# Chapter 1

# Why Do I Need to Write Code?

# Introduction

This introductory chapter looks at what SAS is and how SAS Enterprise Guide fits into the overall SAS product set. It then explores with a few examples some of the effects that can be achieved by adding small amounts of code to SAS Enterprise Guide tasks.

This book assumes a good working knowledge of SAS Enterprise Guide and familiarity with basic SAS concepts such as tables, libraries, naming standards, SAS formats, and SAS functions. An extremely good introduction to SAS Enterprise Guide is *The Little SAS Book for Enterprise Guide 4.2* by Susan Slaughter and Lora Delwiche. Alternatively, you can use the built-in SAS Enterprise Guide Help Tutorial, SAS training, or SAS Enterprise Guide online documentation at **`support.sas.com/eguide`** to review any areas you are not familiar with.

This book does not assume any prior knowledge of SAS coding.

# A Brief History of SAS

Let's look first at what we mean when we talk about SAS.

## SAS: A Complete Programming Language

The name "SAS" is used to refer to SAS the company, the core products offered by the company, and the underlying programming language that lets all the SAS products do their work.

The SAS programming language and core products were created at North Carolina State University in the early 1970s to help with analyzing agricultural research data. In 1976, SAS Institute Inc. was formed to bring the language and products to the wider world.

Computing in the early seventies was very different from the world of computers we know today. To get any serious processing power required a mainframe computer, a physically large machine requiring strict environmental control and a team of operators working in shifts to keep it serviced and running.

There was no spare capacity for the niceties of providing a friendly usable interface to make the life of computer users easier.

If you wanted to make a computer do anything, you had to learn the commands it understood and a programming language to make it work the way you wanted.

At least by then most programming languages were based on something that could be recognized as English, making them easier to learn and follow. The SAS programming language was a 4th Generation Programming Language (4GL), though it would be another six years before that term was first used. A 4GL allowed basic results to be achieved very easily, but also allowed those with greater knowledge of the language to make it do exactly what they needed.

**4GL? Was there a 5GL?**
Each generation of programming languages was one step further removed from zeros and ones that are all a computer understands, and one step closer to a human viewpoint. Though a fifth generation was often heralded, it never really arrived, partly because Graphical User Interfaces (GUI) came along instead.

Working with these computers was not easy. Punched cards were still in use (in fact, the SAS language statement CARDS still exists) but terminals were starting to become more common. These were low-resolution, monochrome, text-only devices that allowed users to feed commands and programs into a computer and see the results. If printed results were needed, the output was black and white monospace text only.

SAS started the year before the Apple II was launched; the first Macintosh was eight years in the future. IBM PCs would not be along for five years, and the first Microsoft Windows operating system was not available until two years after that. A product such as SAS Enterprise Guide was not even possible back then.

Between then and now computers and SAS have undergone massive changes.

- The power of computers escalated beyond all expectations, allowing massively complex programs to run on even a small computer. This allowed SAS to present even more challenging statistical techniques through their simplified 4GL.
- Manufacturers of serious computers gradually standardized on the UNIX operating system. SAS ported its products to run there too.

- The IBM PC and its clones became the main desktop tool, also making Microsoft Windows the de facto standard. Again, massive increases in power, along with the addition of color screens, pointing devices, and versatile printers, moved the user interface and presentation to the forefront. SAS produced tools to keep up with these developments—first an interactive workbench for programmers, and then SAS Enterprise Guide.
- The World Wide Web changed the usage of computers back towards a more centralized client/server environment, this time with the client available through a standardized browser or via a truly machine-independent Java language. Once again, SAS embraced the change with its SAS Intelligence Platform.

So SAS started from a strong position with its programming language and has evolved as computers evolved. Everything you have ever been able to do with SAS can still be done via the programming language, but that is a lot to learn and even more to master. A new approach was needed.

## SAS Enterprise Guide: A Graphical User Interface for SAS

The key problem for a new SAS user before the introduction of SAS Enterprise Guide was the time it took to become proficient in using the programming language. A few attempts were made to provide a better interface, but they were all a little cumbersome and tended to be focused on a specific area of SAS. For example, there was (and still is) a very unusual interface to the SAS reporting routines discussed in the "One-Stop Reporting" section of Chapter 7.

Some interface improvements, such as that for SAS Enterprise Miner, were much more successful, but were still aimed at achieving specific objectives—data mining in this case.

The key tool in opening up the power of SAS to a wider audience of power users is SAS Enterprise Guide—currently on release 4.2.

