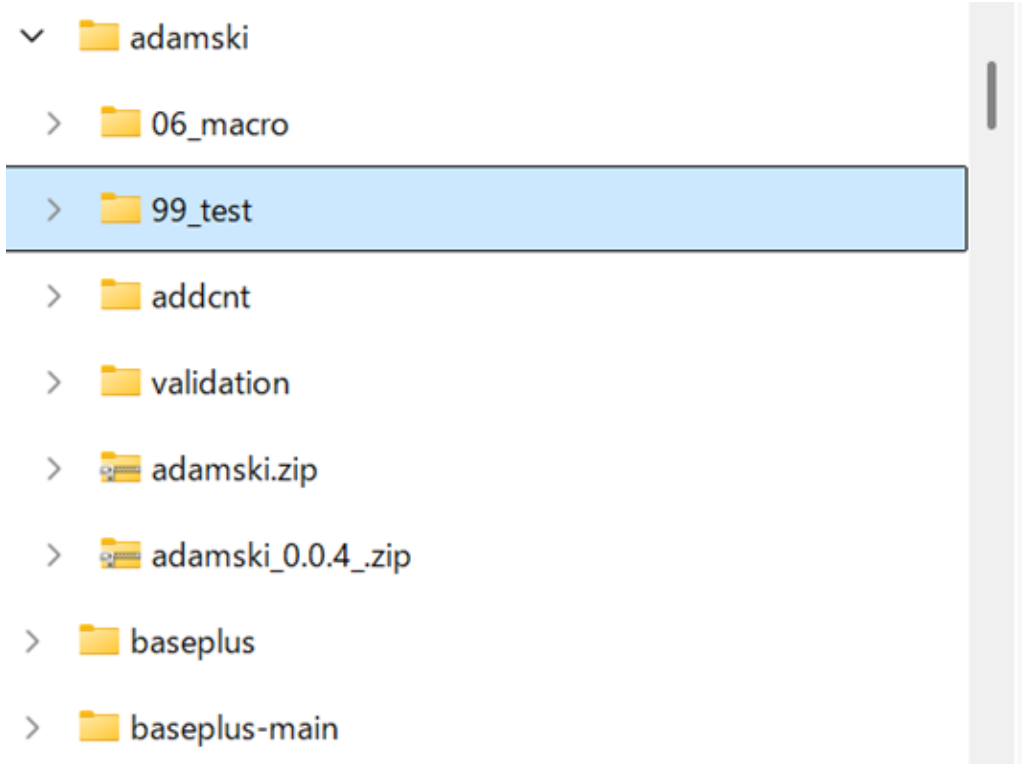
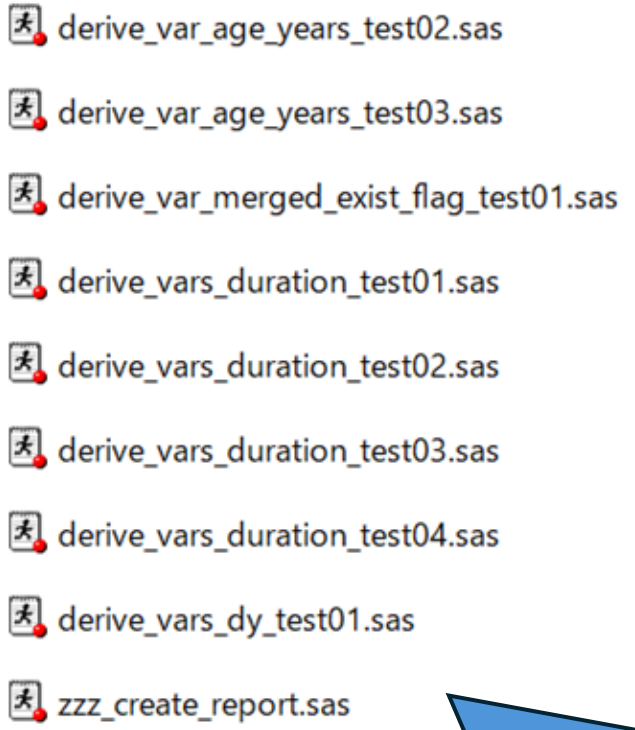
How to use Valivali

- Overall test flow in %generatePackage()
- How to write test script
- How to create report

