Generating Component Documentation with SAS/AF® Software

Marty Tomasi, SAS Institute Inc., Cary, NC
Chris Barrett, SAS Institute Inc., Cary, NC

ABSTRACT

As applications developers continue to adopt object-oriented programming techniques in their applications, documenting those objects becomes an increasingly important task. SAS/AF software includes GenDoc, a documentation utility that enables you to generate HTML files that document class, interface, resource, and frame entries. You can view the HTML files through most standard Web browsers. This paper describes how to use GenDoc to create HTML pages that document the classes, interfaces, and frames you develop. This paper also presents strategies and processes for generating documentation for entire resources and catalogs.

INTRODUCTION

One of the most important aspects of object-oriented development is re-use. But as organizations adopt object-oriented development and implement components, the ability to re-use objects can be limited if it is not clear what an object does and how it can be used. It is important, then, that component developers document their objects to provide other developers with information that facilitates component re-use.

With Version 8, SAS/AF software includes an experimental documentation utility called GenDoc that enables component developers to generate HTML files that document their classes, interfaces, resources, and frames. You can use it to generate documentation for entries within a catalog. This documentation can provide assistance to developers who use the components that you create.

For complete information on the GenDoc documentation utility, see the SAS/AF online help or *SAS Guide to Applications Development*.

GENERATING COMPONENT DOCUMENTATION

The GenDoc utility in SAS/AF software can be accessed several ways. To create documentation using the GenDoc dialog box (see Figure 1 below), you can:

- select the **Save As HTML...** item from the File menu in the Class Editor, Interface Editor, Resource Editor, or Build window
- enter the command

```
afa c=sashelp.aftools.scl2html.frame
```

to invoke the GenDoc dialog box. You can use the interface to select the entry and entry type you want to document, then specify the output directory and click **Start**.

You can also use the GenDoc utility from within your SCL programs or in batch processing.

- Use the experimental SCL function CreateDoc: `CreateDoc('catentry-name', 'out-dir');` where *catentry-name* is the name of the class, interface, frame, resource, or catalog you want to document, and *out-dir* is the local directory to which you want to direct the HTML output.
- Use the SCL Static Analyzer and add the HTMLDIR= option to the CROSSREF statement. For example:

```
proc build batch;
     crossref proj=lib.cat HTMLDIR='out-dir';
  run;
```

where *out-dir* is the local directory to which you want to direct the HTML output.

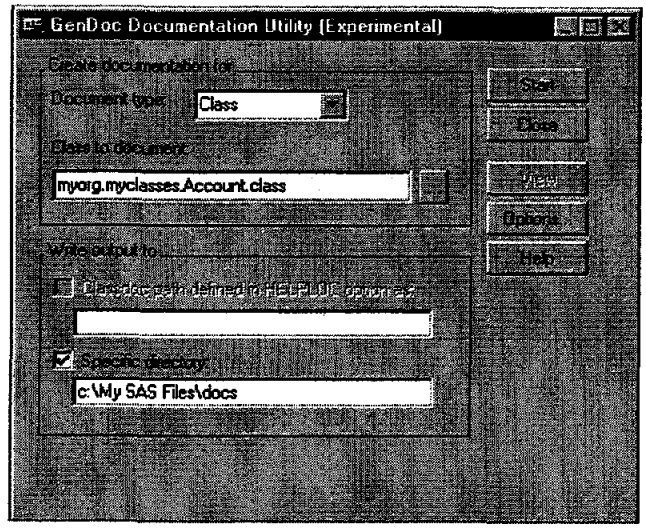

***Figure 1.** The GenDoc dialog box*

DOCUMENTING CLASSES AND INTERFACES

You can generate documentation for a class directly from within the SAS/AF Class Editor. In the Class Editor, select **File > Save As HTML...** to start GenDoc. Specify an output directory or use the default HELPLOC path (see "GenDoc and the HELPLOC Option," below), then click **Start** to create an HTML document for the current class.

For classes, GenDoc generates an HTML file that contains parent and ancestor information and all properties (attributes, methods, events, event handlers, and supported or required interfaces) defined for the class. The document is output to a file named `classname.htm`. See Figure 2 below for a screen shot of a browser that is displaying a GenDoc-generated document.

To generate documentation for an interface, open the Interface Editor, then select **File > Save As HTML...** to start GenDoc. Specify an output directory (or use the default HELPLOC path), then click **Start** to create an HTML document for the current interface.

