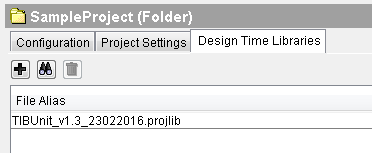
TIBUnit – Developer’s Guide

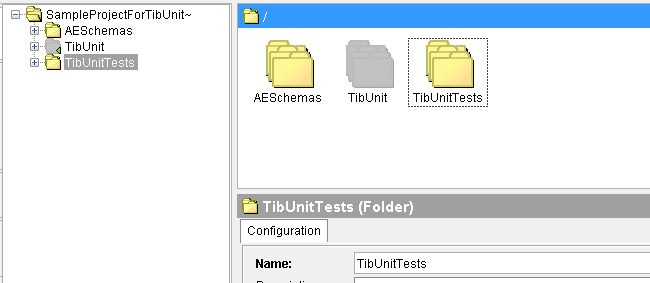
* In this guide, we are calling each unit test process as a TibUnit.
* Input data for a TibUnit can be given in two two ways-
  + Without any input data file. In this case input data will be hard coded inside the TibUnit.
  + With one or more input data file. In this case, input will be kept in a file. File data will be available inside the TibUnit and developer should know how to map the file data inside the TibUnit assertions.
  + Optionally, expected output data file can also be defined for a TibUnit. This file data will be available inside TibUnit, and developer should know how to map file data in TibUnit assertions.
* The naming convention for input file is “**input\_*TibUnitName\_*<id>.xml**”, where *TibUnitName* is the name of the TibUnit corresponding to which, this file has to act as input. **e.g**..: input\_TestLogin\_1.xml
* The naming convention of the output file is “**output\_*InputFileName***”, where *InputFileName* is the name of the input file corresponding to which, this file has to act as output. **e.g**. : output\_ input\_TestLogin\_1.xml
* For creating a new TibUnit, there is a SampleTest TibUnit provided in the library, and developer can copy this process to create new TibUnits.

Following section describes what steps a developer has to perform to write and run TibUnits.

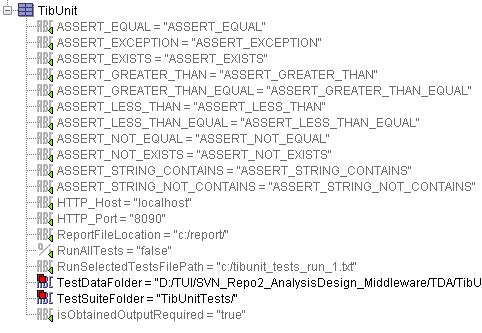
1. Import the **TibUnit\_<version>.projlib** in the project’s design Time Libraries.



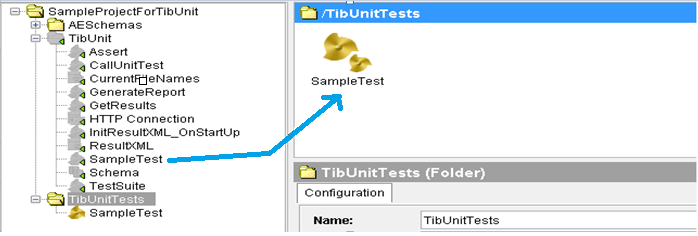
1. In the project at root level, create a folder named: **‘TibUnitTests’**. This name is also specified in the global variables list.

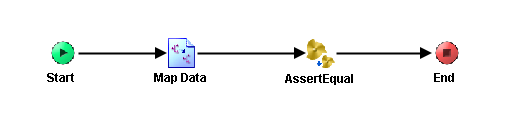


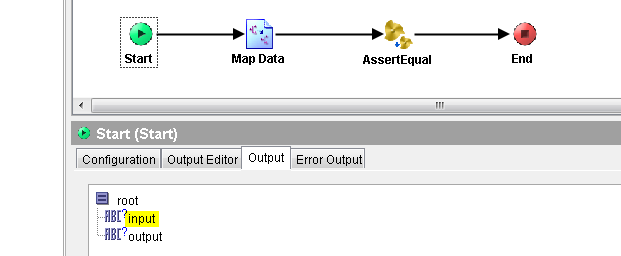
1. All TibUnits are to be created inside this folder. Any hierarchy can be followed inside this folder to systematically categorize and organize the TibUnits.
2. In the global variables, there is a folder named ‘TibUnit’ which contains the variables which configure the TibUnit test suite. Location for test data (input and output files) and report files are mentioned here. They can be changed if required by overriding.

