PharmaSUG China 2025 - Paper CC-130

From Circles to Custom Icons: Controlling Symbols in SAS GTL

Lin Jiang, BeOne

ABSTRACT

Effective symbol control is essential for producing clear and informative statistical graphics. The SAS Graph Template Language (GTL) offers powerful and flexible options for customizing symbol display in both simple and complex plots. This paper introduces a range of techniques for controlling symbol appearance, including basic MARKERATTRS usage, group-specific customization using attribute maps, and integration of special symbols through Unicode characters. It also explores advanced methods such as the SYMBOLCHAR and SYMBOLIMAGE statements, enabling the use of font-based glyphs and image files as plot markers. By combining these approaches, users can achieve highly tailored visualizations that meet the demands of complex graphics.

INTRODUCTION

Effective data visualization is essential for communicating insights clearly and accurately. The SAS Graph Template Language (GTL) empowers you to create highly customizable and reusable statistical graphics by defining plots through a structured set of declarative statements. One key aspect of creating informative and professional plots is controlling the appearance of symbols (also called markers) used to represent data points.

In GTL, symbols can vary automatically based on group values, or you can take full control and define their shape, color, and style explicitly. This flexibility enables you to maintain visual consistency, highlight specific data groups, or apply branding requirements across multiple plots.

This paper introduces the fundamental concepts and syntax used in symbol control, and guides you through practical examples—from simple defaults to advanced techniques using attribute maps, Unicode characters, and image-based markers.

Whether you're building production-quality reports or fine-tuning research plots, understanding these symbol control methods helps you produce clear, consistent, and visually compelling graphics using SAS GTL.

A QUICK EAMPLE

Before diving into advanced symbol control techniques, it's helpful to understand the basic structure of the Graph Template Language (GTL). GTL allows you to define rich, customizable statistical graphics using a clear set of declarative statements.

The following example demonstrates a minimal working GTL graph that:

- Uses a scatter plot to display data.
- Introduces the essential GTL statements and structure.
- Applies basic symbol control using the MARKERATTRS option.

```
/* Step 1: Define a GTL template using PROC TEMPLATE */
proc template;
  define statgraph basic_symbol_plot;
    begingraph;                          /* Start the graph definition */
      layout overlay;                   /* Define the layout area */
        scatterplot x=weight y=height /          /* Specify the plot type */
          markerattrs=(symbol=circlefilled size=10); /* Basic symbol control
*/
      endlayout;
    endgraph;
```

```
  end;
run;

/* Step 2: Render the graph using PROC SGRENDER */
proc sgrender data=sashelp.class template=basic_symbol_plot;
run;
```

This example highlights several key GTL concepts:

- **PROC TEMPLATE**: Used to define a reusable graph template.
- **DEFINE STATGRAPH**: Declares the name and scope of a graph.
- **BEGINGRAPH ... ENDGRAPH**: Marks the beginning and end of the graph structure.
- **LAYOUT OVERLAY**: Defines the plotting space; most plots are created within an overlay.
- **SCATTERPLOT**: A plot statement used here to visualize points.
- **MARKERATTRS**: A plot option that controls the symbol's appearance (shape, size, color).
- **PROC SGRENDER**: Applies the data to the template to generate the actual plot.

LEVELS OF SYMBOL CONTROL

GTL, as an extension of the Output Delivery System (ODS), allows you to create sophisticated graphics using template definitions that govern both structure and appearance. GTL provides a layered approach for marker (symbol) customization: global level, template level, group level, and plot level. Understanding these four levels helps you control marker appearance effectively—from quick styling to detailed customization.

Level	Scope	Use Case	Overrides
ODS Style	Global / session-wide	Use default appearance with minimal setup	Lowest
DATASYMBOLS/DATACOLORS option of BEGINGRAPH	Template-wide	Customize group symbols for a whole graph	Overrides ODS
Attribute Map	Per group value	Precise symbol mapping by group value	Overrides above
MARKERATTRS option of plot statement	Per plot statement	Set exact symbols regardless of grouping	Highest

Table 1 Summary of Symbol Control Levels

ODS STYLE

When you generate any graphic, an ODS style controls its appearance. At this level, marker symbols, colors, and line styles come from the current ODS style (e.g., HTMLBlue, Journal, or Listing). These styles assign default properties based on data grouping order. Use this level when you want consistent visuals without extra configuration.

