

mkheader: An R Package for Automated Generation and Management of Program Headers in Clinical Trial Programming

Laura Frederick, Merck & Co., Inc., Rahway, NJ, USA

Gabriela Piasecki, Merck & Co., Inc., Rahway, NJ, USA

PHUSE US Connect 2026 Paper OS16

26 March 2026



Agenda



Introduction



Package Overview



Core Addins



Quality Checks



Future Improvements



Conclusion

Introduction

Introduction

Why Do Program Headers Matter?

- Contain critical metadata
 - Project (Compound, study, indication)
 - Program name and function
 - Author
 - Date
 - Dependencies and outputs
 - Revision History
- Part of Good Programming Practice (GPP)

```
*****;
* Project      : Sample Drug, Sample Indication, Study1
*
* Program name : sample.sas
*
* Author       : smithb
*
* Date created  : 20100816
*
* Purpose      : Summarize demographics data for the study.
*
* Revision History :
*
* Date      Author    Ref    Revision (Date in YYYYMMDD format)
* 20100818  smithb     1      Removed subjects with who have not been dosed per spec.
*
*****;
```

Introduction

The Problem with Manual Headers

- Manual updates are time-consuming and error-prone
- Common Issues
 - Copy-paste errors
 - Inconsistent formatting
 - Incomplete and outdated info
 - Weak revision tracking
- Documentation errors increase compliance risk

SAS Program-2

Issue

Lack of necessary information in the program header

- Missing SAS macro dependency

Recommendations

- Provide comprehensive information in the program header, such as the datasets used and the names of all macros called in the program
- Ensure the program's structure and dependencies are easy to understand



Slide excerpt from: Liping Sun, PharmaSUG 2023
 "Experiences of CDER Statistical Analysts with NDA/BLA reviews: Some Helpful Tips for Sponsors"

MERCK

SAS Program-2

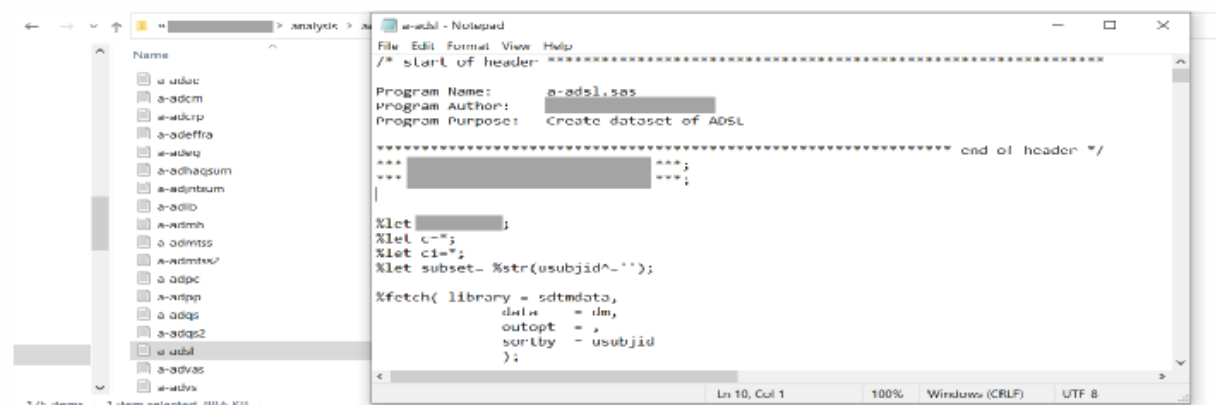
Issue

Lack of necessary information in the program header

- Missing SAS macro dependency

Recommendations

- Provide comprehensive information in the program header, such as the datasets used and the names of all macros called in the program
- Ensure the program's structure and dependencies are easy to understand



13

Slide excerpt from: Liping Sun, PharmaSUG 2023

“Experiences of CDER Statistical Analysts with NDA/BLA reviews: Some Helpful Tips for Sponsors”