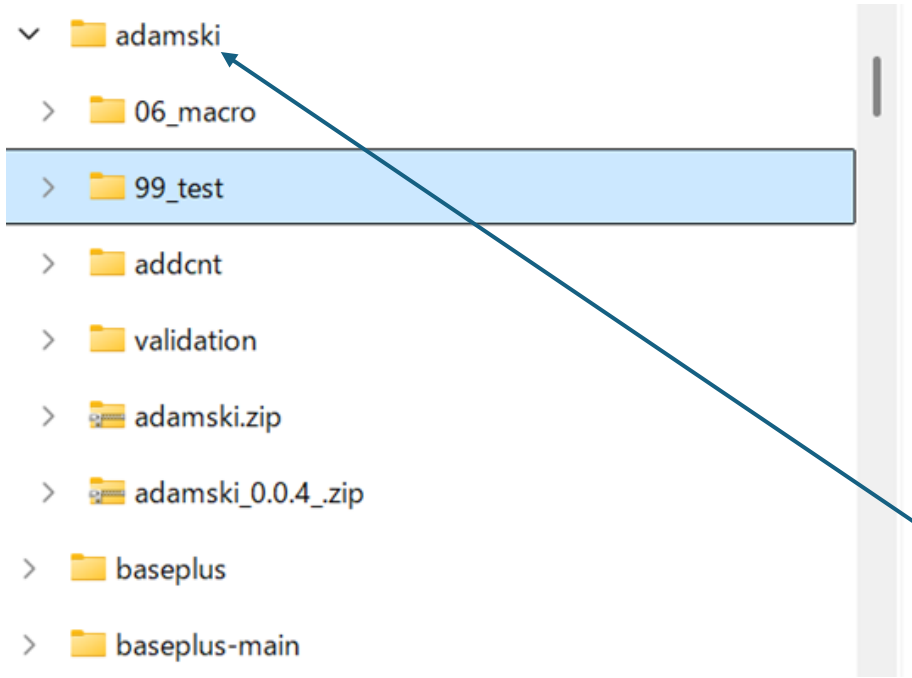
Test flow in SPF (done in %generatePackage)

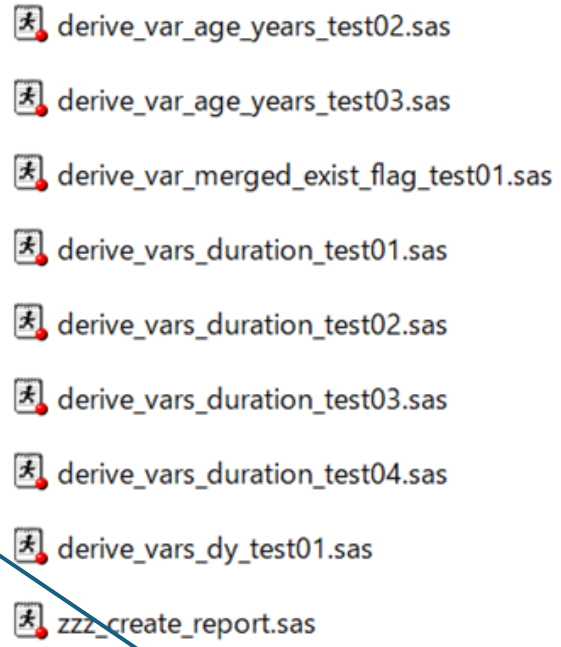




Test scripts(.sas) placed in test folder(e.g. 99_test) will be executed when you generate package using %generatePackage().

