKeys(Keys.control+"c"); // copy the text.
* driver.findElement(By.id(“value”)).sendKeys(Keys.control+"v"); // paste the text.

=========================> **Broken Links** <==========================

=> List < WebElement > allLinks = driver.findElements(By.tagName("a"); // to get all links.

for(WebElement link : allLinks){

System.out.println("the links are :"+ link);

public class VerifyLinks {

public static void main(String[ ] args)

{

WebDriver driver=new FirefoxDriver();

driver.manage().window().maximize();

driver.get("http://www.google.co.in/");

List<WebElement> links=driver.findElements(By.tagName("a"));

System.out.println("Total links are "+links.size());

for(int i=0; i<links.size(); i++)

{

WebElement element= links.get(i);

String url=element.getAttribute("href");

verifyLinkActive(url);

}

}

public static void verifyLinkActive(String linkUrl)

{

try

{

URL url = new URL(linkUrl);

HttpURLConnection httpURLConnect=(HttpURLConnection)url.openConnection();

httpURLConnect.setConnectTimeout(3000);

httpURLConnect.connect();

if(httpURLConnect.getResponseCode()==200)

{

System.out.println(linkUrl+" - "+httpURLConnect.getResponseMessage());

}

if(httpURLConnect.getResponseCode()==HttpURLConnection.HTTP\_NOT\_FOUND)

{

System.out.println(linkUrl+" - "+httpURLConnect.getResponseMessage() + " - "+ HttpURLConnection.HTTP\_NOT\_FOUND);

}

} catch (Exception e) {

System.out.println("Exception cause : "+ e.getCause());

System.out.println(" Exception is : " + e.getMessage());

}

}

* **Data base test**

Class.forName("com.mysql.jdbc.driver"); // driver loaded.

Connection con = DriverManger.getConnection("jdbc:mysql://hostName:port/dbName","uid","pass");

Statement smt = con.createStatement();

ResultSet rs = smt.executequery("select \* from Customer");

while(rs.next()){

String name = rs.getString("name");

String email = rs.getString("email");

String address = rs.getString("address");

System.out.println(name + email+ address);

}

int invalidImageCount = 0;

WebDriver driver = new FirefoxDriver();

driver.get("[http://google.com](http://google.com/)");

List<WebElement> imagesList = driver.findElements(By.tagName("img"));

System.out.println("Total no. of images are :" + imagesList.size());

for (WebElement imgElement : imagesList) {

if (imgElement != null) {

try {

HttpClient client = HttpClientBuilder.create().build();

HttpGet httpGet = new HttpGet(imgElement.getAttribute("src"));

HttpResponse response = client.execute(httpGet);

**// verifying response code the HttpStatus should be 200 if not,**

**// increment as invalid images count**

if (response.getStatusLine().getStatusCode() != 200)

invalidImageCount++;

} catch (Exception e) {

e.printStackTrace();

}

}

}

System.out.println("Total no. of invalid images are " + invalidImageCount);

* **Cross Browser test**

|  |  |
| --- | --- |
| **package crossbrowserTest**  **public class VerifyTitle {**  WebDriver driver;  @Test  @Parameter("browser")  public void crossTest( String browser){  if(browser.equalsIgnoreCase("firefox"){  driver = new FirefoxDriver();  }  else if(browser.equalsIgnoreCase("chrome"){  driver = new ChromeDriver();  }  else if(browser.equalsIgnoreCase("ie"){  driver = new InternetExplorerDriver();  }  driver.manage().window().maximize();  driver.get("http://www.abc.com");  System.out.println(driver.getTitle());  driver.quit(); | ?xml version="1.0" encoding="UTF-8"?>  <!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">  <suite name="TestSuite" parallel="tests" >  <test name="FirefoxTest">  <parameter name="browser" value="Firefox" />  <classes>  <class name="crossBrowserTest.VerifyTitle">  </class>  </classes>  </test>  <test name="ChromeTest">  <parameter name="browser" value="Chrome" />  <classes>  <class name="crossBrowserTest.VerifyTitle">  </class>  </classes>  </test>  <test name="ie">  <parameter name="browser" value="ie" />  <classes>  <class name="crossBrowserTest.VerifyTitle">  </class>  </classes>  </test>  </suite> |

**===================apache commons email========================**

If you want to send email from a **Java**application that uses **Gmail/yahoo**, you need to allow: less secure apps:

On your browser, sign in to your **Gmail**, and acccess the following link:

<https://www.google.com/settings/security/lesssecureapps>

**A simple text email**

Email email = new SimpleEmail();

email.setHostName("smtp.googlemail.com");

email.setSmtpPort(465);

email.setAuthentication( new DefaultAuthentication("userName","passWord"));

email.setSSLOnConnect(true);

email.setFrom("user@gmail.com");

email.setSubject("testMail");

email.setMsg("this is a test mail");

email.addTo("foo@bar.com");

email.send();