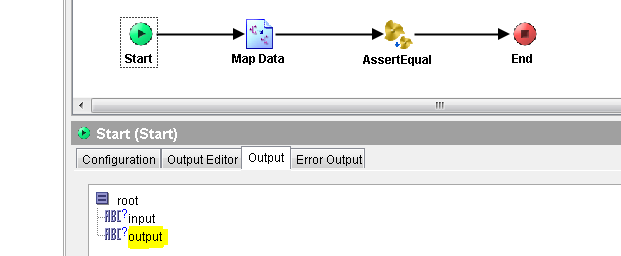
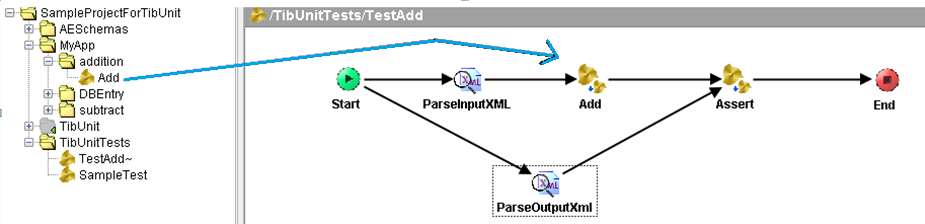


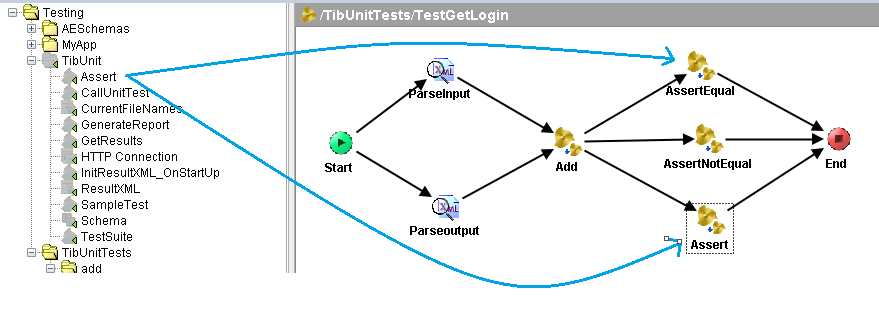
1. A TibUnit is to be created to test any process. For this, copy the sample test provided in TibUnit and rename it as per requirement.