Test flow in SPF

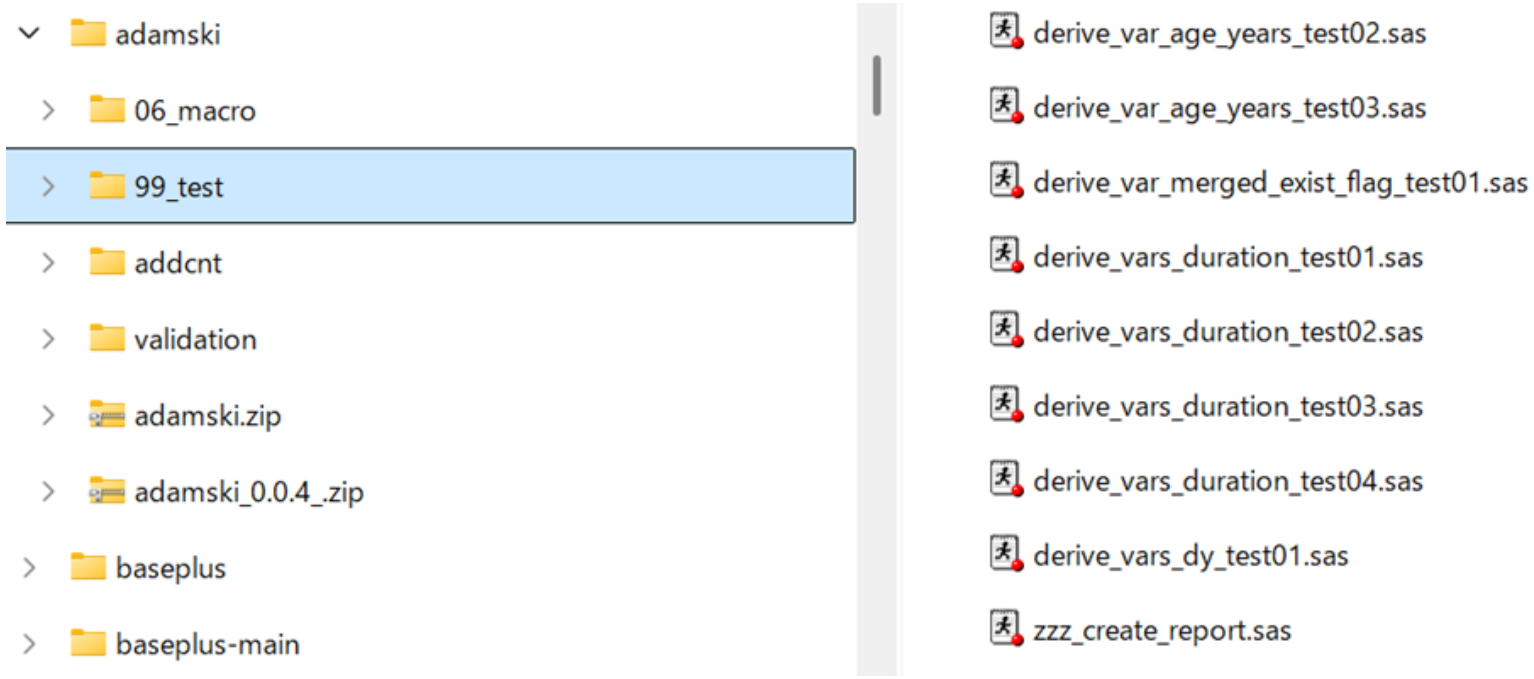




Location of source package folder

```
%generatePackage(
    filesLocation=C:¥Temp¥SAS_PACKAGES¥packages¥adamski ,
    testPackage=Y,
    testResults=C:¥Temp¥SAS_PACKAGES¥packages¥adamski,
    delTestWork=0,
    markdownDoc=1 , easyArch=1 ,
    workInTestResults=1
)
```

