IBM Business Analytics Proven Practices: How to Implement a Corporate-Wide Reporting Style Using IBM Cognos BI

Products: IBM Cognos Report Studio, IBM Cognos Administration, IBM Cognos Visualization Customizer Utility, and IBM Cognos Workspace Advanced

Steve Macko
Advisory Designer, IBM BA Samples Development
IBM

06 March 2015

Many companies would like to implement their own corporate style guide across all of their reports and dashboards, truly engaging their users. This can be a huge undertaking, especially when you have many existing reports and/or intend to create many more new reports. What is the best way to approach this task? The traditional method is to modify each report one by one via the Properties pane in Report Studio. This document will describe multiple alternatives, providing advice on how to approach this problem more efficiently and effectively.

View more content in this series

Introduction

Purpose of Document

Many companies would like to implement their own corporate style guide across all of their reports and dashboards, truly engaging their users. This can be a huge undertaking, especially when you have many existing reports and/or intend to create many more new reports. What is the best way to approach this task? The traditional method is to modify each report one by one via the Properties pane in Report Studio. This document will describe multiple alternatives, providing advice on how to approach this problem more efficiently and effectively.

Chapter 1 provides guidance on what to include and how to structure a corporate style guide so that it can be implemented within an IBM Cognos 10 BI reporting environment.
Chapter 2 identifies multiple strategies for applying a corporate style guide to new and existing reports and dashboards. The more reports you currently have (or intend to create), the more you will need an effective strategy to style them.

Chapter 3 explains how to create and modify classes that apply to all reports by modifying the global Cognos style sheet.

Chapter 4 discusses how report authors can create and use templates in conjunction with layout component references, local classes, and global class extensions to save time and promote consistency.

Chapter 5 provides advice on how to quickly style lists and crosstabs by modifying the default table styles provided by Cognos, or create new ones.

Chapter 6 explains how to modify the default chart styles provided by Cognos, or create new ones.

Chapter 7 provides advice on how to modify the default JSON files provided by Cognos for IBM Rapidly Adaptive Visualization Engine (RAVE) charts.

Chapter 8 provides a link to an existing Proven Practice for adding a custom palette to the IBM Cognos Report Studio palette dialog.

Applicability

The style guide elements outlined in this document can be successfully implemented in IBM Cognos BI 10.2 and 10.2.1.x. In 10.2.2 there are new features which simplify styling reports as well as customizing RAVE charts and will be highlighted where appropriate in the document.

Exclusions and exceptions

This document does not cover every possible aspect of implementing a corporate style guide with IBM Cognos BI. The examples chosen are intended to be representative of core use cases.

Assumptions

This document assumes readers are advanced Report Authors with IBM Cognos Report Studio and IBM Cognos Workspace Advanced and are familiar with common styling and formatting terms.

Chapter 1 - Create a Corporate Style Guide

This chapter will show you how to create and structure a corporate style guide specification that can be implemented in IBM Cognos Report Studio.

Consistency is very important when you are specifying a common corporate look and feel across many reports. Your users will often view more than one report in their roles, and could become confused if the reports all look and behave differently. The exception to this would be a user who views reports and dashboards from different departments. In this case, each major department could have its own unique header, but the charting style remains constant across the organization.
If you don’t have a visual designer within your Cognos BI department, you can obtain design ideas by examining your company’s website. This is an especially useful exercise for obtaining logos, brand styles, and corporate color schemes.

**Prototype**

We can use IBM Cognos Report Studio itself as a quick and easy prototyping tool. For the data, we can use a real data set, a small snapshot of real data, or the Great Outdoors sample data provided with IBM Cognos BI.

First, we identify a list of the core objects that will appear on our reports and dashboards, such as:

- Lists
- Crosstabs
- Bar charts
- Line charts
- Pie charts
- Bullet charts
- By the Numbers (section to provide a quick summary that is easy to see)
- Header
- Footer
- Prompts
- Dashboard layouts and styles (make some sketches and prototypes)

Second, we open Report Studio and start building prototypes of these objects. We experiment with various color schemes, borders, padding, fonts, and other formatting options. I like to build each object on its own report with a small data set so the report will run quickly, run it, make changes, and run it again. This type of iterative design provides rapid results and many versions.