BEGINGRAPH STATEMENT

BEGINGRAPH statement defines the outermost container for a graph template that is defined with GTL-statements. One of its appearance options, DATASYMBOLS=(marker-symbol-list), specifies the list of marker symbols that will replace the graph data marker symbols from the marker symbols that are defined

in the style elements. It allows users to override the default symbol sequence defined by the ODS style, on a per-template basis. This setting provides template-wide customization of symbol order for grouped data, offering more control while maintaining consistency across plots in the same template.

ATTRIBUTE MAP

By default, GTL assigns graphical properties (like marker shape or color) based on group order in your data. This means changes in data order can lead to inconsistent visuals. Attribute maps let you explicitly associate group values with specific symbols and colors. This ensures consistent styling, no matter how the data are sorted or subsetted.

PLOT STATEMENT

At the most granular level, the MARKERATTRS option allows users to directly define the appearance of markers in a specific plot statement, independent of group values or higher-level settings. This approach is useful when a uniform marker style is desired, or when precise override of all other styling is required.

2Above are the 4 levels to control marker appearance. You can combine these levels. For example, use an attribute map for group-specific markers and still set a fallback order with DATASYMBOLS. Use MARKERATTRS when you need full control or a consistent symbol across all points.

In the rest of this paper, we'll explore how these levels work in real examples — from default settings to advanced symbol customization using font glyphs and images.

EXPLORE IN REAL EXAMPLES

Use dataset CLASS in the following examples:

```
proc sort data=sashelp.class(keep=height weight sex age) out=class;
    by sex age;
run;
```

NON-GROUPED DATA

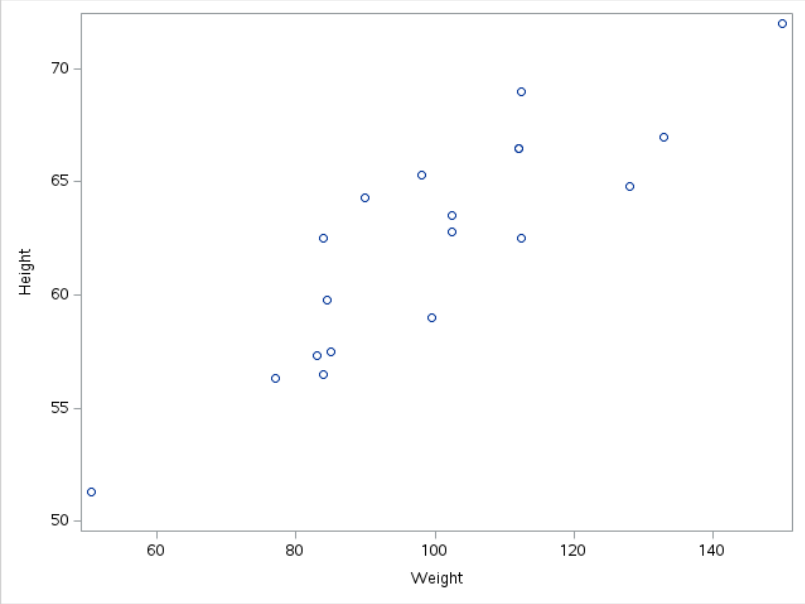
Let's start from the simplest plot with non-grouped data. Because only one group exists, complex control methods aren't needed, you can directly apply the MARKERATTRS option.

Example 1 generates a simple scatter plot, without group and style setting.

```
/* EX 1 default symbol (without group and style)*/
proc template;
    define statgraph testSymbols;
        begingraph;
            layout Overlay;
                scatterPlot y=height x=weight / name="symbols";
            endlayout;
        endgraph;
    end;
run;
ods _all_ close;
ods html path=odsout file="ex1.html";
ods listing gpath="&path.";
ods graphics on / reset imagename="ex1" ;

proc sgrender data=class template=testSymbols;
run;
```

Example 1 default symbol (without group and style)



