Switching from XLS to XLSX or XLSM

In ExcelWriter 8, support for the new Open Office XML (OOXML) file formats (XLSX, XLSM) was introduced in ExcelApplication. Since XLSX files have different file format defaults and a different file structure than XLS, ExcelWriter's behavior can change between the two file formats in specific situations. These differences are outlined below.

Using the Color Palette

XLS files use a 56 color palette (see Excel's Color Palette Explained). When using ExcelApplication's `Workbook.Palette`, ExcelWriter will throw an error if `Palette.GetColor` is called on an XLS file and the color doesn't exist in the palette. Customers are encouraged to use `Palette.GetClosestColor` to avoid calling for colors that don't exist in the palette. ExcelWriter will approximate which color from the 56 color palette to use.

In XLSX files, there are an unlimited number of colors available. `Palette.GetClosestColor` will return *the exact color*, rather than an approximation based on the 56-color palette. Customers may need to adjust the colors to retain the original look and feel of the workbook.

Note: XLSX files also have an underlying 56-color palette for the purposes of rendering XLSX files in Excel 2003 (with compatibility pack). This palette can still be modified by ExcelApplication v8. See this post for information on designing Excel 2007/2010 reports that might be viewed in Excel 2003.

Creating Files from Scratch

ExcelApplication can generate a spreadsheet either by opening an existing file with `ExcelApplication.Open` or by creating a new file in memory using ExcelApplication.Create(FileFormat). Starting in v8, ExcelWriter can create both XLS and XLSX files, but the file format must be specified when calling Create.

```
//Specify either FileFormat.Xlsx or FileFormat.Xls to create a file
Workbook wb = ExcelApplication.Create(ExcelApplication.FileFormat.Xlsx);
```

The `Create` method uses an underlying template with defaults associated with the file format. XLS files use the defaults from Excel 2003; XLSX files use the defaults from Excel 2007. Switching to XLSX means the default template will change and the new defaults may change how output files look.

The largest impact from this relates to named styles. In Excel 2003, the default Normal style is Arial, size 10. In Excel 2007, the default Normal style is Calibri, size 11. Settings that are based on the Normal style will be different between XLS and XLSX files:

- Default row height and column height in a file - These values are based on the font type and style of the Normal style.
- Setting `ColumnProperties.Width` - ExcelWriter sets this to a fixed number (e.g. 38). Excel uses this number and the Normal style to when rendering the width of the column.

To ensure that output remains consistent while switching to XLSX, we recommend creating a blank XLSX file in Excel that has any desired defaults set, such as the Normal style. Then call `ExcelApplication.Open` on the Excel file, rather than calling `Create`.

Chart defaults

Some very minor charting defaults have changed, such as:

- Distance between ticks on an axis
- Default text color for some areas of the chart is white

The workaround is to set these properties to the desired values in ExcelWriter code. Most ExcelWriter-generated charts will appear the same regardless of whether they are in XLS or XLSX files.

New Excel features

New features were introduced in Excel 2007 and 2010. ExcelWriter does not currently support all the new features that are available in 2007/2010. We are constantly working to incorporate support for these features in future releases of OfficeWriter.

ExcelWriter can largely manipulate elements in XLS files without having to completely parse them, which includes features that were introduced in Excel 2007/2010. For example, Office introduced a series of new formulas (e.g. RANDBETWEEN, NORMAL.DIST). These formulas are mostly ignored by ExcelWriter in XLS files.

XLSX files are parsed much more thoroughly and ExcelWriter will sometimes throw exceptions if it encounters features that have not been implemented. In the case of new 2007/2010 formulas in XLSX files, ExcelWriter will throw a *'cannot find function in table'* exception if support for the new formula hasn't been added yet.

We are adding support for new features, including formulas, with every release. Customers are encouraged to upgrade to the latest version of OfficeWriter for the most recent features and to contact support if upgrading doesn't resolve the issue.