SAS Enterprise Guide is a Microsoft Windows tool developed using Microsoft .NET. Although the client can run only on Windows, it can still utilize SAS running on any of the operating systems it supports. Using .NET as a development tool means that SAS Enterprise Guide can be extended to support new tasks, either developed by SAS, by third-parties, or by users wanting to tailor SAS Enterprise Guide to their own needs.

As described earlier, SAS is a comprehensive programming language that can be used in a huge number of ways. Although it is true that

**Power Users?**
SAS Enterprise Guide is a tool for users who understand data structures and have the time and skills to develop sophisticated models and reports. More recently, SAS also introduced their SAS®9 software based on Web technologies and targeted towards business users.

some of the language capabilities are so rooted in the past that you will probably never want to use them (look at the EXPLODE procedure mentioned in Appendix B), most are still very relevant. SAS Enterprise Guide was designed to give easy access to some of the most frequently used options. Fortunately, after you understand the way SAS Enterprise Guide works, you can learn some programming techniques that make use of this hidden functionality.

## So How Does SAS Enterprise Guide Work?

SAS Enterprise Guide handles two things for the user:

### SAS Enterprise Guide Interacts Directly with Microsoft Windows

SAS Enterprise Guide works directly with Windows to deal with tasks that are purely related to Windows, such as using the Windows scheduler and prompting for run-time parameters. As a SAS programmer, there is little you can do to interact with activities that are specific to Windows. It is possible to develop your own add-ins, but that would be a subject for another book and another programming language.

### SAS Enterprise Guide Interacts with SAS

In virtually all cases, SAS Enterprise Guide generates SAS code for you, submits it to a SAS server for processing, and then updates the SAS Enterprise Guide workspace with the results (see Figures 1.1 through 1.4). The SAS server can be SAS software running locally on your machine, or SAS software running on another machine that you have access to. The other machine can be any platform that SAS supports—most likely a Windows or UNIX server, but possibly a mainframe system.