Third, once we have a set of multiple prototypes for each of the core objects (for example, 3 different pies, 4 bar chart designs and so on), we show them to stakeholders and end users to get their feedback. We incorporate the feedback, revise the designs, take a vote, and finalize each object.

**Identify design principles**

As you create your designs, it is useful to specify some overall design principles in your corporate style guide. In the examples in this document, our design principles are:

1. The report or dashboard page background will be white, and the text will be dark grey rather than black.
2. It must be easy to read for an extended period of time.
3. We will leverage the look and feel of the company’s website. It will be clean, minimalist, and professional.
4. We will not use excessive graphics.
5. The focus will always be on the data. The UI chrome (e.g. axes, gridlines, and labels on charts) will be muted in light shades of grey so that the data stands out.
6. The text/background contrast ratio will be greater than 4.5:1 in all cases.
Design for accessibility

Each company will have their own accessibility plan. A common criterion is minimum contrast. In your IBM Cognos reporting style guide, it is important to ensure that you have specified sufficient contrast between the colors of your text and the colors of their immediate background. The text/background contrast ratio should be greater than 4.5:1 in all cases. Full details can be found in Section 1.4.3 of the W3C Web Content Accessibility Guidelines. You can download or use a free color contrast ratio tool from the Internet.

Specify fonts and colors

When it comes time to write the actual corporate style guide document, I always start by specifying the fonts and colors in a table. Table 1 provides an example of the core font elements that we will need to specify in Report Studio, and Table 2 provides the core colors. I have used color banding in the tables to indicate groupings.

Table 1. Font specification

<table>
<thead>
<tr>
<th>Item</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report/Dashboard title (in report Header)</td>
<td>Normal Bold 16pt Arial</td>
</tr>
<tr>
<td>Product name (in report Header)</td>
<td>Normal Bold 12pt Arial</td>
</tr>
<tr>
<td>General text</td>
<td>Normal Normal 9pt Arial</td>
</tr>
<tr>
<td>General titles</td>
<td>Normal Bold 10pt Arial</td>
</tr>
<tr>
<td>Tab text</td>
<td>Normal Normal 12pt Arial</td>
</tr>
<tr>
<td>By The Numbers</td>
<td>Normal Bold 16, 24, or 32pt Arial</td>
</tr>
<tr>
<td>Prompt labels</td>
<td>Normal Bold 10pt Arial</td>
</tr>
</tbody>
</table>

Once specified, your corporate reports and dashboards will only use the fonts in the table above. For example, 11pt Arial would never be used. This ensures consistency, and eliminates guesswork on the part of the report authors.

Table 2. Color specification

<table>
<thead>
<tr>
<th>Item</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Axis lines (2px)</td>
<td>#C2C2C2</td>
</tr>
<tr>
<td>Minor gridlines</td>
<td>#E6E6E6</td>
</tr>
<tr>
<td>Axis labels</td>
<td>#676767</td>
</tr>
<tr>
<td>Chart titles</td>
<td>#676767</td>
</tr>
<tr>
<td>Regular data/text font</td>
<td>#444444</td>
</tr>
<tr>
<td>Regular title font</td>
<td>#676767</td>
</tr>
<tr>
<td>Highlighted data/text font</td>
<td>#1B6A87</td>
</tr>
<tr>
<td>Prompt label font</td>
<td>#676767</td>
</tr>
<tr>
<td>Prompt area background</td>
<td>#F7F7F7</td>
</tr>
</tbody>
</table>
Specify property values

When the time comes to author the reports and dashboards in Report Studio, we will need to specify a set of specific property values for each core reporting object. Let’s use a list as an example. When an author first creates a list, the list will adopt the default Cognos style. One way to modify this style is to manually make adjustments to the list’s properties by selecting an item in the report and then adjusting the desired properties as shown in Figure 1. There are other ways to automate this process that are the subject of this document and are described in later chapters.

**Figure 1. Table Cell properties**
How do we specify for our report authors which properties they need to override or set? Building on our font and color specification outlined above, Table 3 presents an example of a property specification table for corporate lists.

