eTaap Keyword Driven Model

**User Guide**

*Created by*

eTouch Systems Corp

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| * **Revision No.** | * **Date** | * **Name (Alias)** | * **Description of Change** |
| * 1.0 | * 03/07/2016 | * eTouch | * Created Keyword Driven user guide |



**Overview**

The Keyword Driven Framework is designed to minimize the manual testing efforts. The framework should help to reduce the efforts for regression testing. For this purpose, an excel file is created, with separate sheets as Test Suites defining individual test cases to execute. In addition, data set can be configured to hold and supply the data corresponding to a test case.

The purpose of this document is to elaborate the structure of the excel sheets, categorizing and identifying as Test Suites, Test Cases and Data Set. Defining the columns present in them, along with the values required to enter and the functionality behind it.

# Getting Started with KD (KeywordDriven)

The excel sheet can have following sheets present:

1. Test Suite Sheet: Identified by suffix ‘\_Suite’
2. Test Data Sheet: Identified by suffix ‘\_Data’
3. Global Test Data Sheet: Identified by suffix ‘\_Global\_Data’

Multiple test suites can be maintained in one excel file, separated by the sheets. If so, unique names should be maintained for each of them. The data present in the Global Data Sheet can be used by all of these Suites.

For the data, specific to a certain suite a separate sheet can be created, apart from the Global Data Sheet. Generally, this sheet should be named in such a way that the user is able to correspond the data present in Data Sheet with the Suite.

e.g. If the name of the Test Suite is ‘RegressionTest\_Suite’ the corresponding Data Sheet name should be: ‘RegressionTest\_Data’.

## Config setting

The Config.yml file needs to add the following code.

## Execution flow for TestBeds / Test Suites / Test Cases

kdConfig:

execute: true

filePath: '../TestProject/src/test/resources/testdata/TestSuites.xlsx'

#Parallel- All Combination

parallelMode: testsuites

**Property values**

1. **execute**
2. true: Test cases will execute in Keyword Driven mode when set to true.
3. false: Test cases will no execute in Keyword Driven mode.
4. **parallelMode**
5. testbeds: Executes all the test beds (browsers) parallel.
6. testsuites: Executes all the test suites (across the excel file) parallel.
7. testcases: Executes all the test cases (across the test suites combined) parallel.
8. false: Execute all the test cases sequentially.

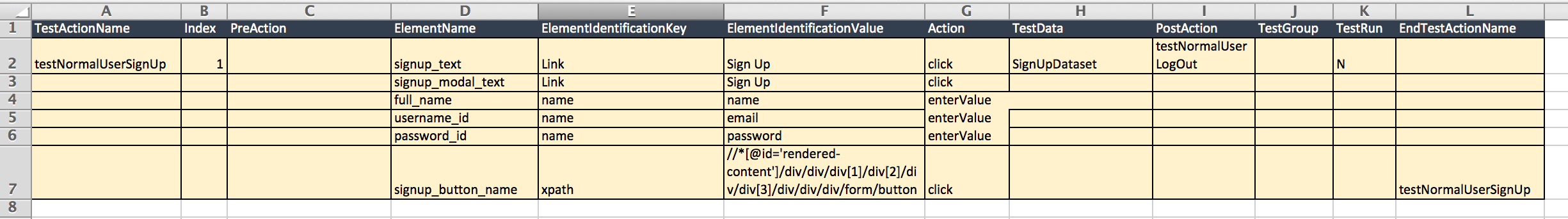
*Things to Remember:*

* *parallelMode values are case sensitive.*
* *Only listed values are supported, any other values will not guarantee proper execution flow.*

## Test Case setup

#### Configuring Test Suites

#### The sheet for suite appears like this:



**Columns:**

1. **TestActionName**

This column mentions the name of the test cases to be executed. The name can be chosen by the user. The same name must be used to start and end the test case.

*Things to Remember:*

* *Only at the first row of the test case can have TestActionName*
* *Mandatory value*
* *Must be unique across the Test Suites*

1. **Index**

Shows the Priority of the Test Cases to be executed. The decimal numbers are also accepted for this purpose, and a whole integer is not needed.

*Things to Remember:*

* *Only at the first row of the test case can have Index value*
* *Mandatory value*
* *Numeric – 1, 2, … or 1.2, 2.1, …*

1. **PreAction**

Lists the test cases, that are required to be executed before the current test case is executed. The name of the test case is entered in the first cell of the column, and the rest of the column is left blank.

In case of multiple test cases are required to be executed, they should be separated by a comma.