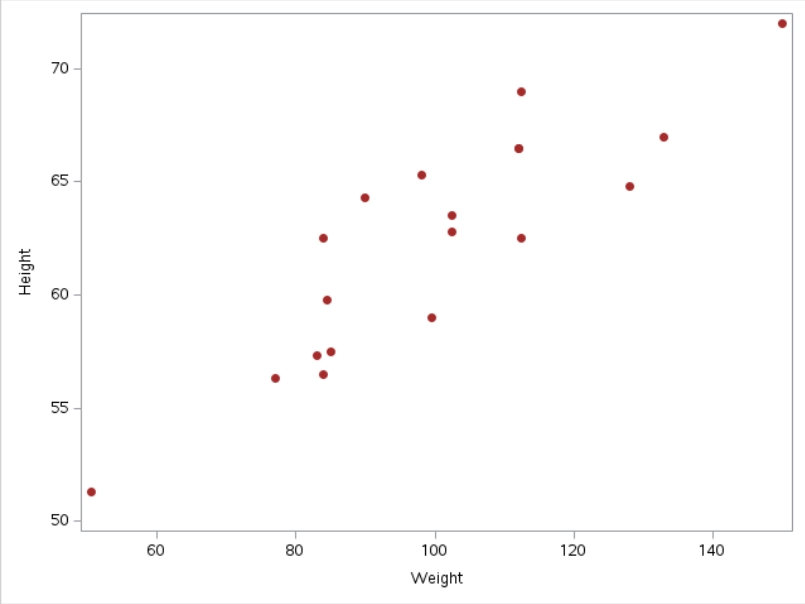
Output for Example 1

Example 2 use option MARKERATTR in plot statement SCATTERPLOT to change the symbol and color..

```
/* EX 2 non-grouped data: markerattr symbol */
proc template;
    define statgraph testSymbols;
        begingraph;
            layout Overlay;
                scatterPlot y=height x=weight /
markerattrs=(symbol=CircleFilled color=brown) name="symbols";
            endlayout;
        endgraph;
    end;
run;
ods _all_ close;
ods html path=odsout file="ex2.html";
ods listing gpath="&path.";
ods graphics on / reset imagename="ex2" ;

proc sgrender data=class template=testSymbols;
run;
```

Example 2 non-grouped data (markerattr)



Output for Example 2

Below are markers and their names supported in SAS:

GROUPED DATA

When working with grouped data, assigning distinct visual attributes—such as marker shape, color, or line style—to each group, is one of the most common tasks.

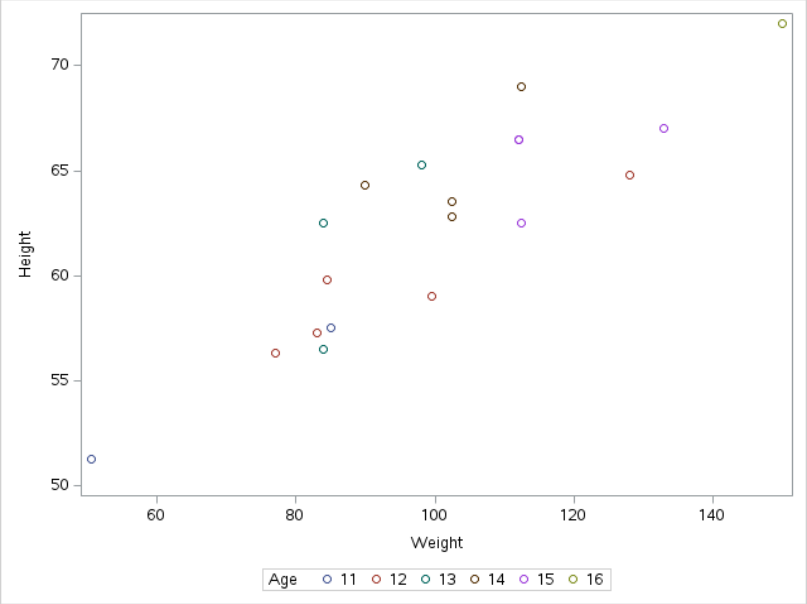
The following examples demonstrate various customization levels for grouped symbols. These examples progress from simple default behavior to explicit group-specific styling using attribute maps, showcasing how each approach influences the appearance and interpretability of the plot.

Example 3 generates a scatter plot grouped by variable AGE in dataset CLASS, using default symbols from the active ODS style.

```
/* EX 3 grouped data: default symbol in default style */
proc template;
    define statgraph testSymbols;
        begingraph;
            layout Overlay;
                scatterPlot y=height x=weight / group=age name="symbols";
                discretelegend "symbols" / title="Age";
            endlayout;
        endgraph;
    end;
run;
ods _all_ close;
ods html path=odsout file="ex3.html";
ods listing gpath="&path.";
ods graphics on / reset imagename="ex3" ;

proc sgrender data=class template=testSymbols;
run;
```

Example 3 grouped data (default symbol in default style)



