

Inhaltsverzeichnis

Terminology	2
Installation	2
<i>Prerequisites</i>	2
<i>Download and install ZIP-file</i>	2
<i>Configuration</i>	4
Documentation	5
<i>Running SASUnit examples</i>	5
<i>SASUnit Test Documentation</i>	7
<i>Logfiles</i>	10
SASUnit Macros	10
<i>Controller Macros</i>	10
initSASUnit	11
runSASUnit	11
reportSASUnit	11
initTestcase	11
endTestcall	11
endTestcase	11
<i>Asserts</i>	11
assertColumns	11
assertEquals	12
assertForeignKey	12
assertLibrary	12
assertLog	12
assertLogMsg	12
assertPerformance	12
assertRecordCount	12
assertRecordExists	13
assertReport	13
assertRowExpression	13
assertTableExists	13
Getting started	13
<i>Write a simple test</i>	13
<i>Execute the test</i>	15
<i>Look at the results</i>	15
<i>Creating your own project</i>	16

Terminology

Test Macro

SAS Code that needs to be tested.

Test Case

A call of a test macro in a controlled environment to test a single(!) feature of the test macro.

Assert

A macro that ensures the adherence to a single feature of the test macro.

Test Scenario

The collection of all test cases for one or more test macro. A scenario covers all test cases for a specific business case (see examples scenario database_test.sas)

Test driver

A program that represent a test scenario.

Installation

Prerequisites

Windows or Linux

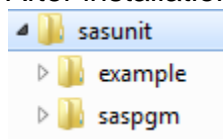
SAS® 9.4, 9.3 or 9.2 in English or German

Download and install ZIP-file

Download the current SASUnit-Version from sourceforge.net/projects/sasunit/ Install the downloaded ZIP-file at your preferred location. For example C:\sasunit.

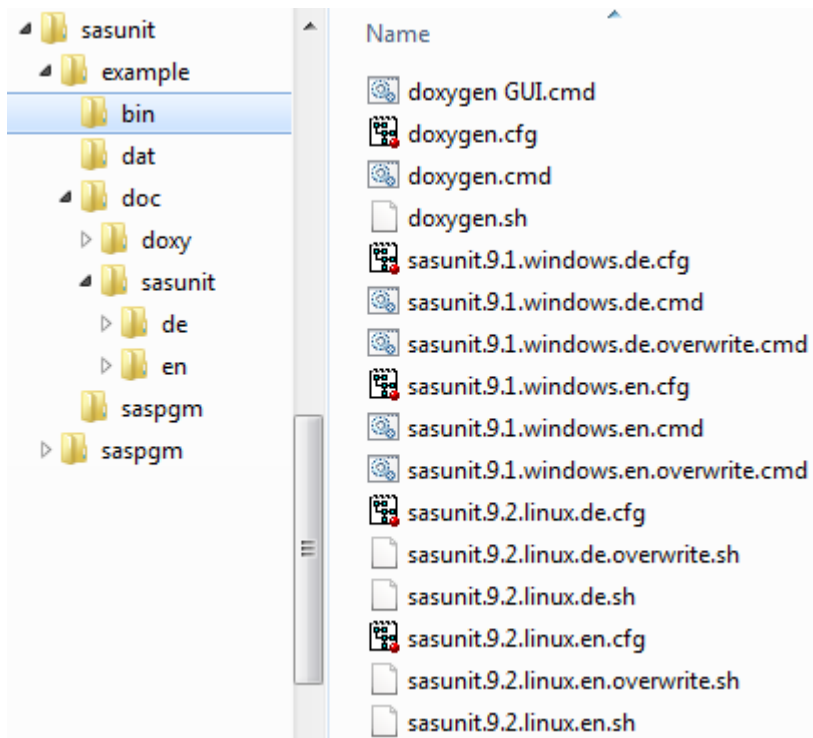
For non-Windows users: Since the zip-file was packed in Windows you need to unpack it on a windows client and copy the files to your target platform using WinSCP or similar. This ensures correction of CR/LF in text and program files. Unpacking the archive directly on LINUX requires additional steps to clean CR/LF in text and program files.

After installation your folder structure should look like this:

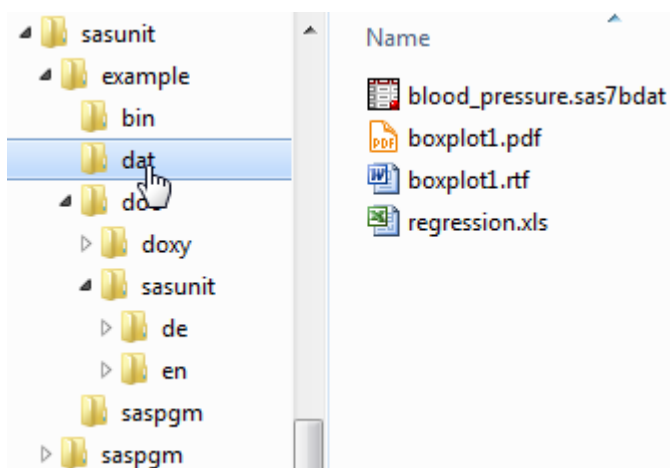


example: includes a sample program to practice SASUnit and to ensure the proper functionality of SASUnit (OQ). Inside the example folder there is a predefined example to test SASUnit and to learn what SASUnit is all about.

