Layout Builder Sections and Subroutines

Two subroutines are created for each layout section created in Layout Builder. One subroutine is named using the section name as the subroutine name. The second subroutine is created using the section name with **ABS** appended to it as the subroutine name.

A third subroutine is created if HTML is selected as the output type using the section name with HTML appended to as the subroutine name. The subroutine takes zero parameters because there is no height calculation involved for HTML. An HTML section will render immediately after the previous HTML section. There is no concept of absolute X and Y offsets.

The first subroutine that is named using the section name takes an integer parameter of **RPT_RENDER** (value of 0) or **RPT_CALCHEIGHT** (value of 1), which is used to designate whether the section should be rendered. This subroutine is created for ease of use when the margins and automatic y offset incrementation are needed. This automatic incrementation is held in the generated _**YOffset** variable. If the **RPT_CALCHEIGHT** is passed in, the calculated section height is returned without the section being rendered to the page and without the _**YOffset** variable being incremented. When **RPT_RENDER** is passed in, the calculated section height is returned, the section is rendered, and the _**YOffset** variable is incremented.

The second subroutine is named using the section name with **ABS** appended. The first subroutine contains a call to execute the second subroutine. This is the absolute subroutine call and contains all of the logic making the calls to the Report API for the report section generation. The three parameters that this subroutine will take are the calculation flag (**RPT_RENDER** or **RPT_CALCHEIGHT**), as well as the X and Y offsets for the section as real numbers in terms of the measure of units selected for the report (inches or centimeters).
The X,Y offsets are relative to the 0,0 origin at the top left corner of the page. A section can be rendered at any location, but the developer should ensure that the section fits on the page. Any items that do not fit on the page are not printed or displayed.

The third subroutine that is named using the section name with HTML appended, takes zero parameters because there is no height calculation involved for HTML. An HTML section will render immediately after the previous HTML section. There is no concept of absolute X and Y offsets.

Layout Builder represents sections as the full width of the page, based on orientation, although the intended section may not consume the full width of the section and may actually only apply to a column of data in the section that is repeatable across the width of the section. This does not pose any problems since the sections are virtual. It is just a draw back of representing sections as full page widths in Layout Builder.
Layout Builder allows the selection of a page size, orientation, type of report (Postscript, .PDF, Zebra Stripe (ZPL) 200dpi, or Intermec 3400 (IPL) 200 dpi), margins, and unit of measure (inches or centimeters).

Sections are displayed in the orientation selected; page width and height are based on a portrait report. Only left and right margins are represented in Layout Builder based on the orientation of the report since Layout Builder only uses these guides for item placement and visual representation.

The top and bottom margins are not represented in the builder since these margins do not directly apply to each section, but instead to the report in general. The Report API does use these margins in limited circumstances for generation of items that may grow either in width, height, or both, in which case the margins are the boundary in which the item can grow.
Layout Builder also generates other subroutines necessary for report generation, **InitializeReport**, **FinalizeReport**, **PageBreak**, **_LoadImages**, **_CreatePens**, and **_CreateFonts**.

InitializeReport takes an integer dummy parameter reserved for future use which must currently be 0 or unexpected results can occur. **InitializeReport** sets the RPT properties, creates the report context, loads any images, creates any pens needed, creates any fonts used, starts the report, and begins the page. It does so by calling the necessary Layout Builder created subroutines beginning with an underscore ('_').

After the **InitializeReport** has been called, the sections created with Layout Builder can be called in the order they are to be rendered. The _**YOffset** is incremented by the height of the section when it is rendered. By using the _**YOffset** and adding the result from the section to be rendered using the **RPT_CALCHEIGHT** as the parameter, you can compare against the appropriate value to determine page break.

To create a page break for Layout Builder sections, call the **PageBreak** subroutine with a 0 for the reserved parameter (0 must be used or unexpected results can occur). This subroutine calls the appropriate Report API routines to end the page and begin a new page.
When no more pages are to be generated and the report is ready to be finalized, call the **FinalizeReport.** This report passes in a string that contains the queue to print to or the file to generate. This subroutine ends the final page, ends the report, and writes the report to a temporary file. If the report is to be printed to a queue, the file is spooled and deleted.

What You See Is What You Get (WYSIWYG):

Layout Builder attempts to provide a WYSIWYG display. However, the standard fonts that are available in a Windows system and the fonts that are available on a printer can be different. To achieve true WYSIWYG, the printer fonts and the windows display fonts must be the same. Most organizations do not purchase additional display fonts to match the fonts that are shipped with a printer. To help match printer fonts to display fonts Layout Builder provides the Map Fonts option on the Edit menu. This option can be used to manually associate an existing display font to a True Type printer font. For example, the default font for a Zebra printer is most likely not available as a display font, but the Arial Black display font is very close to the Zebra's default font. When creating a layout for a Zebra printer, you should select Map Fonts from the Edit menu and modify the Display font to Arial Black. This provides a very near WYSIWYG display; what you see on your screen when creating the layout is almost exactly what is printed by the Zebra printer.

WYSIWYG does not apply for HTML. Layout Builder sometimes has to *massage* the items in order to make them render. For example, HTML cannot stack one text item on top of another. So, Layout Builder may need to *scoot* a time to create a valid arrangement.