**Sending email with attachments**

EmailAttachment attachment = new EmailAttachment();

attachment.setPath("path of attachment");

attachment.setDisposition(EmailAttachment.ATTACHMENT);

attachment.setdescriotion("picture of john");

attachnent.setName("john");

MultiPartEmail email = new MultiPartEmail();

email.setHostName("mail.myserver.com");

email.addTo("examplemail.com");

email.setFrom("user@gmail.com");

email.setSubject("picture");

email.setMsg("this is a picture");

email.attach(attachment);

email.send();

----------------------------------------------------------------------------------------------------------------------

**Extent Report version- 3.1.5**

public class Main {

public static void main(String[] args) {

// start reporters

ExtentHtmlReporter htmlReporter = new ExtentHtmlReporter("extent.html");

// create ExtentReports and attach reporter(s)

ExtentReports extent = new ExtentReports();

extent.attachReporter(htmlReporter);

// creates a toggle for the given test, adds all log events under it

ExtentTest test = extent.createTest("MyFirstTest", "Sample description");

// log(Status, details)

test.log(Status.INFO, "This step shows usage of log(status, details)");

// info(details)

test.info("This step shows usage of info(details)");

// log with snapshot

test.fail("details", MediaEntityBuilder.createScreenCaptureFromPath("screenshot.png").build());

// test with snapshot

test.addScreenCaptureFromPath("screenshot.png");

// calling flush writes everything to the log file

extent.flush();

**//open Extent report after test is done**

String pathOfExtentReport = System.*getProperty*("user.dir")+"\\pathOfReport";

File ExtentReportFile = **new** File(pathOfExtentReport);

**try** {

Desktop.*getDesktop*().browse(ExtentReportFile.toURI());

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

------------------------------------------------------------------------------------------------------

**File upload / download : by autoIt and Robot class API.**

**//upload to...**

**-** driver.get("http://demo.guru99.com/test/upload/ ");

driver.get("http://demo.guru99.com/test/upload/");

WebElement uploadElement = driver.findelement(By.id("value"));

uploadElement.sendKeys("file path");

NB: in autoIT for compiling file select '**compile script**' option.

**//download : by autoIt and Robot class API.**

in autoIt help ... type .. InetGet.. then use the command and parameters.

we can take template from the InetGet.

**Example:**

#include <InetConstants.au3>

#include <MsgBoxConstants.au3>

#include <WinAPIFiles.au3>

; Download a file in the background.

; Wait for the download to complete.

Example()

Func Example()

; Save the downloaded file to the temporary folder.

Local $sFilePath = "C:\Users\Elias Karim\Downloads\FileDownLoad\IEdriver.zip"

; Download the file by waiting for it to complete. The option of 'get the file from the local cache' has been selected.

Local $iBytesSize = InetGet("https://selenium-release.storage.googleapis.com/3.150/IEDriverServer\_x64\_3.150.1.zip", "C:\Users\Elias Karim\Downloads\FileDownLoad\IEdriver.zip", $INET\_FORCERELOAD)

EndFunc ;==>Example

**java reflection:**

different ways to create an object of a Class.

**class Animal{……}**

Class c = Class.forName(“Animal”); // doesn’t work !!!

**Animal a = new Animal(); // works fine ….**

**Class c = a.getClass();**

**Class c = Animal.class(); // works okay….**

|  |  |
| --- | --- |
| **public class Animal {**  public void dog(){  System.out.println(" this is dog method");  }  public void cat(){  System.out.println(" this is cat method");  }  public void cow(){  System.out.println(" this is cow method");  } | **public class ReflectionGuide {**  **@Test**  **public void TestMethod(){**  Class cls = Animal.class();  Method methodList [ ] = cls.getDeclairedMethods();  for(int i = o; i < methodlist.length; i++){  String methodNames = methodList[i].getName();  }  String className = methodlist[0].getDeclaringClass();  } |

---------------------------------------------------------------------------------------------------------------------------

**Takes Screenshot**

**public class** Utility

**public static void** capturScreenShot(WebDriver driver, String screenshotName){

try{

TakesScreenshot ts = (TakesScreenshot), driver;

File source = ts.getSceenshotAs(OutputType.FILE);

FileUtils.copyFile(source, new File("./Screenshot/"+screenshotName+".png"));

System.out.println("screenshot taken");

}

catch (Exeption e)

{

System.out.println("exeption is: "+ e.getMessage());

}

NB: add apache FileUtils maven to POM.