The documentation for interfaces is very similar to the documentation for classes. With an interface, GenDoc generates an HTML file that contains the parent interface information and all methods defined for the interface. The document is output to a file named `interfacename-intrface.htm`.

Because GenDoc reads the metadata that is associated with a class (or interface), it is important that you include as much information as possible when you define your class. Before you generate documentation, you should make sure that you have included values for the **Description** metadata items for attributes,

methods, method arguments, method return arguments, events, and event handlers. (Note that by default, "attribute-Changed" events automatically receive a description that reads, "Occurs when the *attribute-name* attribute is changed.") Refer to the *SAS Guide to Applications Development* for additional tips on writing meaningful metadata descriptions.

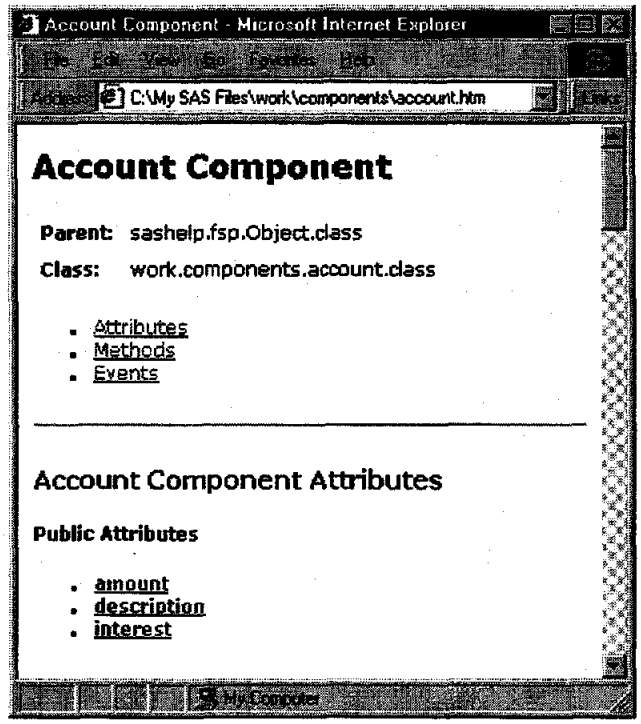

Figure 2. *Class documentation displayed in a Web browser*

ADDING GENDOC COMMENTS TO YOUR SCL CODE

If you create a SAS/AF class using the SCL CLASS/ENDCLASS statements, you can embed additional documentation in the class. The GenDoc utility supports special comments in your SCL code that are delimited by the `/**` and `*/` delimiters. GenDoc processes these comments and includes the information they contain in the generated HTML file.

It is recommended that the SCL entry that contains the definition of the class use a naming convention that concatenates the CLASS entry name and the suffix "class". For example, if you create a class named `salesObject.class` using SCL, you should name the SCL entry `salesObjectclass.scl`. GenDoc only locates comments in SCL entries that contain class definitions and are named using the form <class-entry><suffix>.scl.

You can set the suffix you want to use as part of the file that contains GenDoc comments in the GenDoc Options dialog box. From the GenDoc dialog box, select **Options...**. On the Content tab, select the "Include comments from SCL class file" item and enter the suffix you use. By default, the suffix is **class**.

A GenDoc comment consists of two parts: descriptive text followed by optional tags. All items must be contained within the GenDoc comment delimiters:

```
/**
 *  This is some descriptive text.
 *
 *  @author Rose Budd
 *  @version
 */
```

- The first line must follow the CLASS statement and starts with the begin-comment symbol (`/**`). End the line with a return.
- Subsequent lines begin with a single asterisk (*). They are typically indented an additional space to line up with the asterisks in the first comment. Use at least one space to separate the asterisk from the descriptive text that follows it.
- Insert a blank line between the description and the @ tags as shown above.
- The last line begins with the end-comment (`*/`) indented so that the asterisks line up. Note that the end-comment contains only a single asterisk.
- You can use the @author tag to include author information. Multiple @author tags are permitted.
  Example: `@author Jane Doe`
- You can use the @version tag to indicate version information about the class. If you include the @version on a line by itself, GenDoc inserts the date that the CLASS entry was last modified. You can also add a string of text to indicate a version.
  Example: `@version` returns a date in MM/DD/YYYY format, or `@version 1.0` simply displays "Version 1.0".