Introduction

R Package Motivation

- Benefits of automation
 - Improved efficiency
 - Fewer documentation gaps
 - Increased compliance
 - Ability to leverage standard metadata
- Automated SAS macros and tools exist
- Increased R adoption
 - Lack of automated header tools in R = manual work

The screenshot displays the (CPI) Programming Compliance Toolbox interface. The main window contains a form for generating a program header. The form includes the following fields and sections:

- Choose Template:** A dropdown menu set to "Program Template".
- Study Path:** A text input field.
- Directory:** A text input field with a "Directory..." button.
- Program Name:** A text input field.
- Program Function:** A text input field.
- Keywords:** A text input field.
- SAS Version:** A text input field set to "9.4".
- Platform:** A text input field set to "LINUX".
- Macro Parameters:** A table with columns "parameter", "value", and "description". It contains one row with a "*" in the "parameter" column.
- User Name:** A text input field set to "fredlau (Laura Frederick)".
- Input Data:** A text input field.
- Macros Called:** A text input field.
- Proc Printto:** A section with a "Yes" radio button selected and a "No" radio button.
- Program Log File:** A text input field with a "Directory..." button.
- Program Output File:** A text input field with a "Directory..." button.
- Buttons:** "Program Flow...", "OK", and "Cancel" buttons at the bottom.

Package Overview

Package Overview

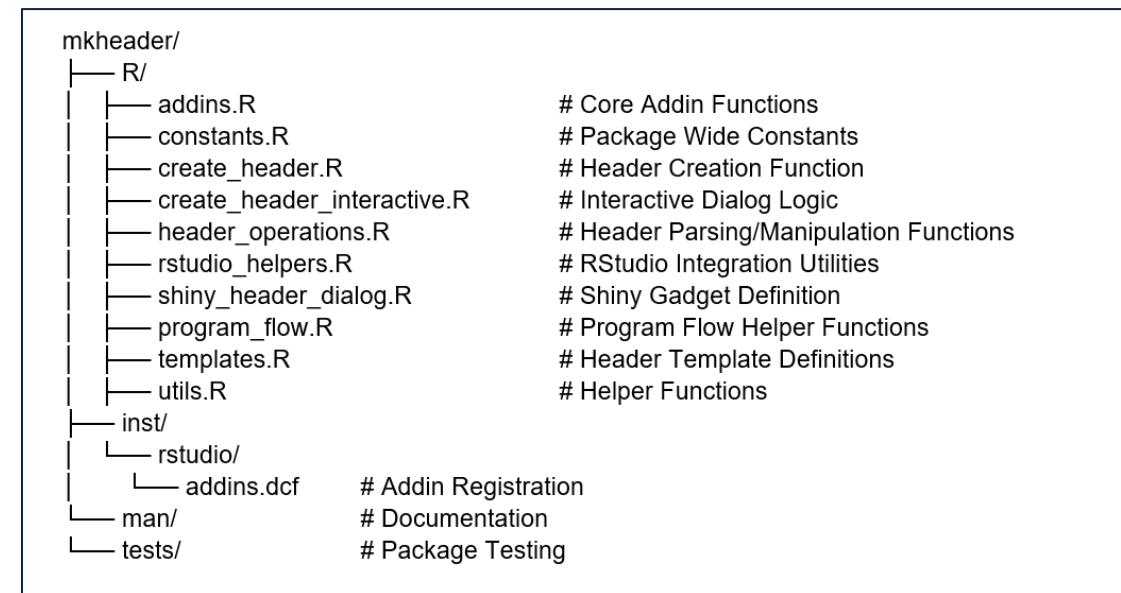
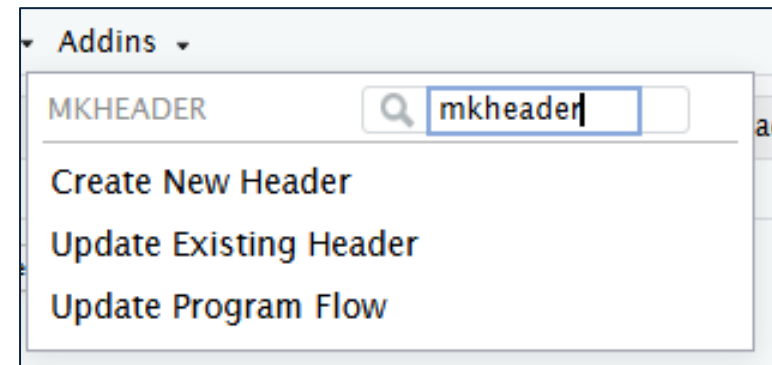
Introducing mkheader