**Keyword driven test**

|  |
| --- |
| **public class ExcelUlity {**  File file;  FileInputStream fis;  XSSFWorkbook wb;  **public string InputData(**String sheetNo, String rowNo, Stringt cellNo**){**  file = new File("file path");  fis = new FileInputStream(file);  wb = new XSSWorkbook(fish);  XSSFSheet sheet1 = wb.getSheetAt(sheetNo);  String data1 = sheet1.getRow(rowNo).getCellNo(cellNo).getStringCellValue();  return data1;  }  **public void KeyData(**String sheetNo, String rowNo, Stringt cellNo**){**  file = new File("file path");  fis = new FileInputStream(file);  wb = new XSSWorkbook(fish);  XSSFSheet sheet1 = wb.getSheetAt(sheetNo);  String data2 = sheet1.getRow(rowNo).getCellNo(cellNo).getStringCellValue();  return data2;  } |

|  |  |
| --- | --- |
| public class ExecutionEngine{  @Test  public void testmethod(){  ActionKeyword keyword = new ActionKeyword();  for(int i = 0; i<=5; i++){  String key = ExcelUtility.keyData(1,i,2);  if(key.equalsIgnoreCase("openBrowser")){  keyword.openBrowser();  }  if(key.equalsIgnoreCase("gotoUrl")){  keyword.gotoUrl();  }  if(key.equalsIgnoreCase("enterUId")){  keyword.enterUid();  }  if(key.equalsIgnoreCase("enterPass")){  keyword.enterPassword();  }  if(key.equalsIgnoreCase("clickSubmit")){  keyword.enterSubmitButton();  }  driver.manage().window().maximize();  driver.manage().timeOuts().implicitlyWait(10, TimeUnit.SECONDS); | **public class ActionKeyword {**  **public void openBrowser(){**  webDriver driver = new FirefoxDriver();  **}**  **public void gotoUrl(){**  driver.get("http://www.facebook.com");  **}**  **public void enterUid(){**  driver.findElent(By.xpath("value");  **}**  **public void enterPassword(){**  driver.findElent(By.xpath("value");  **}**  **public void clickSubmitButton(){**  driver.findElement(By.xpath("value");  **}** |

Page Object Model with Page Factory

|  |  |
| --- | --- |
| **public class** LoginPage  {  WebDriver driver;  By username = By.id("value");  By password = By.xpath("xpath");  By loginButton = By.name("value");  **public** loginPage(WebDriver driver)  {  this.driver = driver;  }  **public void** typeUserName(){  driver.findElement(username).sendKeys("admin");  }  **public void** typePassWord(String pass){  driver.findElement(password).sendKeys("demo1");  }  public void clickLoginButton(){  driver.findElement(loginButton).click();  } | **public class** LoginWordPress{  **public void** verifyLogin()  {  WebDriver driver = new FirefoxDriver();  driver.get("http://www.facebook.com");  LoginPage login = new LoginPage(driver);  login.typeUserName();  login.typePassWord();  login.clickLogin();  driver.quit();  } |
| **public class** LoginPage  {  WebDriver driver;  By username = By.id("value");  By password = By.xpath("xpath");  By loginButton = By.name("value");  **public** loginPage(WebDriver driver)  {  this.driver = driver;  }  **public void** typeUserName(String uid){  driver.findElement(username).sendKeys("uid");  }  **pubic void** typePassWord(String pass){  driver.findElement(password).sendKeys("pass");  }  public void clickLoginButton(){  driver.findElement(loginButton).click();  } | **public class** LoginWordPress{  **public void** verifyLogin()  {  WebDriver driver = new FirefoxDriver();  driver.get("http://www.facebook.com");  LoginPage login = new LoginPage();  login.typeUserName("admin");  login.typePassWord("demo1");  login.clickLogin();  driver.quit();  } |

|  |  |
| --- | --- |
| **public class** LoginPageNew  {  WebDriver driver;  **public** LoginPageNew(WebDriver driver)  {  this.driver = driver;  }  @FindBy(id = "value")  WebElement username;  @FindBy(how = How.XPATH, using = "value")  WebElement password;  @FindBy(how = How.XPATH, using = "value")  WebElement clickloginbutton;  **public void** loginWordPress(String uid,String pass){  username.sendKeys(uid);  password.sendKeys(pass);  clickloginbutton.click();  } | **public class** BrowserFactory{  **static** WebDriver driver;  **public static** WebDriver startBrowser(String browser, String url){  if (browser.equalsIgnoreCase("firefox")){  driver = new FirefoxDriver();  }  else if(browser.equalsIgnoreCase("chrome")){  driver = new ChromeDriver();  }  else if(browser.equalsIgnoreCase("ie")){  driver = new InternetExplorerDriver();  }  driver.manage().window().maximize();  driver.get(url);  return driver;  } |
| **public clas**s ExecutionLoginPage{  **@Test**  **public void** testMethod(){  WebDriver driver = BrowserFactory.startbrowser("firefox", "http://www.facebook.com");  LoginpageNew login = PageFactory.initElements(driver, LoginPageNew.class);  login.loginWordPress("admin", "demo1");  }  } | |