**Figure 1.1** Use tasks to define what you want

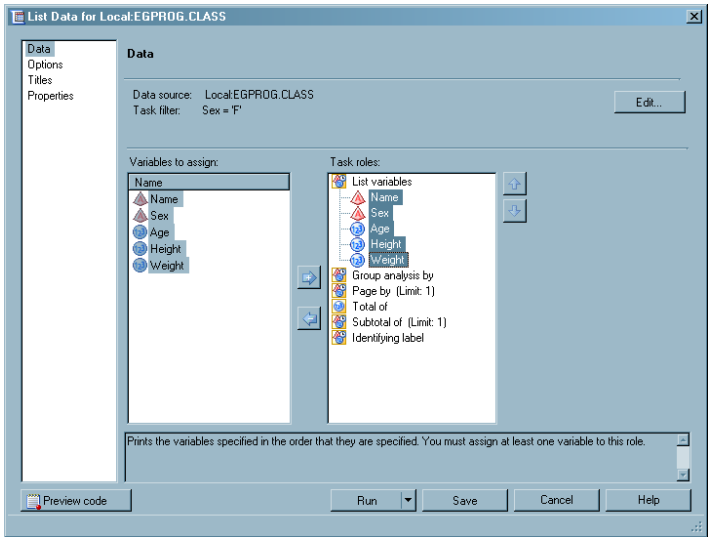



**Figure 1.2** The SAS code is written for you

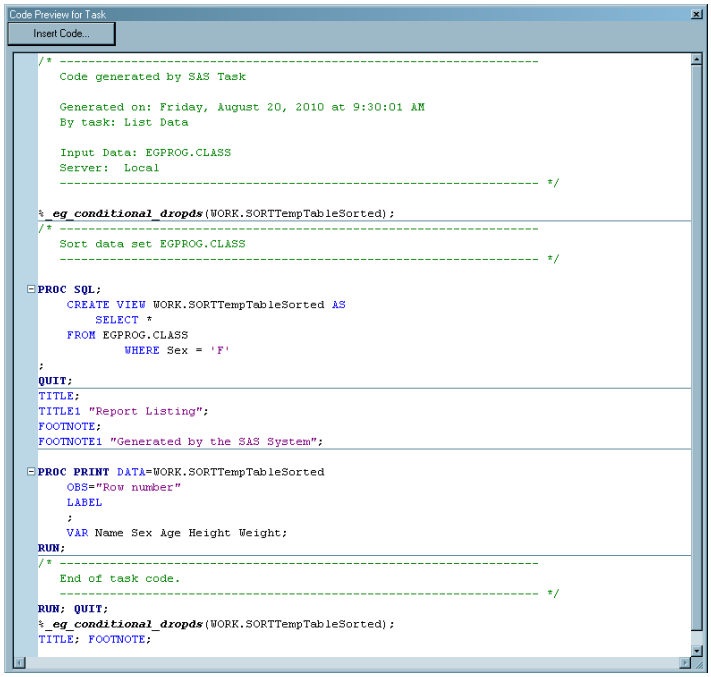


**Figure 1.3** SAS works out the answer …

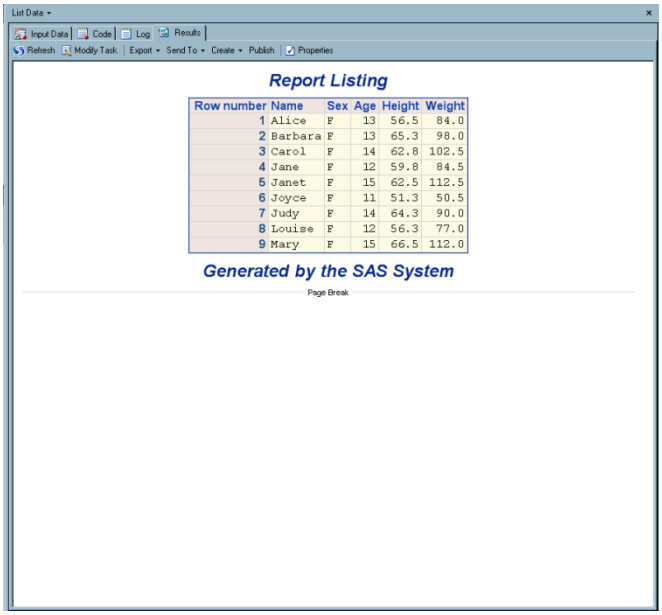
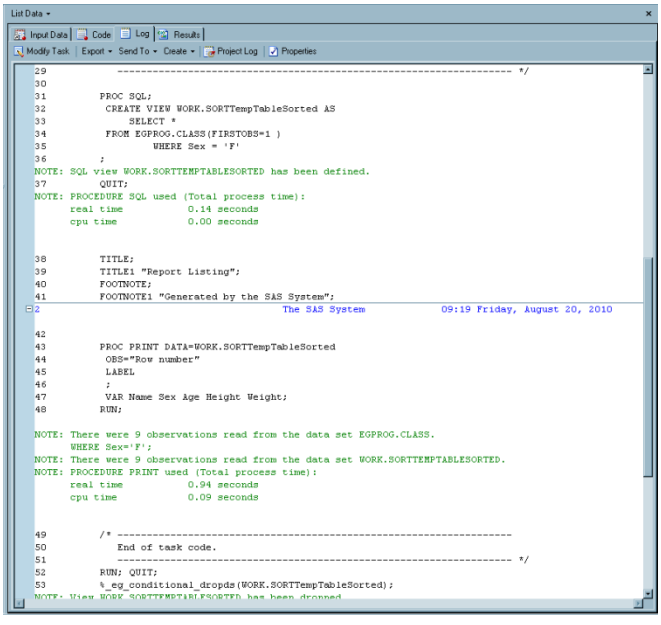



**Figure 1.4** … and the log tells you how it was done



When SAS Enterprise Guide was developed, it was recognized that it would not be practical to make everything that SAS can do available through tasks. This would have made the SAS Enterprise Guide application extremely large and much harder to use. So decisions were made about what to put in and what to leave out. Just to make sure that everything was still possible, the SAS Enterprise Guide developers included several ways for people to insert SAS code into their work.

So people were introduced to SAS Enterprise Guide as the easy way into SAS. When they found out that they needed to get more out of SAS, they were sent on to SAS programming courses. What this book seeks to do is introduce the parts of SAS code that complement SAS Enterprise Guide, without re-inventing the things that SAS Enterprise Guide does well.

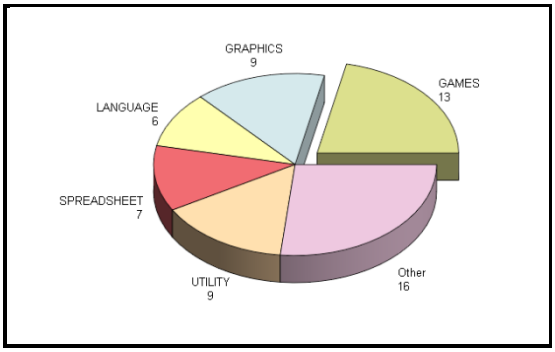
In the rest of this chapter we will look at just a few examples of the ways in which you can extend SAS Enterprise Guide by the addition of SAS code. You will see many more examples as you work through the book.

