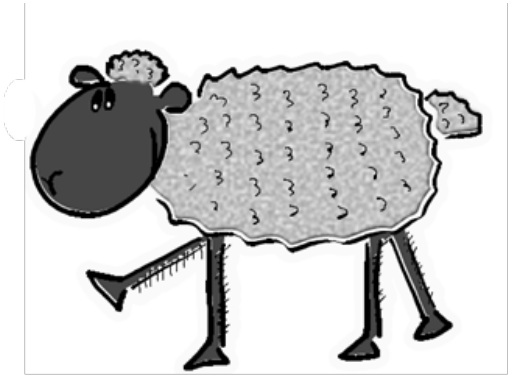
How to Create SAS packages

Ryo Nakaya



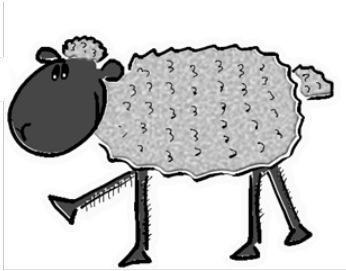
What is SAS Packages?

- SAS macros, functions, data, etc. are packaged in a unified form using SAS Packages Framework (SPF).
  https://github.com/yabwon/SAS_PACKAGES
- Package can be created and shared like R
- SASPAC (SAS Packages Archive) Repository
  https://github.com/SASPAC

SAS packages were developed by Bartosz Jablonski * in 2019 and the number of users has been increasing in recent years

*Takeda pharmaceutical, Warsaw University of Technology, yabwon

How to Create SAS Packages



developer

1. Loading SPF (Setup)
   - > next slide(3 lines only!)

1.5. Create a hex logo with ChatGPT, etc.!!

2. Creating Source Files and Folders for a Package
   - > To be explained in the following slides

3. Execute the% generatePackage () macro

This will result in the creation of a [packagename] .zip file (package file)

```
%generatePackage(
        filesLocation = C:¥your¥packages¥folder, /* location of the files/folders created above */
        markdownDoc = 1,                         /* Create .md file*/
        easyArch = 1                             /* create archive files of .zip and .md */
)
```

There are other parameters related to tests, etc. at the time of package preparation.

SPF Setup

```
filename packages "¥Your¥Folder¥";
filename SPFinit url "https://raw.githubusercontent.com/yabwon/SAS_PACKAGES/main/SPF/SPFinit.sas";
%include SPFinit;
%installPackage(SPFinit)
```

Once SPFinit.sas is downloaded to the location of packages using the above% installPackage (SPFinit), the following is OK

```
filename packages "¥Your¥Folder¥";
%include packages(SPFinit);
```

https://github.com/yabwon/SAS_PACKAGES/blob/main/SPF/Documentation/HelloWorldPackage.md

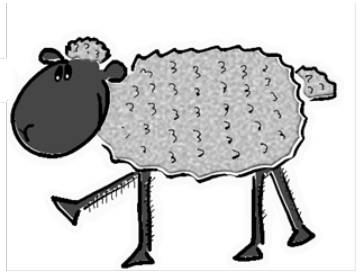
Installation and Loading Examples

```
%installPackage(baseplus)
%installPackage(SASPACer, sourcePath = https://github.com/Nakaya-Ryo/SASPACer/raw/main/)
%installPackage(RWDExpress(0.0.1), github = Narusawa-T)

%loadPackage(baseplus)
%loadPackage(SASPACer)
```

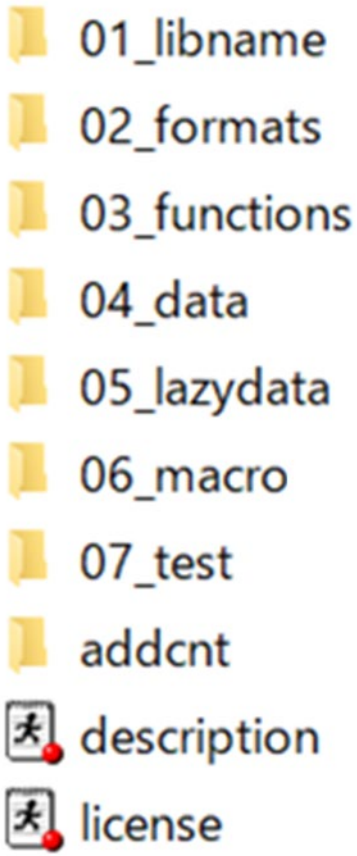
%loadPackage () looks like library () in R.

Source file folder for package

- Standard Configuration



developer

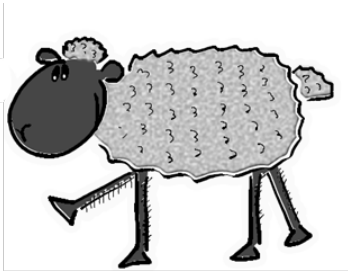


- Sequential number (for order of execution at the time of package creation)
- Object type (e.g. macro) at end
- Test is the test code at the time of package creation
- Only numbers, “_ ” and lower case letters
- Any files other than .sas files are ignored (except for addcnt).
- Addcnt (Additional contents) will be included in the package
- “description.sas ” should be an essential file, and“ license.sas ” should be created (to be explained on a later page)

https://github.com/yabwon/SAS_PACKAGES/blob/628641a6cfb6c5cf900283e9db450c7fe0e386e9/SPF/Documentation/Paper_1079-2021/My%20First%20SAS%20Package%20-%20a%20How%20To.pdf