Table 3. List properties specification

<table>
<thead>
<tr>
<th>Item</th>
<th>Property</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>List</td>
<td>Border</td>
<td>&lt;delete&gt;</td>
</tr>
<tr>
<td></td>
<td>Size &amp; Overflow</td>
<td>&lt;do not set&gt;</td>
</tr>
<tr>
<td>List Columns Title Style</td>
<td>Border</td>
<td>Top&lt;br&gt;• Style: Solid line&lt;br&gt;• Width: 1pt&lt;br&gt;• Color: #DADADA&lt;br&gt;Bottom&lt;br&gt;• Style: Solid line&lt;br&gt;• Width: 1pt&lt;br&gt;• Color: #CCCCCC&lt;br&gt;No Left or Right Border</td>
</tr>
<tr>
<td>Font</td>
<td>Family: Arial&lt;br&gt;Size: 10pt&lt;br&gt;Weight: Bold&lt;br&gt;Style: Normal&lt;br&gt;Foreground Color: #444444</td>
<td></td>
</tr>
<tr>
<td>Padding</td>
<td>Top: 10&lt;br&gt;Left: 15&lt;br&gt;Right: 15&lt;br&gt;Bottom: 10</td>
<td></td>
</tr>
<tr>
<td>List Column Title (each column)</td>
<td>Allow Sorting</td>
<td>Choose Yes or No</td>
</tr>
<tr>
<td></td>
<td>Background Image</td>
<td>grey_gradient.png&lt;br&gt;Tile in horizontal direction</td>
</tr>
<tr>
<td></td>
<td>Classes</td>
<td>&lt;Delete&gt;</td>
</tr>
<tr>
<td></td>
<td>Horizontal Alignment</td>
<td>Left for string columns&lt;br&gt;Right for number columns&lt;br&gt;Center for text only status columns&lt;br&gt;Left for icon status columns</td>
</tr>
<tr>
<td>List Columns Body Style</td>
<td>Border</td>
<td>Bottom&lt;br&gt;• Style: Solid line&lt;br&gt;• Width: 1pt&lt;br&gt;• Color: #D9D9D9&lt;br&gt;No Top, Left, or Right border</td>
</tr>
<tr>
<td></td>
<td>Font</td>
<td>Family: Arial&lt;br&gt;Size: 9pt&lt;br&gt;Weight: Normal&lt;br&gt;Style: Normal&lt;br&gt;Foreground Color: #333333</td>
</tr>
</tbody>
</table>
Figure 2 is an example image of a list that has been styled in the corporate style guide. Some things to note are the consistent font styles, a gradient list header, and specific grey subtotal and total rows.

**Figure 2. List with the corporate style guide applied**

![List with the corporate style guide applied](image-url)

**Create cheat sheets**

One method for increasing adoption of your corporate style guide is to provide cheat sheets for your report authors. Cheat sheets provide a quick reference to the core style elements. Figure 3 shows an example of a cheat sheet for clustered column charts which labels each element with the style to be used.
Create standard page layouts

A rule of thumb is always place the most important information at the top left and in the centre of the dashboard or report. It is a good idea to decide on a few common layout grids. Figure 4 shows multiple examples of common layout grids that can be used with dashboards or large reports. These are examples for one, two, three, or four columns of charts.

Figure 4. Common dashboard layout grids

Figure 5 shows effective screen architecture for a dashboard. At the top is a header that includes the dashboard’s title, a prompt filter that applies to the entire dashboard, and a company logo. Below the header is a set of tabs for grouped content. Each tab has an area for two static reports. Below that is a bar containing local prompts that filter the dynamic charts below them.
Chapter 2 - Determine Strategy

Applying a corporate style guide to many reports is a big job! This chapter will describe implementation strategies for new and existing reports. The more reports that you currently have and/or intend to create, the more you will need an effective strategy to style them.

How does a report object obtain its style?

As shown in Figure 6, the style of a Cognos report or report object can be affected by many sources such as:

- Properties
- Global Cognos style sheet such as GlobalReportStyles_10.xml
- Templates
- Local Classes
- Layout Component References
- TableStyles.xml
- ChartTemplates.xml
- RAVE visualizations
- ReportPresets.xml
Show XML

An IBM Cognos BI 10.2 report is specified in XML. To see how the styles have been implemented within this XML, we can click on the **Show Specification** toolbar button in Report Studio for any open report, as shown in Figure 7.

