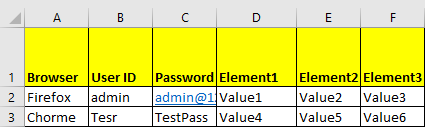
### **Test Data Creation**

**Step 1:** The first and the foremost step is to create the test data with which we would be executing the test scripts. Considering the aforementioned test data format, let us create an excel file named as “TestScript1”. Furnish the values in the elements.



**Step 2:** The next step is to download a standard java based API/Library named as “Java excel Library” (jxl) to be able to access the already created generic methods for Excel Manipulation.

**Step 3:** Create a generic excel reader class named as “ExcelReader.java”. Copy the below code in the ExcelReader.java.

|  |  |
| --- | --- |
| 1 | package utilities; |
| 2 | import java.io.File; | |

|  |  |
| --- | --- |
| 3 | import java.io.IOException; |
| 4 | import java.util.Hashtable; |

|  |  |
| --- | --- |
| 5 | import jxl.Sheet; |
| 6 | import jxl.Workbook; | |

|  |  |  |
| --- | --- | --- |
| 7 | import jxl.read.biff.BiffException; | |
| 8 |  |

|  |  |  |
| --- | --- | --- |
| 9 | /\*\* | |
| 10 | | \* This is a utility class created to read the excel test data file before performing the test steps. | |

|  |  |  |
| --- | --- | --- |
| 11 | \* This class loads the excel file and | |
| 12 | \* reads its column entries. |

|  |  |
| --- | --- |
| 13 | \* |
| 14 | \*/ | |

|  |  |
| --- | --- |
| 15 |  |
| 16 | public class ExcelReader { | |

|  |  |
| --- | --- |
| 17 | /\*\* |
| 18 | \* The worksheet to read in Excel file | |

|  |  |  |
| --- | --- | --- |
| 19 | \*/ | |
| 20 |  |

|  |  |  |
| --- | --- | --- |
| 21 | public static Sheet wrksheet; | |
| 22 | /\*\* |

|  |  |  |
| --- | --- | --- |
| 23 | \* The Excel file to read | |
| 24 | \*/ |

|  |  |
| --- | --- |
| 25 |  |
| 26 | public static Workbook wrkbook = null; | |

|  |  |
| --- | --- |
| 27 | /\*\* |
| 28 | \* Store the column data | |

|  |  |  |
| --- | --- | --- |
| 29 | \*/ | |
| 30 |  |

|  |  |  |
| --- | --- | --- |
| 31 | public static Hashtable<String, Integer> dict = new Hashtable<String, Integer>(); | |
| 32 | /\*\* |

|  |  |  |
| --- | --- | --- |
| 33 | \* Create a Constructor | |
| 34 | \* |

|  |  |
| --- | --- |
| 35 | \* @param ExcelSheetPath |
| 36 | \* @throws BiffException |

|  |  |  |
| --- | --- | --- |
| 37 | \* @throws WeblivException | |
| 38 | \*/ |

|  |  |
| --- | --- |
| 39 |  |
| 40 | public ExcelReader(String ExcelSheetPath)throws IOException, BiffException { | |

|  |  |
| --- | --- |
| 41 |  |
| 42 | // Initialize | |

|  |  |
| --- | --- |
| 43 | try { |
| 44 | wrkbook = Workbook.getWorkbook(new File(ExcelSheetPath)); | |

|  |  |  |
| --- | --- | --- |
| 45 | wrksheet = wrkbook.getSheet("Sheet1"); | |
| 46 | } catch (IOException e) { |

|  |  |  |
| --- | --- | --- |
| 47 | throw newIOException(); | |
| 48 | } |

|  |  |
| --- | --- |
| 49 | } |
| 50 | /\*\* | |

|  |  |  |
| --- | --- | --- |
| 51 | \* Returns the Number of Rows | |
| 52 | \* |

