**Booking a vehicle:**

Now let’s see the special function userclickvehicledetails. We will pass the vehicle number in the car listings page to this function so that the vehicle can be booked.

First, we add this as a keyword in properties file.

config.properties file:

WB\_PATH\_TESTS = C:\\Users\\padal\\eclipse-workspacenew\\HybridKeywordDriven\\tests\\tests\_main.xlsx

WB\_PATH\_TESTS\_SHEET = TestCases

WB\_PATH\_TEST\_CASES = C:\\Users\\padal\\eclipse-workspacenew\\HybridKeywordDriven\\tests\\

MASTERKW\_PATH = C:\\Users\\padal\\eclipse-workspacenew\\HybridKeywordDriven\\data\\masterkeywords.xlsx

MASTERKW\_SHEET = globalkeywords

DATA\_PATH = C:\\Users\\padal\\eclipse-workspacenew\\HybridKeywordDriven\\data\\data.xlsx

SPECIAL\_KEYWORDS = adminpostvehicle,adminlogin,adminpostbrand,registeruser,userlogin

CHECK\_KEYWORDS = checkusersdifference,checkvehiclesdifference,checkbookingsdifference,checkbrandsdifference,checksubscribersdifference,checkqueriesdifference,checktestimonialsdifference

**BOOKING\_VEHICLES = userclickvehicledetails**

ADMIN\_UNAME = admin

ADMIN\_PWD = Test@12345

USER\_UNAME = subbu123

USER\_PWD = subbu123

We added a new variable here BOOKING\_VEHICLES and stored the new function here.

Now let’s modify the executeTC() method in DriverScript to check this keywod. The code is highlighted in red color.

**executeTC() method in DriverScript.java:**

**public** **static** **void** executeTC(List<String> originalkeywords, Set<String> actionclass, Map<String, String> keywordvsac, Set<String> objectrepository, Map<String, String> keywordvsor) **throws** ClassNotFoundException, NoSuchMethodException, SecurityException, IllegalAccessException, IllegalArgumentException, InvocationTargetException, InstantiationException, IOException {

System.*setProperty*("webdriver.gecko.driver", "C:\\BrowserDrivers\\geckodriver.exe");

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

Properties gldata = **new** Properties();

InputStream input = **new** FileInputStream("src/executionEngine/config.properties");

gldata.load(input);

**for**(String str : originalkeywords) {

String keyword = str;

String[] specialkeywords = gldata.getProperty("SPECIAL\_KEYWORDS").split("\\,");

**boolean** skstatus = **false**;

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

**if**(keyword.indexOf(specialkeywords[i]) != -1) {

skstatus = **true**;

**break**;

}

}

String[] checkkeywords = gldata.getProperty("CHECK\_KEYWORDS").split("\\,");

**boolean** ckstatus = **false**;

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

**if**(keyword.indexOf(checkkeywords[i]) != -1) {

ckstatus = **true**;

**break**;

}

}

**String[] bookingvehicles = gldata.getProperty("BOOKING\_VEHICLES").split("\\,");**

**boolean bkstatus = false;**

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

**if(keyword.indexOf(bookingvehicles[i]) != -1) {**

**bkstatus = true;**

**break;**

**}**

**}**

**if**(skstatus) {

**if**(keyword.indexOf("(") != -1){

String[] parts = keyword.split("\\(");

String[] dataelements = (parts[1].split("\\)"))[0].split("\\,");

*specialfunction*(parts[0],dataelements,driver);

}

**else** {

String parts = keyword;

String[] dataelements = **null**;

*specialfunction*(parts,dataelements,driver);

}

}

**else** **if**(ckstatus) {

**if**(keyword.indexOf("(") != -1){

String[] parts = keyword.split("\\(");

String checkkeyword = parts[0];

**int** expectedvalue = Integer.*parseInt*((parts[1].split("\\)")[0]));

//System.out.println("check keyword: "+checkkeyword);

//System.out.println("expected value: "+expectedvalue);

*specialcheckfunction*(checkkeyword,expectedvalue,driver);

}

}

**else if(bkstatus) {**

**if(keyword.indexOf("(") != -1){**

**String[] parts = keyword.split("\\(");**

**String bookingkeyword = parts[0];**

**int vehno = Integer.*parseInt*((parts[1].split("\\)")[0]));**

**//System.out.println("check keyword: "+checkkeyword);**

**//System.out.println("expected value: "+expectedvalue);**

***bookfunction*(bookingkeyword,vehno,driver);**

**}**

**}**

**else** {

//System.out.println("going to normal");