**Figure 7. Show Specification (XML) toolbar button**

Tip: If you select a specific report object (e.g. a bar chart), hold down the **Shift** key and click the **Show Specification** toolbar button and you will see the XML for only the selected object and not the whole report. This trick works great with charts!

Which strategy should I use?

Again, as shown in Figure 6 above, there are many options for report styling that are available to report authors:

- Assign object properties in Report Studio
- Modify the global Cognos style sheet (see **Chapter 3**)
- Create templates (see **Chapter 4**)
- Create local classes and global class extensions (see **Chapter 4**)
- Create layout component references (see **Chapter 4**)
- Modify TableStyles.xml (see **Chapter 5**)
- Modify ChartTemplates.xml (see **Chapter 6**)
- Create RAVE visualizations (see **Chapter 7**)
- Modify ReportPresets.xml (see **Chapter 8**)

Note that some options only apply when we are creating new objects, and other options only apply when we are modifying existing objects. Table 4 identifies common report objects and the best strategies for styling each one, divided into new and existing sections.

**Table 4. Options for applying styles to new and existing reports**

<table>
<thead>
<tr>
<th>Report Object</th>
<th>Strategies</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
New

Chart
- Manually adjust properties as required
- Create local classes and modify global class extensions as required
- Modify ChartTemplates.xml prior to authoring
- Modify GlobalReportStyles_10.xml prior to authoring
- Use a modified RAVE visualization

List, Crosstab, Table, Repeater Table
- Manually adjust properties as required
- Create local classes and modify global class extensions as required
- Modify TableStyles.xml and apply a table style
- Modify GlobalReportStyles_10.xml prior to authoring

Report
- Manually adjust properties as required
- Create local classes and modify global class extensions as required
- Use a template for the layout and add local classes, global class extensions, and layout component references

Existing

Chart
- Manually adjust properties as required
- Create local classes and modify global class extensions as required
- Modify ChartTemplates.xml, rebuild the existing chart beside the original chart on the same report (this is to preserve the query), and delete the original chart
- Modify ReportPresets.xml and change the palette
- Modify GlobalReportStyles_10.xml and delete existing local properties that override the modifications
- Insert a modified RAVE chart, rebuild the existing chart beside the original chart on the same report (this is to preserve the query), and delete the original chart

List, Crosstab, Table, Repeater Table
- Manually adjust properties as required
- Create local classes and modify global class extensions as required
- Modify TableStyles.xml and apply a table style
- Modify GlobalReportStyles_10.xml and delete existing local properties that override the modifications

Report
- Manually adjust properties as required
- Create local classes and modify global class extensions as required

Scope

Some strategies are local to a specific report (e.g. manually adjusting the properties, creating local classes, and manually adjusting global class extensions) and are best suited to situations where you only need to style one or a few reports. However, this strategy is time consuming, prone to errors and omissions and only applicable to the current report.

Global strategies such as RAVE (Rapidly Adaptive Visualization Engine) or modifying the various XML files listed above are time-consuming to set up initially but are extremely useful when you need to modify large numbers of reports. However, any modifications to these XML files will be lost if the IBM Cognos BI software is re-installed or upgraded. These modified XML files will need to be backed up and restored in these situations.

Example

Here is an example of how to put several strategies together to come up with a plan to style 150 new reports. Suppose each report will have a title and two charts (or lists or crosstabs), one on top of the other vertically. The charts will have titles, grey gutters, and padding around them. The plan would be:
1. Create a template for the basic report layout that includes a layout component reference for the report’s title and each chart’s title. In the template, create and apply local classes and global class extensions for objects that have multiple instances.

2. Modify `ChartTemplates.xml` only for the chart types that we will need (e.g. clustered column, clustered line with markers, clustered bar, and pie).

3. Modify a RAVE visualization bundle for a treemap.

4. Modify `TableStyles.xml` for the lists and crosstabs.

**Chapter 3 - Modify Global Style Sheet**

If we want to create and modify classes that apply to all reports, we can modify the global Cognos style sheet. This is a useful strategy if we only have one corporate style for all reports. If we have multiple styles, such as a “blue style” and a “green style”, then some of the other strategies described later in this document will apply.

There are four versions of the global Cognos style sheet as shown in Table 5.