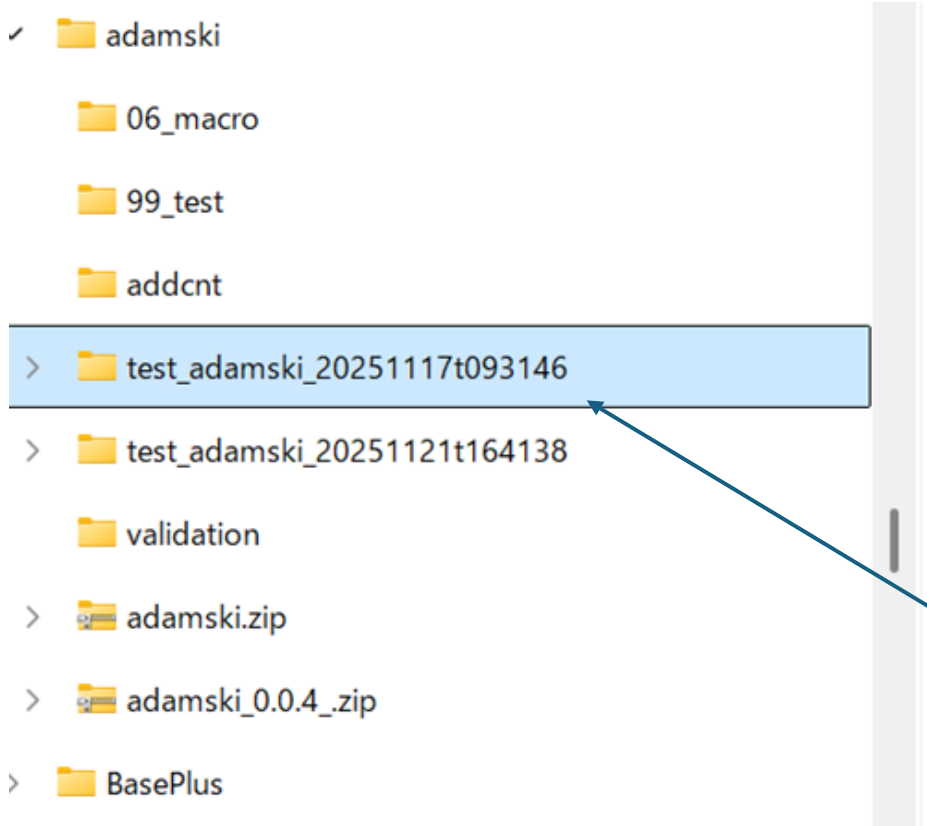
Test flow in SPF

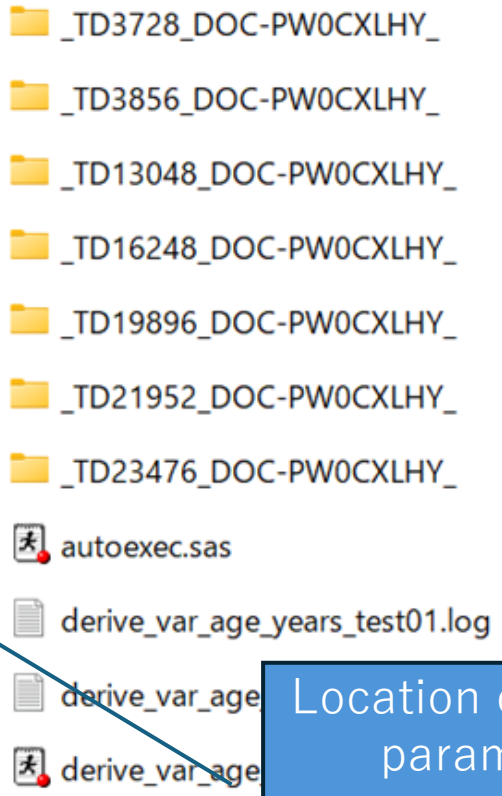



```
%generatePackage(
    filesLocation=C:¥T
    testPackage=Y,
    testResults=C:¥Temp¥SAS_PACKAGES¥packages¥adamski,
    delTestWork=0,
    markdownDoc=1 , easyArch=1 ,
    workInTestResults=1
)
```

Y to execute test scripts, otherwise tests will not run.

Test flow in SPF






```
%generatePackage(
    filesLocation=C:¥Temp¥SAS_PACKAGES¥packages¥adamski ,
    testPackage=Y,
    testResults=C:¥Temp¥SAS_PACKAGES¥packages¥adamski,
    delTestWork=0,
    markdownDoc=1 , easyArch=1 ,
    workInTestResults=1
)
```

Location of test results created. If testResults parameter is not set, results will be in temporary work directory.

By combination of delTestWork=0(do not delete work in each test session) and workInTestResults=1(work folder created in folder specified in testResults=)

How to write test scripts

CKAGES > packages > adamski > 99_test　　99_testの検索

並べ替え　表示　…

名前	更新日時	種類
derive_var_age_years_test01.sas	2025/11/08 13:24	SAS System Program
derive_var_age_years_test02.sas	2025/11/08 13:24	SAS System Program
derive_var_age_years_test03.sas	2025/11/08 13:25	SAS System Program
derive_var_merged_exist_flag_test01.sas	2025/11/08 13:27	SAS System Program
derive_vars_dy_test01.sas	2025/11/08 13:27	SAS System Program
zzz_create_report.sas	2025/11/08 14:46	SAS System Program

```
%loadPackage(valivali)
%set_tmp_lib(lib=TEMP, winpath=C:¥Temp, otherpath=/tmp, newfolder=adamski)

test...

/*Compare*/
%mp_assertdataset(
  base        = e_data,                /* parameter in proc compare */
  compare     = o_data,                /* parameter in proc compare */
  desc        = (%nrstr(%derive_var_age_years))[test01] Compare expected and test
  id =,                                /* parameter in proc compare(e.g. id=USUBJID) */
  by =,                                /* parameter in proc compare(e.g. by=USUBJID VISIT) */
  criterion   = 1e-8,                  /* parameter in proc compare */
  method      = absolute,              /* parameter in proc compare */
  outds       = TEMP.adamski_test
);
```

1. Load {valivali}
2. Set up libname (can use valivali)
3. Create your test contents
   (e.g. expected data and test data)
4. Use assert macros in valivali

Set up a consistent lib reference across all test scripts is key to integrate results into a dataset.

This is because test scripts will be executed in separate sub-sessions in %generatePackage().

