T-base WATS Logger

Manual

Status:	Approved
Version:	1.0
Creator:	Torben Christensen
Approver:	Michael Rothmeier
File name:	T-base WATS Logger Manual.dotx
Doc Location:	Rebase SVN server

Change History

Status	Version	Date	Handled by	Comments
Approved	1.0	2023-05-24	Torben Christensen	Initial version

Contents

1. Purpose of this document

This describes the implementation of the T-base WATSLogger.dll and how to use T-base to get structured result data in WATS. See also WATS and the T-base Framework documentation.

2. High level description

The WATS Logger uses the WATS API. Data is collected by T-base Dataservice while the test is running. When the sequence is done, all data is transferred to the WATS API. The implementation supports results of type passed/failed, numeric and string.

3. WATS Service Address

The connection to the WATS cloud is configured in the WATS Client. See WATS documentation.

4. Naming differences between WATS and T-base

Naming differences:

T-base	WATS
Dut.Product	Part number
Dut.Note/Description	Comment
Dut.SwVersion	Revision
JobNumber	BatchSerialNumber
Fixture	TestSocketIndex

The WATS logger converts T-base verdict enum to the WATS verdict enum:

```
switch (result.Verdict)
{
    case DataService.Verdict.Passed:
         stepVerdict = StepStatusType.Passed;
         break;
    case DataService.Verdict.Failed:
         stepVerdict = StepStatusType.Failed;
         break;
    case DataService.Verdict.Interrupted:
         stepVerdict = StepStatusType.Terminated;
         break;
    case DataService.Verdict.ErrorInExecution:
         stepVerdict = StepStatusType.Error;
         break;
    case DataService.Verdict.NotApplicable:
    case DataService.Verdict.None:
    default:
         // None is not allowed for Test in WATS
         stepVerdict = StepStatusType.Passed;
         noVerdict = true; // Only logging
         break;
 }
```

Other items have similar naming in WATS and T-base.

5. WATS OperationType

The logger and WATS require that OperationType is set. Use following code snippet:

Log.UpdateTestRunInfoDictonary("WatsOperationType", “10”);

6. WATS ReportText

If the ResultMessage is set for a T-base result object the message will show up in the WATS Report.

7. File attachments

All types of results can have a file attached, e.g., an image. The file will be shown as a link in the result report. The T-base result object has the property ReferenceToDataFile.

DataStorageResult result = new DataStorageResult(Description, ResultString, "string", "SUCCESS", verdict, null);
result.ReferenceToDataFile = FilePathManager.testSystemBitMapPath + "green_smiley.jpg";

8. Sequence looping

T-base supports advanced step looping, testing steps at different test conditions. It is not directly supported by WATS that results has test conditions. This logger adds the test conditions to the

result description in the format: Condition:Description1=Value1 | Description2=Value2… | ResultDescription.
Note that the WATS Result Description is limited to 100 characters! So, this will not work with many test conditions and long descriptions. ‘See also T-base Result Group’

9. T-base Test Cases

T-base has a result level called test case. This can hold many result rows. When test cases are used, the result will be shown under a nice expander in the WATS report:

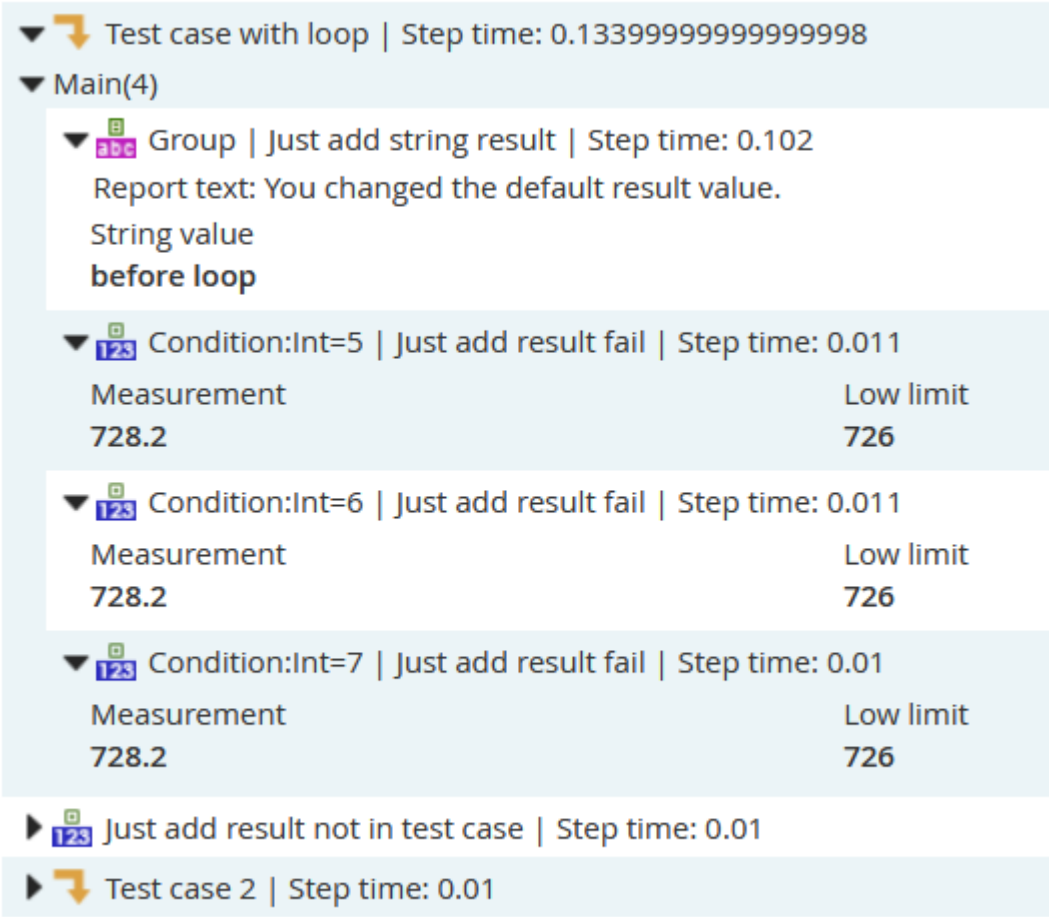


Figure 1. WATS report example

Note that the sum of test times for all test steps in the test case is shown.

See also ‘T-base Result Group’.

10. T-base Result Group

T-base has an additional result level called ‘Group’. One or many results can be marked with a Group description to indicate thar these results are related. The Group name is added to the result description in the WATS report in the format: ‘GroupName | ResultDescription’.
Note that the WATS Result Description is limited to 100 characters!

See also T-base Test Cases’ and Figure 1. WATS report example.

11. Device Under Test Sub-units’ information

The logger supports information about the device under tests sub-units.
Use the following code snippet to add any number of sub-units:

```
Log.UpdateDutInfoDictonary("Module=Heater", "SerialNo=123:FWVersion=1.2.3");
```