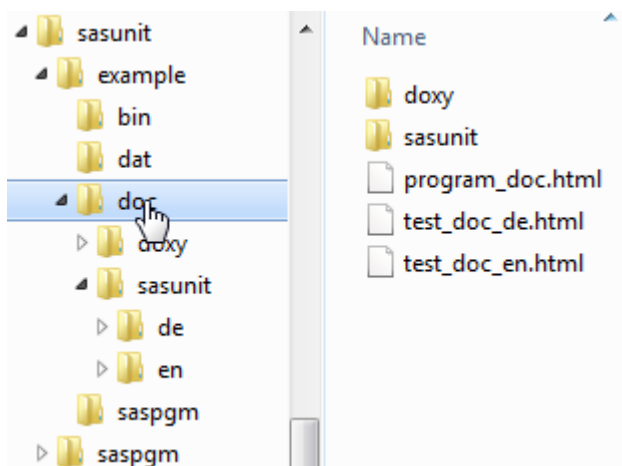
saspgm: includes the macro code for the Unit Testing Framework



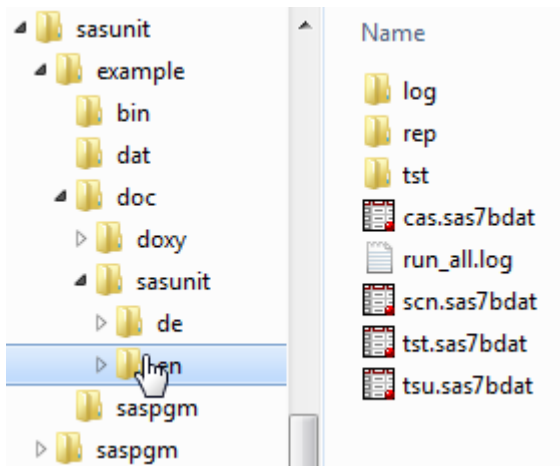
bin: Includes all the scripts to run the SASUnit-code depending on which version of SAS® and which operating system you are running.



dat: Contains test- and reference data that is used in the SASUnit examples.



doc: Contains the SASUnit- and code documentation including links that refer to the results page. The SASUnit documentation is stored in subdirectories depending on the chosen language.



en: includes the sample database and the control modul logfile

en\log: includes logfiles of test scenarios

en\rep: includes the SASUnit documentation

en\tst: includes the intermediate results of the test scenarios

Configuration

Windows Users need to adapt paths in case their SAS®-Installation does NOT use the standard installation path.

First the installation path in the shell script needs to be adapted to the used folder structure. Here for example you can see sasunit.9.3.windows.en.cmd

```
@echo off
rem          Copyright 2010, 2012 HMS Analytical Software GmbH.
rem          This file is part of SASUnit, the Unit testing framework for SAS(R) programs.
rem          For terms of usage under the GPL license see included file readme.txt
rem          or https://sourceforge.net/p/sasunit/wiki/readme.v1.2/.
@echo on

cd ..
SET SASUNIT_ROOT=.\
SET SASUNIT_OVERWRITE=0
SET SASUNIT_LANGUAGE=en
SET SASUNIT_HOST_OS=windows
SET SASUNIT_SAS_VERSION=9.3
SET SASUNIT_COVERAGEASSESSMENT=1
"C:\Program Files\SASHome\SASFoundation\9.3\sas.exe" -CONFIG "bin\sasunit.%SASUNIT_SAS_VERSION%.%SASUNIT_HOST_OS%

@echo off
if %ERRORLEVEL%==0 goto normalexit
@echo.
@echo Exit code: %ERRORLEVEL%
@echo.
if %ERRORLEVEL%==1 @echo SAS ended with warnings. Review run_all.log for details.
if %ERRORLEVEL%==2 @echo SAS ended with errors! Check run_all.log for details!
@echo.
pause
:normalexit
```

Second the installation path in the config file needs to be changed. For example sasunit.9.3.windows.en.cfg

```

/**
\copyright Copyright 2010, 2012 HMS Analytical Software GmbH.
           This file is part of SASUnit, the Unit testing framework for SAS(R) programs.
           For terms of usage under the GPL license see included file readme.txt
           or https://sourceforge.net/p/sasunit/wiki/readme.v1.2/.
*/

-CONFIG "C:\Program Files\SASHome\SASFoundation\9.3\nls\en\SASV9.CFG"
-sysin "saspgm\test\run_all.sas"
-log "doc\sasunit\en\run_all.log"
-print "doc\sasunit\en\run_all.lst"

```

Linux-Users need to adapt one path in any case.