It is recommended that you limit your GenDoc comment lines to a width of no more than 80 characters, if possible. If you have more than one paragraph in the comment, separate the paragraphs with the HTML paragraph tag (<p>). For example:

```
/**
 *  The Sales Object class provides a
 *  component that retrieves, calculates,
 *  and stores sales information.
 *  <p>Sales data must be stored in the
 *  saledata library.
 *  @author Rose Budd
 *  @version
 */
```

The <p> tag in the fourth line of comments will insert an HTML paragraph tag into the output. It is also possible to embed other HTML formatting codes, such as tables or images. For example:

```
/**
 *  The Site Objet can correctly extract
 *  information from the following locations:
 *
 *  <table>
 *  <tr>
 *  <th>Location</th><td>Site Code</td>
 *  </tr>
 *  <tr><td>Parma</td><td>OH1007</td></tr>
 *  <tr><td>Euclid</td><td>OH1980</td></tr>
 *  <tr><td>Mayfield</td><td>OH1743</td></tr>
 *  </table>
 *
 *  @author Jim Shorts
 *  @version 2.2
 */
```

This comment embeds a standard table in the generated HTML.

DOCUMENTING FRAMES

The GenDoc utility also enables you to provide documentation for a frame entry. Developers -- especially those who inherit the maintenance of a frame -- often must spend a significant amount of time deciphering the relationships between objects on a frame. GenDoc generates an HTML file that contains frame information and general attributes, the associated resource, and instance information for all visual and nonvisual components on the frame. The documentation also includes reports that list all linked

attributes as well as all model/view connections.
To generate frame documentation, open the frame and select **File > Save As HTML...** to start GenDoc. Specify an output directory, then click **Start** to create an HTML document. The document is output to a file named *framename*`-frame.htm`.

The "component reports" included in the frame documentation list the connections between components. The "Attribute Links" report lists all attributes that send and receive information. The "Model/View" report lists all views and their attached models.

The documentation for frames that make extensive use of legacy objects (that is, those objects that originally shipped with Version 6 SAS/AF) may not include as much detail simply because legacy components do not contain as much metadata information. However, some legacy objects, such as data form and data table, are specifically queried by GenDoc for important information when you document a frame.

DOCUMENTING RESOURCES AND CATALOGS

The output produced by GenDoc for resources or catalogs is a useful way to provide documentation for a project or application. If you have a Web page for your project or site, you can create links to the index files for information on all related classes.

In the Resource Editor, select **File > Save As HTML...** to start GenDoc. Specify an output directory (or use the default HELPLOC path), then click **Start** to create an HTML document for each class and interface included in the resource.

When you generate documentation for a resource, an HTML file is generated for each of the class entries and interface entries included in the resource. An index file that lists and provides links to all of these files is output to *resourcename*`-resource.htm`.

You can also use GenDoc to generate documentation for all parts of a project that are stored in a catalog. Start the GenDoc utility, select **Catalog** as the Type, then enter the two-level name of the SAS catalog you want to document. Click **Start**.

When you generate documentation for a catalog, an HTML file is generated for each class, interface, and frame entry stored in the catalog. An index file that lists and provides links to all of these files is output to *catalogname*`-index.htm`.

GENDOC AND THE HELPLOC OPTION

By default, the documentation generated by GenDoc is stored in a directory that is specified in the HELPLOC option. The HELPLOC option contains one or more paths to online help files through which SAS software will search when help is requested. To work with GenDoc, at least one directory path in the HELPLOC option must end with a **classdoc** subdirectory. For example:

```
C:\SAS\classdoc\
```

SAS/AF must be able to write to this directory. On most platforms, this directory is created under your SASUSER directory when you install SAS software. For example, if your SASUSER directory on Windows is set to `C:\My SAS Files\V8`, the HELPLOC option includes a path to `!sasuser\classdoc`, which resolves to `C:\My SAS Files\V8\classdoc`.

Each entry that you document is output to an HTML file that includes the same name as the entry. Since you cannot have two entries with the same name within a catalog, the GenDoc utility creates subdirectories under the **classdoc** directory for the library and catalog that contain the specified entry. The first subdirectory specifies the library. A second directory is created under the library directory and is named for the catalog containing the entry that you want documented. For example, if you document the class entry `myorg.classes.Account.class`, the output is written to

```
C:\SAS\classdoc\myorg\classes\Account.htm
```

Likewise, the documentation for the B.intrface entry stored in the same catalog would be written to

```
C:\SAS\classdoc\myorg\classes\B-intrface.htm
```

Note the two subdirectories that are created under the classdoc path.