*Things to Remember:*

* *The value must be available in the TestActionName list*
* *Multiple PreAction values are allowed with comma separator*

1. **ElementName**

This column contains the name of the element, on which the action is performed. The name present in this column needs to be accurately mapped with the name in the corresponding dataset

*Things to Remember:*

* *Must match with the Data set column if any data set is present.*

1. **ElementIdentificationKey**

The attribute by which the element is identified.

|  |  |  |
| --- | --- | --- |
| ELEMENT IDENTIFICATION KEYS | | |
| id | pageTitle | multiValueCss |
| xpath | window | multiValueLink |
| Link | methodInvoke | multiValuePartialLink |
| name | css | multiValueClass |
| URL | multiValueXpath | multiValueTag |

*Things to Remember:*

* *Only Key provided in the Library are supported*
* *Case sensitive*

1. **ElementIdentificationValue**

Actual value of the element to be identified.

1. **Action**

This column indicates the action to be performed on the element selected. It can be selected from a list of pre-determined actions. The library as shown below:

|  |  |  |
| --- | --- | --- |
| WINDOW ACTIONS | | |
| swipe | scroll | pinch |
| zoom | tap | resize |
| getCoordinates | getSize | OpenBrowser |
| openNewTab | navigateToUrl | getBackToURL |
| scrollUp | zoomIn | zoomOut |
| toggleWindow | zoomTo100 | closeBrowser |
| scrollDown | scrollLeft | scrollRight |
| scrollBottom | close\_ToggleWindow | scrollTop |

|  |  |  |
| --- | --- | --- |
| PAGE ACTIONS | | |
| click | hover | doubleClick |
| clear | submit | check |
| uncheck | dragAndDrop | selectDropDownList |
| getText | getCssValue | getWebElement |
| clickEvent | loadPage | scrollToElement |
| executeJava |  |  |

|  |  |  |
| --- | --- | --- |
| FORM INPUT ACTIONS | | |
| enterValue | sendKeys |  |

|  |  |  |
| --- | --- | --- |
| ASSERT ACTIONS | | |
| hasValue | contains | hasContent |
| hasName | hasSrc | hasHref |
| isElementVisible | isEnabled | isSelected |
| isDisplayed | isChecked | hasChildCount |
| isElementVisible | isSelected | isGreaterThan |
| hasImageAltText | isLessThan | hasImageTitle |
| isSuccess | hasStyle | styleContains |
| isURLMoved | isCreated | isAccepted |
| isInformationPartial | hasNoResponse | isBadRequest |
| isUnauthorized | isPaymentRequired | isForbidden |
| isNotFound | hasInternalError | hasNotImplemented |
| isServiceOverloaded | isGatewayTimeout | isURLRedirected |

*Things to Remember:*

* *Only Key provided in the Library are supported*
* *Case sensitive*

1. **TestData**

This column lists the dataset name, from which the data is to be used for the test case. The name for the columns in the dataset should be exactly the same as the ‘ElementName’ of the element for which the data is to be populated

*Things to Remember:*

* *Not mandatory.*
* *If TestData has value, it must be available in the TestData sheet.*

1. **PostAction**

Lists the test cases that are to be executed after the main test case is executed.

*Things to Remember:*

* *The value must be available in the TestActionName list*
* *Multiple PostAction values are allowed with comma separator*

1. **TestGroup**

Contains the name of the TestGroup, to which the test case belongs.

1. **TestRun**

This is the flag, that determines if the test case is to be executed or not.

If the test case is executed as part of PreAction or PostAction of another test case, this flag does not affect the execution. Valid value is “Y/y” or “N/n”

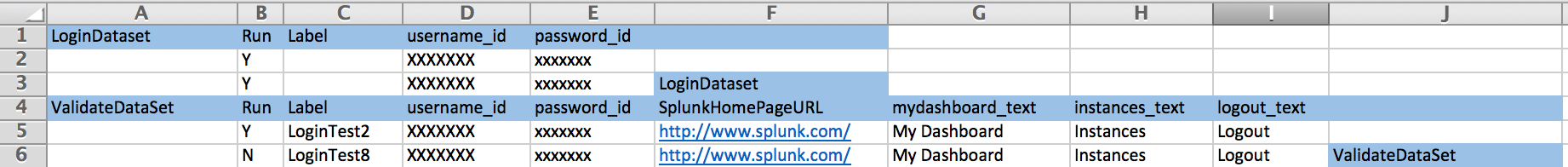
1. **EndTestActionName**

This column signifies the end of the test case, and lists the name of the same in the bottom cell

*Things to Remember:*

* *Must match with the TestActionName value*
* *Only at the last row of the block that belongs to the TestActionName*
  + 1. **Configuring Test Data**

The sheet for data looks like this:



In the Data Sheet, there are datasets present, which hold the data that can be passed to the test cases. The dataset name is unique, and marks the beginning and end of the dataset.