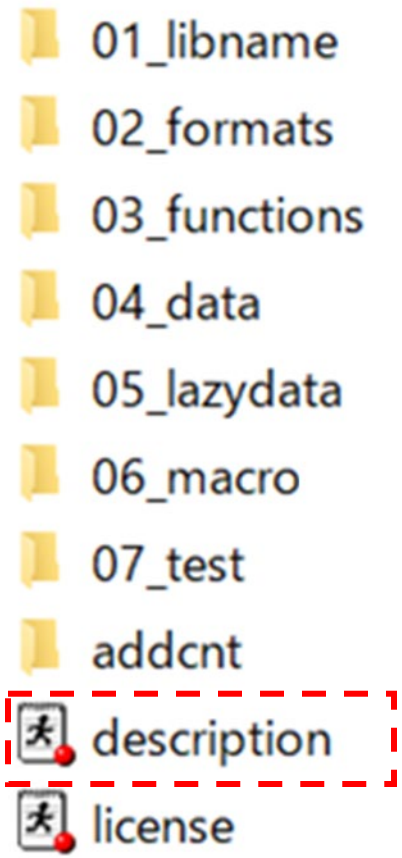
https://github.com/yabwon/SAS_PACKAGES/blob/main/SPF/Documentation/HelloWorldPackage.md

Source file folder for package



developer

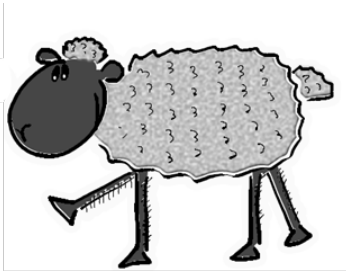
- Examples of sas file content in each folder




```
Type : Package
Package : testPackage
Title : Test package
Version : 0.0.1
Author : Taro Sasu(taro.sasu@mail.com)
Maintainer : Taro Sasu(taro.sasu@mail.com)
License : MIT
Encoding : UTF8
Required : "Base SAS Software"
ReqPackages : "List any SAS packages required here"

DESCRIPTION START:

## The testPackage ##
The test package is to show how the package is created
You can describe package information here

DESCRIPTION END:
```

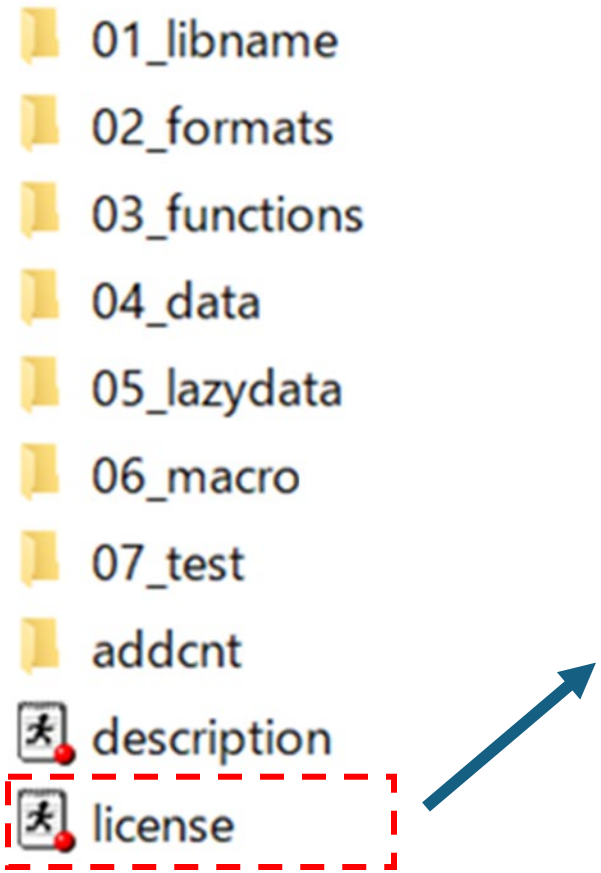
Fill in the package information in this format such as type

Source file folder for package



developer

- Examples of sas file content in each folder



Copyright (c) [YEAR] [Owner Name]

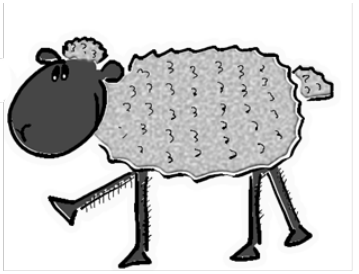
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