|  |  |  |
| --- | --- | --- |
| 53 | \* @return Rows | |
| 54 | \*/ |

|  |  |
| --- | --- |
| 55 |  |
| 56 | public static int RowCount() { | |

|  |  |  |
| --- | --- | --- |
| 57 | return wrksheet.getRows(); | |
| 58 | } |

|  |  |
| --- | --- |
| 59 | /\*\* |
| 60 | \* Returns the Cell value by taking row and Column values as argument | |

|  |  |
| --- | --- |
| 61 | \* |
| 62 | \* @param column | |

|  |  |
| --- | --- |
| 63 | \* @param row |
| 64 | \* @return Cell contents | |

|  |  |  |
| --- | --- | --- |
| 65 | \*/ | |
| 66 |  |

|  |  |
| --- | --- |
| 67 | public static String ReadCell(int column,int row) { |
| 68 | returnwrksheet.getCell(column, row).getContents(); | |

|  |  |
| --- | --- |
| 69 | } |
| 70 | /\*\* | |

|  |  |  |
| --- | --- | --- |
| 71 | \* Create Column Dictionary to hold all the Column Names | |
| 72 | \*/ |

|  |  |
| --- | --- |
| 73 | public static void ColumnDictionary() { |
| 74 | // Iterate through all the columns in the Excel sheet and store the | |

|  |  |
| --- | --- |
| 75 | // value in Hashtable |
| 76 | for (int col = 0; col < wrksheet.getColumns(); col++) { | |

|  |  |  |
| --- | --- | --- |
| 77 | dict.put(ReadCell(col,0), col); | |
| 78 | } |

|  |  |
| --- | --- |
| 79 | } |
| 80 | /\*\* | |

|  |  |  |
| --- | --- | --- |
| 81 | \* Read Column Names | |
| 82 | \* |

|  |  |  |
| --- | --- | --- |
| 83 | \* @param colName | |
| 84 | \* @return value |

|  |  |  |
| --- | --- | --- |
| 85 | \*/ | |
| 86 |  |

|  |  |  |
| --- | --- | --- |
| 87 | public static int GetCell(String colName) { | |
| 88 | try { |

|  |  |
| --- | --- |
| 89 | int value; |
| 90 | value = ((Integer) dict.get(colName)).intValue(); | |

|  |  |
| --- | --- |
| 91 | returnvalue; |
| 92 | } catch(NullPointerException e) { | |

|  |  |  |
| --- | --- | --- |
| 93 | return(0); | |
| 94 | } |

|  |  |  |
| --- | --- | --- |
| 95 | } | |
| 96 | } |

**Step 4:** Create a generic class –“CommonMethods.java”. Create a common method within the class that would read the cells from the excel sheet using the methods implemented in ExcelReader.java.

|  |  |
| --- | --- |
| 1 | /\*\* |
| 2 | \* Read the test data from excel file | |

|  |  |
| --- | --- |
| 3 | \* |
| 4 | \* @param data The TestData data object | |

|  |  |  |
| --- | --- | --- |
| 5 | \*/ | |
| 6 |  |

|  |  |
| --- | --- |
| 7 | public void readExcelData (TestData data) { |
| 8 | ArrayList<String> browser = new ArrayList<String>(); | |

|  |  |  |
| --- | --- | --- |
| 9 | ArrayList<String> username = new ArrayList<String>(); | |
| 10 | | ArrayList<String> password = new ArrayList<String>(); | |

|  |  |
| --- | --- |
| 11 | ArrayList<String> element1 = new ArrayList<String>(); |
| 12 | ArrayList<String> element2 = new ArrayList<String>(); |

|  |  |  |
| --- | --- | --- |
| 13 | ArrayList<String> element3 = new ArrayList<String>(); | |
| 14 |  |

|  |  |
| --- | --- |
| 15 | // Get the data from excel file |
| 16 | for (int rowCnt = 1; rowCnt < ExcelReader.RowCount(); rowCnt++) { | |

