

# Improved Style Sheets for Better Looking Apps

## Introduction

The purpose of a style sheet is to let you quickly create professional-looking documents whose appearance underlines purpose.

Just like a newspaper, you want the reader's eye drawn to headlines and subtitles, before they get down to devouring the detail of the body text.

You want to apply tasteful and consistent design to let the consumers of your applications see that you are a professional who knows you are doing.

And you want to do all that as simply as possible.

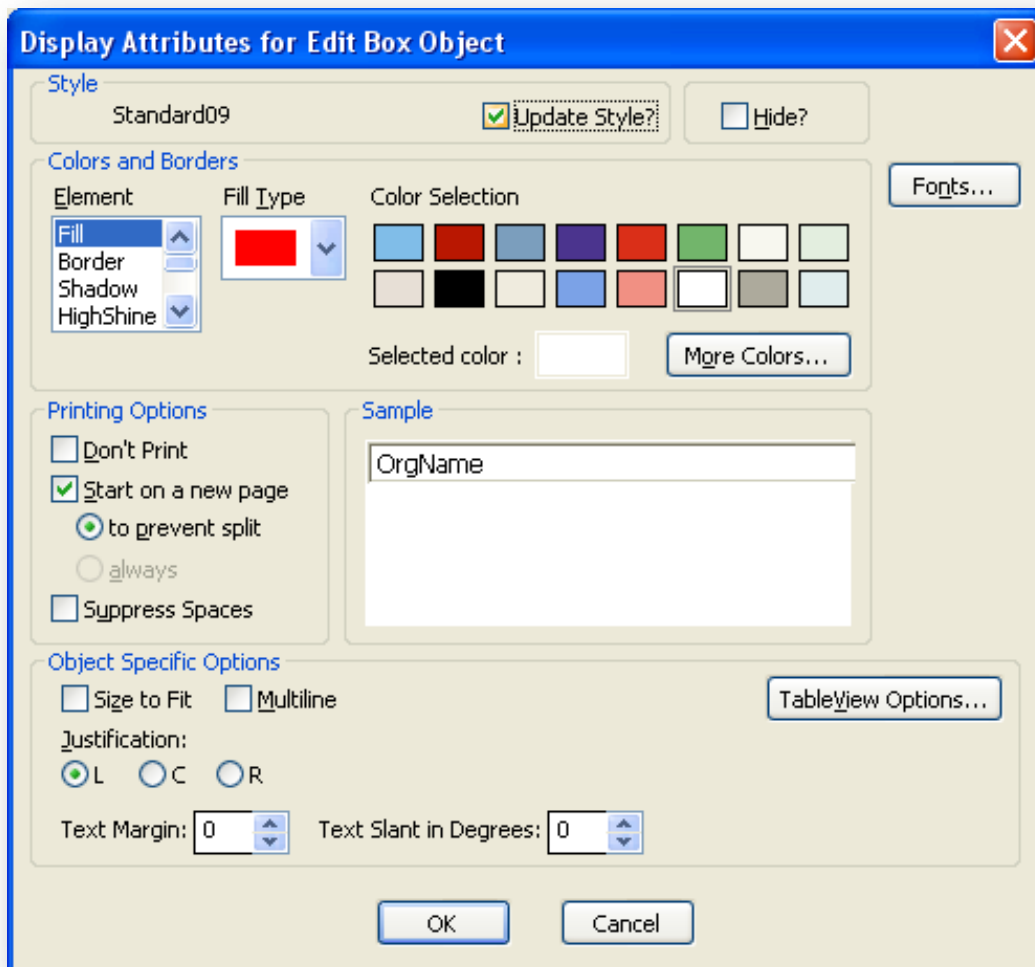
Ffenics style sheets have always let you do a lot of this. But we feel that work is needed to make them truly the design aid they should be. 1.53 introduces the first set of changes on the road to the perfect style!

## Updating Styles

In order to update a style in versions up to 1.52, you had to:

- Make changes via the display dialog. These changes would result in the field losing its style name, forcing you to ...
- Select Object → Style → Update ...
- Find the name of the style this object previously had (assuming you could remember it), and then ok the dialog to update the style.

1.53 introduces the ability to update the style via the display dialog, missing out the need to separately update it. We've added an 'Update Style' checkbox to the dialog. If you want the style for the currently selected object to be updated, just check it!



Now when you OK this dialog, all objects with the same style will get the new display details.