The script file for example `sasunit.9.3.linux.en.sh` contains a path to the SAS® executable. Please adapt this path so it suits your folder structure.

```

#!/bin/bash
# Copyright 2010, 2012 HMS Analytical Software GmbH.
# This file is part of SASUnit, the Unit testing framework for SAS(R) programs.
# For terms of usage under the GPL license see included file readme.txt
# or https://sourceforge.net/p/sasunit/wiki/readme.v1.2/.

cd ..
export SASUNIT_ROOT=$(readlink -f .)
export SASUNIT_OVERWRITE=0
export SASUNIT_COVERAGEASSESSMENT=1
export SASUNIT_LANGUAGE=en
export SASUNIT_HOST_OS=linux
export SASUNIT_SAS_VERSION=9.3
export SASCFGPATH=./bin/sasunit.${SASUNIT_SAS_VERSION}.${SASUNIT_HOST_OS}.${SASUNIT_LANGUAGE}.cfg
echo "Starting SASUnit ..."
/usr/local/SASHome/SASFoundation/${SASUNIT_SAS_VERSION}/bin/sas ${SASUNIT_LANGUAGE} -nosyntaxcheck -noovp

# Show SAS exit status
RETVAL=$?

```

Documentation

Running SASUnit examples

First let the SASUnit examples script run so you can have a look at the generated example files and the documentation. You find the script files under `sasunit/example/bin`. There you have different versions of script files depending on which version of SAS® you are using and which operating system you are working on.

The naming convention is as follows:

Sasunit.<SASVersion>.<OS>.<Language>

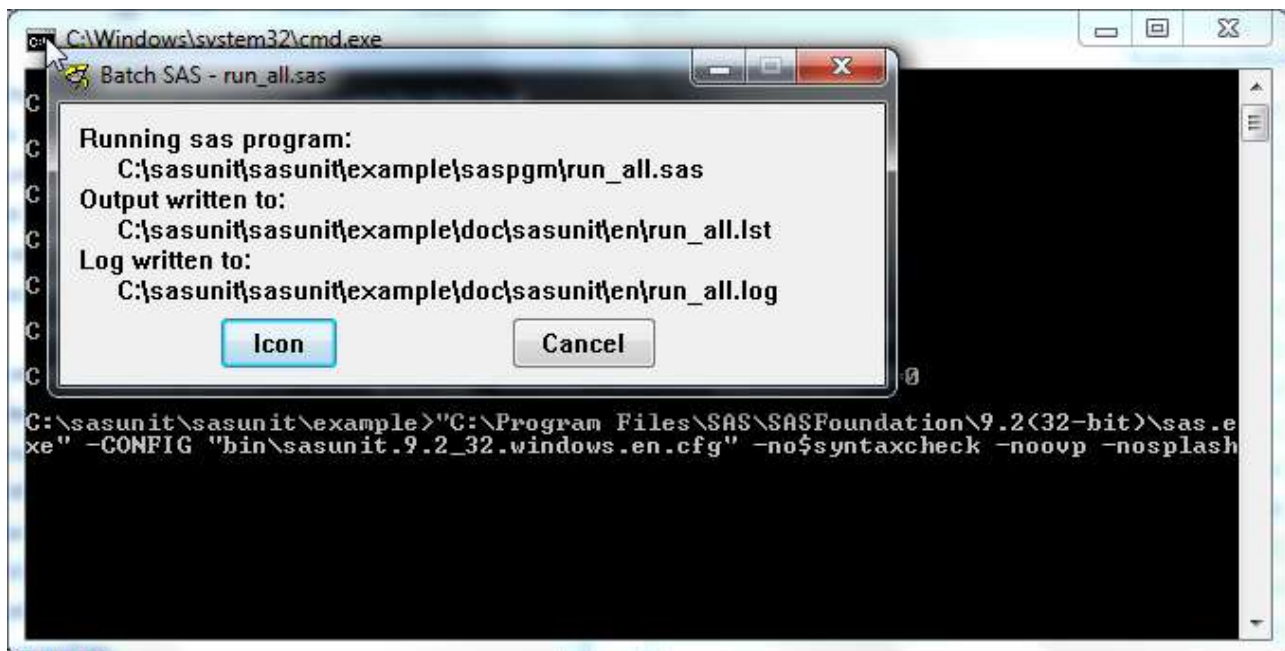
Filename	SAS® Version	Operating System	Report Language
<code>sasunit.9.2.windows.de.cmd</code>	9.2 64bit	Windows	German
<code>sasunit.9.2.windows.en.cmd</code>	9.2 64bit	Windows	English
<code>sasunit.9.2_32.windows.de.cmd</code>	9.2 32bit	Windows	German
<code>sasunit.9.2_32.windows.en.cmd</code>	9.2 32bit	Windows	English
<code>sasunit.9.3.windows.de.cmd</code>	9.3 64bit	Windows	German