|  |  |
| --- | --- |
| 17 | browser.add(ExcelReader.ReadCell(ExcelReader.GetCell("Browser"), rowCnt)); |
| 18 | username.add(ExcelReader.ReadCell(ExcelReader.GetCell("User ID"), rowCnt)); |

|  |  |
| --- | --- |
| 19 | password.add(ExcelReader.ReadCell(ExcelReader.GetCell("Password"), rowCnt)); |
| 20 | element1.add(ExcelReader.ReadCell(ExcelReader.GetCell("Element1"), rowCnt)); |

|  |  |  |
| --- | --- | --- |
| 21 | element2.add(ExcelReader.ReadCell(ExcelReader.GetCell("Element2"), rowCnt)); | |
| 22 | | element3.add(ExcelReader.ReadCell(ExcelReader.GetCell("Element3"), rowCnt)); |

|  |  |
| --- | --- |
| 23 | } |
| 24 | data.setBrowser(browser); | |

|  |  |  |
| --- | --- | --- |
| 25 | data.setLoginUser(username); | |
| 26 | data.setPassword(password); |

|  |  |
| --- | --- |
| 27 | data.setElement1(element1); |
| 28 | data.setElement2(element2); |

|  |  |  |
| --- | --- | --- |
| 29 | data.setElement3(element3); | |
| 30 | } |

**Step 5:** Create a new java class named as “TestData.java”. This class would act as a getter and setter for excel data. Copy and paste the following code in the TestData.java class.

|  |  |  |
| --- | --- | --- |
| 1 | package Utilities.dataSetters; | |
| 2 | import java.util.ArrayList; |

|  |  |
| --- | --- |
| 3 | public class TestData { |
| 4 | private ArrayList<String> loginUser = null; | |

|  |  |  |
| --- | --- | --- |
| 5 | private ArrayList<String> password = null; | |
| 6 | private ArrayList<String> browser = null; |

|  |  |
| --- | --- |
| 7 | private ArrayList<String> element1 = null; |
| 8 | private ArrayList<String> element2 = null; |

|  |  |  |  |
| --- | --- | --- | --- |
| 9 | private ArrayList<String> element3 = null; | | |
| 10 | | /\*\* |

|  |  |  |
| --- | --- | --- |
| 11 | \* @return loginUser | |
| 12 | \*/ |

|  |  |  |
| --- | --- | --- |
| 13 | public ArrayList<String> getLoginUser() { | |
| 14 | return loginUser; |

|  |  |
| --- | --- |
| 15 | } |
| 16 | /\*\* | |

|  |  |  |
| --- | --- | --- |
| 17 | \* @param loginUser | |
| 18 | \*/ |

|  |  |  |
| --- | --- | --- |
| 19 | public void setLoginUser(ArrayList<String> loginUser) { | |
| 20 | this.loginUser = loginUser; |

|  |  |
| --- | --- |
| 21 | } |
| 22 | /\*\* | |

|  |  |  |
| --- | --- | --- |
| 23 | \* @return password | |
| 24 | \*/ |

|  |  |  |
| --- | --- | --- |
| 25 | public ArrayList<String> getPassword() { | |
| 26 | return password; |

|  |  |
| --- | --- |
| 27 | } |
| 28 | /\*\* | |

|  |  |  |
| --- | --- | --- |
| 29 | \* @param password | |
| 30 | \*/ |

|  |  |  |
| --- | --- | --- |
| 31 | public void setPassword(ArrayList<String> password) { | |
| 32 | this.password = password; |

|  |  |
| --- | --- |
| 33 | } |
| 34 | /\*\* | |

|  |  |  |
| --- | --- | --- |
| 35 | \* @return browser | |
| 36 | \*/ |

|  |  |  |
| --- | --- | --- |
| 37 | public ArrayList<String> getBrowser() { | |
| 38 | return browser; |

