Learning SAS: A Beginner’s Guide to Using Comments in Your Code

Authored by

Mohammed looti

November 16, 2025

The Indispensable Role of Comments in SAS Programming

In the expansive realm of modern programming language development, comments are not merely optional features; they serve as absolutely essential annotations within your executable code. These textual snippets are designed to be entirely ignored by the compiler or interpreter, yet they prove invaluable for human readers, facilitating understanding and maintenance. Within SAS, a powerful and widely adopted statistical software suite, mastering the effective use of commenting is fundamental for developing programs that are maintainable, lucid, and inherently collaborative.

Effective commenting significantly enhances code readability, simplifying the process for both you and your colleagues to quickly grasp the underlying logic, the programmer's intent, and any specific considerations driving the program's design. Furthermore, well-placed comments are crucial for efficient debugging, offering a non-destructive method to temporarily disable or isolate sections of code for testing. Collectively, they form the backbone of sustainable software documentation.

This comprehensive guide will systematically explore the various established and recommended methods for creating and managing comments in SAS, ranging from convenient integrated keyboard shortcuts to precise manual syntax application. By implementing these robust techniques, you can ensure that your statistical programs are consistently clear, well-structured, and fully documented, thereby promoting accuracy and long-term project viability.

Understanding the Two Primary Comment Styles in SAS

Before delving into the practical mechanics of implementation, it is beneficial to clearly distinguish between the two primary styles of comments supported natively by the SAS environment: block comments and line comments. Each style is optimized for a slightly different application and possesses its own distinct structural characteristics and required delimiters.

Block comments, as their designation suggests, are specifically engineered to encapsulate and enclose multiple contiguous lines of text or code. They are optimally suited for providing extensive high-level explanations, outlining the overall purpose of a major code segment, or, most critically, temporarily disabling a significant chunk of your active program without deleting the source material. In SAS, these comments must initiate with the delimiter `/*` and must terminate with the corresponding closing delimiter `*/`, creating a defined boundary for the non-executable text.

Line comments, conversely, are designed exclusively for concise, single-line annotations. They are the perfect tool for adding brief notes, clarifying a specific variable assignment, or providing immediate context for a single programmatic step. SAS supports two main methods for line commenting: the use of the traditional C-style block comment (`/* ... */`) applied to a single line, and a unique, highly utilized SAS-specific line comment (`* ... ;`). This specialized SAS syntax is

always delimited by an asterisk at the beginning and, most importantly, requires a mandatory semicolon at the end of the line.

Method 1: Leveraging the Efficient Keyboard Shortcut (Ctrl + /)

For the majority of SAS developers working within the integrated SAS environment (such as SAS Studio or the SAS Display Manager), the fastest and most highly recommended way to insert or remove comments is by utilizing a simple, integrated keyboard shortcut. This method is exceptionally convenient and efficient for toggling the comment status of one or more contiguous lines of selected code, dramatically speeding up the development process.

To instantly transform existing executable code into a non-executable comment, you must first **highlight the desired text block** within the SAS editor pane. Once the text is selected, simply press the key combination **Ctrl** + **/**. The SAS environment automatically recognizes this command and inserts the necessary block comment delimiters (`/*` at the start and `*/` at the end of the selected region) around your highlighted selection, effectively commenting out the entire block.

The process of removing these comments, often referred to as 'uncommenting,' is equally swift and simple. Highlight the previously commented text once more and press **Ctrl** + **/** again. The SAS editor is intelligent enough to detect the existing comment syntax and will instantly remove the `/*` and `*/` characters, thereby restoring your code to its active, executable state. This seamless toggle functionality dramatically streamlines the iterative development and debugging workflow, allowing rapid testing of different code segments.

Method 2: Manual Block Comments for Extensive Documentation (/* ... */)

While the keyboard shortcut offers unparalleled convenience, possessing a thorough understanding of the manual syntax for block comments remains fundamental for robust SAS programming. The universally recognized C-style block comment is highly flexible and robust, enabling you to comment out anything from single statements to entire procedures, regardless of the line count.

To manually initiate a block comment, you begin by typing the opening delimiter `/*` at the precise location where you intend for your annotation to commence. Crucially, all text immediately following this opening delimiter will be treated as a non-executable comment until SAS encounters the corresponding closing delimiter, `*/`. This design allows your comments to logically span across multiple lines and paragraphs without the need to individually mark each line, making it perfect for lengthy descriptions.