**Table 5. Four versions of the global Cognos style sheet**

<table>
<thead>
<tr>
<th>CSS Version</th>
<th>Report Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlobalReportStyles.css</td>
<td>8.x styles</td>
<td>Classes that were used in IBM Cognos 8 BI</td>
</tr>
<tr>
<td>GlobalReportStyles_none.css</td>
<td>Simplified styles</td>
<td>Classes that have minimal styling defined, useful for financial reports</td>
</tr>
<tr>
<td>GlobalReportStyles_1.css</td>
<td>1.x styles</td>
<td>Classes that were used in IBM Cognos ReportNet</td>
</tr>
<tr>
<td>GlobalReportStyles_10.css</td>
<td>10.x styles</td>
<td>Classes in the default style sheet for IBM Cognos 10 BI</td>
</tr>
</tbody>
</table>

The exact CSS file used will be chosen based on how the report style is set for a given report. The default setting for IBM Cognos BI 10 is **10.x styles**, as shown in Figure 8. You can open this dialog by selecting **Report Properties** from the **File** menu in IBM Cognos Report Studio.

**Figure 8. Report Properties dialog**

Each of the four CSS files identified above is installed by Cognos in four different locations, as shown in Table 6.

**Table 6.**

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;c10_install&gt;/bin</code></td>
<td>The file in this location is used by Report Server for PDF and Microsoft Excel spreadsheet software outputs.</td>
</tr>
</tbody>
</table>
Appendix B in the attached Appendices.zip file provides a snapshot of GlobalReportStyles_10.css for lists and crosstabs. By modifying any of the style elements in this file, we can change the appearance of all lists and crosstabs that are authored.

It is important to remember that local styles that have been set in the properties of our individual reports will override the global Cognos style sheet, so if you are experiencing unexpected results while modifying the style sheet, this could be the case. We can't modify the global Cognos style sheet and expect it to restyle at runtime our library of existing reports that have previously been styled via their properties or other means. For more information about modifying the global style sheet, please see the section Changing the Global IBM Cognos Style Sheet in the IBM Cognos BI Administration and Security Guide.

In the following example we will modify the IBM Cognos 10 BI global style sheet GlobalReportStyles_10.css. This style sheet is used by Cognos Viewer for HTML output. Figure 9 shows a default list report.

Figure 9. Default list report

![Default list report](image.png)

The following steps are a simple modification where we will modify GlobalReportStyles_10.css to increase the font size.

1. On your IBM Cognos BI 10 server, find the file named `<c10_install>\webcontent\schemas\GlobalReportStyles_10.css`
2. Copy this file and save it as `GlobalReportStyles_10_BU.css`
3. Open the original GlobalReportStyles_10.css in a text editor.
4. Find the text `.ls /* list */`
5. Scroll down to `font-size: 8pt;` and change it to `font-size: 28pt;`
6. Save the file and close it.
7. Clear your browser's cache and restart Report Studio.
8. Create a new List report with any data source. Select the List object and view its properties.
   You will notice that the Font property is empty.

IBM Business Analytics Proven Practices: How to Implement a Corporate-Wide Reporting Style Using IBM Cognos BI
Figure 10. Properties pane with Ancestor icon and Font property highlighted

9. Click the Run Report toolbar button to preview the HTML output in Cognos Viewer. It will pick up the new 28pt font size at runtime.

Figure 11. List report with new formatting

There are a couple of caveats to this approach.

- When we make changes to GlobalReportStyles_10.css, we can only see them at runtime when we run the report. They are not visible in Report Studio itself.
- We can override the font setting in the file GlobalReportStyles_10.css by setting the Font property of the List object in Report Studio itself.
- Any modifications to GlobalReportStyles_10.css will be lost if the IBM Cognos 10 BI software is reinstalled or upgraded. This file will need to be backed-up and restored in these situations.

Chapter 4 - Leverage Layout Component References, Local Classes and Global Class Extensions in a Template

This chapter describes how to use templates in conjunction with layout component references, local classes, and global class extensions to save time and promote consistency while applying your corporate style guide to your new reports.

The strategy described in the following sections is:

- Create a template.
• Create local classes as part of the template.
• Modify global class extensions as part of the template.
• Add layout component references to the template.