- Purpose: Automated header creation/maintenance
- Interactive user interface
- Extensible template system
- Three core addins
 - **Create New Header** – Automatically generate standardized headers
 - **Update Existing Header** - Maintain headers with revision tracking
 - **Update Program Flow** - Auto-sync header with code step comments

Package Overview

mkheader Package Architecture

- Shiny gadgets-based popup windows
- Rstudio integration via Addins
- Core functions for each addin
 - `addin_create_header()`
 - `addin_update_header()`
 - `addin_update_program_flow()`
- Support functions to process headers
 - Ex: `.check_header()`

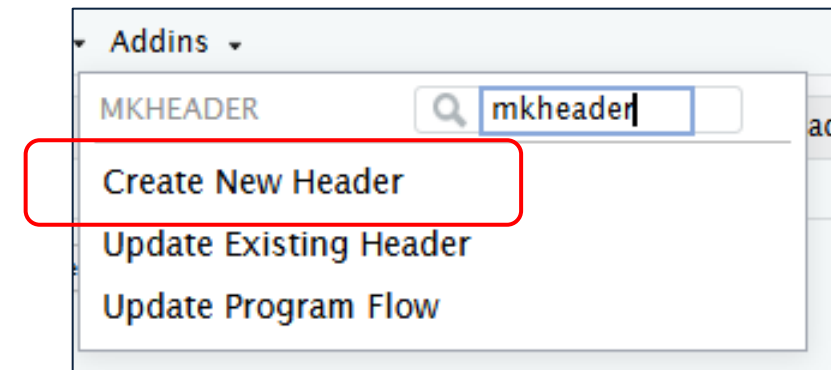


Core Addins

Core Addins

Create New Header - Overview

- Identifies active R program
- Minimal User Input
 - Automatically extracts:
 - Program name
 - Study ID
 - Author
 - Current date and R version
 - Pre-fills defaults
- Guidance / examples for each entry field
- Outputs standardized header per template



Core Addins

Create New Header - Demo

mkheader

Cancel mkheader: Create / Update Program Header Done

Note: Required fields are marked with *

Program Directory:*
/opt/bardsar/test/mkXXXX/protXXX/pgmsetup

Study:
mkXXXX/protXXX

Program Name:*
ADSL.R

Keywords:
analysis, data processing, reporting

Program Function:*
Describe what this script does...