This manual method is particularly suitable when you are required to provide comprehensive, detailed explanations for intricate algorithms, specify external data sources, or temporarily

deactivate a substantial segment of a SAS DATA Step or PROC without actually deleting the code. The following example demonstrates its powerful application in a typical SAS environment, showing how descriptive text and even code lines can be safely enclosed:

```
/* This block of code performs data cleaning
and transformation steps for the raw_data dataset.
It handles missing values and outliers based on
interquartile range calculations. */
DATA clean_data;
SET raw_data;
/* More specific comments could go here */
IF var1 > 100 THEN var1 = .; /* Outlier handling: set values > 100 to missing */
RUN;
```

Method 3: The SAS-Specific Line Comment (* ... ;)

SAS also furnishes a unique, proprietary line comment syntax that differs structurally from the traditional C-style block comment. This specific approach is ideally utilized for adding concise, immediate, and single-line annotations directly adjacent to your code, thereby supplying instant context for specific commands or individual statements that require brief clarification.

A SAS-specific line comment starts with a single asterisk (**`*`**) and, unlike many other programming languages, must conclude with a semicolon (`;`). Everything positioned between the asterisk and the terminating semicolon on that specific line will be rendered non-executable and treated as a comment. It is absolutely vital to remember the closing semicolon, as its accidental omission will invariably result in a SAS syntax error, halting program execution and requiring immediate corrective action.

This style of comment is frequently employed for quick, focused notes, such as explaining the precise purpose of a variable, detailing a specific option utilized within a procedure, or marking a specific point for future revision or refinement. Because of its brevity and required termination, it fits naturally at the end of a line of code. Here is a practical illustration demonstrating its use within a standard SAS program structure:

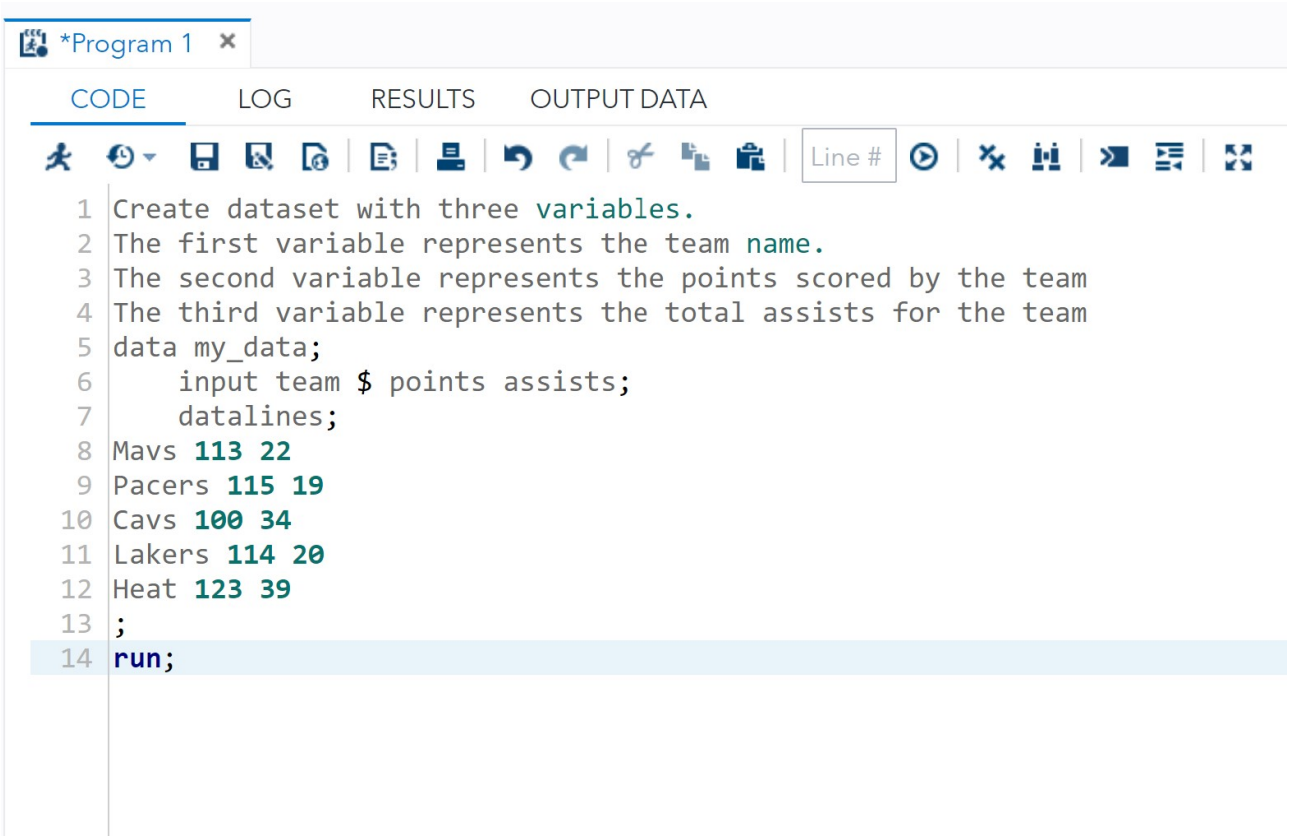
```
DATA my_data;
SET original_data;
new_var = old_var * 2; * Calculate a derived variable for analysis;
DROP temp_var; * Remove temporary variable to save memory;
RUN;
```

```
PROC PRINT DATA=my_data;
VAR new_var; * Display only the newly created variable;
RUN;
```