Output for Example 3

ODS Style

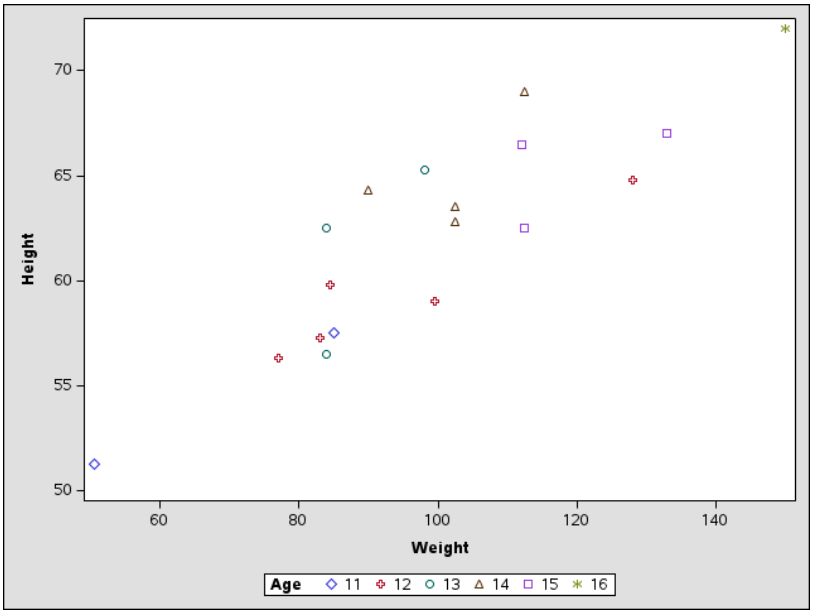
Example 4 create a custom style MYDEFAULT from STYLES.DEFAULT, and use this custom style in ODS.

```
/* EX 4 grouped data: user-defined style */
proc template;
    /* Create custom style MYDEFAULT from STYLES.DEFAULT. */
    define style MyDefault;
        parent=Styles.Default;
        style GraphData1 from GraphData1 / MarkerSymbol="DIAMOND";
        style GraphData2 from GraphData2 / MarkerSymbol="CROSS";
        style GraphData3 from GraphData3 / MarkerSymbol="CIRCLE";
    end;

    /* Create the plot template. */
    define statgraph testSymbols;
        begingraph;
            layout Overlay;
                scatterPlot y=height x=weight / group=age name="symbols";
                discretelegend "symbols" / title="Age";
            endlayout;
        endgraph;
    end;
run;
/* Generate the plot. */
ods _all_ close;
ods html path=odsout file="ex4.html" style=MyDefault;
ods listing gpath="&path.";
ods graphics on / reset imagename="ex4" ;

proc sgrender data=class template=testSymbols;
run;
```

Example 4 grouped data (user-defined style)