Note that once a new report is created from a template, any template that was used to define the initial layout/structure of the report is not identified in the new report’s XML. If required, report authors will need to manually track which templates have been used by which reports.

Report authors can access templates in Cognos by navigating to them in IBM Cognos Connection or via the New dialog in Report Studio. The access method is determined by how the template is finalized, as described below in the Finalize template section.

Create a new template

We can use a template when we need to create multiple corporate-wide reports that have the same basic layout. To create a new template, we will start by creating a new report in Report Studio. It doesn’t matter which data source package we use because we will be removing all references to it later.

In our current example, we will create a template for a corporate report that contains a header and four charts with titles (Figure 12). If we switch to the Page Structure view (select View > Page Structure from the menu bar), we can see that these objects are arranged in a Table (Figure 13).

Figure 12. Report template in Report Studio
Figure 13. Outer layout table in the template

If we expand the page structure even more, we can see the inner Tables and their contents (Figure 14).
Figure 14. Inner layout tables in the template

As described in the Create local classes and global class extensions section and the Create layout component references section below, this template will leverage local classes for all table cells and layout component references for the header and chart titles. Figure 15 shows the various sections of the template including 1) the header (at the top of the page), 2) the chart titles (at the top of each chart), 3) the outer gutters (the space between the tables that contain the charts and the parent table cell borders, and 4) the interior gutter (the space between the chart titles and the charts and the borders of the table that contain them).
Figure 15. Layout component references and local classes used in the template

![Diagram of layout component references and local classes](image)

Legend:
1 = Layout Component Reference: Header
2 = Layout Component Reference: ChartTitle
3 = Local Class: Outer_Gutter
4 = Local Class: Interior_Gutter

Finalize template

There are two ways that report authors can access templates in IBM Cognos 10 BI - via IBM Cognos Connection or the New dialog in Report Studio. The access method is determined by how the template is finalized, as described in the following sections.

Access templates via IBM Cognos Connection

Report authors can navigate to and open their report templates directly from IBM Cognos Connection just like any other report. The steps involved in finalizing a template that is accessed in this way are:

1. When we are finished designing our template in Report Studio, we open the File menu and select Convert to Template. Report Studio will remove several XML elements from the report’s specification including <modelPath>, which is the reference to a specific package.
2. Open the File menu, select Save As, and save the report with a new filename. I like to prefix the name of all of my templates with the text “Template_“. For example, a template that includes a header and four charts might be named “Template_Header and 4 Charts”.
3. When it is time to actually use the template, navigate to the template location and open it in Report Studio.
4. Click the Change Report Package toolbar button in the Source pane and select a package.

Access templates via Report Studio’s New dialog

Report authors can access their report templates via the New dialog in Report Studio. There are five steps involved in finalizing a template that is accessed in this way.

Step 1: Modify the XML

When we are finished designing our template, we open the File menu and select Convert to Template. Report Studio will remove several XML elements including <modelPath>, which is
the reference to a specific package. Then we open the **Tools** menu and select **Copy Report to Clipboard**. We paste the report specification into an XML editor and find the `<report>` element.

```
<report xmlns="http://developer.cognos.com/schemas/report/11.0/" expressionLocale="en"
    useStyleVersion="10" template="true">
```

Delete the `xmlns` attribute and the `template` attribute. The `<report>` element should appear as follows.

```
<report expressionLocale="en" useStyleVersion="10">
```

Search for any instances of the `query` element and the `refQuery` element and delete the value of the `name` attribute, leaving the double quotation marks.

```
<query name="">
```

Search for any instances of the `page` element and delete the value of the `name` attribute, leaving the double quotation marks, like this:

```
<page name="">
```

Now we are going to add a new `template` element to the top and bottom of the specification, making it a parent of the `report` element. The first line of the XML should read `<template name="Corporate_Template_1">` and the last line in the XML should read `</template>`. The value in the `name` attribute of the new `template` element will be the name of the template in Report Studio’s **New** dialog. In this example, the name of the template is Corporate_Template_1. Do not put any spaces in this name.

Save the file in your XML editor.

Note that if you try to open this modified XML directly in Report Studio (using the **Tools > Open Report from Clipboard** menu), it will give you the following error:

```
The name of the top most element must match the name of the DOCTYPE declaration.
```

This is caused by the addition of the `<template>` element and you must proceed with the following steps in order to make the template available to your report authors.