Practical Demonstration: Toggling Comments in SAS Code

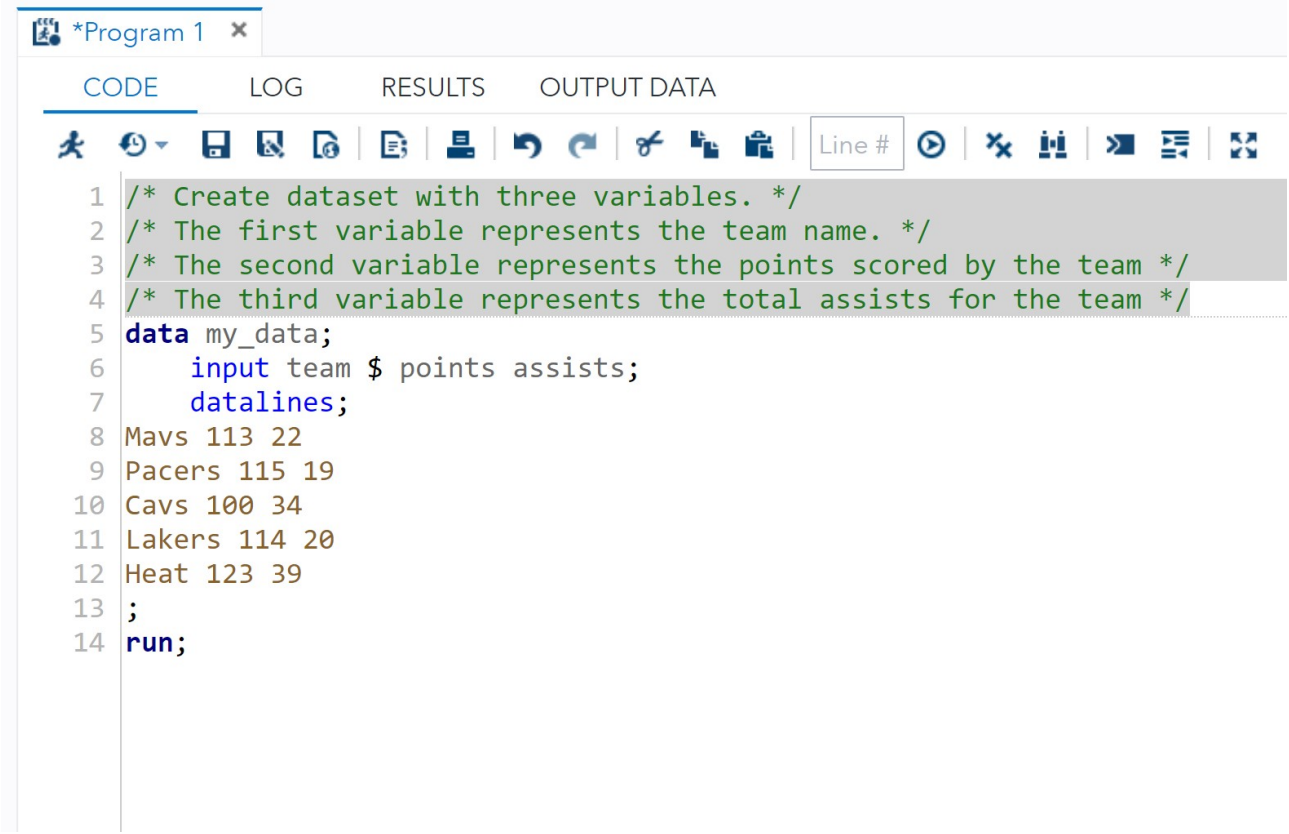
To solidify these concepts, let us walk through a practical scenario illustrating how to effectively deploy and manage comments within a real-world SAS programming context. Imagine we are tasked with documenting a SAS program designed to process preliminary data, and we need to insert descriptive notes to dramatically improve its overall code readability and maintainability.