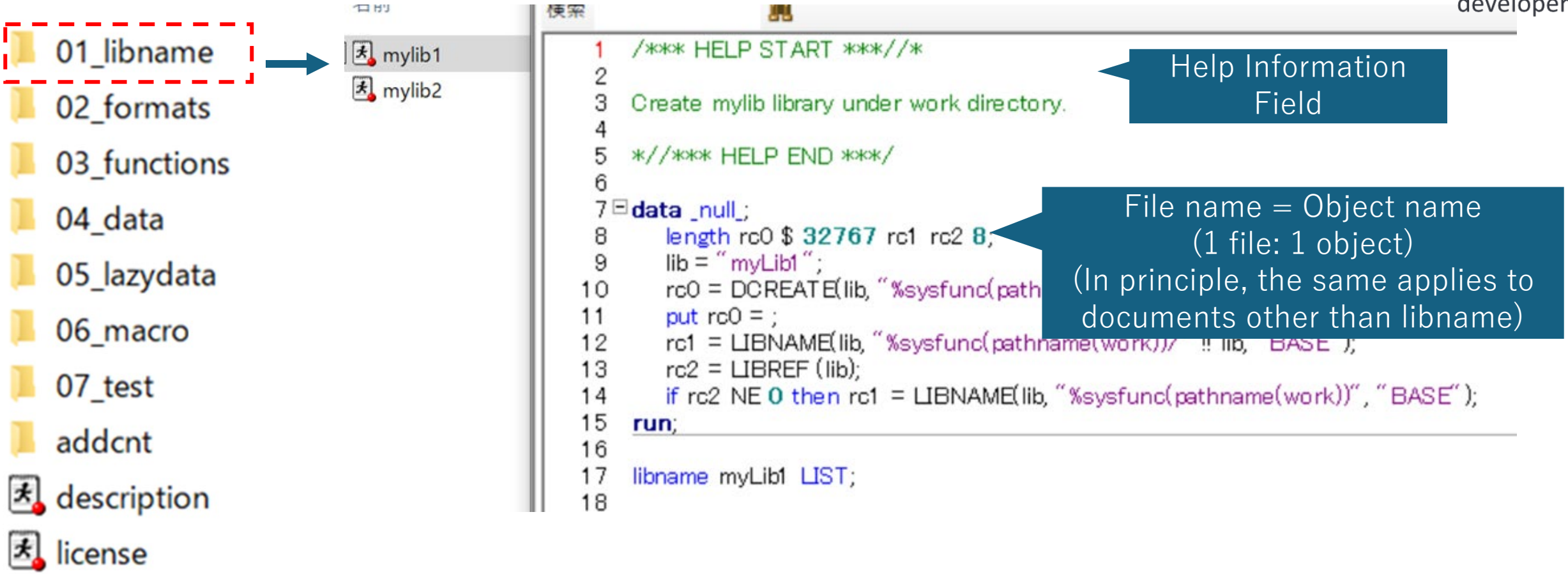
Standard text in the MIT License (if liscense.sas file is not present, the MIT standard text is prepopulated.)

Source file folder for package

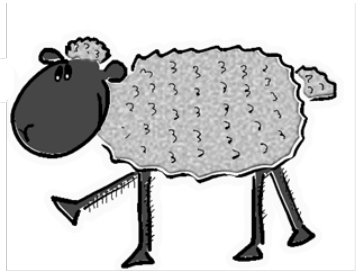


developer

- Examples of sas file content in each folder

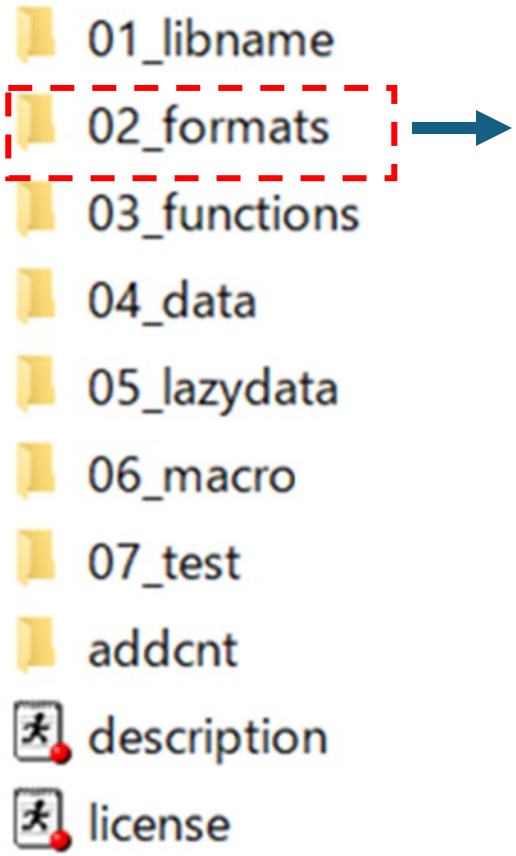


Source file folder for package



developer

- Examples of sas file content in each folder




```
1  /*** HELP START ***//*
2
3  This is a format.
4  Proc format ; and run ; are not needed within SASPAC framework.
5
6  *//*** HELP END ***/
7
8      value fmtNum
9          low -< 0 = "negative"
10         0 = "zero"
11         0 <- high = "positive"
12         other = "missing"
13     ;
```

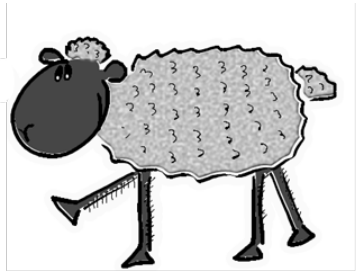
Help Information Field

It is impossible to describe Proc format; and run; (Describe only the contents)

Formats does not need to be 1 file: 1 object.