How to write test scripts

CKAGES > packages > adamski > 99_test　　99_testの検索

並べ替え　表示　…

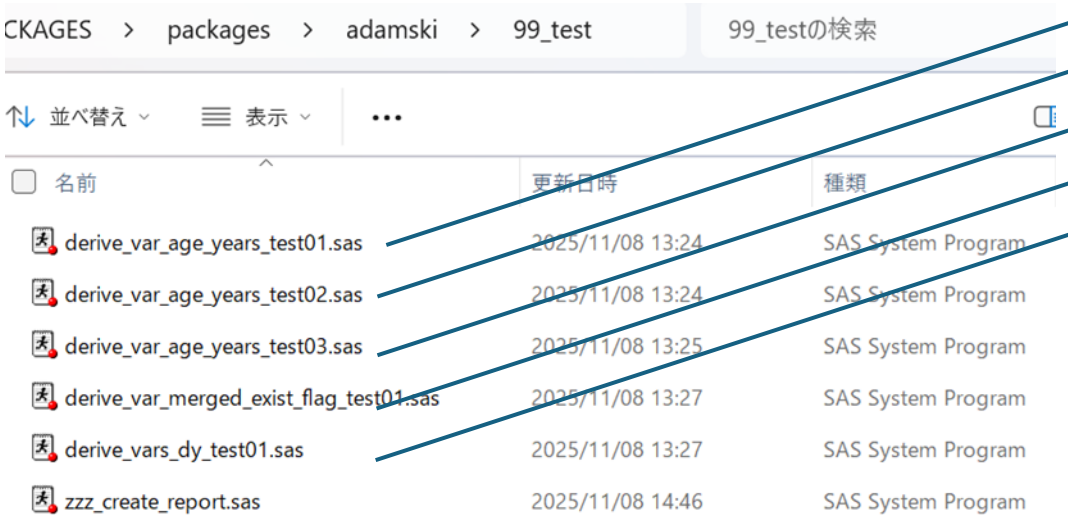
名前	更新日時	種類
derive_var_age_years_test01.sas	2025/11/08 13:24	SAS System Program
derive_var_age_years_test02.sas	2025/11/08 13:24	SAS System Program
derive_var_age_years_test03.sas	2025/11/08 13:25	SAS System Program
derive_var_merged_exist_flag_test01.sas	2025/11/08 13:27	SAS System Program
derive_vars_dy_test01.sas	2025/11/08 13:27	SAS System Program
zzz_create_report.sas	2025/11/08 14:46	SAS System Program

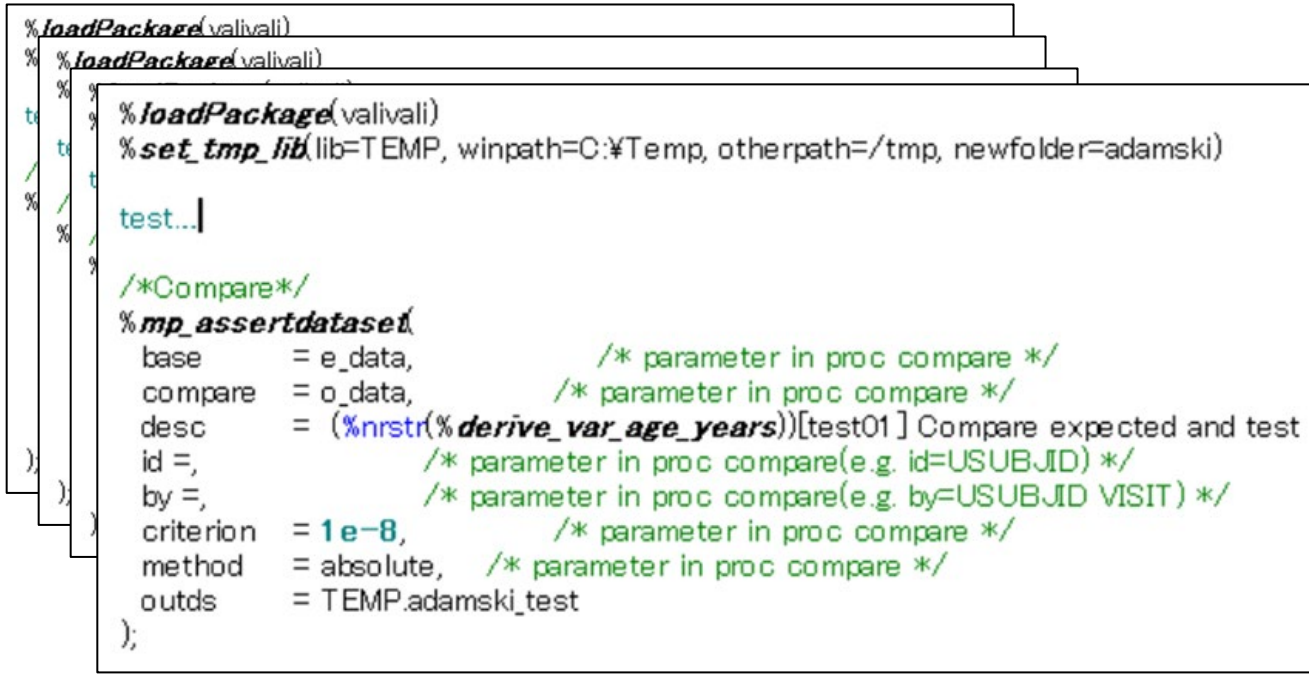
```
%loadPackage(valivali)
%set_tmp_lib(lib=TEMP, winpath=C:¥Temp, otherpath=/tmp, newfolder=adamski)

test...

/*Compare*/
%mp_assertdataset(
  base       = e_data,              /* parameter in proc compare */
  compare    = o_data,              /* parameter in proc compare */
  desc       = (%nrstr(%derive_var_age_years))[test01] Compare expected and test
  id =,                             /* parameter in proc compare(e.g. id=USUBJID) */
  by =,                             /* parameter in proc compare(e.g. by=USUBJID VISIT) */
  criterion  = 1e-8,                /* parameter in proc compare */
  method     = absolute,            /* parameter in proc compare */
  outds      = TEMP.adamski_test
);
```

1. Load {valivali}
2. Set up libname (can use valivali)
3. Create your test contents
   (e.g. expected data and test data)
4. Use assert macros in valivali

%set_tmp_lib is a very small macro to create libname for Windows/Linux.
You may or may not use.