Consider the following SAS code snippet. The initial descriptive lines convey the general overview of the program's function. These lines are perfect candidates for conversion into comments to clearly distinguish them from the actual executable code:



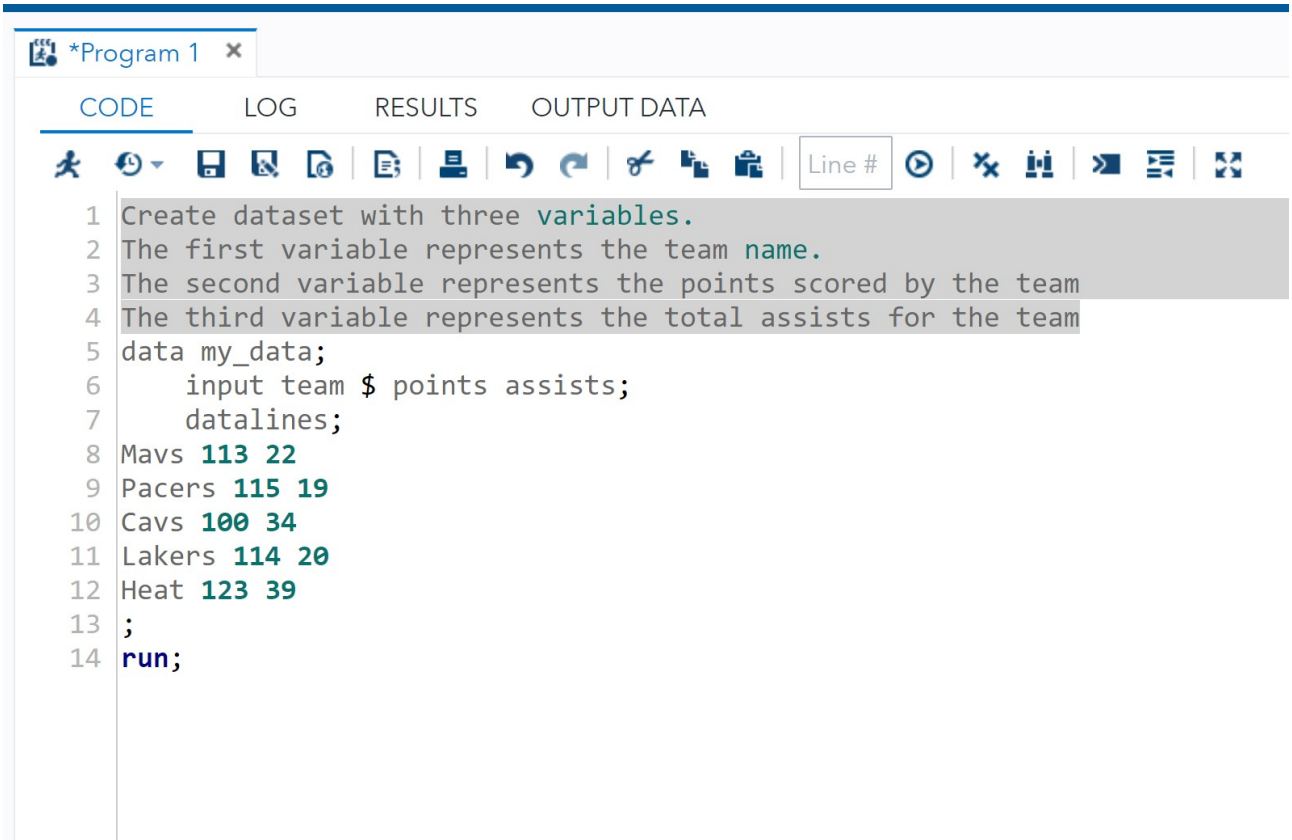
Our immediate objective is to transform those first four descriptive lines into non-executable comments, thereby making them purely for documentation purposes. The most efficient and recommended way to achieve this is by utilizing the keyboard shortcut discussed earlier. We begin by carefully highlighting each of these four lines within the SAS editor.