When you document a catalog, the files are output in the same way, and the catalog index file (*catalogname*`-index.htm`) is also stored in the directory named for the catalog.

SPECIFYING AN OUTPUT DIRECTORY

You can also specify a different output directory each time you run the GenDoc utility. Although you specify an output directory, GenDoc actually creates two subdirectories under that directory, following the same rules as defined above for the **classdoc** directory.

If you are using the CreateDoc function or the Static Analyzer with the HTMLDIR= option, you can direct the output to the HELPLOC directory by specifying `'!HELPLOC'` as the output directory. For example, if the HELPLOC option includes the path `C:\My SAS Files\V8\classdoc`, the function

```
CreateDoc('work.a.a.class', '!helploc');
```

creates the documentation for work.a.a.class in the file `C:\My SAS Files\V8\classdoc\work\a\a.htm`.

For more information about the HELPLOC option and how it can be used with your class documentation files -- as well as other application help -- see *SAS Guide to Applications Development*.

DEPLOYING COMPONENT DOCUMENTATION

Several build-time windows in SAS/AF enable you to view class documentation in context with the current active class or property. For example, you can select a class in the Components window, then right-click and select Help on Class from the pop-up menu to view the documentation for that class. The Class Editor, Properties window, and Class Browser provide similar features.

In Version 8, you can make documentation available in the same way for components that you create. After you have used GenDoc to create documentation for your components, you can make that documentation available to users of your components. When a user of the class selects "Help on Class" from a build-time window such as the Class Editor or Components window, SAS searches the HELPLOC path for the appropriate class HTML file.

If your organization maintains a library of SAS/AF classes, you can make the documentation for those classes available for all developers. To provide access to the documentation, you could copy the HTML files to a network location or intranet and have developers add that path to their HELPLOC options. For example, if the documentation is available on an intranet, the HELPLOC path might include:

```
http://myorg.com/mis/docs
```

You can also copy the generated HTML files to a Web server to make the class documentation available to any developer at any time, regardless of whether SAS is running. If you copy the HTML files, be sure to preserve the directory structure that GenDoc creates for the **library-name** and **catalog-name** that are associated with the class, interface, or resource. Also, make sure that each user sets a HELPLOC path that points to the Web server.

In addition, if you specify that you want to include a common style sheet file, that information is output to `gendoc.css` in the root **classdoc** path. If you move the HTML files for your classes, you will also have to move the `gendoc.css` file and maintain the directory hierarchies.

You can also create your own HTML files to document your classes without using GenDoc and make them accessible to the SAS/AF build-time windows. Simply store the files in the same directory structure with the same naming conventions that GenDoc uses.

GENDOC TIPS AND HINTS

While GenDoc creates an index file that contains links to other files when you generate documentation for a resource or catalog, you can create your own HTML file to use as an index if you want a more customized collection. Make sure that any links you make include the appropriate relative path to the files. For example, if you want to add a link from an index page created by GenDoc to a document from a class in another library, the link would use the form

```
<a href="../../lib/cat/classname.htm">
```

SPECIFYING GENDOC OPTIONS

The GenDoc Options window enables you to specify limited content and format options for the HTML files that you generate for classes, frames, and interfaces. The options are saved in your SASUSER.PROFILE and are automatically applied each time you run the GenDoc utility. To view or set the options defined for your SAS/AF session, open the GenDoc utility and select the **Options...** button. For more information on specific options, see the online help for the GenDoc Options window.

CONCLUSION

By enabling users to quickly generate HTML files that are viewable on any standard web browser, the GenDoc utility helps meet the immediate documentation needs of a project as well as the longer-term goal of object re-use.

REFERENCES

SAS Institute Inc. (1999), *SAS Guide to Applications Development, First Edition*, Cary, NC: SAS Institute Inc.

Tomasi, Marty (1999), "Adding Help and Other User Assistance to SAS/AF Applications," *Proceedings of the Twenty Fourth Annual SAS Users Group International Conference*, 24.

ACKNOWLEDGMENTS

The authors would like to acknowledge the efforts of Glen Walker, Tammy Gagliano, and Yao Chen of SAS Institute for their support of the GenDoc initiative. Also, Jeff Diamond and David Henderson of SAS Institute were instrumental in implementing new help features in SAS software that make it possible to provide component documentation as user assistance within SAS/AF software.

SAS and SAS/AF are registered trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

CONTACT INFORMATION

Marty Tomasi
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
Marty.Tomasi@sas.com

Chris Barrett
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
Chris.Barrett@sas.com