|  |  |
| --- | --- |
| **package** test;  **public class** RetryClass **implements** IRetryAnalyzer{  int counter = 1;  int retryLimit = 4;  **public** **boolean** retry(ITestResult result){  if(result.getStatus == ITestResult.FAILURE){  if(counter < tryLimit){  counter ++ ;  **return** **true**;  }} **return** **false**;  }  // it will re-run only annoted method. | **package** test;  **public class** Test001{  @Test(retryAnalyzer = test. RetryClass.class)  **public void** test1(){  String name = "jawad karim";  Assert.assertTrue(name.contains("jawad"));  }  @Test  **public void** test2(){  String name = "jawad karim";  Assert.assertTrue(name.contains("fawad"));  }  } |

|  |
| --- |
| **//** it will re-run for all test method.  **package** test  **public class** RetryListenerClass **implements** IAnnotationTransformer{  //go to source > override/implement Methods import unImplemented transform method.  **public void** transform((ITestAnnotation annotation, Class testClass, Constructor testConstructor, Method testMethod){  annotation.setRetryAnalyzer(retryClass.class);  }  }  -------------------------------------------------------------------------------------------------------------------  **package** test;  **public class** Test001{  @Test  **public void** test1(){  String name = "jawad karim";  Assert.assertTrue(name.contains("jawad");  }  @Test  **public void** test2(){  String name = "jawad karim";  Assert.assertTrue(name.contains("fawad");  }  } |

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE suite SYSTEM "<http://testng.org/testng-1.0.dtd>">

<**suite** name="Suite">

<**listeners**>

<**listener** class-name="test.RetryListenerClass" />

</**listeners**>

<**test** name="Test">

<**classes**>

<**class** name= "test.Test001"/>

</**classes**>

</**test**>

</**suite**> <!-- Suite -->

**Read Data from Excel:**

package utility;

public class ExcelDataProvider {

public File file;

public FileInputStream fis;

public XSSFWorkbook wb;

public ExcelDataProvider() {

try {

file = new File("./utility.ExcelDataProvider");

fis = new FileInputStream(file);

wb = new XSSFWorkbook(fis);

//sheet = wb.getSheetAt(sheetNo);

} catch (Exception e) {

Reporter.log(e.getMessage(), true);

}

}

public String getUserName(int sheetNo, int rownum, int cellnum) {

return wb.getSheetAt(sheetNo).getRow(rownum).getCell(cellnum).getStringCellValue();

}

public String getPassword(int sheetNo, int rownum, int cellnum) {

return wb.getSheetAt(sheetNo).getRow(rownum).getCell(cellnum).getStringCellValue();

}

}

**package** utility;

**public** **class** ReadExcel {

**public** **void** xlData() **throws** Exception {

File file = **new** File("filePath");

FileInputStream fis = **new** FileInputStream(file);

XSSFWorkbook wb = **new** XSSFWorkbook(fis);

XSSFSheet s1 = wb.getSheetAt(0);

**int** rowCount = s1.getLastRowNum();

**int** cellCount = s1.getRow(0).getLastCellNum();

**for**(**int** i=0; i<=rowCount; i++) {

**for**(**int** j=0; j<=cellCount; j++) {

String data1 = s1.getRow(i).getCell(j).getStringCellValue();

//**int** data2 = (**int**) s1.getRow(i).getCell(j).getNumericCellValue();

System.***out***.println("data : "+data1);

}

}

}

}

**package** testCases;

**public class DataDrivenTest** {

WebDriver driver;

@BeforeMethod

**public** **void** launchBrowser() {

WebDriverManager.*chromedriver*().setup();

driver = **new** ChromeDriver();

driver.get("https://demo.guru99.com/V4/index.php");

driver.manage().window().maximize();

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

}

@Test(dataProvider = "login\_data")

**public** **void** loginPage(String userName, String passWord) **throws** Exception {

driver.findElement(By.*xpath*("//input[@name='uid']")).sendKeys(userName);

driver.findElement(By.*xpath*("//input[@name='password']")).sendKeys(passWord);

driver.findElement(By.*xpath*("//input[@name='btnLogin']")).click();

Thread.*sleep*(2000);

**if**(isAlertPresent()==**true**) {

driver.switchTo().alert().accept();

driver.switchTo().defaultContent();

Assert.*assertTrue*(**false**); //login failed

}

**else** {

Assert.*assertTrue*(**true**); //login successful

driver.findElement(By.*xpath*("//a[normalize-space()='Log out']")).click();