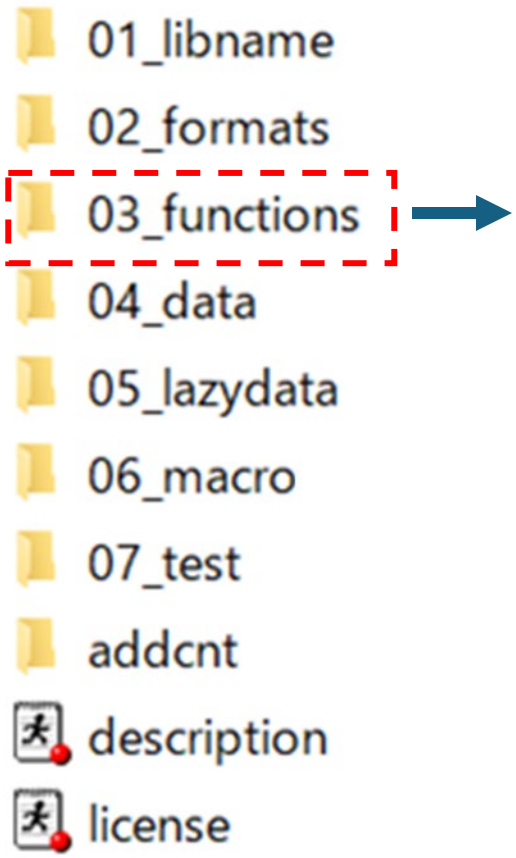
Note: A folder structure of format (without s) is possible, but the format folder was historically created earlier and should contain a proc format lib = work. & packageName.format; and run; (Formats is usually more convenient.)

Source file folder for package



developer

- Examples of sas file content in each folder




```
1   /*** HELP START ***//*
2
3   F1 is an user-defined function to output +1 value.
4   Proc fcmp outlib=work.f.p ; and run ; are not needed in
5   options cmplib=work.f ; is required prior to f1 is used.
6
7   *//*** HELP END ***/
8
9       function F1(n);
10          return (n+1 );
11      endsub;
```

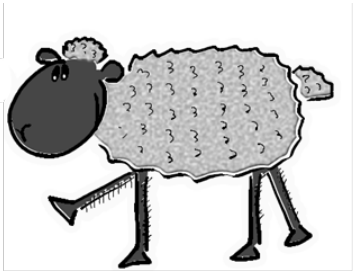
Help Information Field

It is impossible to describe Proc fcmp; and run; (Describe only the contents)

For Functions, 1 file: 1 object is not required.

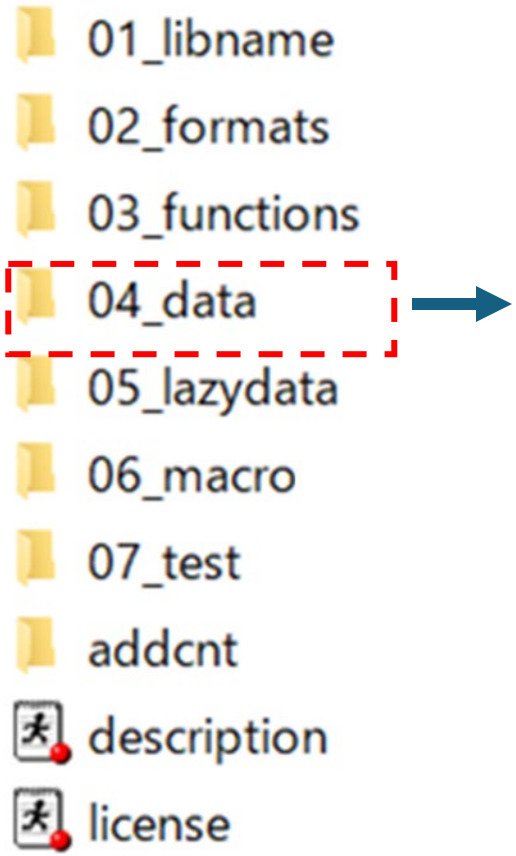
Note: A folder structure of function (without s) is possible, but the function folder was historically created earlier and should contain proc fcmp outlib = work. & packageName.fcmp.package; and run; (I usually prefer the functionss folder.)

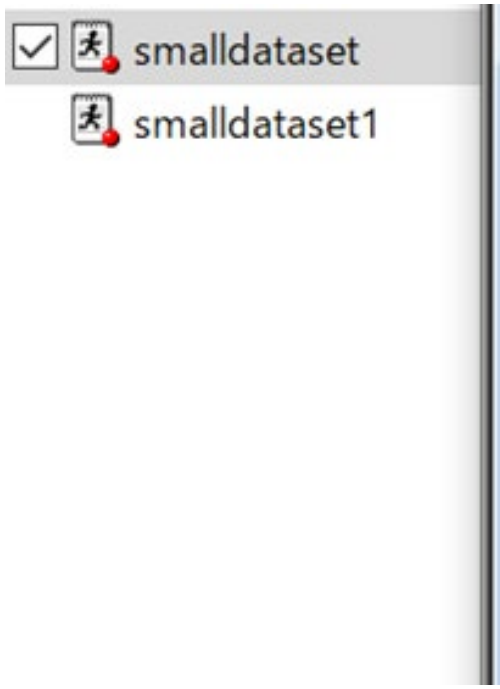
Source file folder for package



developer

- Examples of sas file content in each folder



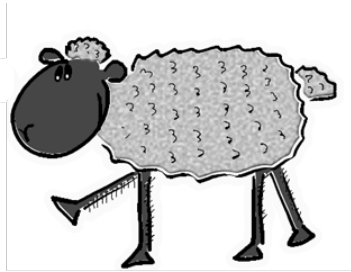



```
1  /*** HELP START ***//
2
3  This is a small dataset.
4
5  *//*** HELP END ***/
6
7  data myLib1.smallDataset;
8      do n = ., -1, 0, 1;
9          m = put(n, fmtNum.);
10     output;
11     end;
12 run;
```