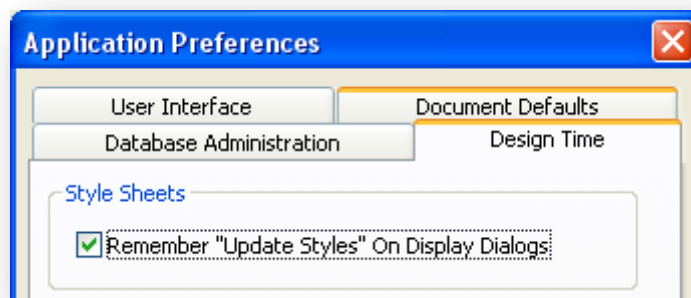
A few things to note:

1. If the object does not have a style, the Update Style option is disabled.



2. If you update an object that has a style, but don't check Update Style, the object will lose its style name (which is the same behaviour as previously).

3. If you have this new option in Application Preferences checked:



Then each time you show the display dialog, the 'Update Styles' setting is remembered. This makes 'Style updating' sessions a lot easier!

4. You still need to save the style sheet for the changes to be permanent. Note that this will affect the styling on all documents that use this style sheet.

## New Default Styles

For each object type – that is, everything on the object palette – you can create a default style that will be used and assigned when the object is first created.

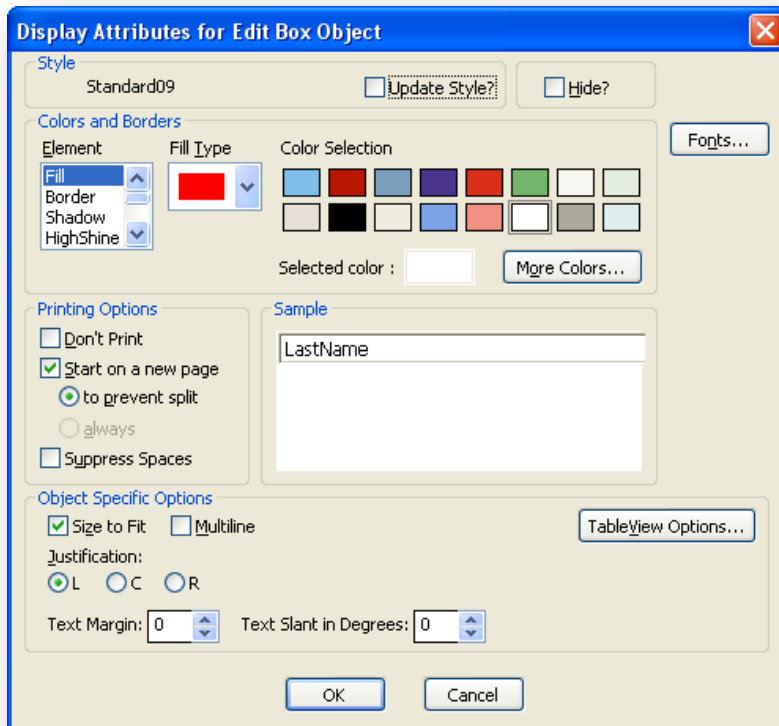
In certain circumstances, there are also what we might call secondary styles that relate to tabular layout. These all use pre-set names. For example, if you create a part of your document as Table Field Layout, AND if your style sheet contains styles with specific names for those generated items, then that style is automatically applied and used. The style names all start with the words 'Table Grid', and continue 'Form', 'Record', 'Field', 'Text' and so on.

These options have always been there. In 1.53 we've added two new ones: for the main form and any subform title.

If you check 'Add Title' on the Layout dialog for a main form, Ffenics will apply a style called 'Form Title', if one exists, to the label it creates for the title. If this is a subform, it looks for a style name of 'Sub form Title'.

## Tidy-Up

You can think of a style sheet as a way to automatically apply most of the settings on the display dialog. Here's that dialog for an editbox object:



The sections on this dialog that style sheets address are: Colors and Borders, the Object Specific Options, and then the dialogs accessed via the fonts and Table View options buttons.

Due to various changes over the years, improvements in what you could control with the display dialog were not always added to the style sheet as well. For example, the size and stretch to fit options for images and image fields were not style settings. Well, they now are, and generally all the options (including the new justification settings for labels) can be saved to a style.

The main exception is label justification. This was planned to be changed, but rather late in the beta testing, we realised that the code was setting justification based on field type – such as centring date and time values. Had we implemented our changes, existing documents would have been affected, and you'd probably also need a separate editbox style for each field type.

We didn't want to force either of these on you, so we took editbox justification out of style sheets again. We'll address this again in the next version instead.

### No Default

Individual tabs on the tab control were not being assigned a default style. This has now been corrected.