Output for Example 4

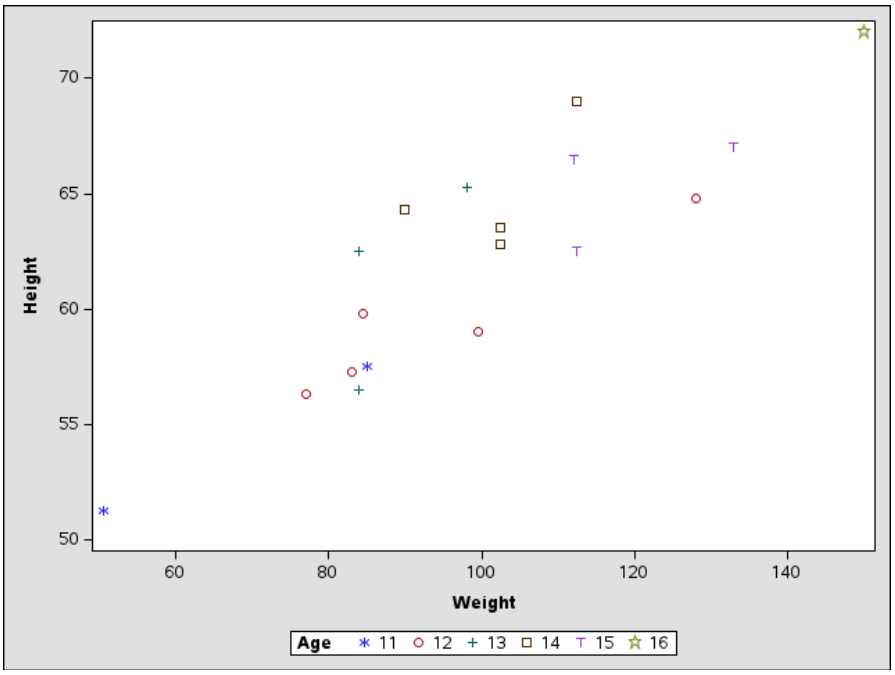
BEGINGRAPH Statement

Example 5 use option DATASYMBOLS and DATACOLORS of BEGEINGRAPH statement to override the default symbol and color in Styles.Default. Be sure to specify the style in ODS, or it might not show symbols as you intended.

```
/* EX 5 grouped data: begingraph option*/
proc template;
    define statgraph testSymbols;
        begingraph / datasymbols=(asterisk circle plus square tack star x y)
datacolors=(black blue brown yellow red pink);
            layout Overlay;
                scatterPlot y=height x=weight / group=age name="symbols";
                discretelegend "symbols" / title="Age";
            endlayout;
        endgraph;
    end;
run;
ods _all_ close;
ods html path=odsout file="ex5.html" style=Styles.Default;
ods listing gpath="&path.";
ods graphics on / reset imagename="ex5" ;

proc sgrender data=class template=testSymbols;
run;
```

Example 5 grouped data (begingraph statement)



Output for Example 5

Attribute Map

There are 2 ways to define and use the attribute map:

1. Define the attribute map in DISCRETEATTRMAP block, and link the map and variables using DISCRETEATTRVAR
2. Save the attribute map in a SAS dataset, then link the map and variables in proc sgrender using dattrmap and dattrvar.

DISCRETEATTRMAP and DISCRETEATTRVAR

Example 6 generates a scatter plot grouped by variable SEX in dataset CLASS. We define the attribute map in DISCRETEATTRMAP block, and link variables by DISCRETEATTRVAR.

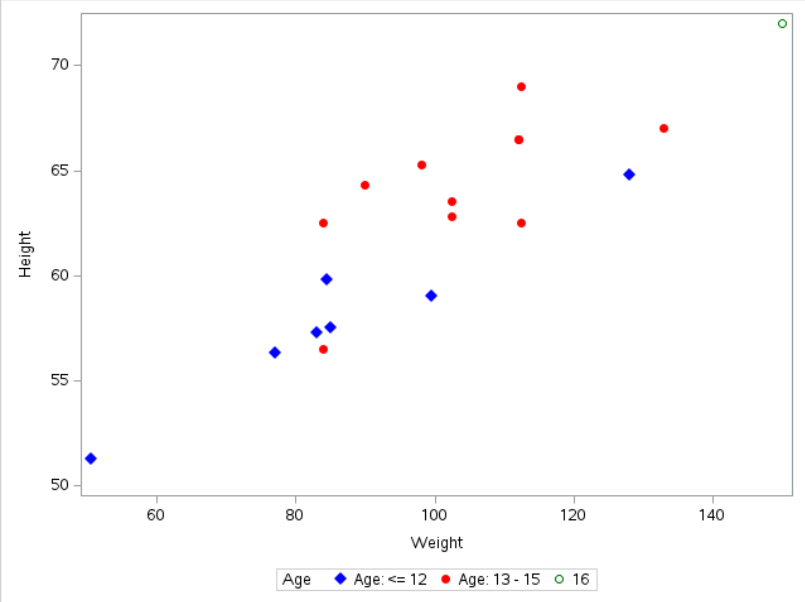
For statement DISCRETEATTRVAR, argument ATTRVAR specifies a SAS name for this association between the attribute map and the group variable; argument VAR specifies an variable to be associated with an attribute map at run time. In a word, VAR should be the group variable in input dataset used in render, ATTRVAR is a new variable, and it should be used in plot statement GROUP option

```
/* EX 6 grouped data: DISCRETEATTRMAP block in template */
proc template;
    /* Create the plot template. */
    define statgraph testSymbols;
        begingraph;
            discreteattrmap name="symbols" / ignorecase=true;
                value "m" / markerattrs=(color=blue symbol=diamondfilled) ;
                value "f" / markerattrs=(color=red  symbol=circlefilled) ;
            enddiscreteattrmap;
            discreteattrvar attrvar=groupmarkers var=sex attrmap="symbols";
            layout Overlay;
                scatterPlot y=height x=weight / group=groupmarkers
name="scatter";
                discretelegend "scatter" / title="Sex";
            endlayout;
```

```
        endgraph;
    end;
run;
/* Generate the plot. */
ods _all_ close;
ods html path=odsout file="ex6.html";
ods listing gpath="&path.";
ods graphics on / reset imagename="ex6" ;

proc sgrender data=class template=testSymbols;
run;
```

Example 6 grouped data (DISCRETEATTRMAP block in template)



Output for Example 6

Example 7 shows how to use DISCRETEATTRMAP with format: the DISCRETEATTRMAP is created with the formatted value of AGE and 'other' to suit the cases that not in the specified values, DISCRETEATTRVAR use variable AGE, and variable AGE is formatted with format agegrp in proc render.