Thread.*sleep*(2000);

driver.switchTo().alert().accept();

driver.switchTo().defaultContent();

}

}

@DataProvider(name = "login\_data")

**public** Object[][] testData(){

**return** **new** Object[][] {

{"mngr455792", "UsEzadU"},

{"thjhffjfdj", "hjlillh"},

{"shsdgffflh", "hliltlh"},

{"mngr455792", "UsEzadU"},

{"shgffddslh", "hlillfffh"}

};

}

**public** **boolean** isAlertPresent() {

**try** {

driver.switchTo().alert();

**return** **true**;

}

**catch**(NoAlertPresentException e) {

**return** **false**;

}

}

@AfterMethod

**public** **void** tearDown() {

driver.close();

}

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* testNG data provider with excel data \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**public** **class** ExcelDataProvider {

String filePath = "C:\\Users\\jawad\\test\_data\\input\_data.xlsx";

@DataProvider(name = "exceData");

@Test

**public** Object[][] GetDataFromExcel() **throws** Exception{

File file = **new** File(filePath);

FileInputStream fis = **new** FileInputStream(file);

XSSFWorkbook wb = **new** XSSFWorkbook(fis);

XSSFSheet s1 = wb.getSheetAt(0);

System.***out***.println("data1: "+s1.getRow(0).getCell(0).getStringCellValue());

//getLastRowNum() returns total rows 0 based. add 1 to get actual no of rows.

//getLastCellNum() returns actual total no of columns.

**int** rowNo = s1.getLastRowNum()+1;

**int** cellNo = s1.getRow(0).getLastCellNum();

System.***out***.println("total rows: "+rowNo+" and total columns: "+cellNo);

String[][] data = **new** String[rowNo][cellNo];

**for**(**int** i=0; i<rowNo; i++) {

**for**(**int** j=0; j<cellNo; j++) {

//store datas into 2 dimensional array

data[i][j] = s1.getRow(i).getCell(j).getStringCellValue();

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\* print data \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**for**(**int** i=0; i<rowNo; i++) {

**for**(**int** j=0; j<cellNo; j++) {

System.***out***.println("data: "+data[i][j]);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**return** data;

}

}

@Test(dataProvider = "excelData", dataProviderClass = ExcelDataProvider.**class**)

**public** **void** search(String keyWord1, String keyWord2) {

WebElement txtBox = driver.findElement(By.id("sb\_form\_q"));

txtBox.sendKeys(keyWord1, " ", keyWord2);

System.***out***.println("Keyword entered is : " + keyWord1 + " " + keyWord2);

txtBox.sendKeys(Keys.ENTER);

System.***out***.println("Search results are displayed.");

System.***out***.println("RESULT: "+ driver.getTitle());

Assert.assertTrue(driver.getPageSource().contains(keyWord1));

}

**Read object repository:**

// creat a repository folder under project in eclipse.

//right click on folder and create a file ( name.properties ).

fb.uid.xpath = .//input[@name ='email']

fb.pass.xpath = .//input[@id = 'pass']

@Test

**public** **void** test001() **throws** Exception {

System.*setProperty*("webdriver.chrome.driver", "C:\\Users\\Elias Karim\\Downloads\\Driver's\\chromedriver\_win32.80\\chromedriver.exe");

WebDriver driver = **new** ChromeDriver();

driver.manage().window().maximize();

driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

driver.get("http://www.facebook.com");

File file = **new** File("./Repo/facebookData.properties");

FileInputStream fis = **new** FileInputStream(file);

Properties pro = **new** Properties();

pro.load(fis);

driver.findElement(By.xpath(pro.getProperty("fb.uid.xpath"))).sendKeys("Sabah" driver.findElement(By.xpath(pro.getProperty("fb.pass.xpath"))).sendKeys("karim");

}

------------------------------------------------------------------------------------------------------------------------

**Log4j**

# Log4j is a framework which helps the java user to record the messages (logs) during the execution time. Logs are nothing but the detailed execution steps  
  
# Log4j write all the messages into files, files could be in many formats like a text file, log file, html file. Users have to configure the log4j before using it.

We can configure the log4j in two ways :

* Properties file configuration
* XML file configuration

**Components of log4j :**   
Logger, Appender, Layout.

**Logger:** Logger captures the log messages generated by the framework, and it passes the logs to appends.

Logger Priority Levels :

1. DEBUG - To debug the application, developers give the priority to debug. Level of message logged will be focused on providing support to an application developer.

2. INFO - Provides informational messages that highlight the progress of the application execution.

3. WARN - The messages coming out of this level may not halt the progress of the system, but informs the user something unexpected is happening.