Filename	SAS® Version	Operating System	Report Language
sasunit.9.3.windows.en.cmd	9.3 64bit	Windows	English
sasunit.9.3_32.windows.de.cmd	9.3 32bit	Windows	German
sasunit.9.3_32.windows.en.cmd	9.3 32bit	Windows	English
sasunit.9.4.windows.de.cmd	9.4	Windows	German
sasunit.9.4.windows.en.cmd	9.4	Windows	English
sasunit.9.2.linux.de.sh	9.2	Linux	German
sasunit.9.2.linux.en.sh	9.2	Linux	English
sasunit.9.3.linux.de.sh	9.3	Linux	German
sasunit.9.3.linux.en.sh	9.3	Linux	English

Double click on the filename that fits your system environment. A console and a batch version of SAS® will start. If you start one of the above script files only the changed test scenarios will be executed.

The suffix *.overwrite* will execute all test scenarios independent of any changes made in the test macros.

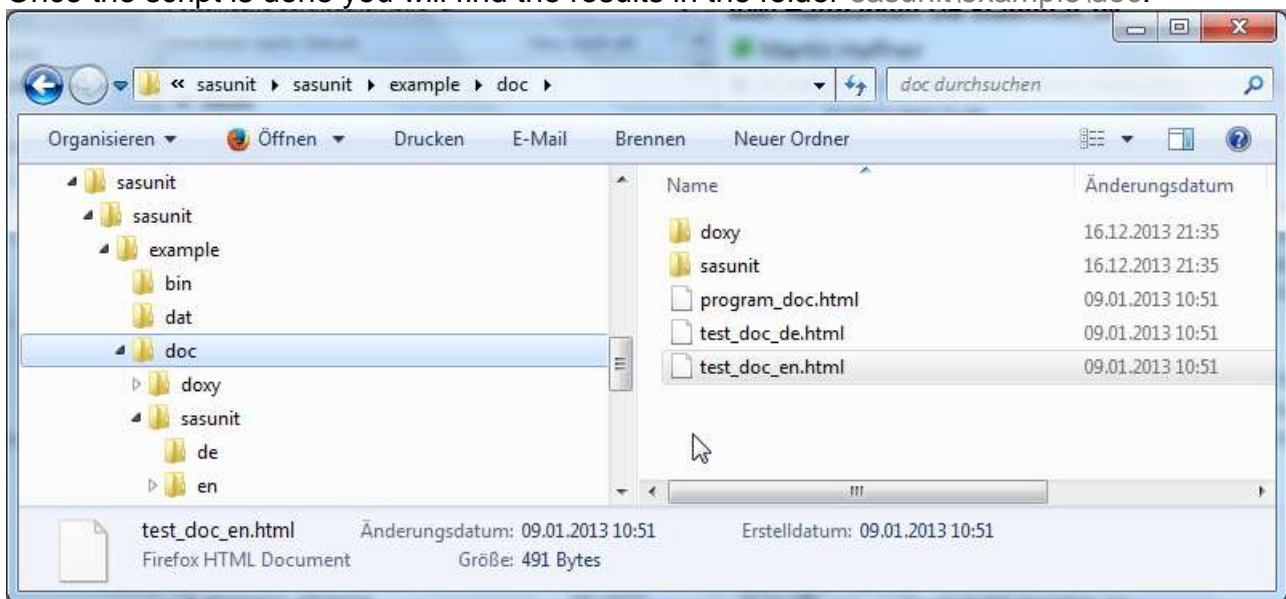
The suffix *.ci* contains scripts for requests from ci-server without user interaction.



SASUnit now checks the example code, which you find in `sasunit\example\saspgm`. This folder contains a Macro and testcase for each example. The testcase is always named `_test.sas` at the end.

SASUnit Test Documentation


Once the script is done you will find the results in the folder `sasunit\example\doc`.



Have a look at the documentation. Depending on the language settings you chose you double click on the equivalent HTML-file. Your standard browser will open and forward you to the main page of SASUnit Test Documentation.


SASUnit Examples

- Scenarios
- Units under Test

SASUnit Examples - SASUnit  Test Documentation

Properties of this test suite:

Name of project	&g_project	SASUnit Examples
Root directory	&g_root	C:\sasunit\sasunit
Path to test repository	&g_target	example\doc\sasunit\en
Program libraries (macro autocall paths)	&g_sasautos	example\saspgm
SAS configuration file for test scenarios	&g_sascfg	example\bin\sasunit 9.2_32.windows.en.cfg
Folder for test data	&g_testdata	example\dat
Folder for reference data	&g_refdata	example\dat
Path to SASUnit macros	&g_sasunit	saspgm\sasunit
SAS log of reporting job		example\doc\sasunit\en\run_all.log
Platform	&SYSCPL	W32_VSPRO
SAS Version	&SYSVLONG4	9.02.02M3P04132010
User ID	&SYSUSERID	kirchmann
SASUnit Language	SASUNIT_LANGUAGE	en
Number of test scenarios		5
Number of test cases		41
Number of assertions		84

Generated on Friday, 21 February 2014, 22:02:45 by SASUnit  Version 1.2.1 (155)

The Tab „Test Scenarios“ shows an overview of the different tests, where you can find them and when they were executed.

