

Understanding DataEase Style Sheets

Graham Smith
PLM Consulting, Inc.
gsmith@plmconsulting.com
October 2005

What is a Style Sheet?

Style sheets in DataEase are, at best, badly named. In actual fact, a style sheet is a file with an STY extension that contains individual Styles. If you look in your DataEase directory, you will find a directory called Styles that contain a set of Style Sheets.

If you create a new DataEase document (form, report, proc) or look in the properties of an existing document, you will see a scrollable list of Style Sheets available. This is a dynamic list and will include any STY files that can be found in the Styles directory, the DataEase directory, or the application directory. It is not the file name of the STY that is displayed, but a name that is internal to the file. So, you can have a file with a name of XXX.STY and have the actual name of the Style Sheet be PLMForm1. The reason this is important to know is that the same Style Sheet Name will only be displayed once in the list. IOW, if you make a copy of one of the existing STY files and give the file a different name, it will still only appear once in the list because it has the same internal name.

The smallest of these is the Style Sheet called Template because only a couple of styles are included in it. The larger files contain styles for just about any object you are likely to encounter. If you want to create a new style sheet from scratch, start a new document and choose either Template or None as the Style Sheet. Then choose Save Style Sheet from the File menu and give it a unique name. The new STY file will be put in the application directory.

You also have to be careful here to know where the file is that you are acting on. If you apply an existing Style Sheet such as Corporate or <Normal> to a document then Save the Style Sheet, you will create a new STY file in the application directory. After that, this file is the one that will be modified whenever you change that style. Where this can be confusing is if you are using the same Style Sheet in different applications. Changing a Style Sheet in one application will not change the Style Sheet in another because they are two separate files.

Modifying Style Sheets

It is unfortunate that there is no "editor" for Style Sheets. Changes to styles are made by example. That is, you pick an object on a document and change it's properties then either create a new style for that object or update an existing style. You can see, then, that the only way to view and edit all the styles contained in a Style Sheet is to have a document or

documents with one example of each object and each style for that object. IOW, if you have 5 different styles for Text Fields, you would need to have a document with 5 Text Fields on it, each with a different style applied in order to see all your Text Field styles. You can quickly see how this could get out of hand real quick when you consider that you probably use 8 – 10 different object Classes and have several different styles for each.

Using Style Sheets and Styles

Any Style can be specified as the Default Style for an object. This means that when you create a new form, Styles will automatically be applied to the various objects.

The whole idea behind Style Sheets is to make it easier to apply a consistent look and feel to an application. When you are creating a new form or report, you can select a group of objects and change them all by selecting a style from the list. This can also speed up application development.

The biggest advantage to using styles, however, comes if and when you decide to change the properties of an object Class and you want that to be applied universally throughout the application. If you change a style, then every object in the application that uses that style will reflect that change. The reason is that styles are “attached” to objects in documents – you can actually see the names when you look in the FRMs. When a document is opened, the particular object properties are “looked up” from the style sheet and displayed.

You will likely want to have at least two different Style Sheets in an application. One for forms and one for reports. In actual fact, it can be helpful to have multiple report Style Sheets that default to different font sizes since not all reports can be laid out using the same size fonts.

Object Classes and Types

This is probably the least clear aspect of Style Sheets and the primary reason why it would be nice to have a true editor. At least then it might be possible to understand this topic better.

Every object is part of a larger Class of and this is how Style Sheets are grouped. You can see this if you view one of the STY files. These Classes are apparently labeled “Template” since you see things like LabelTemplate, EditTextTemplate, ButtonTemplate, etc.

Within each Class are the various Styles that are associated with that Class. What can be confusing is that some of the objects have two different, for lack of a better word, Types. One Type is created on a Single Record layout and the other is created in a columnar layout.

To see what I mean by this, create a new form over a small existing table using the <Normal> style sheet and choose the layout as Field Per Line. Now, click on a text field and you will see that the *Field* Style was automatically applied to it. This is the default Style for

this Object Type. Now, change the layout to Table and you will see that the appearance of the text field completely changes. You will also see that it now has a style of Table Grid Field.

This happens despite the fact that Field is the Default Style for a Text Edit Box and there doesn't seem to be any way around it. So, I guess the best way to deal with this is to accept it as a feature and deal with it. That means that you can either let a table layout default to Table Grid Field and then change them all to something else, or you can change the two styles so that they are the same. I had been doing the former but have realized that I am just creating extra work for myself and am going to change to the latter.

Tricks with Style Sheets

The way in which Style Sheets work also make it possible to play games with Style Sheets. You can have two different Style Sheets with the same named styles but with different properties. This means that you can completely change the look of a document simply by changing the Style Sheet it uses.

You can also have a separate Style Sheet with the same name on each workstation. For example, you could have a style called AdminOnly that displays a field using colors that make it visible. Copy that style sheet to the Styles folder of the database administrator. Make a copy of the same style sheet in which the AdminOnly style uses colors that make it invisible and put it on the user computers. Then remove the STY files from the database directory on the server. Each user will then see the fields using the colors supplied by their local copy of the STY sheet. Of course, trying to manage this during development is a real pain, but once the STY sheets are fixed, it is easy enough to implement.

Style Sheet Gotcha

The biggest problem with styles is that they do not get carried along when you copy an object from one document to another. If you have a text object on one form with a style of BigRed and you copy it to another form, it will still look the same and have the same properties but there will not be any style attached to it. That means that if you later change the font of the BigRed style, you will not see that change in the copied object unless you reapplied the style after copying it.