4. ERROR - The ERROR level designates error events that might still allow the application to continue running, means a particular part is failing but not a showstopper.

5. FATAL - The FATAL level designates very severe error events that will presumably lead the application to abort, means it is a showstopper.

**Appender:** Appenders writes the log messages decided by the logger into a file or to a database. Appenders delegates the message to the formatter. There are different kind of appenders.

**. FileAppender :** It appends the message to a file.

* **RollingFileAppender :** It also appends the messages to the file with file size limit, when the file reaches the size limit automatically it creates another file and adds the messages.
* **DailyRollingFileAppender :** It is Similar to Rolling File Appender, but it creates the files based on the given frequency.
* **ConsoleAppender :** This appends to the console of the system.
* **SMTPAppender :** This sends mail to the specific mail when a priority level occurs, example : sends mail when a FATAL error occurs.
* **SysyLogAppender :** Appends to a remote syslog domain.

**Layout** **:** The Layout component defines the format in which the log statements are written to the log destination by appender, like html or .log so on. There are different kind of layout are present

**SimpleLayout :** it presents the log level - log messages format

**PatternLayout :** formats the output based on a conversion pattern specified, or if no conversion specified, it takes the default conversion pattern

**HTMLLayout :** present the log messages as HTML table to the user.

**XMLLayout :** It provides the logs in the format of XML.

Now install Log4J.......

1. Goto apache log4j site :[**https://logging.apache.org/log4j/1.2/**](https://logging.apache.org/log4j/1.2/)  
2. Click download Link.

3. Click log4j.version.zip file from the table for windows.

4. Click on the recommended mirror site link.

5. Extract the downloaded zip file, and you can find the log4j.version.jar file.

6. Open Eclipse and go to build path of the project  
7. Add the log4j jar using **Add External jar** option.

8. Inside java class try to type Logger, you should get a suggestion from the Log4j. If you are seeing below suggestion your installation is successful.

**Properties File :** We can configure the log4j2 using a properties file, the properties file is nothing but hold the key-value pair.

# Create a file named '**log4j2.properties**' in '**source/main/resources**' then copy the below properties and paste to the file.

Example : from- <https://springframework.guru/log4j-2-configuration-using-properties-file/>

name=PropertiesConfig

property.filename = logs

appenders = console, file

appender.console.type = Console

appender.console.name = STDOUT

appender.console.layout.type = PatternLayout

appender.console.layout.pattern = [%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{1} - %msg%n

appender.file.type = File

appender.file.name = LOGFILE

appender.file.fileName=${filename}/propertieslogs.log // file\_name/ report\_name.log

appender.file.layout.type=PatternLayout

appender.file.layout.pattern=[%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{1} - %msg%n

loggers=file

logger.file.name=reoporing // would be package\_name.

logger.file.level = debug

logger.file.appenderRefs = file

logger.file.appenderRef.file.ref = LOGFILE

rootLogger.level = debug

rootLogger.appenderRefs = stdout

rootLogger.appenderRef.stdout.ref = STDOUT

:

Practical Example :

Step 1:Add Log4j libraries in the java project Can add jar files or maven dependency [https://logging.apache.org/log4j/2.x/...](https://www.youtube.com/redirect?q=https%3A%2F%2Flogging.apache.org%2Flog4j%2F2.x%2Fdownload.html&v=rbuR9Q_55h4&event=video_description&redir_token=mLkfGmhR3Yvy2ksHlQznLKyW54d8MTU4NjkzNzQzMkAxNTg2ODUxMDMy)

Step 2 : Create reference for Logger in the class Class where you want to add log statements

Step 3 : Create log4j.xml or log4j.properties file What is the difference ?

Step 4 : Run and Validate

Log4j - log levels

TRACE Most detailed information. Expect these to be written to logs only. Since version 1.2.12

DEBUG Detailed information on the flow through the system. Expect these to be written to logs only. Generally speaking, most lines logged by your application should be written as DEBUG

INFO Interesting runtime events (startup/shutdown). Expect these to be immediately visible on a console, so be conservative and keep to a minimum.

WARN Use of deprecated APIs, poor use of API, 'almost' errors, other runtime situations that are undesirable or unexpected, but not necessarily "wrong". Expect these to be immediately visible on a status console.

ERROR Other runtime errors or unexpected conditions. Expect these to be immediately visible on a status console.

FATAL Severe errors that cause premature termination. Expect these to be immediately visible on a status console.

OFF The highest possible rank and is intended to turn off logging. \*\* built-in log levels with increasing order of severity helpful tips You can configure to append or overwrite logs in file You can set logging OFF when not required in case heavy logging is making your runs slower... Log4j - Configuration File .  
  