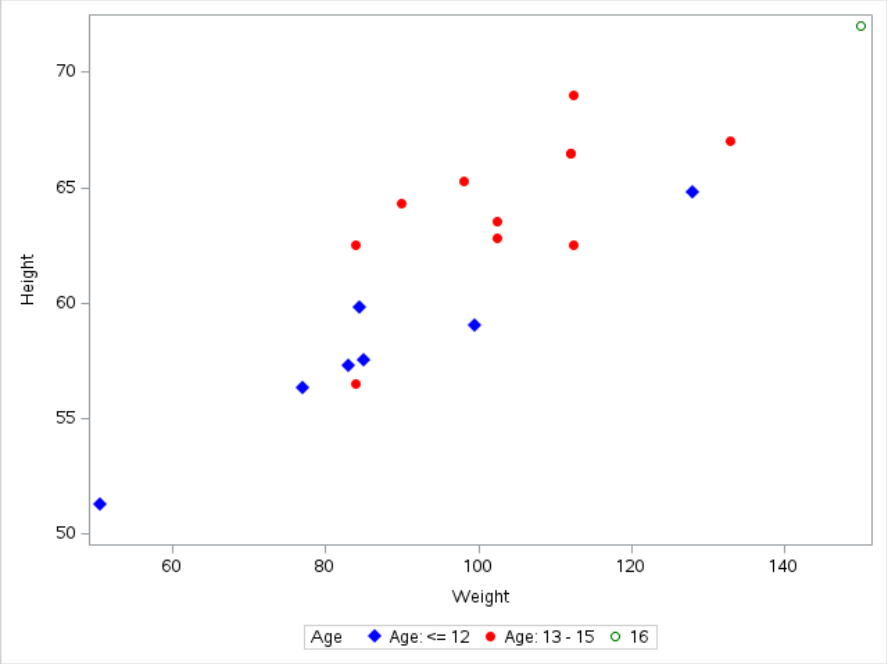
```
/* EX 7 grouped data: DISCRETEATTRMAP block in template, format the age
group */
proc format;
    value agegrp
        11-12 = 'Age: <= 12'
        13-15 = 'Age: 13 - 15'
    ;
run;

proc template;
    /* Create the plot template. */
    define statgraph testSymbols;
        begingraph;
            discreteattrmap name="symbols" / ignorecase=true;
                value "Age: <= 12" / markerattrs=(color=blue
symbol=diamondfilled) ;
```

```
                value "Age: 13 - 15" / markerattrs=(color=red
symbol=circlefilled) ;
                value other / markerattrs=(color=green  symbol=circle) ;
            enddiscreteattrmap;
            discreteattrvar attrvar=groupmarkers var=age attrmap="symbols";
            layout Overlay;
                scatterPlot y=height x=weight / group=groupmarkers
name="scatter";
                discretelegend "scatter" / title="Age";
            endlayout;
        endgraph;
    end;
run;
/* Generate the plot. */
ods _all_ close;
ods html path=odsout file="ex7.html";
ods listing gpath="&path.";
ods graphics on / reset imagename="ex7" ;

proc sgrender data=class template=testSymbols;
format age agegrp.;
run;
```

Example 7 grouped data (DISCRETEATTRMAP block in template, format the age group)



Output for Example 7

Attribute Map Dataset

In SAS 9.4, you can specify the attribute mapping information for a discrete attribute map in a SAS data set instead of in a DISCRETEATTRMAP block in the template. Each observation in the data set maps a discrete value to one or more specific graphical properties. For details of Columns That Are Required in Each Observation, you can read the Defining a Discrete Attribute Map Part in Graph Template Language Reference.