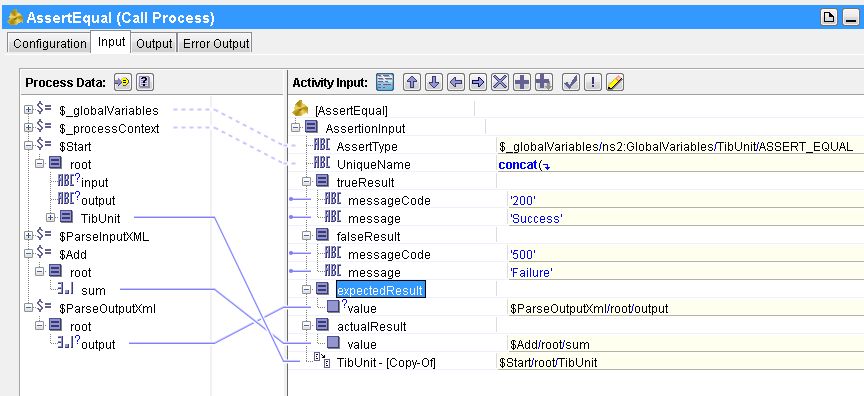


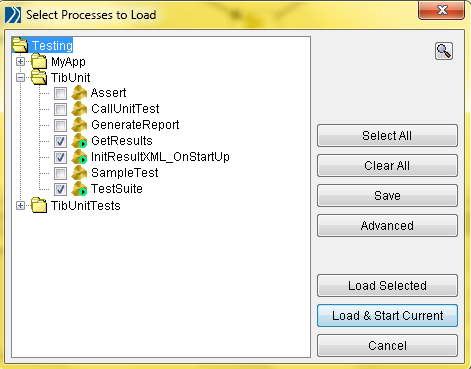
1. By default, the SampleTest will look like this: 
2. For every TibUnit you create, input and output files are to be provided (Follow the naming conventions mentioned at the start of this document). Input Files and Output files are not mandatory for each TibUnit. But to run a TibUnit for more than one test data, Input Files and Expected Output files should be setup. So for different test data, different input files are to be provided in the location mentioned in the global variables. Input file has to contain the input data for the process, the content of this file will be available in the TibUnits Start’s output, highlighted in yellow. You may need to parse this input as xml and then can simply map it to the input of your process.



1. Start’s Output also contains ‘output’, which contains the content of the output file (if provided) corresponding to the current input file. This provides as the expected output of your process. You may also need to parse this according to the schema you want and per the assertion you need to apply. This need not necessarily be the exact schema as of the output of your process.
2. Once the sample test is copied, renamed and provided with the test data, call the process which needs to be tested and provide the input from the file or hard code it if required. 

1. By default, one assert process is available in one sample test. Multiple Assert processes can also be used in a single TibUnit. For this, assert process can be dragged and dropped inside the TibUnit 
2. Providing Inputs for Assert:
   1. AssertType: It has to take a value from global variables, depending on what the user wants to assert. Variables are self-explanatory.
   2. UniqueName: To be provided by user to uniquely identify the TibUnit from the report file.
   3. trueResult: Message code and Message to be displayed in report file in case the assert is a success.
   4. falseResult: Message code and Message to be displayed in report file in case the assert is a success.
   5. expectedResult: To be mapped from the expected output obtained from ‘Start’ of the process.
   6. actualResult: To be mapped from the output obtained from the original process.
   7. TibUnit: This is required for TibUnit framework to work smoothly, so just copy the TibUnit node from Start into this element.



1. Once you have provided all the required inputs for all the TibUnits and input and output files in their respective locations, the test Suite is set to run. 3 Processes need to be run from the TibUnit Folder:
   1. TestSuite: Which will run all the TibUnits
   2. InitResultXML\_OnStartUp: Initializing Shared Variables.
   3. GetResults: To see the result after the test has been successfully run.
   4. Note: Starter Processes of the actual project also need to be started if internally they are getting used inside any of the TibUnits.
2. Once the TestSuite run has completed, the report of the TestSuite can be seen at the URL: ‘http://localhost:8090/results’
3. The same report is also saved as an html file in the report file location mentioned in the global variables. The URL will give the result of the last run. Whereas the report is saved for every run executed and can be seen any time later on.
4. If any of the assert fails, the obtained output from the actual process is also saved in the location where expected output is saved with the same file name appended by ‘obtainedOutput’. To check and compare where the issue might have aroused.
5. There are two GVs that allows us to control which TibUnits to run and which not to run.
   1. TibUnit/RunAllTests : If this is set to true, all the TibUnits created in the project are executed and asserted. If the value of this GV is false, then only those TibUnits are executed whose names are mentioned in the configuration file. The path of configuration file is defined in below mentioned GV.
   2. TibUnit/RunSelectedTestsFilePath : This GV contains the full file path to the configuration file that has the comma separated names of all TibUnit processes to be executed. Only those TibUnits will be loaded and executed whose names are mentioned in the file. Also, if in this file, any TibUnit name has “- -“ (minus minus) appended in the start, then that TibUnit is not executed. See the example below. As per this configuration file, 3 TibUnits will be executed and 1 TibUnit will not be executed as it has - - before its name. The name of TibUnit in the file should have names of the all the parent folders starting from the TibUnits folder name in the project.