Test Scenarios | SASUnit Examples - SASUnit Test Documentation

No.	Test Scenario	Program	Last Run	Duration	Result
001	Tests for boxplot.sas	example/saspgm/boxplot_test.sas	21FEB2014:22:02:21	9.2s	<input type="checkbox"/>
002	Tests for generate.sas	example/saspgm/generate_test.sas	21FEB2014:22:02:30	3.9s	<input checked="" type="checkbox"/>
003	Tests for getvars.sas	example/saspgm/getvars_test.sas	21FEB2014:22:02:34	1.6s	<input checked="" type="checkbox"/>
004	Tests for nobis.sas - has to fail!	example/saspgm/nobis_test.sas	21FEB2014:22:02:36	2.9s	<input checked="" type="checkbox"/>
005	Tests for regression.sas	example/saspgm/regression_test.sas	21FEB2014:22:02:39	3.3s	<input type="checkbox"/>

Generated on Friday, 21 February 2014, 22:02:44 by SASUnit ☒ Version 1.2.1 (155)

In SASUnit you see at a glance if your code is correct. The checkbox shows the results.



- test turned out correct



- test needs a manual check



- test result is not correct

When clicking on the failed scenario (No. 004) you can see that the test scenario contains 6 tests and one of them failed. This is also the tab "Test cases" which lists all test cases for each test scenario with detailed information and a link to the log file when clicking on the Last Run.

SASUnit Examples

- Scenario
- Units under Test

No. 004

Scenario Tests for nobis.sas - has to fail!

Program example/saspgm/nobis_test.sas

Last Run 21FEB2014:22:02:36

Duration 2.9s

Test Cases

No.	Test Case	Unit under Test	Last Run	Duration	Result
001	simple example with sashelp class	nobis.sas	21FEB2014:22:02:37	0.0s	<input checked="" type="checkbox"/>
002	failed test - must be red!	nobis.sas	21FEB2014:22:02:37	0.0s	<input checked="" type="checkbox"/>
003	example with big dataset	nobis.sas	21FEB2014:22:02:37	0.2s	<input checked="" type="checkbox"/>
004	example with empty dataset	nobis.sas	21FEB2014:22:02:38	0.3s	<input checked="" type="checkbox"/>
005	dataset not specified	nobis.sas	21FEB2014:22:02:38	0.0s	<input checked="" type="checkbox"/>
006	invalid dataset	nobis.sas	21FEB2014:22:02:38	0.0s	<input checked="" type="checkbox"/>

No. 005

Scenario Tests for regression.sas

Program example/saspgm/regression_test.sas

Last Run 21FEB2014:22:02:39


Duration 3.3s

Test Cases

No.	Test Case	Unit under Test	Last Run	Duration	Result
001	compare linear regression between Excel and SAS	regression.sas	21FEB2014:22:02:39	1.4s	<input type="checkbox"/>


You will get a detailed description of every executed assertion when you click on a test scenario and then on a test case. Here for example you find an overview of test scenario 1 and test case 3. As you can see it shows which assertion was executed, some details about the ran test and the expected and actual results.

Details for Test Case 001.003 SASUnit Examples - SASUnit Test Documentation					
Scenario No.	001				
Scenario	Tests for boxplot.sas				
Program	example/saspgm/boxplot_test.sas				
Last Run	21FEB2014 22:02:21				
Duration	9.2s				
Test Case No.	003				
Test Case	standard case with reference, missing values for Y				
Unit under Test	boxplot.sas				
Last Run	21FEB2014 22:02:23				
Assertions					
No.	Assertion	Description	Expected	Actual	Result
001	assertReport	please compare the two charts, no changes produced by missing values in the y variable	.pdf	.pdf	<input type="checkbox"/>
002	assertLogMsg	regular expression used to support different languages	Message NOTE: 240 observation(s) contained a MISSING value for the SBP 1 st Visit = Med requestNOTE: 240 Beobachtung(en) in fehlendem Wert enthalten für den Befehl SBP 1 st Visit = Med present	Message found	<input checked="" type="checkbox"/>
003	assertLog	scan log	Errors: 0, Warnings: 0	Errors: 0, Warnings: 0	<input checked="" type="checkbox"/>

Generated on Friday, 21 February 2014, 22:02:45 by SASUnit  Version 1.2.1 (155)

When clicking on the last tab you will find an overview of all units that were tested in this test scenario. There is a Column “Test Coverage” – this is only shown when using SAS9.3 or higher.