The same dataset can be used for multiple test cases.

**Dataset**

In the dataset, there are a few mandatory columns, which contains the information regarding the dataset. The other columns, which contain the data for the test cases, can vary in the number according to the values required for test case/s using the dataset.

**Columns**:

1. **Beginning of the dataset**

This column contains the name of the dataset, in the first cell. This column does not have a heading. The dataset name serves as the heading instead

1. **Run**

This column contains the flag, that determines if the values listed in the column will be considered for test case execution or not

1. **Label**

This column contains the label for the data in the subsequent row. The values in this column are optional.

1. **Data Columns**

These columns, which vary in number contain the actual data that is passed to the test cases.   
The names of these column headers should be exactly as the corresponding values in the ‘ElementName’ column of the test case, which is using the data for execution

1. **End of the dataset**

This Column indicates the end of the dataset. It contains the name of the dataset in the last cell, to mark the end. This column, like the first column does not have a header.

*Things to Remember:*

* *The first column value must be match with the TestData value in the TestSuite Sheet*
* *The second column header must have data “Run”. It must has value as “y”, or “n”. It’s case insensitive*
* *Each ElementName must have the matching TestData value in the Data sheet for which input data is required.*
* *The TestData name must match with one of the Data Sheet*
* *The TestData Set needs to have all the input values which belongs to the PreAction/PostAction and combine with the Elements needs input from the TestCase*
* *Multiple test case can reuse the same TestData if it has the matching Data Set*

**Integration with Defect tools**

The Keyword Driven Framework provides extensive support to complete Defect Management cycle along with Integration to any Defect management tools available today.

This document illustrates below integration with JIRA and Rally.

# Jira Integration

## Config Settings:

The Config.yml file needs to add the following code.

defectManagementTool: Jira

#Jira Config

jiraConfig:

username: testUser

password: testPassword

url: https://mycompany.atlassian.net/

key: PROJ

projectName: MyProject

environment: QA

screenshotAttachment: true

videoAttachment: false

**Property values**

1. **defectManagementTool**: Jira.

*(Name of defect tool to be integrated)*

1. **jiraConfig**
2. username: Provide the user login name
3. password: Provide the user password
4. [url: Jira](url:Jira) URL for integration
5. key: Project key for which defect to be logged and managed.
6. projectName: Name of the project.
7. environment: Environment in which defect to be logged. It should match with the configured environment in Jira
   1. QA
   2. Prod
8. screenshotAttachment:
   1. true: Will attach the screenshot for failed test cases everytime a defect if created/reopened/updated.
   2. false: Will not attach any screenshot.
9. videoAttachment:
   1. false: Will not attach any Videos to the defect.

*Things to Remember:*

* *Configuration setting should be valid and match to the corresponding property field.*
* *Video Attachment is not currently supported hence it should always contain default value as ‘false’*

# Rally Integration

## Config Settings:

The Config.yml file needs to add the following code.

defectManagementTool: Rally

#Rally config

rallyConfig:

projectId: 34867036301

workspaceId: 34866816045

defectOwner: 16693935729

build: 1.0

username: testuser@mycompany.com

password: UmFsbHkxMjMh

url: https://rally1.rallydev.com

rallyApp: MyProject

screenshotAttachment: true

videoAttachment: false

**Property values**

1. **defectManagementTool**: Rally.

*(Name of defect tool to be integrated)*

1. **rallyConfig**
2. projectId: It should be a unique number
3. workspaceId: Should be unique number
4. defectOwner: Unique number provided for logged user
5. build: Version of the Rally to be configured.
6. username: Registered user email or id
7. password: Password needs to be encrypted
8. url: Rally URL
9. rallyApp: Project name under which defect will be logged and manged.
10. screenshotAttachment:
    1. true: Will attach the screenshot for failed test cases every time a defect if created/reopened/updated.
    2. false: Will not attach any screenshot.
11. videoAttachment:
    1. false: Will not attach any Videos to the defect.

*Things to Remember:*

* *Configuration setting should be valid and match to the corresponding property field.*
* *Video Attachment is not currently supported hence it should always contain default value as ‘false’*

**Reporting**

Custom Report is generated providing following details:

1. File Name of the excel sheet containing the test cases.
2. Execution style.
3. Test case executed per Test Suites and their Test Result (passed/failed) on Test beds (browser)
4. Test case which are excluded.
5. Summary of test cases

Report generated can be located as:

1. File Name: KDTestNgCustomReport.html
2. Path: ..\MyProject\test-output\

*Things to Remember:*

* *All the format error in Excel sheet will be validated before actual execution of the test cases.*
* *In the event there are any excel validation error found it will displayed in this report.*

**Sample Test Case Report**

