**Screen shot**

File scrFile = ((TakesScreenshot)driver).getScreenshotAs(OutputType.***FILE***);

FileUtils.*copyFile*(scrFile, **new** File("c:\\Invalid Login.jpg"));

Git commands for local repositiry:

Git init

Git status

Git add src/

Git commit –m “First Commit”

Github is basically a git repository web hosting service

Generating keys: **ssh-keygen -t rsa**

ssh folder will get generated in user folder

open the public file- copy key

Goto setting🡪**SSH and GPG Keys**

To store key permanently : ssh -T [git@github.com](mailto:git@github.com)

Create a Project in Git Hub

Open it—>**Select SSH**->copy the location from new repository box

Then push the code to remote by **$ git push -u origin master**

**Selenium Commands:**

**Thread.sleep(2000)**

selenium.setSpeed(“2000”)-

clickAt

element.submit () ;//submit a form

<http://creyate:tom@www.gmail.com>

Use getCssValue(arg0) function to get the colors by sending ‘color’ string as an argument

* **Syntax- Actions act = new Actions (driver);**
* **act.doubleClick(webelement);**
* driver.switchTo().frame(“frameName”);
* To specify a frame you can use index number
* driver.switchTo().frame(“parentFrame.4.frameName”);
* This would bring control on frame named- “frameName” of the 4th sub frame names “parentFrame”

driver.switchTo().defaultContent(); Switch back from a frame

* By.id()
* By.name()
* By.tagName()
* By.className()
* By.linkText()
* By.partialLinkText()
* By.xpath
* By.cssSelector()

**Config File**

**public** **static** **final** String ***IhaveGotIt*** = "//\*[@id=\"go-to-help-center\"]";

**public** **static** **final** String ***browser***="Chrome";

Page

**public** Serviceability chkServiceablity(String StAddress, String Zip) {

strAddress.sendKeys(StAddress);

zipCode.sendKeys(Zip);

lookUpAccount.click();

**return** PageFactory.*initElements*(driver, Serviceability.**class**);

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

**@Before test**

**@test(**dataProvider = "getData")

**public** **static** **void** main( String StAddress, String Zip, String Expected) **throws** InterruptedException, IOException {

HomePage searchAddress = PageFactory.*initElements*(*driver*, HomePage.**class**);

Serviceability chkServc = searchAddress.chkServiceablity(StAddress, Zip);

@DataProvider

Xpath

//input[@id=’name’]

//a[text()='Allow']

//div[text()='Later']

In Chrome

$x(“//input[id=’Dil’]) Type the same in console and enter

x("//input[@type='search'][@id='searchInput']")- Real Xpath is **//input[@type='search'][@id='searchInput']**

$x(“//input[stars-with(@id,’Seacrh’)]”)

//div[contains(@class,'chf\_left\_portion\_hdr chf\_flL')]

//img[contains(@class,'for\_large')]

//img[contains(@class, "for\_large" src="v2/images/responsive/error/error\_mascot.png" alt="mascot"/)]

//article[contains(@class,'my-res new-theme my-result-list animation-on item-0')]//label[contains(@id,'fare')]

Starts-with

substring

Switching between Tabs/Windows/PopUps

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

Iterator<String> iterate = WindowsId.iterator();

System.***out***.println(iterate.next());//Will print 1st window id home page

WindowsId = *driver*.getWindowHandles();

iterate = WindowsId.iterator();

String MainWind =iterate.next();

String TabWind =iterate.next();

*driver*.switchTo().window(TabWind);//Switch to 2nd window

*driver*.findElement(By.*linkText*("dhdhhd")).click();

WindowsId = *driver*.getWindowHandles();

iterate = WindowsId.iterator();

System.***out***.println(iterate.next());//Will print 1st window id home page

System.***out***.println(iterate.next());//Will print 2nd window id home page

String PopWind =iterate.next();

*driver*.switchTo().window(PopWind);//Switch to 2nd window

*driver*.close();

*driver*.switchTo().window(TabWind);

*driver*.close();

*driver*.switchTo().window(MainWind);

*driver*.close();

Static/NonStatic object Main Rules

With In Same Class🡪

If we want to access variable (glob)/method that is **Non static** and if want to call it in **Static class** we have to create an object the same class

If we want to access variable (glob)/method that is **Non static** and if want to call it in **Non** **Static method** we can direct call it

If we want to access variable (glob)/method that is S**tatic**, and if we want to call it in **Non** **Static method** we can direct call it

If we want to access variable (glob)/method that is S**tatic**, and if we want to call it in **Static class/method** we can direct call it

Outside a Class

If we want to access variable (glob)/method that is **Non static**, in a different class, we have to create an object of that class first

If we want to access variable (glob)/method that is **Static**, and if want to call it in **Static class** we can direct call it(Classname.methodname)

Instance variable- variable declared inside the main class body- Throuh out class(should be non static0