Help Information Field

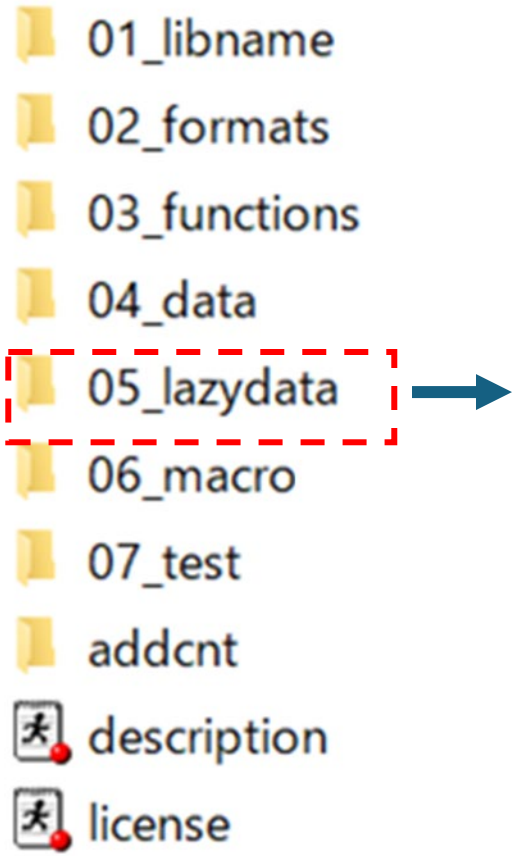
Using mylib1 library defined in 01_libname and fmtNum format defined in 02_formats (so folders are numbered sequentially)

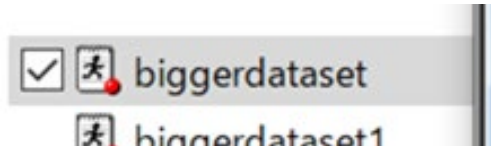
Source file folder for package



developer

- Examples of sas file content in each folder



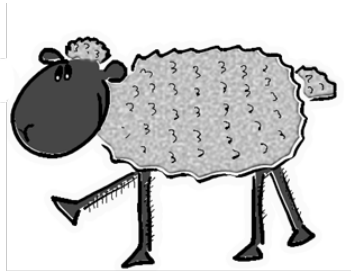



```
1  /*** HELP START ***//*
2
3  This is a bigger dataset.
4
5  *//*** HELP END ***/
6
7  data myLib1.biggerDataset;
8      do i = ., -1e6 to 1e6;
9          j = put(i, fmtNum.);
10         k = ranuni(17);
11     output;
12     end;
13 run;
```

Help Information Field

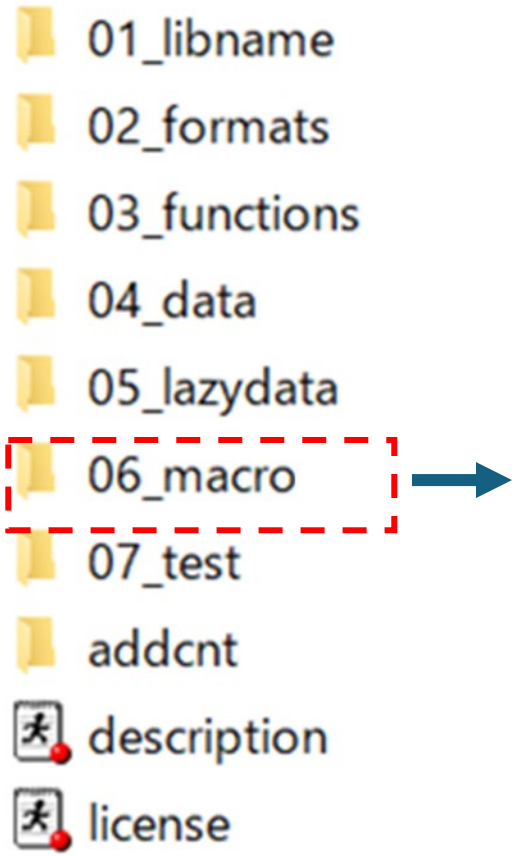
The file is not loaded unless lazyData = is specified in% loadPackage ().

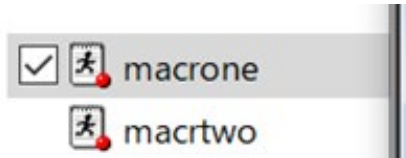
Source file folder for package



developer

- Examples of sas file content in each folder



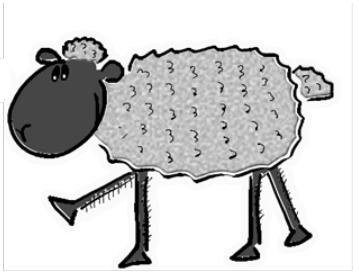



```
1  /*** HELP START ***//*
2
3  This is mcrOne macro.
4
5  *//*** HELP END ***/
6
7  %macro mcrOne();
8      %put **Hi! This is macro &sysmacroname.**;
9      data _null_;
10         set myLib1.smallDataset;
11         p = f1(n);
12         p + f2(n);
13         put (n p) (= fmtNum.);
14     run;
15 %mend mcrOne;
16
```