## Examples of Enhancing the Generated Code

As previously mentioned, the tasks in SAS Enterprise Guide generate SAS code. The majority of the tasks allow users to insert extra code at predetermined points in the generated code, enhancing the functionality of the tasks without affecting the user's ability to manipulate the task through the user interface. We will look in detail at the generated code and methods of inserting code in Chapter 2. Let's have a quick look now at some of the things you can do by inserting code into a task.

Look at the pie chart in Figure 1.5. Can you see something there that you cannot do using the point-and-click Pie Chart task or Pie Chart wizard?

**Figure 1.5** Pie chart with exploded slice




**Trivia**
It is thought that Florence Nightingale was the person who first came up with the idea of showing proportions as segments of a pie. She used it to illustrate mortality rates from poor sanitation during the Crimean war.

Notice the way that the GAMES slice is exploded out from the rest of the pie. This is a great way to emphasize a specific point. The SAS Enterprise Guide task does not enable you to do this, but SAS code does. All it takes is the insertion of two words into your task.

We will look at ways to improve graphical output in Chapter 8, but there are many examples throughout the book of other hidden bits of SAS functionality that you can use, often with only a few words of code inserted.

What about this simple-looking listing report?

**Figure 1.6** List data with conditional highlighting

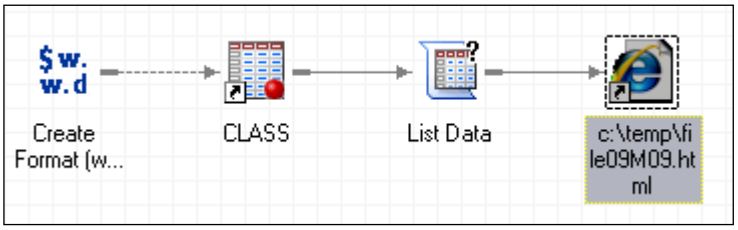
**Listing of students aged 13 or older**

| Name | Sex | Age | Height | Weight |
|---|---|---|---|---|
| Alfred | M | 14 | 69.0 | 112.5 |
| Alice | F | 13 | 56.5 | 84.0 |
| Barbara | F | 13 | 65.3 | 98.0 |
| Carol | F | 14 | 62.8 | 102.5 |
| Henry | M | 14 | 63.5 | 102.5 |
| Janet | F | 15 | 62.5 | 112.5 |
| Jeffrey | M | 13 | 62.5 | 84.0 |
| Judy | F | 14 | 64.3 | 90.0 |

The first thing you might notice is that the highlighting on the Weight column differs from cell to cell. The color of the background is actually dependent on the value in the cell; in this case it is set to have a red background only if the weight is greater than 100. This technique of conditional highlighting, sometimes known as *trafficlighting*, can be used to focus people's attention on certain areas so that they are less likely to overlook important information. This effect is slightly more complicated to achieve than the previous one, but still quite straightforward. We will see how we can do this in Chapter 7.

There are two other techniques used in the production of this report that only become apparent if you look at the process flow used to create the report.

**Figure 1.7** Named output in a process flow




The Create Format task at the start of the process flow is where the rules are defined for the conditional highlighting.

Look back at the report and you will see that it is only for students aged 13 or over. Look at the process flow and you will see a question mark on the List Data icon. The question mark indicates that you will be prompted for information at run time. In this case, you will be prompted for the age limit. The **`Edit`** button in the Data section of a task enables you to do some filtering of your data. It will not allow you to do complex filtering of the sort that can be done in the Query Builder. With a bit of persuasion, it will enable you to prompt for character values, but it will not allow you to prompt for numeric vales as in this example. You can get these extra capabilities by adding a single extra line of code into almost any SAS Enterprise Guide task. Although it is not significant on a small data set like this, on a large data set the potential savings from not reading the data twice are enormous.

Finally, look at the name of the HTML document produced. Exporting a document in SAS Enterprise Guide enables you to completely overwrite the document each time, or create a completely new version each time by adding a date-and-time stamp to the name. If you look closely, you will see that the name of this document includes only the year and month. Inserting two lines of code increases the flexibility you have when naming your output files. As with the data extraction, you can use this technique in any task that creates a report.