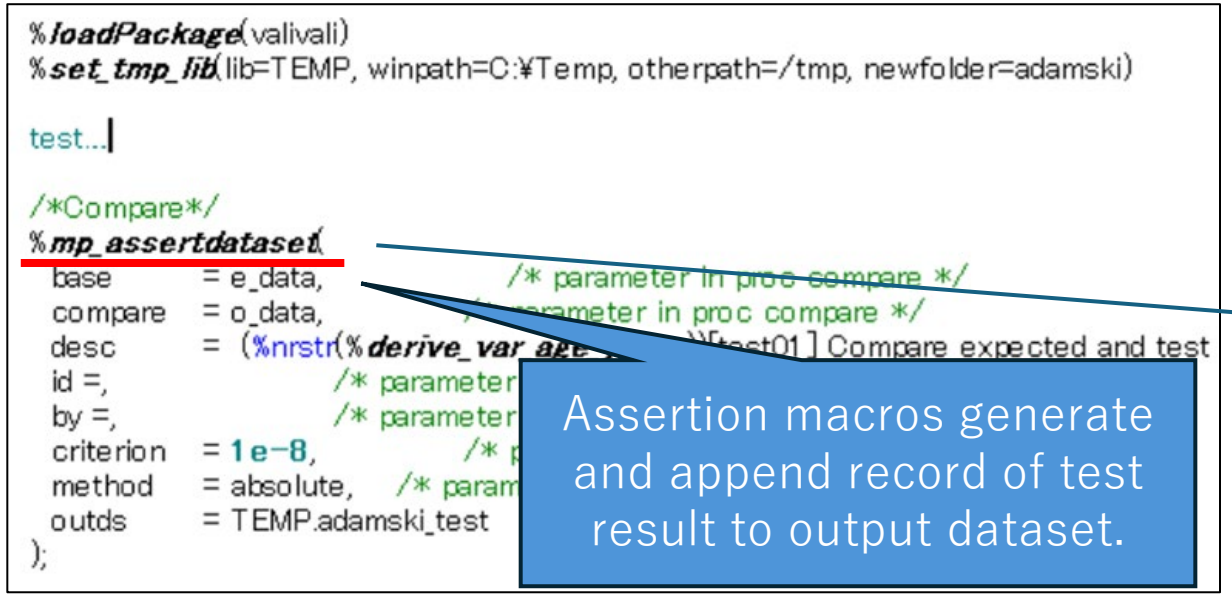
How to write test scripts





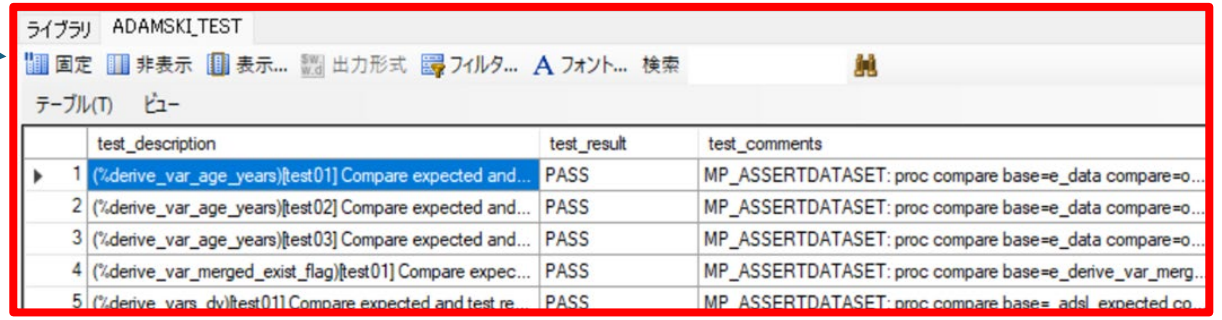
```
%loadPackage(valivali)
%set_tmp_lib(lib=TEMP, winpath=C:¥Temp, otherpath=/tmp, newfolder=adamski)

test...

/*Compare*/
%mp_assertdataset(
  base       = e_data,              /* parameter in proc compare */
  compare    = o_data,              /* parameter in proc compare */
  desc       = (%nrstr(%derive_var_age_years))[test01] Compare expected and test
  id =,                             /* parameter in proc compare(e.g. id=USUBJID) */
  by =,                             /* parameter in proc compare(e.g. by=USUBJID VISIT) */
  criterion  = 1e-8,                /* parameter in proc compare */
  method     = absolute,            /* parameter in proc compare */
  outds      = TEMP.adamski_test
);
```