**Complete program for Log4j with properties file**

public class FirstLog {

static Logger log = Logger.getLogger(FirstLog.class.getName);

public static void main(String[] args) {

//PropertiesConfigurator is used to configure logger from the properties file

PropertyConfigurator.configure("C:\\~\\log4j.properties");

//log the message to file

log.debug("This is a debug message");

log.info("This is an info message");

}

}

**XML CODE CONFIGURATION :**

We can also configure the log4j logging system with xml configuration file. Please follow below steps to configure the log4j with xml.

<appender name="fileAppender" class="org.apache.log4j.RollingFileAppender">

<param name="File" value="C:\\~\\firstXMLLog.log"/>

<layout class="org.apache.log4j.PatternLayout">

<priority value ="debug">

**Integrate XML with Log4j**

create a java class under project 2.

public class LOgWithXML

{

static Logger log = Logger.getLogger(LOgWithXML.class);

public static void main(String[] args)

{

//DOMConfigurator is used to configure logger from xml configuration file

DOMConfigurator.configure("C:\\~\\log4j.xml");

//Log to file

log.fatal("Log4j XML configuration is successful !!");

}

}

HTML Layout :

this layout provides outputs events in a HTML table.

=> add html properties to the log4j properties.

# Define the root logger with file appender

log4j.rootLogger = DEBUG, HTML

# Define the file appender

log4j.appender.HTML=org.apache.log4j.FileAppender

log4j.appender.HTML.File=C:\\~\\HTMLlog.html

# Define the html layout for file appender

log4j.appender.HTML.layout=org.apache.log4j.HTMLLayout

log4j.appender.HTML.layout.Title=HTML log

log4j.appender.HTML.layout.LocationInfo=true

log4j.appender.HTML.Threshold=DEBUG

in class----

public class HTMLLog {

static Logger log = Logger.getLogger(FirstLog.class);

public static void main(String[] args) {

//PropertiesConfigurator is used to configure logger from properties file

PropertyConfigurator.configure("C:\\~\\log4j-html.properties");

//log the message to file

log.trace("This is a trace message");

log.info("This is an info message");

log.warn("This is an warn message");

log.error("This is an error message");

log.fatal("This is an fatal message");

}

Advantages :

- log4j is optimized for speed, the burden of logging is minimized.

- log4j is thread-safe.

- We can have logs in different format.

- log4j is not restricted to a predefined set of facilities.

- log4j properties can be set at runtime.

- Supports XML and Properties files configuration.

- log4j can direct its output to a file, the console, a java.io.OutputStream, java.io.Writer, a remote server using TCP, a remote Unix Syslog daemon, to a remote listener using JMS, to the NT EventLog or even send e-mail.

- log4j supports multiple output appenders per logger.

**TestNG**

testNG report : TestNG Reports come in to the picture once we execute the test cases using TestNG. Once we execute test cases using TestNG, it will generate a default HTML report.

report will be available in test-output folder under project and name emailable-report.html.

|  |  |
| --- | --- |
| public class DefaultReport {  //To make it pass  @Test  public void passTest(){  Assert.assertTrue(true);  }  //To make it fail  @Test  public void failTest(){  Assert.assertTrue(false);  }  //To make it skip  @Test  public void skipTest(){  throw new SkipException("Skipping ");  }  } | <?xml version="1.0" encoding="UTF-8"?>  <!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">  <suite name="softwaretestingmaterial" parallel="methods">  <test name="testngTest">  <classes>  <class name="softwareTestingMaterial.DefaultReport" />  </classes>  </test>  </suite> |

**testNG Reporter class:** testNG provides a Reporter class for reporting.

report will be added to emailable-report automatically.

Syntax :

Reporter.log(String s); example : **Reporter.log(“browser started”);**

Reporter.log(String s, Boolean logToStandardOut);

Reporter.log(String s, int lebel);

Reporter.log(String s, int lebel, Boolean logToStandardOut);

**testNG Listener interface (ITestListener) :**

Listeners “listen” to the event defined in the selenium script and behave accordingly. The main purpose of using listeners is to create logs. Using TestNG listeners we could generate logs and customize TestNG Reports.

Step 1: Create a Class “**ListenerTestNG**” to implement **ITestListener**

Add unimplimented methods.

**package listeners;**

public class ListenerTestNG implements ITestListener{

@Override

public void onTestStart(ITestResult result) {

// TODO Auto-generated method stub

}

@Override

public void onTestSuccess(ITestResult result) {

// TODO Auto-generated method stub

}

@Override

public void onTestFailure(ITestResult result) {

// TODO Auto-generated method stub

}

@Override

public void onTestSkipped(ITestResult result) {

// TODO Auto-generated method stub

}

@Override

public void onTestFailedButWithinSuccessPercentage(ITestResult result) {

// TODO Auto-generated method stub

}

@Override

public void onStart(ITestContext context) {

// TODO Auto-generated method stub

}

@Override

public void onFinish(ITestContext context) {

// TODO Auto-generated method stub

}

}

**package listeners;**

@Listeners(listeners.ListenerTestNG.class)

public class ListenerTestNGTestCase {

WebDriver driver= new FirefoxDriver();

// Test to pass as to verify listeners .