|  |  |
| --- | --- |
| 39 | } |
| 40 | /\*\* | |

|  |  |  |
| --- | --- | --- |
| 41 | \* @param browser | |
| 42 | \*/ |

|  |  |  |
| --- | --- | --- |
| 43 | public void setBrowser(ArrayList<String> browser) { | |
| 44 | this.browser = browser; |

|  |  |
| --- | --- |
| 45 | } |
| 46 | /\*\* | |

|  |  |  |
| --- | --- | --- |
| 47 | \* @return element1 | |
| 48 | \*/ |

|  |  |  |
| --- | --- | --- |
| 49 | public ArrayList<String> getElement1() { | |
| 50 | return element1; |

|  |  |
| --- | --- |
| 51 | } |
| 52 | /\*\* | |

|  |  |  |
| --- | --- | --- |
| 53 | \* @param element1 | |
| 54 | \*/ |

|  |  |  |
| --- | --- | --- |
| 55 | public void setElement1(ArrayList<String> element1) { | |
| 56 | this.element1 = element1; |

|  |  |  |
| --- | --- | --- |
| 57 | } | |
| 58 | /\*\* |

|  |  |  |
| --- | --- | --- |
| 59 | \* @return element2 | |
| 60 | \*/ |

|  |  |  |
| --- | --- | --- |
| 61 | public ArrayList<String> getElement2() { | |
| 62 | return element2; |

|  |  |
| --- | --- |
| 63 | } |
| 64 | /\*\* | |

|  |  |  |
| --- | --- | --- |
| 65 | \* @param element2 | |
| 66 | \*/ |

|  |  |  |
| --- | --- | --- |
| 67 | public void setElement2(ArrayList<String> element2) { | |
| 68 | this.element2 = element2; |

|  |  |
| --- | --- |
| 69 | } |
| 70 | /\*\* | |

|  |  |  |
| --- | --- | --- |
| 71 | \* @return element3 | |
| 72 | \*/ |

|  |  |  |
| --- | --- | --- |
| 73 | public ArrayList<String> getElement3() { | |
| 74 | return element3; |

|  |  |
| --- | --- |
| 75 | } |
| 76 | /\*\* | |

|  |  |  |
| --- | --- | --- |
| 77 | \* @param element3 | |
| 78 | \*/ |

|  |  |  |
| --- | --- | --- |
| 79 | public void setElement3(ArrayList<String> element3) { | |
| 80 | this.element3 = element3; |

|  |  |  |
| --- | --- | --- |
| 81 | } | |
| 82 | } |

**Step 6:** The next step is to create instances of “TestData.java” and “CommonMethods.java” java classes within the test script in order to access and populate the test data. Refer the below code snippet for object initialization, reading excel data and populating the values wherever required.

|  |  |
| --- | --- |
| 1 | // Create Objects |
| 2 | public ExcelReader excelReaderObj; | |

|  |  |  |
| --- | --- | --- |
| 3 | CommonMethods commonMethodobj = new CommonMethods(); | |
| 4 | TestData td = new TestData(); |

|  |  |
| --- | --- |
| 5 |  |
| 6 | // Load the excel file for testing | |

|  |  |  |
| --- | --- | --- |
| 7 | excelReaderObj = new ExcelReader(Path of the excel); | |
| 8 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 9 | // Load the Excel Sheet Col in to Dictionary for use in test cases | | |
| 10 | | excelReaderObj.ColumnDictionary(); |

|  |  |
| --- | --- |
| 11 |  |
| 12 | // Get the data from excel file | |

|  |  |  |
| --- | --- | --- |
| 13 | commonMethodobj.readExcelData (td); | |
| 14 |  |

|  |  |  |
| --- | --- | --- |
| 15 | | // Populate the username |
| 16 | driver.findElement(By.id("idofElement")).sendKeys(data.getLoginUser().get(0)); | | |