1. Load {valivali}
2. Set up libname (can use valivali)
3. Create your test contents
   (e.g. expected data and test data)
4. Use assert macros in valivali



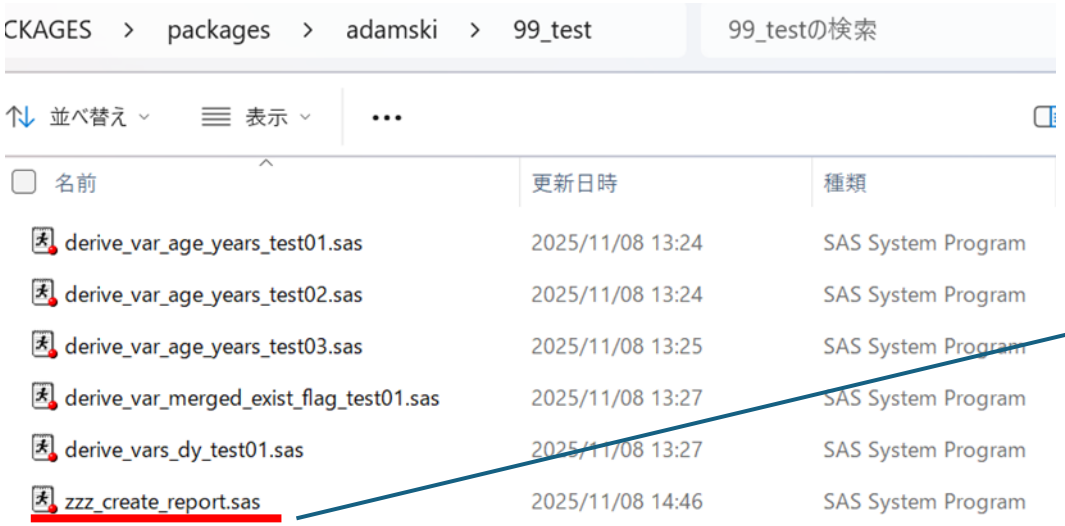
```
%loadPackage(valivali)
%set_tmp_lib(lib=TEMP, winpath=C:¥Temp, otherpath=/tmp, newfolder=adamski)

test...

/*Compare*/
%mp_assertdataset(
  base       = e_data,              /* parameter in proc compare */
  compare    = o_data,              /* parameter in proc compare */
  desc       = (%nrstr(%derive_var_age_years))[test01] Compare expected and test
  id =,                             /* parameter
  by =,                             /* parameter
  criterion  = 1e-8,                /* p
  method     = absolute,            /* param
  outds      = TEMP.adamski_test
);
```

Assertion macros generate and append record of test result to output dataset.



	test_description	test_result	test_comments
1	(%derive_var_age_years)[test01] Compare expected and...	PASS	MP_ASSERTDATASET: proc compare base=e_data compare=o...
2	(%derive_var_age_years)[test02] Compare expected and...	PASS	MP_ASSERTDATASET: proc compare base=e_data compare=o...
3	(%derive_var_age_years)[test03] Compare expected and...	PASS	MP_ASSERTDATASET: proc compare base=e_data compare=o...
4	(%derive_var_merged_exist_flag)[test01] Compare expec...	PASS	MP_ASSERTDATASET: proc compare base=e_derive_var_merg...
5	(%derive_vars_dy)[test01] Compare expected and test re...	PASS	MP_ASSERTDATASET: proc compare base=_adsl_expected co...

How to create report



Create report at last.
(zzz is to run it lastly)

```
%loadPackage(valivali)
%set_tmp_lib(lib=TEMP, winpath=C:¥Temp, otherpath=/tmp, newfolder=adamski)

/*Create report*/
%create_report(
  sourcelocation = C:¥Temp¥SAS_PACKAGES¥packages¥adamski,  /* for package information */
  reporter = Ryo Nakaya,

  general = %nrstr(
Adamski is blah blah
  ),  /* for general description of package */

  requirements = %nrstr(
- %derive_vars_dy :  ^{newline}
  Generates study day (DY) variables from given date variables using a specified reference date, fo
- %derive_var_merged_exist_flag :  ^{newline}
  Creates a character flag variable indicating whether the current DATA step row`s keys exist in a
  ),
  results = TEMP.adamski_test, /* validation results dataset */
  additional = %nrstr(
    NA
  ),  /* Any additional information */
  references = %nrstr(
    https://github.com/PharmaForest/adamski ^n
  ),  /* reference information */
  outfilelocation = C:¥Temp¥SAS_PACKAGES¥packages¥adamski¥validation  /* location for output
);
```

1. Load {valivali}
2. Set up libname (can use valivali)
3. Use %create_report in valivali

Validation Report

Adamski (Version 0.0.4)

Automatically extracted from session info and package source file(description.sas)

Ryo Nakaya

17NOV2025

General Information

Adamski is a SAS package inspired by the R package {admiral}. It aims to bring the same flexible and modular ADaM derivation framework to the SAS environment. The package follows the {admiral} design principles while adapting to SAS syntax and workflows. It enables consistent, reproducible ADaM dataset creation in compliance with CDISC standards. Adamski serves as a bridge between open-source R implementations and traditional SAS programming.