Help Information Field

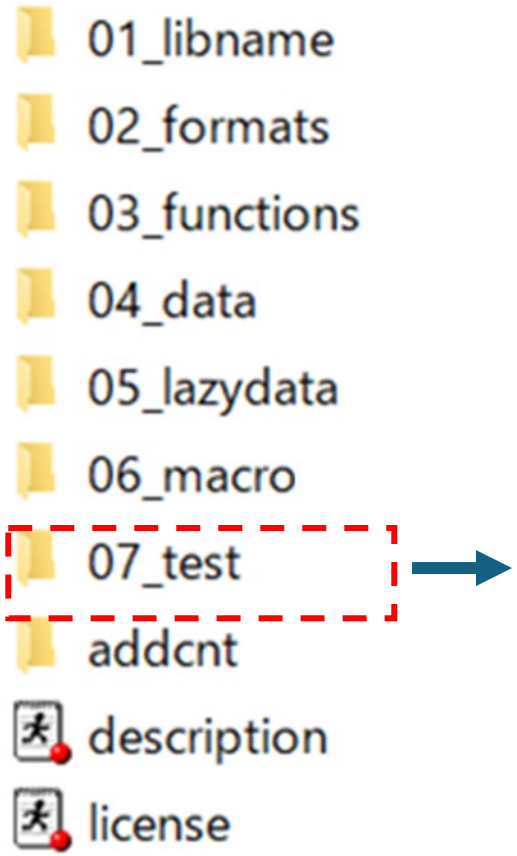
Macro plays a leading role in package!!

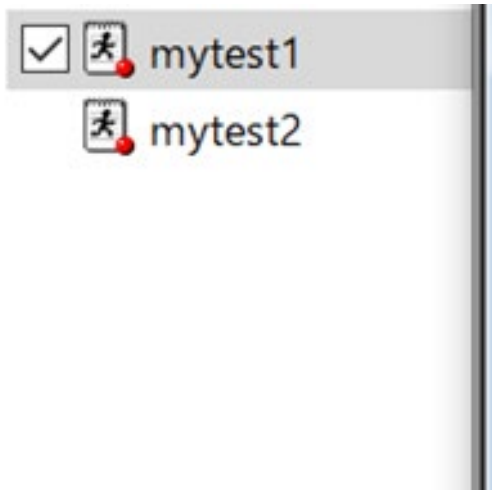
Source file folder for package



developer

- Examples of sas file content in each folder





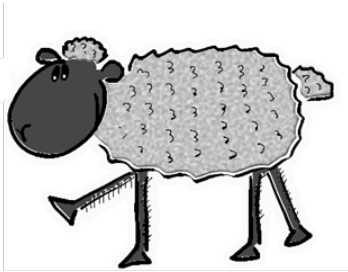

```
1  /*** HELP START ***//*
2
3  This is a test.
4
5  *//*** HELP END ***/
6
7  %mcrTwo(m=mcrOne)
8
```

Help Information Field

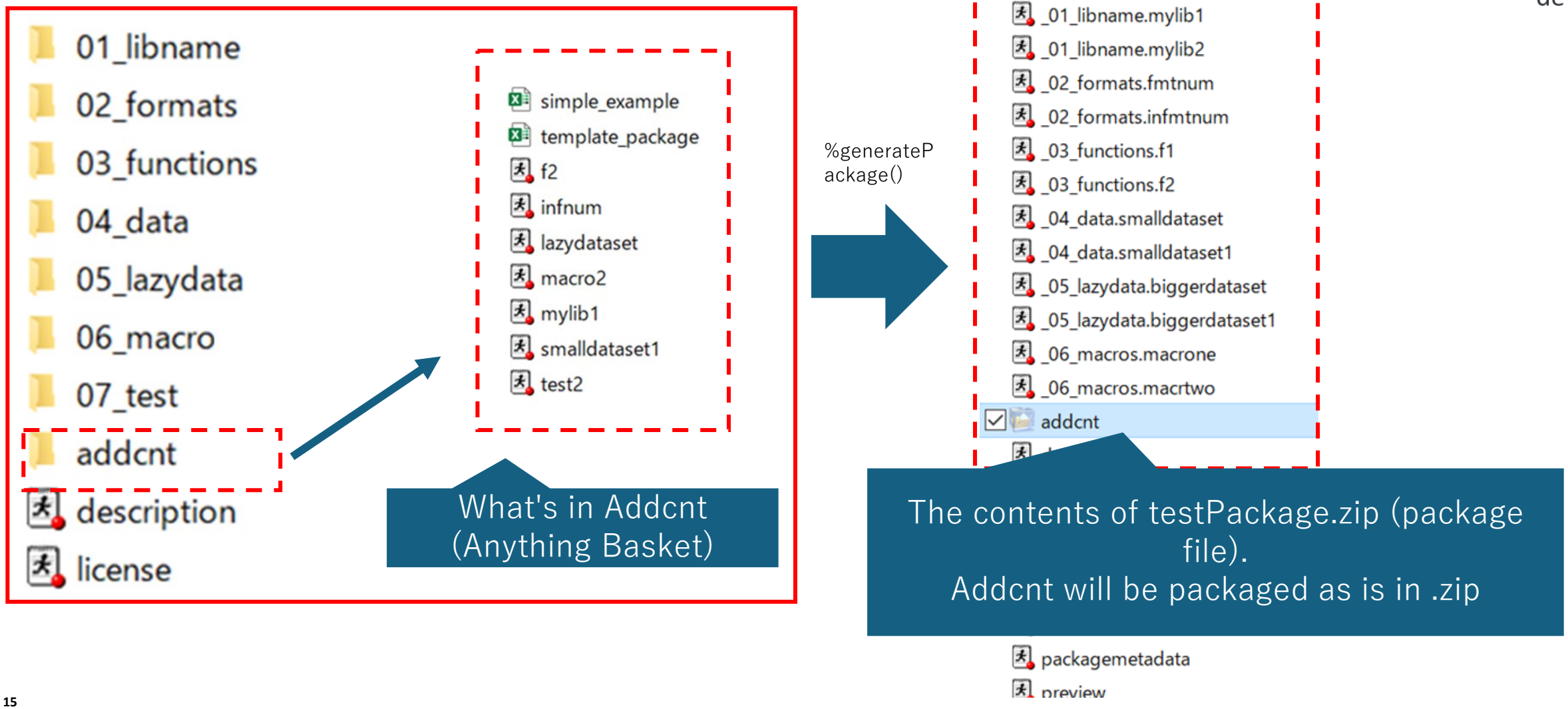
If testPackage = Y (default) is selected in% generatePackage (), the test is executed when the package is created.