@Test(priority=1)

public void TestToPass()

{

System.out.println("This method to pass test");

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

driver.getTitle();

driver.quit();

}

//Used skip exception to skip the test

@Test(priority=2)

public void TestToSkip ()

{

System.out.println("This method to skip test");

throw new SkipException("Skipping - This is not ready for testing ");

}

// In the above method, we have already closed the browser. So we couldnot get the title here.

It is to forcefully fail the test

@Test(priority=3)

public void TestToFail()

{

driver.getTitle();

System.out.println("This method to test fail");

}

<xml version="1.0" encoding="UFT-8"?>

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name="Suite">

<listeners>

<listener class-name="listeners.listenerTestNG"/>

</listeners>

<test name="Test">

<classes>

<class name="listeners.ListenerTestNGTestCase">

</classes>

</test>

</suite>

# we also can create testng.xml **package** level. then add the **<listeners>** under the **<suite>**.

# shortcut for importing packages in eclipse- **CTRL + SHIFT + O**

**testNG annotations**

@Test: Marks a class or a method as a part of the test.

@BeforeMethod: A method which is marked with this annotation will be executed before every

**@test** annotated method.

@AfterMethod: A method which is marked with this annotation will be executed after every @testannotated method.

@BeforeClass: A method which is marked with this annotation will be executed before first @Testmethod execution. It runs only once per class.

@AfterClass: A method which is marked with this annotation will be executed after all the test methods in the current class have been run.

@BeforeTest: A method which is marked with this annotation will be executed before first @Test annotated method.

@AfterTest: A method which is marked with this annotation will be executed when all **@Test** annotated methods complete the execution of those classes which are inside <test> tag in testng.xml file.

@BeforeSuite: A method which is marked with this annotation will run only once before all tests in the suite have run.

@AfterSuite: A method which is marked with this annotation will run once after execution of all tests in the suite have run

@BeforeGroups: This annotated method will run before the first test run of that specific group.

@AfterGroups: This annotated method will run after all test methods of that group completes its execution.

Some other TestNG Annotations, we need to discuss, here are mentioned below:

@Parameters: This annotation is used to pass parameters to test methods.

@DataProvider: If we use @DataProvider annotation for any method that means you are using that method as a data supplier. The configuration of @DataProvider annotated method must be like it always return Object[][] which we can use in @Test annotated method. The @Test method that wants to receive data from this DataProvider needs to use a dataProvider name equals to the name of this annotation.

@Factory: Marks a method as a factory that returns objects that will be used by TestNG as Test classes. The method must return Object[ ].

@Listeners: This annotation is used with test class. It helps in writing logs and results.

**testNG annotations hierarchy levels :**

<suit>

<test>

<class>

<method>

<test>

</method>

</class>

</test>

</suit>

**How to hide projects in eclipse**

Eclipse > click down arrow > select working set > new > java > next > name > add project for working set > finish > select project > ok. // now I can see only the selected project.

= to unselect working set : click down arrow > deselect working set.

**How to create a multi Module Maven project: AJ AUTOMATION**

1. first create simple Maven project

> GroupID: com.SmokeTest > artifactID: ParentProject > packaging: pom > name: registerFunction > description: registerFunction > finish.

2. create another maven module. (We can create multiple module.)

> right click on maven project > maven > maven module > module name: childProject1 > next > packaging: jar > name > description > finish.

3. add all common depedency in parent pom.xml.

<dependencyManagement>

<dependencies>

<dependency>selenium</dependency>

<dependency>testng</dependency>

<dependency>webdriverManager</dependency>

</dependencies>

</dependencyManagement>

4. add the dependency reference from parent project to child project. Just artifact id & group id.

<dependencies>

<dependency>

<artifactid> value </artifactid>

<groupid>value</groupid>

</dependency>

</dependencies>

Also the extra depedencies as needed for this child module.

5. run maven projects.

> right click on parent project > run > run as maven test/install/ build

6. how to create a JAR of a maven project and use it in another project.

> add maven-jar-plugin to the project1 pom.xml . …do update>clean the project.

> run as mvn test. …………. to see code runs good or not.

> run as mvn build > Goals: package or install >apply>run. It will create 2 jars for packages/classes

i. project = contains classes from src/main/java

ii. project-tests = contains classes from src/test/java.

>