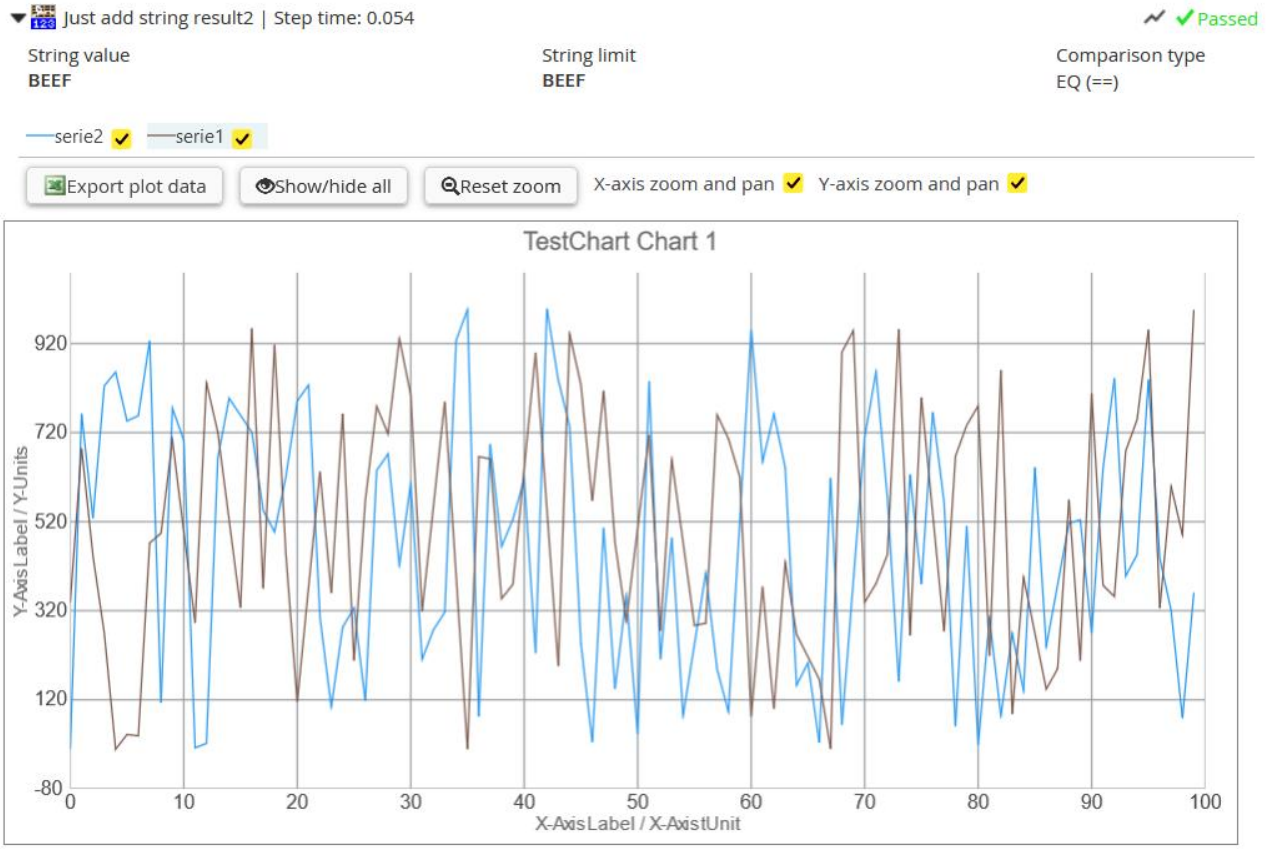
Figure 2. Sub units shown in the WATS report

12. Charts

All types of results can have a chart. The following adds a chart to a string result with two series:

```
DataStorageResult result2 = new DataStorageResult(Description + "2",   dummyResult,
"string", "BEEF", verdict, null);

result2.Chart = new DataStorageChart("TestChart Chart 1");
result2.Chart.ChartType = ChartType.Line; //Optinal. Line is default
result2.Chart.X_AxisLabel = "X-AxisLabel"; //Optinal
result2.Chart.X_AxistUnit = "X-AxistUnit"; //Optinal
result2.Chart.Y_AxisLabel = "Y-AxisLabel"; //Optinal
result2.Chart.Y_AxisUnits = "Y-Units"; //Optinal

List<double> serie1 = new List<double>();
List<double> serie2 = new List<double>(); //Optinal

Random r = new Random();
for (int x = 0; x < 100; x++)
{
   serie1.Add((double)r.Next(1000));
   serie2.Add((double)r.Next(1000)); //Optinal
}

result2.Chart.Series.Add("serie1", serie1);
result2.Chart.Series.Add("serie2", serie2); //Optinal
Log.AddResultInformation(result2);
Log.NextResultRowInformation();
```

And the chart in the WATS report:



13. Test time

A T-base step can generate none, one or more results. Step description and result description may not be the same. So test time linked to a result has been solved in the following way.

WATS now gets StepTimeElapsedSeconds from when the step.execute was called for the first result. If the T-base test step generates more results, the next result time is set to the elapsed time between the last and present results. If the T-base test step has functionality after the last result, this will not be included in the time registration.