Author:*
fredelau

Version Date:*
2026-01-05

R Version:*
4.4.1



ADSL.R

Source on Save

```

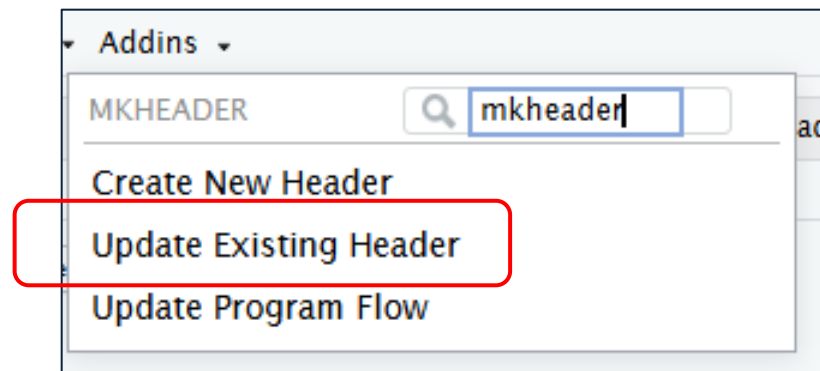
1  #-----#
2  #### PROGRAM HEADER ####
3  #-----#
4  #
5  # Study:          mkXXXX/protXXX
6  #
7  # Program:        ADSL.R
8  #
9  # Keywords:       ADaM ADSL
10 #
11 # Function:        Create ADSL dataset
12 #
13 # Author:          fredelau
14 #
15 # Version Date:    05JAN2026
16 #
17 # R Version:       4.4.1
18 #
19 # Platform:        LINUX
20 #
21 # Input Data:      dm.sas7bdat
22 #
23 # Functions Called: ex_function()
24 #
25 # Program Output:   proj/outlist/ADSL.lst
26 #
27 # Program Log:      proj/outlog/ADSL.log
28 #
29 # Function Parameters: N/A
30 #
31 # Program Flow:
32 #
33 # Limitations/Cautions: N/A
34 #
35 # Comments:         This is an example comment.
36 #
37 # Revisions:
38 # Name              | Date      | Description
39 #
40 # fredelau           | 05Jan2026 | Initial creation
41 #
42 #-----#

```

Core Addins

Update Existing Header - Overview

- Parses current header
- Pre-fills dialog with existing values
- Automatically prompts for new revision entry
 - Ensures complete traceability
- Inserts updated header to active program
 - Preserves existing code



Core Addins

Update Existing Header - Demo

Comments:

This is an example comment.

Revision History

Existing Revisions:

fredelau | 19Jan2026 | Initial creation

☒ Add New Revision Entry

Revision Author:*

fredelau

Revision Date:*

2026-01-06

Revision Description:*

Update FASFL derivation



```

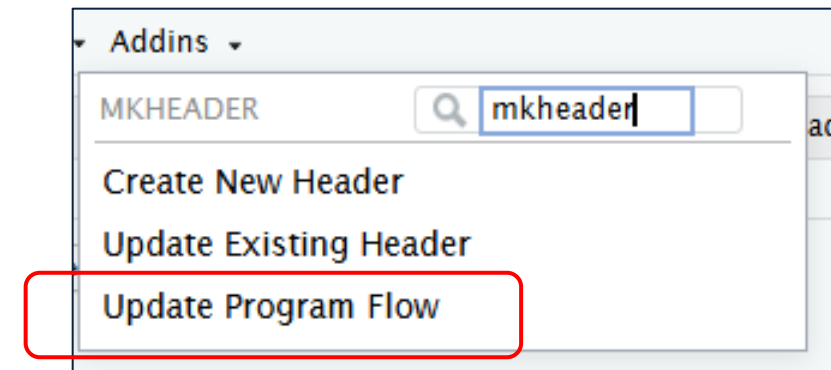
#
# Comments:          This is an example comment.
#
# Revisions:
# Name                | Date       | Description
#
# fredelau           | 05Jan2026 | Initial creation
# fredelau           | 06Jan2026 | Update FASFL derivation|
#
# -----#

```

Core Addins

Update Program Flow- Overview

- Scans code for “Step” comments
- Syncs comments with Program Flow section
- Reflects current code logic
 - Quickly update header flow
 - No manual edits needed



Core Addins

Update Program Flow - Demo

```

45 # Step 1: Load Packages
46 library(dplyr)
47 library(pharmaversesdtm)
48
49 # Step 2: Read in Data
50 dm <- pharmaversesdtm::dm
51
52 # Step 3: Derive Treatment Variables (TRT0xP, TRT0xA)
53 adsl <- dm %>%
54   mutate(TRT01P = ARM, TRT01A = ACTARM)
55

```



```

#
# Program Flow:
# Step 1: Load Packages
# Step 2: Read in Data
# Step 3: Derive Treatment Variables (TRT0xP, TRT0xA)
#
# Limitations/Cautions
#
# Comments:
#
# Revisions:
# Name
#
# fredelau
# fredelau
#
#-----

```

Success

Program Flow updated with detected steps.