Units under Test SASUnit Examples - SASUnit Test Documentation					
Program Library example/saspgm					
Unit under Test	Test Scenario	Test Cases	Assertions	Test Coverage [%]	Result
boxplot.sas	001	22	45	97	<input type="checkbox"/>
generate.sas	002	6	9	100	<input checked="" type="checkbox"/>
getvars.sas	003	6	12	100	<input checked="" type="checkbox"/>
nobs.sas	004	8	12	100	<input checked="" type="checkbox"/>
regression.sas	005	1	6	100	<input type="checkbox"/>

Generated on Sunday, 2 March 2014, 11:03:20 by SASUnit  Version 1.2.1 (155)

Unit under Test: boxplot.sas

Test Coverage: 97%

Color Legend:

- Covered code
- Non covered code
- Comments
- Code marked as non contributing by option mcoverage

```

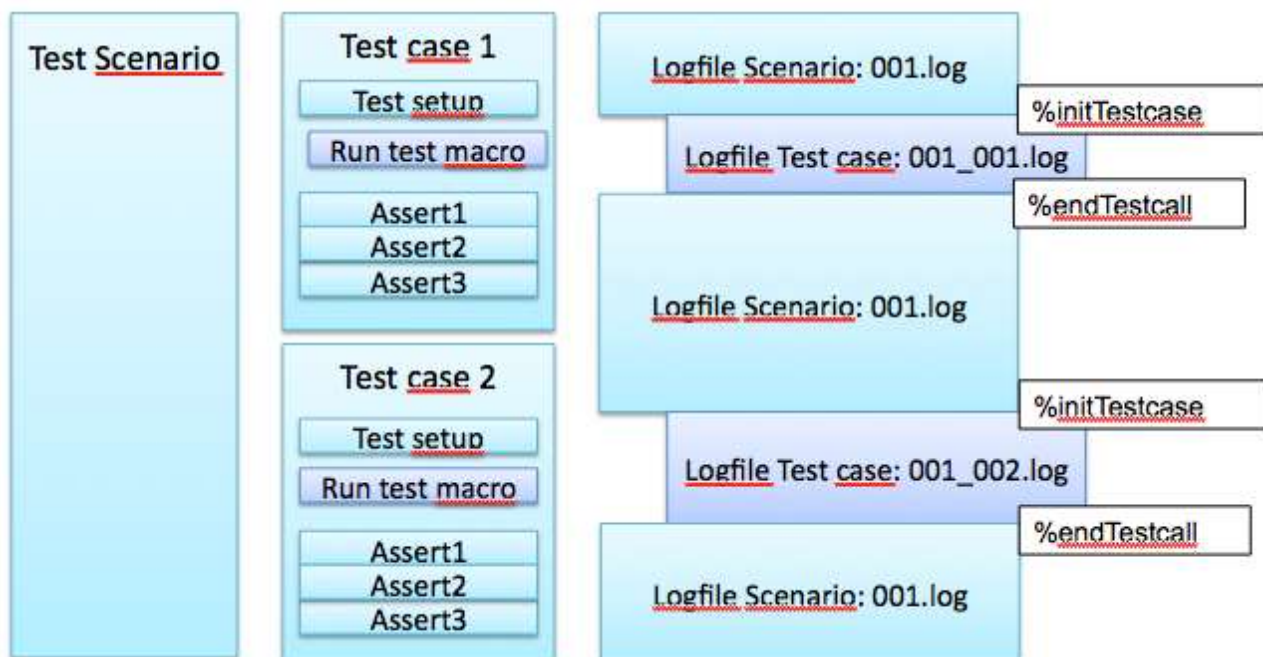
000110  /*-- check for number of groups and determine variable type and group sequence --*/
000111  %let grouptype=%sysfunc(vartype(&dsid,%sysfunc(varnum(&dsid,&group))));
000112  %local count lower;
000113  proc sql noprint;
000114      select count(distinct &group) into :count from &data;
000115      select min(&group) into :lower from &data;
000116  quit;
000117  %if &lower=. %then %do;
000118      %put ERROR: boxplot: Missing values in group variable are not allowed;
000119      %return;
000120  *** This is code not covered by any testcase and left intetionally here to ***;
000121  *** demonstrate the functionality of the test coverage ***;
000122  proc sql noprint; drop table &d_1; quit;
000123  %end;
000124  %if &count NE 2 %then %do;
000125      %put ERROR: boxplot: Variable &group must have exactly two values;
000126      %return;
000127  %end;

```

Logfiles

Each SASUnit Session generates its own logfiles and stores them in doc\sasunit\<language>\log. The log of controller and report are combined into one log which is called run_all.log and can be found directly in the sasunit\<language> folder.

For each test scenario there is one logfile per test case and a different logfile for the rest like the preliminary code and asserts.



SASUnit Macros

Controller Macros

- SASUnit
 - initSASUnit
 - runSASUnit
 - reportSASUnit
- Testscenarios
 - initTestcase
 - endTestcall
 - endTestcase

initSASUnit

Initialization of a test suite that may comprise several test scenarios.
An existing test repository is opened or a new test repository is created.

runSASUnit

Invokes one or more test scenarios.