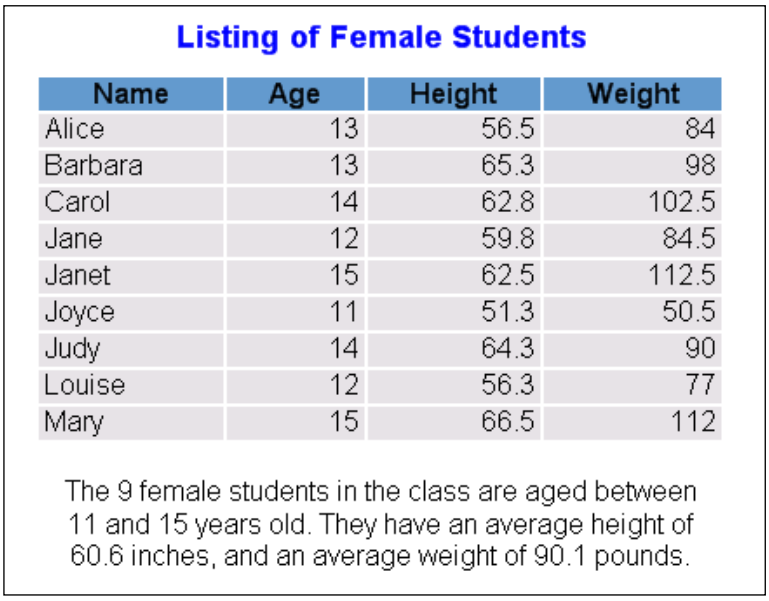
We will look in more detail at both of these techniques and many more in Chapter 3.

## Examples of Writing Your Own Code

In the previous section, we looked at a few of the things you can achieve by adding small pieces of code to the code that is generated by SAS Enterprise Guide. There might be occasions when you need to develop some code of your own in a Program node. In Chapter 2 we will look at the Program node in general, but again here we will just look at a couple of examples of why you might want to do this.

First look at the summary section of this report.

**Figure 1.8** Advanced reporting capabilities



**Listing of Female Students**

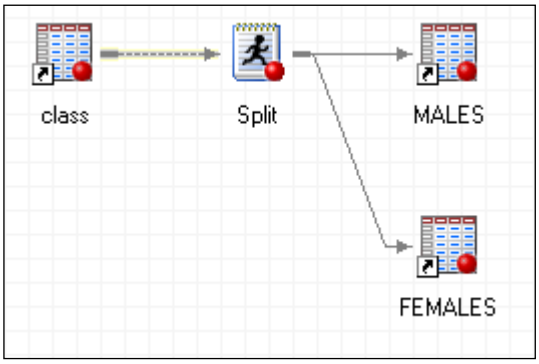
| Name | Age | Height | Weight |
|---|---|---|---|
| Alice | 13 | 56.5 | 84 |
| Barbara | 13 | 65.3 | 98 |
| Carol | 14 | 62.8 | 102.5 |
| Jane | 12 | 59.8 | 84.5 |
| Janet | 15 | 62.5 | 112.5 |
| Joyce | 11 | 51.3 | 50.5 |
| Judy | 14 | 64.3 | 90 |
| Louise | 12 | 56.3 | 77 |
| Mary | 15 | 66.5 | 112 |

The 9 female students in the class are aged between 11 and 15 years old. They have an average height of 60.6 inches, and an average weight of 90.1 pounds.

The narrative description below the table contains various statistics extracted from the table itself. This is an example of the advanced reporting available through SAS that is not fully supported by a SAS Enterprise Guide task. We will look in detail at this in Chapter 7.

Just to finish off, the following process flow shows an example of the different sorts of data manipulation that can be achieved in SAS.

**Figure 1.9** Data being split




In this example, the CLASS data set is being split into MALES and FEMALES. This can be achieved in SAS Enterprise Guide by using two standard Query Builder tasks, but again using a few lines of code—five in this case—saves reading the data twice.

# Summary

SAS started life long before the current fashion for Graphical User Interfaces and can do far more than is available through the SAS Enterprise Guide task interfaces.

By understanding the SAS Programming language, SAS Enterprise Guide users can access the full power of SAS while still maintaining the ease of use of SAS Enterprise Guide.

Occasionally it might be necessary to develop your own code. Even then, a relatively small number of lines can give significant gains in efficiency or in the results achieved.