Once the lines are highlighted, pressing **Ctrl** + **/** instantly converts them into a cohesive block comment. Observe the visual transformation in the image below: `/*` has been automatically prepended to the beginning of the selected lines, and `*/` has been appended to the end of the block. This visual change clearly signifies that these lines are now inactive and will be ignored during program execution.



Should you subsequently decide that these lines must be restored to active code, or if you simply wish to remove the comment wrappers, the reversal process is equally direct and fast. Highlight the same four lines once more and press **Ctrl** + **/**. The SAS editor will intelligently detect the existing comment delimiters and remove both the `/*` and `*/` tags without requiring manual editing.

The final image demonstrates the outcome of successfully uncommenting the lines. Notice that the comment delimiters are gone, indicating that the lines are no longer static annotations and would now be processed as executable code if they contained valid SAS instructions. This seamless, instantaneous toggling capability is an invaluable asset during agile code development and iterative debugging sessions, allowing developers to switch between active and inactive code segments instantly.



Best Practices for Writing High-Quality SAS Comments

While understanding the precise mechanics of creating comments is essential, knowing precisely **when** and **what kind** of information to include in your comments is paramount for developing truly high-quality SAS programs. Adhering to established best practices ensures that your annotations genuinely add significant value to the code base rather than simply cluttering the screen.

Focus on the "Why," Not Just the "What": Resist the urge to simply reiterate what the code mechanically performs. For example, avoid writing `* Add 1 to X;` next to the code `X = X + 1;`. Instead, dedicate your commentary to explaining the rationale, underlying assumptions, or complex business logic that necessitated the code block's presence.

Thoroughly Document Complex Logic: Any section of code that is not inherently obvious, employs complicated statistical algorithms, or utilizes non-standard programming approaches must be extensively commented. This effort is absolutely critical for long-term maintenance, auditing, and efficient troubleshooting by future developers.

Highlight Assumptions and Limitations: If your program's execution or results depend on specific assumptions regarding the input data structure, integrity, or external environment, document these constraints clearly at the top of the program or procedure. Similarly, explicitly note any known limitations or edge cases the current code might not handle robustly.

Use Markers for Future Revisions: Employ standardized comment markers such as `/* TODO: Implement robust error handling for missing dates */` or `* REVIEW: Check performance optimization needed here;` to effectively flag areas that require future development, immediate refactoring, or further attention from the team.
Maintain Up-to-Date Comments: Outdated comments can be highly deceptive and are often worse than having no comments at all, as they actively mislead future users regarding the code's true function. Always commit to updating your comments immediately whenever you modify the corresponding code segment to ensure accuracy.
Avoid Over-Commenting: The excessive placement of comments can clutter the code, making it ironically harder to read and navigate. Strive for an optimal balance where comments serve to clarify specific ambiguity rather than obscure the natural flow of the program structure.
Ensure Consistency in Style: Adopt a standardized, consistent style for all your comments (e.g., consistent capitalization, mandatory indentation, or specific delimiters) across all your projects. This professional consistency dramatically enhances the overall code readability and reduces cognitive load for anyone reviewing the script.

Conclusion

The effective and thoughtful use of comments is a definitive hallmark of proficient SAS programming. Whether you choose to leverage the sheer speed and convenience of the **Ctrl** + **/** shortcut, the structural flexibility of `/* ... */` block comments for large sections, or the precision of the SAS-specific `* ... ;` line comments for immediate annotations, integrating these robust documentation practices into your daily workflow will significantly enhance the clarity, long-term maintainability, and collaborative potential of all your SAS programs.

By investing a relatively small amount of time and effort into meticulously documenting your code now, you will save exponentially greater effort in the future, particularly when revisiting archived projects or engaging in team collaboration and knowledge transfer. Therefore, embrace comments not merely as simple annotations, but as an indispensable and integral component of your comprehensive SAS development process.

Additional Resources for SAS Programming

To further advance your proficiency in SAS programming skills and explore other common operations and advanced techniques, we recommend consulting the following authoritative resources:

SAS Base Products Documentation: This is the official and most comprehensive source for detailed information regarding SAS procedures, functions, and core language elements.
SAS Communities: A highly active and vibrant forum where SAS users globally can pose complex

questions, share accumulated knowledge, and find validated solutions from peers and experts.

SAS Education & Training: Offers a wide spectrum of certified courses and professional training opportunities designed to deepen your expertise in SAS programming and advanced analytics methodologies.

The SAS Dummy Blog: Provides practical, field-tested tips, tricks, and invaluable insights directly from veteran SAS experts on a diverse array of programming and data management topics.