Local variable- variable declared inside the method- Onlt with in method

Class variable- variable declared inside the main class body- values gets changes(Shotld be static)

Int x = (int)(Math.random()\*5)🡪Toget any random numbet beteen 1-5

**Loops:**

While: Number of iterations are unknown

For: Number of iterations are Know

DoWhile: Atleast one

**While Loop**

While repeat he block until the conditions met

**public** **class** While {

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

**int** i = 1;

**while**(i<10){

System.***out***.println(i);

i++;

}

}

}

**DoWhile Loop**

**public** **class** DoWhile {

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

**int** i= 6;

**do**{

System.***out***.println (i);//This will always execute. Write all code here

i++;

}

**while**(i<10);{//Condition to exit

System.***out***.println (i);//will come to this on the condition is met

}

}

}

**ForLoop**

**public** **class** ForLoop {

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

**for**(**int** i=1;i<10;i++){

System.***out***.println(i);

}

}

}

**Break And Continue**

**public** **class** BreakAndContinue {

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

**for**(**int** i=1; i<10;i++){

**if** (i==5)

**break**;//It will break the condition if any of the condition met

System.***out***.println(i);

}

}

}

**Array**

**public** **class** Array {

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

String[]Name = {"Dil", "Raj"};

**for**(**int** i =0;i<Name.length;i++){

System.***out***.println(Name[i]);

}

}

}

//int [] salary;

//salary = new int[10];

**Array 2d**

**public** **class** Array2 {

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

**int** row= 2;

**int** col= 3;

**int** table[][] = **new** **int**[row][col];

table[0][0] = 10;

table[0][1] = 20;

table[0][2] = 30;

table[1][0] = 40;

table[1][1] = 50;

table[1][2] = 60;

System.***out***.println(table.length);

System.***out***.println(table[0].length);

**for**(**int** x=0;x<table.length;x++){

**for**(**int** y=0;y<table[0].length;y++){

System.***out***.print(" "+table[x][y]);

}

System.***out***.println(" ");

}

}

}

**Print Table**

**public** **class** PrintTable {

**public** **static** **void** table(**int** t){

**for**(**int** i=1;i<=10;i++){

System.***out***.println(i\*t);

}

}

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

*table*(10);

}

}

**Constructor**

**public** **class** Const {

String Name;

**int** age;

**int** RollNo;

**public** Const(String Name, **int** age, **int** RollNo)

{

**this**.Name = Name;

**this**.age = age;

**this**.RollNo =RollNo;

}

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

Const c = **new** Const("Dil", 12, 101);

Const d = **new** Const("Dilee", 13, 103);

System.***out***.println(c.Name);

System.***out***.println(c.age);

System.***out***.println(c.RollNo);

System.***out***.println(d.Name);

System.***out***.println(d.age);

System.***out***.println(d.RollNo);

}

}

**Inheritance**

Class A

Class B (public class b extends a)- B inherited to Class A.

Class C- Another class

If we create an object of class b in Class C, all the properties of Class A is accessible in class c.

If we create a same method in child and parent class, and we create object of both class in class C, then if we access the method by calling object b, then it will work as per the logic written in child class

**Interface**

**public** **interface** Employee {

**public** **void** salary();

**public** **void** policy();

}

**public** **class** department **implements** Employee{

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

Employee e = **new** department();

e.policy();

e.salary();

}

@Override

**public** **void** salary() {

System.***out***.println("Salary credited");

}

@Override

**public** **void** policy() {

System.***out***.println("Policy Chnged");

}

}

**public** **class** TestBile {

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

Bike b = **new** Bike();

b.startBike();

b.stopBike();

MakeHonda m = **new** MakeHonda();//Here method overrirden

m.startBike();

m.tubeType();

Bike b1=**new** MakeHonda();

b1.startBike();

b1.stopBike();

//Note: now if we try to access method from child class, it wond work

}

}

**Try catch**

**public** **class** TestException {

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

System.***out***.println("Start");

**try** {

**int** c = 5/0;

System.***out***.println(c);

} **catch** (Exception e) {

e.printStackTrace();

}

System.***out***.println("Start Exe");

System.***out***.println("Start Exe");

}

}

Default Modifile🡪Can access it in samepackage if we create an object(without public)

Public🡪Anywhere in proj by creating an object

**Reading a File**

**package** Properties;

**public** **class** Test {

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

Properties prop = **new** Properties();

//FileInputStream fis = new FileInputStream("C:\\Javabasics\\src\\Properties\\Properties");

FileInputStream fis = **new** FileInputStream(System.*getProperty*("user.dir")+"\\src\\Properties\\Properties");

prop.load(fis);

System.***out***.println(prop.getProperty("Name"));

System.***out***.println(prop.getProperty("Age"));

System.***out***.println(prop.getProperty("Course"));

System.***out***.println(System.*getProperty*("user.dir"));