Source file folder for package

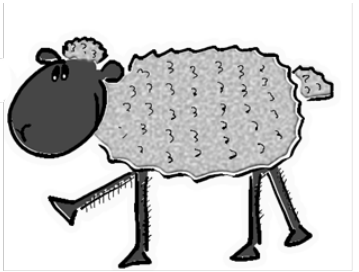
- Additional Contents(Addcnt)



developer



Output of %generatePackage ()



developer

- 01_libname
- 02_formats
- 03_functions
- 04_data
- 05_lazydata
- 06_macro
- 07_test
- addcnt
- description
- license
- testpackage
- testpackage
- testpackage_0.0.1_
- testpackage_0.0.1_

Source folder file (input data to be placed in% generatePackage ())

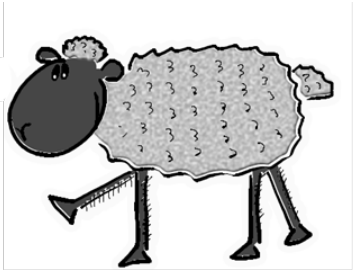
testPackage.md
(Markdown document based on description and help information)

Package body (.zip), instructions (.md), archive copy (_version)
(Output data from% generatePackage ())

- _01_libname.mylib1
- _01_libname.mylib2
- _02_formats.fmtnum
- _02_formats.infmtnum
- _03_functions.f1
- _03_functions.f2
- _04_data.smalldataset
- _04_data.smalldataset1
- _05_lazydata.biggerdataset
- _05_lazydata.biggerdataset1
- _06_macro.macrone
- _06_macro.macrtwo
- addcnt
- description
- help
- iceloadpackage

Contents of testPackage.zip (package body)

Options to create SAS packages



developer

1. Manually create source files and folders and run %generatePackage
   You can copy existing source packages(e.g. Adamski) as a start point
2. Use SASPACer
3. Use SASPACerShiny and run %generatePackage
4. Ask Oba-chan(SAS Package Lady) to cook source files and folders and run %generatePackage

how_to_create_example_SAS_package.sas

SASPACer

- Package for **Creating SAS package in one step**
  (A mechanism that you can generally understand when you look at Sample in Excel)
  https://github.com/PharmaForest/SASPACer
  - **%ex2pac(), %pac2ex()**

Type	Package
Package	testPackage
Title	My first SAS package
Version	0.0.1
Author	John Smith(john.smith@mail.com)
Maintainer	John Smith(john.smith@mail.com)
License	MIT
Encoding	UTF8
Required	"Base SAS Software"
ReqPackage	"Baseplus (2.1.0)"
Description	## The myPackage ## The `myPackage` is my first SAS package. ### References ### 1. Bartosz Jablonski. "My First SAS Package - a How To". SGF

description | license | 01_libname | 02_forma

name	help	body	location
mcrone	This is mcrOne macro. (No need to write location column if content is written in body column.)	%macro mcrOne(); %put **Hi! This is macro &sysmacroname.**; data _null_; set myLib.smallDataset; p = f1(n); p + f2(n); put (n p) (= fmtNum.); run; %mend mcrOne;	
mcrtwo	This is mcrTwo macro. (No need to write body column if content is in a file written in location column, SASPACer reads the file.)		C:¥temp¥addcnt¥mcrtwo.sas