Procedure:

- Check whether test repository was already initialized with %initSASUnit, if not: End.
- Determination of the test scenarios to be invoked.
- For every test scenario:
 - Check whether it already exists in the test repository.
 - if yes: Check whether the test scenario was changed since last invocation.
 - if no: Creation of the test scenario in the test repository.
 - In case the test scenario is new or changed:
The test scenario is executed in an own SAS session which is initialized by [_scenario.sas](#). All test results are gathered in the test repository.

reportSASUnit

Creation of a test report.

initTestcase

Start of a new test case that comprises an invocation of a program under test and one or more assertions.

internally:

- Insertion of relevant data into the test repository
- Redirection of SAS log
- Setting of flag g_inTestcase

endTestcall

Ends an invocation of a program under test.

internally:

- Ensure sequence
- End redirection of SAS log
- Reset ODS destinations

endTestcase

Ends a test case. Result and finish time are added to the test repository.

Asserts

assertColumns

Check whether there are differences between the values of the columns of two sas data sets (PROC COMPARE).

The values of the two data sets are considered to match each other if all columns existing in data set `i_expected` are also existing in data set `i_actual`. Optionally one can define a deviation for numerical values so that all corresponding values can be deviating from each other less than a maximal deviation of `i_fuzz` (Caution: this corresponds to the parameter 'criterion' of PROC COMPARE, the parameter 'fuzz' has a different meaning in the context of PROC COMPARE)

assertEquals

Check whether there are differences between the value of a macro variable and an expected value.

The values can be character string or numerical. Optionally one can define a deviation for numerical values so that the values can be deviating from each other less than a maximal deviation of `i_fuzz`.

assertForeignKey

Checks whether a foreign key relationship between the columns of two data sets exists. This assert supports simple and composite keys. The number of specified columns in parameters `i_mstKey` and `i_lookupKey` must be the same and columns have to be in the same order. If more than one column is specified the parameter `i_cmpKeyLen` has to be provided containing the number of columns. Eventually needed renaming of key variables takes place automatically.

assertLibrary

Check whether all files are identical in the libraries `i_expected` and `i_actual`.

The comparison report is created later, as PROC REPORT does not support ODS Document.

assertLog

Check whether errors or warnings appear in the log.

If number of errors and warnings does not appear in the log as expected, the check of the assertion will fail.

assertLogMsg

Check whether a certain message appears in the log.

If the message does not appear in the log as expected, the check of the assertion will fail.

If `i_not` is set to 1, the check of the assertion will fail in case the message is found in the log.

assertPerformance

Check whether runtime of the testcase is below or equal a given limit.

assertPrimaryKey

Checks whether a set of columns can be used as a primary key for a dataset.

assertRecordCount

This assert checks whether a certain number of records exist in a data set specified by parameters `i_libref` and `i_memname`.

Furthermore a where condition can be specified (if not specified set to 1) as well as the number of expected records in the data set that meet the given where condition.

assertRecordExists

Check whether at least one record exists which satisfies a certain WHERE condition.

assertReport

Check whether a report file exists and was created during the current SAS session.

It is possible to write an instruction into the test protocol indicating the need to perform a manual check of the report. Writes an entry into the test repository indicating the need to perform a manual check of the report and copies the report and a given report template (optional).

assertRowExpression

Checks if all observations meet a given WHERE expression.

assertTableExists

Check whether a certain data set, view or catalogue exists.

Setting the optional parameter `i_not` to 1 allows to test whether a certain data set, view or catalogue does not exist.

Step 1: Check whether library has been assigned successfully.

Step 2: Check for existence with `exist` function

Getting started

Write a simple test

Add two new SAS® files to the folder `sasunit\example\saspgm`.

Call them for our example `getFirstNobs_test.sas` and `getFirstNobs.sas`. The test will check if your SAS® program writes the first 5 observations of a SAS® table to a new table.

Open the test file *getFirstNobs_test.sas* in your preferred SAS® development environment and write the following code into it:

Paste the following code to the *getFirstNobs.sas* file

```
%initTestCase (i_object = getFirstNobs.sas
                , i_desc = %STR(My first Testscenario))

DATA work.expected;
    SET SASHELP.class (obs = 5);
RUN;

%getFirstNobs (i_input = SASHELP.class
                , i_output = work.output)

%endTestcall()

%assertColumns ( i_actual      = work.output
                  , i_expected  = work.expected
                  , i_desc      = compare estimated values)

%assertLog()
%endTestcase
```

```
%Macro getFirstNobs ( i_input =
                     , i_output = );

    DATA &i_output.;
        SET &i_input. (obs = 5);
    RUN;

%Mend getFirstNobs;
```

Execute the test

After you have written and saved your macro and test code start the batch file again. SASUnit now checks for all the test files in the `saspgm` folder and executes new and changed tests.