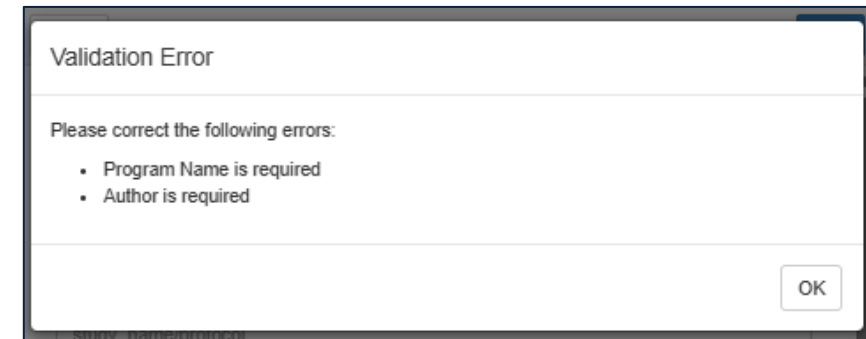
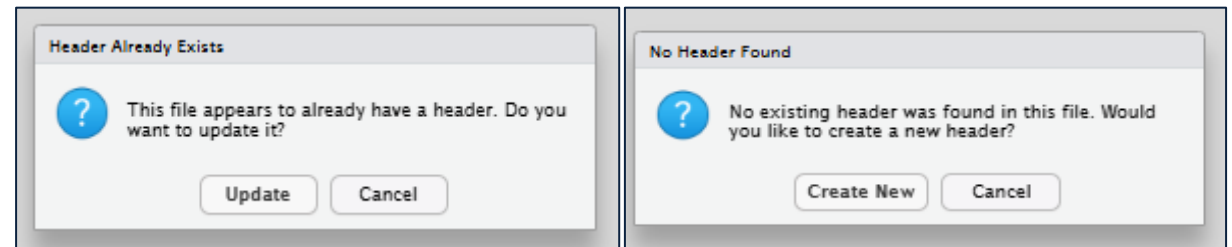
OK

Quality Checks

Quality Checks

Built-in Safety and Validation

- Pre-operation checks
 - Verifies RStudio environment
 - Checks for existing headers
 - Prompts to save unsaved files
- Field validation
 - Required fields enforced
 - Date validation (e.g., no future dates)
 - Clear error messages
- No changes until all validations pass



Future Improvements

Future Improvements

Planned Upgrades and Enhancements

- Validation, testing and user feedback collection
- Internal publication
- Additional Functionality
 - Auto-detect inputs/outputs and dependencies
 - Separate 'Update Revision History' addin
 - More standard program header templates
 - Improved step comment extraction robustness
 - Support for SAS and Python headers

Conclusion

Conclusion

Why mkheader Matters

- **Automation** - Eliminates manual header maintenance
- **Consistency** - Standardized format across all scripts
- **Compliance** - Built-in revision tracking and validation
- **Efficiency** - Reduced manual effort and savings on documentation

Disclaimer

The views and opinions presented in the paper are our own and do not necessarily reflect the official position of the company.

Thank you!

We value your questions and feedback. The authors can be contacted at:

- **Laura Frederick**

Senior Scientist, Statistical Programming

Merck & Co., Inc., Rahway, NJ, USA

E-mail: laura.frederick1@merck.com

- **Gabriela Piasecki**

Senior Scientist, Statistical Programming

Merck & Co., Inc., Rahway, NJ, USA

E-mail: l.gabriela.piasecki@merck.com



Questions?