}

}

**Creating and writing inside a file**

**public** **class** WriteNotePad {

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

//Create a file

/\* File file = new File(System.getProperty("user.dir")+"\\Note2.txt");

// File file = new File("C:\\Javabasics\\Note.txt");

file.createNewFile();

//Write a file

FileWriter fwriter = new FileWriter(file);

BufferedWriter bfwriter = new BufferedWriter(fwriter);

bfwriter.write("Dileep first Line11133");

bfwriter.newLine();

bfwriter.write("Dileep second Line11133");

//bfwriter.flush();

bfwriter.close();

\*/

//Read a File

FileReader fReader = **new** FileReader(System.*getProperty*("user.dir")+"\\Note2.txt");

BufferedReader bReader = **new** BufferedReader(fReader);

String x= "";

**while**((x=bReader.readLine())!= **null**){

System.***out***.println(x);

}

}

}

**ArrayList**

**import** java.util.ArrayList;

**public** **class** TestArrayList {

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

ArrayList<String> myList = **new** ArrayList<String>();

myList.add("Dileep");

myList.add("Ram");

myList.add("Ramu");

System.***out***.println(myList.get(1));

System.***out***.println(myList.size());

**for**(**int** i=1; i<(myList.size());i++)

{

System.***out***.println(myList.get(i));

}

}

}

**Set(iterate)**

**import** java.util.HashSet;

**import** java.util.Iterator;

**import** java.util.Set;

**public** **class** TestSet {

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

Set <Integer> set = **new** HashSet <Integer>();

set.add(1);

set.add(2);

set.add(3);

Iterator<Integer>iterate = set.iterator();

/\* System.out.println(iterate.next());

System.out.println(iterate.next()); \*/

**while**(iterate.hasNext()){

System.***out***.println(iterate.next());

}

}

}

**HashTable**

**import** java.util.Hashtable;

**public** **class** TestHashTable {

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

Hashtable<String, String> ht = **new** Hashtable<String, String>();

ht.put("name", "Dileep");

ht.put("place", "Delhi");

ht.put("Corse", "Mech");

System.***out***.println(ht.get("place"));

}

}

**CSS**

For Id- **#searchInput or input#searchInput** In Chrome it is **$$('#searchInput')**

For class- **.searchButton or .searchButton[name=go] or input.searchButton[name=go]**

**DropDown/Links/WebElements**

**public** **class** DropDown {

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

System.*setProperty*("webdriver.chrome.driver", "chromedriver.exe");

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

driver.get("https://www.wikipedia.org/");

WebElement Drop =driver.findElement(By.*xpath*("//\*[@id=\"searchLanguage\"]"));

Drop.click();

Select select = **new** Select(Drop);

select.selectByValue("az");

List <WebElement> option =select.getOptions();

option.size();

System.***out***.println(option.size());

**for**(**int** i= 0; i<option.size();i++){

System.***out***.println(option.get(i).getText());

}

System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

List <WebElement> option1= driver.findElements(By.*tagName*("option"));

option1.size();

System.***out***.println(option1.size());

**for**(**int** i= 0; i<option1.size();i++){

System.***out***.println(option1.get(i).getText());

System.***out***.println(option1.get(i).getAttribute("lang"));

}

System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

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

links.size();

System.***out***.println(links.size());

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

System.***out***.println(links.get(i).getText());

System.***out***.println(links.get(i).getAttribute("href"));

}

System.***out***.println("\*\*\*\*\*\*Printing Links from a particular Block\*\*\*\*\*\*\*\*\*\*\*");

WebElement Block = driver.findElement(By.*xpath*("//\*[@id=\"www-wikipedia-org\"]/div[6]"));

List <WebElement> links2= Block.findElements(By.*tagName*("a"));

links2.size();

System.***out***.println(links2.size());

**for**(**int** i= 0; i<links2.size();i++){

System.***out***.println(links2.get(i).getText());

System.***out***.println(links2.get(i).getAttribute("href"));

}

driver.close();

}

}

**KeyBoard events**

driver.findElement(By.*xpath*(Pwd)).sendKeys(Keys.***PAGE\_UP***);

Alert alert = driver.switchTo().alert();

**String/Int Conversion**

String a = "12 + 2";

String b = a.substring(0,2);

String c = a.substring(5,6);

**int** d = Integer.*parseInt*(b);//convert string into int

**int** e = Integer.*parseInt*(c);

**int** sum = d+ e ;

String f = String.*valueOf*(sum);

System.***out***.println(f);

**Check Box/Radio Button Selection**

WebElement dd = driver.findElement(By.xpath("//\*[@id=\"PersistentCookie\"]"));

List<WebElement>Chk = driver.findElements(By.xpath("//\*[@id=\"PersistentCookie\"]"));

System.out.println(Chk.get(0).getAttribute("checked"));

Thread.sleep(2000);

Chk.get(0).click();