String actioncl = keywordvsac.get(keyword);

String objectcl = keywordvsor.get(keyword);

Class<?> cls = Class.*forName*("actions."+actioncl);

Class<?> orc = Class.*forName*("objectrepository."+objectcl);

Method[] methodcall = cls.getDeclaredMethods();

**for**(Method m : methodcall) {

**if**(keyword.equalsIgnoreCase(m.getName()) && m.getParameterCount() == 0)

{

Method mc = cls.getDeclaredMethod(keyword);

Constructor<?> constructor = cls.getConstructor(WebDriver.**class**);

mc.invoke(constructor.newInstance(driver));

}

**else** **if**(keyword.equalsIgnoreCase(m.getName()) && m.getParameterCount() == 1)

{

Method morc = orc.getDeclaredMethod(keyword);

Constructor<?> orconstructor = orc.getConstructor(WebDriver.**class**);

WebElement we = (WebElement) morc.invoke(orconstructor.newInstance(driver));

Method mc = cls.getDeclaredMethod(keyword,WebElement.**class**);

Constructor<?> constructor = cls.getConstructor(WebDriver.**class**);

mc.invoke(constructor.newInstance(driver),we);

}

}

}

}

}

We have added the functionality to check if the keyword is a part of BOOKING\_VEHICLES. If this is a part, we will call the function bookfunction by passing the keyword, the vehicle number we would like to book and the driver.

**bookfunction() of DriverScript.java:**

**public** **static** **void** bookfunction(String bookingkeyword, **int** vehno, WebDriver driver) **throws** IOException, ClassNotFoundException, NoSuchMethodException, SecurityException, IllegalAccessException, IllegalArgumentException, InvocationTargetException, InstantiationException{

Properties gldata = **new** Properties();

InputStream input = **new** FileInputStream("src/executionEngine/config.properties");

gldata.load(input);

excelUtilities ecu = **new** excelUtilities();

List<String> actionorcl = ecu.getBookActionORClass(bookingkeyword, gldata.getProperty("MASTERKW\_PATH"), gldata.getProperty("MASTERKW\_SHEET"));

String actioncl = actionorcl.get(0);

String objectcl = actionorcl.get(1);

Class<?> cls = Class.*forName*("actions."+actioncl);

Class<?> orc = Class.*forName*("objectrepository."+objectcl);

Method morc = orc.getDeclaredMethod(bookingkeyword, **int**.**class**);

Constructor orconstructor = orc.getConstructor(WebDriver.**class**);

WebElement we = (WebElement) morc.invoke(orconstructor.newInstance(driver), vehno);

Method mc = cls.getDeclaredMethod(bookingkeyword,WebElement.**class**);

Constructor constructor = cls.getConstructor(WebDriver.**class**);

mc.invoke(constructor.newInstance(driver),we);

}

In the book function we call getBookActionORClass to get the action class and object repository class for the keyword.

**getBookActionORClass() of excelUtilities.java:**

//This function will return the action class and object repository class for booking keyword

**public** List<String> getBookActionORClass(String keyword, String wbpath, String sheetname) **throws** IOException{

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

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

XSSFSheet sheet = workbook.getSheet(sheetname);

//Get all the rows

Iterator<Row> rows = sheet.iterator();

String actionclassvalue = "";

String orclassvalue = "";

List<String> bookingclasses = **new** ArrayList<String>();

**while**(rows.hasNext()) {

**try** {

Row rowno = rows.next();

Cell kwvalue = rowno.getCell(0);

Cell acvalue = rowno.getCell(2);

Cell orvalue = rowno.getCell(3);

String keywordvalue;

DataFormatter df = **new** DataFormatter();

keywordvalue = df.formatCellValue(kwvalue);

**if**(keyword.equalsIgnoreCase(keywordvalue)) {

actionclassvalue = df.formatCellValue(acvalue);

orclassvalue = df.formatCellValue(orvalue);

bookingclasses.add(actionclassvalue);

bookingclasses.add(orclassvalue);

**break**;

}

}

**catch**(Exception e) {

}

}

workbook.close();

**return** bookingclasses;

}

Once we get the action class and object repository information we use java reflections in bookfunctions() method to call the respective methods by passing the vehicle number.

**vehicledetailsOR.java:**

**public** WebElement userclickvehicledetails(**int** vehno) {

WebElement vehdetails = driver.findElements(By.*xpath*("//a[contains(text(),'View Details')]")).get(vehno-1);

**return** vehdetails;

}

**vehicledetails.java:**

**public** **void** userclickvehicledetails(WebElement vehdetails) {

vehdetails.click();

}