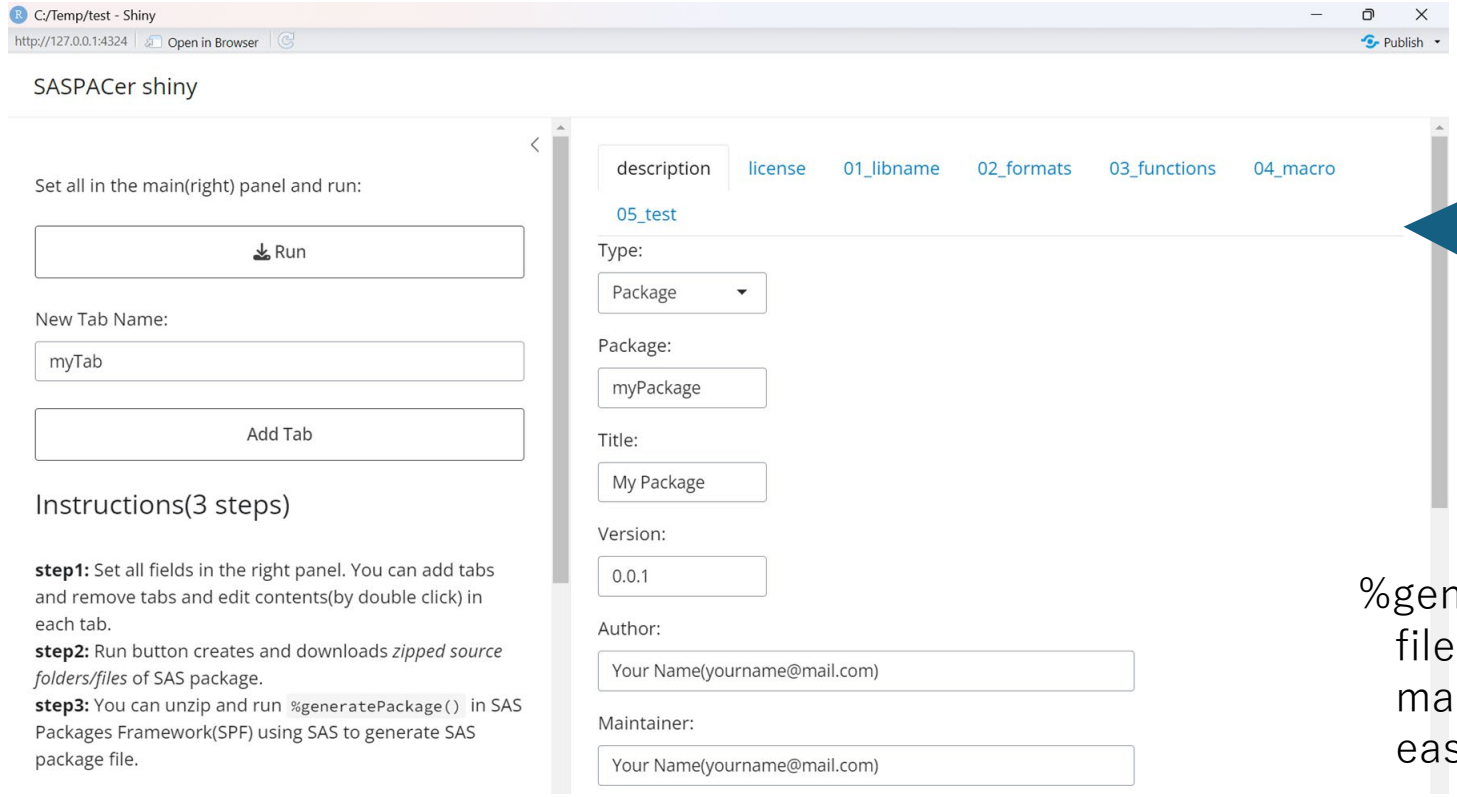
02_formats | 03_functions | 04_data | 05_lazydata | 06_macro | 07_test

```
%ex2pac(
    excel_file = ¥folder¥to¥excel_file.xlsx,
    package_location = ¥folder¥for¥output,
    complete_generation = Y
)
```

Output source folder files and package files as input in Excel.
There is no need to create a folder or file!

SASPACerShiny

- For **SAS Package Creation using Rshiny**
  R package (Shiny application)
  https://github.com/PharmaForest/SASPACerShiny



Set up source folder files in the graphical user interface
Download and run %generatePackage() separately

```
%generatePackage(
  filesLocation = ¥path¥to¥source¥package¥folder,
  markdownDoc = 1,
  easyArch = 1)
```

SAS Package Lady

- The aunty who cooks SAS Package interactively
  https://chatgpt.com/g/g-68be12f679a88191866ef1e9b35be3c4-sas-package-lady

- Support details
  - Listens to package information
  - Can add help information when uploading macro files, etc.
  - The system outputs the source package (folder and file structure)
  - They also help me create logos
- Requires chatGPT sign-up (free version OK)

Your aunt will create the source package, download it, and run %generatePackage() separately.

```
%generatePackage(
  filesLocation = ¥path¥to¥source¥package¥folder,
  markdownDoc = 1,
  easyArch = 1)
```


Enjoy!!

References

- SAS Packages: The Way to Share (a How To), SAS Global Forum 2020, Bartosz Jablonski
  https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2020/4725-2020.pdf
- SASPAC(SAS Packages Repository)
  https://github.com/SASPAC
- Modern graph package “SAS plotter,” SAS User Group Conference Japan 2024, Kosuke Tsutsugo
  https://github.com/SASPAC/sasplotter
- Integration of SAS GRID environment and SF-36 Health Survey scoring API with SAS Packages, PharmaSUG 2024, Bartosz Jablonski
  https://pharmasug.org/wp-content/uploads/2024/06/PharmaSUG-2024-SD-262.pdf
- SASPAC - Introduction of a certain Github archive-, The 10th Osaka SAS Study Group 2024, Ryo Nakaya,
  https://sites.google.com/view/osakasasbenkyokai/%E7%AC%AC10%E5%9B%9E
- SASPAC2 - Examination of Real World Data acceleration tool -, The 11th Osaka SAS Study Group 2025, Teruko Narusawa,
  https://sites.google.com/view/osakasasbenkyokai/%E7%AC%AC11%E5%9B%9E
- PharmaForest
  https://github.com/PharmaForest
- SASPACer
  https://github.com/Nakaya-Ryo/SASPACer
- SASPACerShiny
  https://github.com/Nakaya-Ryo/SASPACerShiny
- SAS Package Lady
  https://chatgpt.com/g/g-68be12f679a88191866ef1e9b35be3c4-sas-package-lady
- RWDExpress
  https://github.com/Narusawa-T/RWDExpress