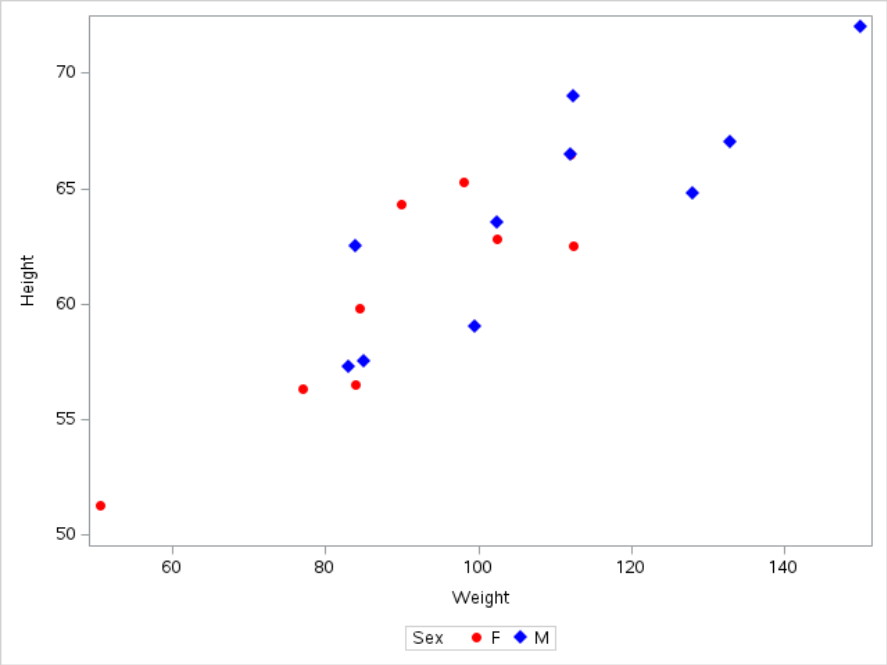
To reference the attribute map, variable used in option group of plot statement should also be used in dattrvar statement in proc sgrender, and the attribute map dataset should be referenced in argument dattrmap in proc sgrender.

```
/* EX 8 grouped data: attrmap dataset */
/* Create the attribute map data set */
data attrds;
    input ID $1-7 VALUE $9 MARKERSYMBOL $11-23 MARKERCOLOR $25-30;
    datalines;
symbols M diamondfilled blue
symbols F circlefilled  red
;
run;

proc template;
    /* Create the plot template. */
    define statgraph testSymbols;
        begingraph;
            layout Overlay;
                scatterPlot y=height x=weight / group=sex name="scatter";
                discretelegend "scatter" / title="Sex";
            endlayout;
        endgraph;
    end;
run;
/* Generate the plot. */
ods _all_ close;
ods html path=odsout file="ex8.html";
ods listing gpath="&path.";
ods graphics on / reset imagename="ex8" ;

proc sgrender data=class template=testSymbols dattrmap=attrds;
dattrvar sex="symbols";
run;
```

Example 8 grouped data (attrmap dataset)



Output for Example 8

SPECIAL SYMBOLS

SAS can only provide the following symbols. If these SAS symbols can not meet your need, there are 2 methods to introduce special symbols in your graph. SYMBOLCHAR and SYMBOLIMAGE.

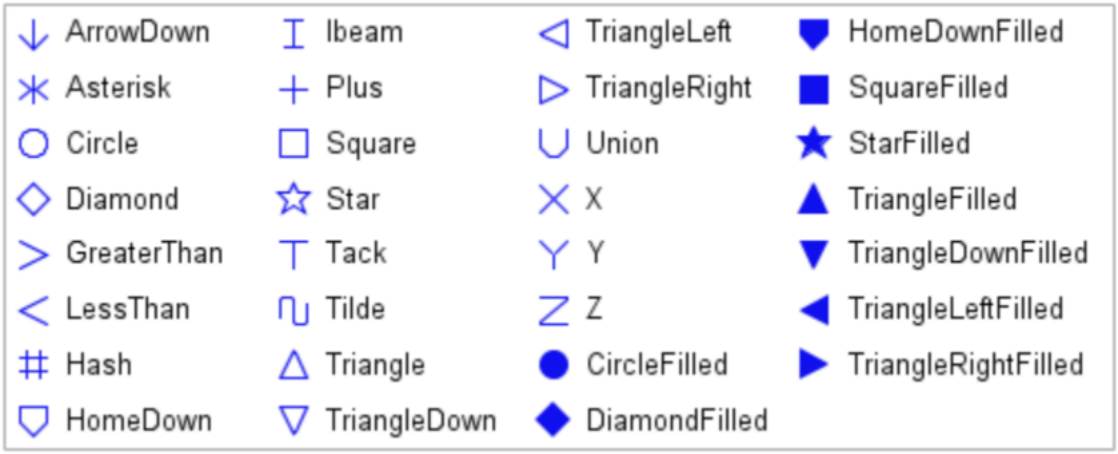


Figure 1 SAS supported markers

SYMBOLCHAR defines a marker symbol using a Unicode character so that the symbol can be referenced in other statements. You can define the name of the Unicode character, also the font and weight, even the horizontal / vertical offset and angle of rotation of it.

The font influences the shape of symbol, choose the right font you want. You can find fonts supplied by SAS in website
https://documentation.sas.com/doc/en/pgmsascdc/9.4_3.5/uprint/n0oqoymgqj2ahrn1l35h2t4tl2k0.htm .