Look at the results

So now you can refresh your browser showing the SASUnit Test Documentation and will find the new generated test in the tab "test scenarios". Here you can see that No. 006 is our newly written SASUnit test and that it works correctly.

Main Page

Test Scenarios

Test Cases

Units under Test

Test Scenarios | SASUnit Examples - SASUnit Test Documentation

No.	Test Scenario	Program	Last Run	Duration	Result
001	Tests for boxplot.sas	example/saspgm/boxplot_test.sas	21FEB2014:22:02:21	9.2s	<input type="checkbox"/>
002	Tests for generate.sas	example/saspgm/generate_test.sas	21FEB2014:22:02:30	3.9s	<input checked="" type="checkbox"/>
003	Tests for getvars.sas	example/saspgm/getvars_test.sas	21FEB2014:22:02:34	1.6s	<input checked="" type="checkbox"/>
004	Tests for nob.sas - has to fail!	example/saspgm/nobs_test.sas	21FEB2014:22:02:36	2.9s	<input checked="" type="checkbox"/>
005	Tests for regression.sas	example/saspgm/regression_test.sas	21FEB2014:22:02:39	3.3s	<input type="checkbox"/>
006		example/saspgm/getFirstNobs_test.sas	22FEB2014:22:58:07	1.2s	<input checked="" type="checkbox"/>

Generated on Saturday, 22 February 2014, 22:58:09 by **SASUnit** ☒ Version 1.2.1 (155)

When you click on No. 006 you will see more details for this test case and that each assertion succeeded.

Main Page

Test Scenarios

Test Cases

Units under Test

Details for Test Case 006.001 | SASUnit Examples - SASUnit Test Documentation

Scenario No. 006

Scenario

Program example/saspgm/getFirstNobs_test.sas

Last Run 22FEB2014:22:58:07

Duration 1.2s

Test Case No. 001

Test Case My first Testscenario

Unit under Test getFirstNobs.sas

Last Run 22FEB2014:22:58:08

Assertions

No.	Assertion	Description	Expected	Actual	Result
001	assertColumns	compare estimated values	Table listing DSLABEL LABEL COMPVAR	Table listing Comparison	
002	assertLog	scan log	Errors: 0, Warnings: 0	Errors: 0, Warnings: 0	

Generated on Saturday, 22 February 2014, 22:58:09 by **SASUnit** ☒ Version 1.2.1 (155)


```

/**
\brief Tests for getFirstNobs.sas

*/ /** \cond */
%Macro getFirstNobs ( i_input =
                    , i_output = );

    DATA &i_output.;
        SET &i_input. (obs = 5);
    RUN;

%Mend getFirstNobs;

```

When looking at the overview of the test scenarios you can see that the description of our test scenario is missing. For this description we need a DoxyGen tag in our macro getFirstNobs.sas So we need to add the following code at the beginnig of our macro

After adding the DoxyGen tag please save your SAS code and execute the batch file again. The overview of the test scenario should look like this now







Main Page


Test Scenarios

Test Cases

Units under Test

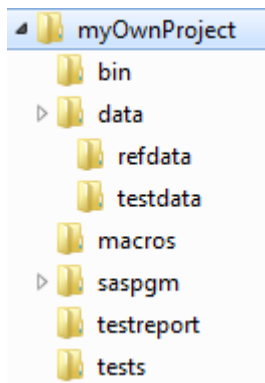
Test Scenarios | SASUnit Examples - SASUnit Test Documentation

No.	Test Scenario	Program	Last Run	Duration	Result
001	Tests for boxplot.sas	example/saspgm /boxplot_test.sas	21FEB2014:22:02:21	9.2s	
002	Tests for generate.sas	example/saspgm /generate_test.sas	21FEB2014:22:02:30	3.9s	
003	Tests for getvars.sas	example/saspgm /getvars_test.sas	21FEB2014:22:02:34	1.6s	
004	Tests for nob.sas - has to fail!	example/saspgm/nobs_test.sas	21FEB2014:22:02:36	2.9s	
005	Tests for regression.sas	example/saspgm /regression_test.sas	21FEB2014:22:02:39	3.3s	
006	Tests for getFirstNobs.sas	example/saspgm /getFirstNobs_test.sas	22FEB2014:23:00:40	1.2s	

Generated on Saturday, 22 February 2014, 23:00:42 by **SASUnit**  Version 1.2.1 (155)

Creating your own project

1. Create a folder structure that includes the following folder:
 - bin
 - a folder for your data, maybe with subfolders for test and reference data
 - a folder for your macro code
 - a folder for your tests
 - a folder for the documentation



2. Copy all script files to your bin folder
 - Adjust SASUNIT_ROOT in the script file
 - Adjust installation paths in script and config file
3. Copy or generate macros to your macros folder
4. Copy the file run_all.sas from example\saspgm to your tests folder
5. Adjust autocall paths in the run_all file