**Grouping test cases like smoke tests, regression tests etc:**

If you use individual methods as tests, then you can use grouping like @Test(groups={“smoke”}) for the methods you want to run as part of smoke testing. Similarly you can do for regression as well. But since in keyword driven framework, it is just one method, we can’t do it. We have to drive it from main test cases excel sheet.

In main test cases excel sheet, create a column naming smoke. For some of the test cases, that we think part of smoke test add “Yes” and for others add “No”. Similarly add another column “Regression” and select some cases for regression.

In the properties file add two parameters.

SMOKE\_TEST = YES

REGRESSION\_TEST = NO

Here the idea is, if both the above are No, then all the test cases are executed. If SMOKE\_TEST is Yes then only smoke test cases will be run. If REGRESSION\_TEST is Yes then only regression test cases are run.

For now we have selected SMOKE\_TEST as YES.

Now we have to change the method getTestCases() in ExcelUtilities.java program a little to read the column values accordingly based on the property set in properties file.

The following is the piece of code we made changes to.

Cell rvalue;

**if**(gldata.getProperty("SMOKE\_TEST").equalsIgnoreCase("yes")) {

rvalue = rowno.getCell(6);

}

**else** **if**(gldata.getProperty("REGRESSION\_TEST").equalsIgnoreCase("yes")) {

rvalue = rowno.getCell(7);

}

**else** {

rvalue = rowno.getCell(4);

}

Earlier we are reading the cell 4 value into rvalue. Now based on the condition we are reading cell 6 or cell 7 or cell 4 value.

**getTestCases() method of ExcelUtilities class:**

//This function will return all the main test cases like TC01, TC02 etc where run = "yes"

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

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

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

gldata.load(input);

//This list holds all the test cases

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

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

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

XSSFSheet sheet = workbook.getSheet(sheetname);

//Get all the rows

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

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

//This list holds each test case info test case no, workbook, sheet

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

Row rowno = rows.next();

Cell tcvalue = rowno.getCell(0);

Cell wbvalue = rowno.getCell(1);

Cell shvalue = rowno.getCell(2);

**Cell rvalue;**

**if(gldata.getProperty("SMOKE\_TEST").equalsIgnoreCase("yes")) {**

**rvalue = rowno.getCell(6);**

**}**

**else if(gldata.getProperty("REGRESSION\_TEST").equalsIgnoreCase("yes")) {**

**rvalue = rowno.getCell(7);**

**}**

**else {**

**rvalue = rowno.getCell(4);**

**}**

Cell prvalue = rowno.getCell(5);

String testcasevalue;

String workbookvalue;

String sheetvalue;

String runvalue;

String priorityvalue;

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

runvalue = df.formatCellValue(rvalue);

**if**(runvalue.equalsIgnoreCase("yes")) {

testcasevalue = df.formatCellValue(tcvalue);

testcaseinfo.add(testcasevalue);

workbookvalue = df.formatCellValue(wbvalue);

testcaseinfo.add(workbookvalue);

sheetvalue = df.formatCellValue(shvalue);

priorityvalue = df.formatCellValue(prvalue);

testcaseinfo.add(sheetvalue);

testcaseinfo.add(priorityvalue);

testcases.add(testcaseinfo);

}

}

workbook.close();

testcases.sort((l1, l2) -> l1.get(3).compareTo(l2.get(3)));

**return** testcases;

}