SYMBOLIMAGE defines a marker symbol using an external image file so that the image can be referenced in other statements.

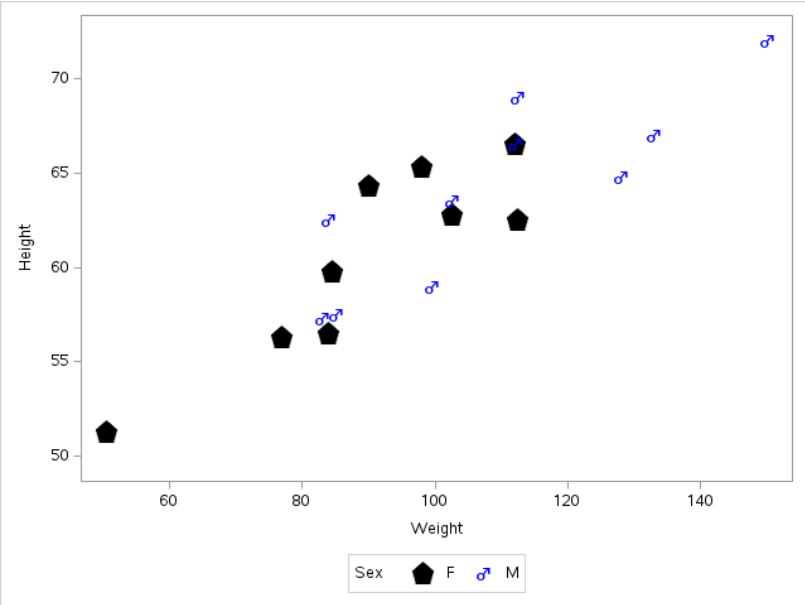
In general, SYMBOLCHAR is more recommended as color and size of symbols created by it can be more flexible (related MARKERATTRS can change some display attributes of the symbol), while the color and size of symbols created by SYMBOLIMAGE is fixed.

```
/* EX 9 SYMBOLCHAR and SYMBOLIMAGE*/
proc template;
    /* Create the plot template. */
    define statgraph testSymbols;
        begingraph;
            symbolchar name=male char="2642"x / textattrs=(family='Noto Sans
KR Regular' weight=bold);
            symbolimage name=female image="&path./pentagon.png";
            discreteattrmap name="symbols" / ignorecase=true;
                value "m" / markerattrs=(color=blue symbol=male size=5pt) ;
                value "f" / markerattrs=(color=red  symbol=female) ;
            enddiscreteattrmap;
            discreteattrvar attrvar=groupmarkers var=sex attrmap="symbols";

            layout Overlay;
                scatterPlot y=height x=weight / group=groupmarkers
name="scatter" markerattrs=(size=15pt);
                discretelegend "scatter" / title="Sex";
            endlayout;
        endgraph;
    end;
run;
/* Generate the plot. */
ods _all_ close;
ods html path=odsout file="ex9.html";
ods listing gpath="&path.";
ods graphics on / reset imagename="ex9" ;

proc sgrender data=class template=testSymbols;
run;
```

Example 9 SYMBOLCHAR and SYMBOLIMAGE



Output for Example 9

CONCLUSION

Controlling symbol appearance in SAS Graph Template Language (GTL) is a powerful way to enhance the clarity, consistency, and effectiveness of your visualizations. By understanding the different levels of symbol control—ranging from global ODS styles to granular plot-level settings—you can tailor your graphics to meet both technical and aesthetic goals.

You've seen how to:

- Apply default styles for quick and consistent output
- Use DATASYMBOLS/DATACOLORS in BEGINGRAPH to customize markers across a template
- Define attribute maps to ensure group values are always represented consistently
- Override all other styling with MARKERATTRS in plot statement for full visual precision
- go beyond standard markers by using Unicode characters or external images to define custom symbols.

As you build and refine your SAS plots, remember that thoughtful symbol control doesn't just improve aesthetics—it strengthens the way your data tells its story.

REFERENCES

SAS Institute Inc. *SAS® 9.4 Graph Template Language: Reference, Fifth Edition*

Abhinav Srivastva, 2017, "Special Symbols in Graphs: Multiple Solutions", https://pharmasug.org/proceedings/2017/TT/PharmaSUG-2017-TT05.pdf

ACKNOWLEDGMENTS

I would like to express my very great appreciation to Yan Qiao for her valuable and constructive suggestions during the planning and generating of this paper.

RECOMMENDED READING

- *SAS® 9.4 Graph Template Language: Reference*

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Lin Jiang
BeOne, Inc.
13051616639
lin.jiang@beigene.com