Validation Environment

OS: WIN
SAS: 9.4
Required Packages: -
Execution Datetime: 17NOV2025:09:35:00

Authors

[Yutaka Morioka],[Hiroki Yamanobe],[Ryo Nakaya],[Sharad Shhetri]

Requirements

- %derive_vars_dy :
Generates study day (DY) variables from given date variables using a specified reference date, for example TRTSDT.
- %derive_var_merged_exist_flag :
Creates a character flag variable indicating whether the current DATA step row`s keys exist in another dataset.
- %derive_var_age_years :
Converts a set of age values from the specified time unit to years.
- %derive_vars_duration :
Derives duration between two dates, specified by the variables present in the input dataset, for example duration of adverse events, relative day, age, etc..

Validation Records

Fixed

Bring from result dataset

Test Description	Result	Comments
(%derive_var_age_years)[test01] Compare expected and test results for case of age_unit=variable name	PASS	MP_ASSERTDATASET: proc compare base=e_data compare=o_data
(%derive_var_age_years)[test02] Compare expected and test results for case of age_unit=character literal	PASS	MP_ASSERTDATASET: proc compare base=e_data compare=o_data
(%derive_var_age_years)[test03] Compare expected and test results for case of digits=2	PASS	MP_ASSERTDATASET: proc compare base=e_data compare=o_data
(%derive_var_merged_exist_flag)[test01] Compare expected and test results	PASS	MP_ASSERTDATASET: proc compare base=e_derive_var_merged_exist_flag compare=o_derive_var_merged_exist_flag
(%derive_vars_dy)[test01] Compare expected and test results	PASS	MP_ASSERTDATASET: proc compare base=_adsl_expected compare=_adsl_test

Additional comments

NA

References

https://github.com/PharmaForest/adamski
https://github.com/PharmaForest/adamski/blob/main/Adamski_and_Admiral.md

Manually describe in macro parameters

Graph

test_description	test_result	test_comments	gpath1	gpath2	test_target	test_ID
<Dummy descripti...	PASS	<Dummy comme...				
<Dummy descripti...	FAIL	<Dummy comme...				
<Dummy descripti...	CHECK	<Dummy comme...	C:¥Temp¥test¥expected¥testplot1.png	C:¥Temp¥test¥output¥testplot2.png	<macroname>	test01

- %mp_assertgraph() creates record with test_result="CHECK" and paths for previous and current graphs

- Graph should be created in test contents part before running %mp_assertgraph.

- %create_report() creates Appendix with graphs placed based on gpath1/gpath2 with test_target and test_ID.
(e.g. test_target=%swimmer_plot, test_ID=test01)

- test_target and test_ID to be created automatically if test_description starts from (xxx) [yyy].
(xxx=test_target, yyy=test_ID)

- Users should visually review graph outputs

Validation Records

Test Description	Result	Comments
<Dummy description. Check macro variable resolution>	PASS	<Dummy comments. Macro resolved correctly without warning messages.>
<Dummy description. Validate date format conversion>	FAIL	<Dummy comments. Conversion failed when input date was missing.>
<Dummy description. Validate date format conversion>	CHECK	<Dummy comments. Check appendix.>

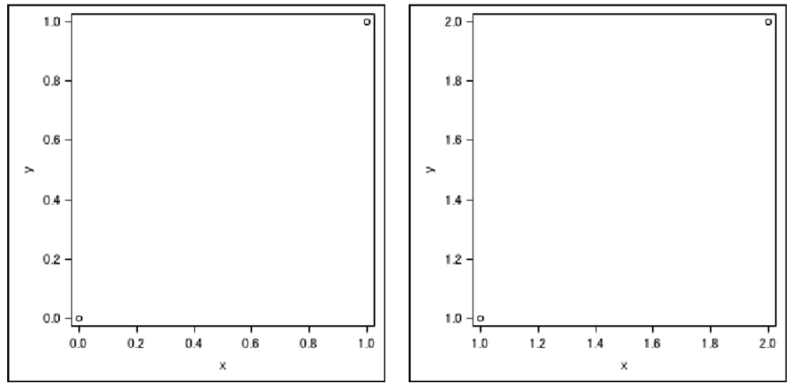
Additional comments

No additional comments.

References

https://company.example/validation
Document reference

Appendix



(<macroname>) [test01] Previous (<macroname>) [test01] Current