**Step 2: Modify templates.xml**

A file on the IBM Cognos 10 BI server called identifies the templates that the user can select in the Report Studio **New** dialog (Figure 16).
Step 3: Designate an icon

We can create our own 32px by 32px icon for our new template or re-use an existing one. The existing icons are as follows and can be found on the IBM Cognos 10 BI server in the images directory `<c10_install>\webcontent\pat\images`.

- icon_blank.gif
- icon_list.gif
- icon_crosstab.gif
- icon_chart.gif
- icon_map.gif
- icon_financial.gif
- icon_repeater.gif
- icon_blankApplication.gif
- icon_application.gif
- icon_template_qs.gif
- browse_32x32.gif

Newly created icons should be placed in the same images directory and not contain spaces in the file name.

Step 4: Modify Resources.xml

A file on the Cognos server called `Resources.xml` identifies the content and its order that appears in the Report Studio New dialog, as shown in Figure 14 above. We need to complete the following steps in order to add our new template into `resources.xml`:
1. On your IBM Cognos 10 BI server, open the file `<c10_install>\webcontent\pat\res\Resources.xml`
2. Copy this file and save it as `Resources_BU.xml`.
3. Open the original Resources.xml in an XML editor.
4. Find the text `<listView id="New" view="icon" clipLabels="false">`.
5. Add the following XML in the position you would like your template to appear in the Report Studio New dialog:
   ```xml
   <listItem idsLabel="IDS_LBL_NEW_CORPORATE_TEMPLATE_1" icon="icon_template_qs.gif"
     templateName="Corporate_Template_1"/>
   ```
6. Save the Resources.xml file and close it.

In the `<listItem>` element that was added, the associated attributes are:

- `idsLabel` specifies an ID in a Report Studio string resource file (such as reportstudio_en.xml) that correlates to the label for the template within the Report Studio New dialog. This attribute is only required if you will be using multiple locales.
- `label` specifies the label for the template within the Report Studio New dialog if you are only using a single locale and are not going to translate the label.
- `icon` specifies an image file located in the IBM Cognos 10 BI images directory that correlates to the icon for the template within the Report Studio New dialog.
- `templateName` specifies the name of the template in the file templates.xml.

**Step 5: Modify reportstudio_en.xml**

A file on the IBM Cognos 10 BI server called `reportstudio_en.xml` supplies the translated English name of the template in the Report Studio New dialog. There are similar files for each supported locale. We need to complete the following steps in order to add our new template into reportstudio_en.xml:

1. On your IBM Cognos 10 BI server, find the file `<c10_install>\webcontent\pat\res\reportstudio_en.xml`.
2. Copy this file and save it as `reportstudio_en_BU.xml`.
3. Open the original reportstudio_en.xml in an XML editor.
4. Add the following line of XML. It doesn't matter where it is added.
   ```xml
   <string id="IDS_LBL_NEW_CORPORATE_TEMPLATE_1" type="Control Label">Corporate Template 1</string>
   ```
5. Save the reportstudio_en.xml file and close it.
6. Repeat steps 1-5 for any locales you intend to support by editing the corresponding `reportstudio_<locale>.xml` files.

After following these steps, our new corporate template will be available for use in Report Studio, as shown in Figure 17.
Create local classes and global class extensions

A template provides a great opportunity to create some local classes and/or modify global class extensions. These can be applied only to the current report or template and become part of the report or template XML specification. In comparison, layout component references (discussed in the next section) can be applied to any report or template. Both local classes and global class extensions have the same list of available properties to set. However, you can provide your own names for local classes as can be seen in the snippet of XML below where we have Outer_Gutter and Interior_Gutter classes. Conversely, the global class extension names (those with the name attribute having a value of “ch”, also seen below) are set by IBM Cognos Report Studio and map to the provided global style sheets.

```xml
<classStyles>
<classStyle name="cls1" label="Outer_Gutter" description="This is the padding etc in the outer table cells.">
  <CSS value="padding:20px;background-color:#DFDFDF;text-align:left;vertical-align:top"/>
</classStyle>
<classStyle name="cls2" label="Interior_Gutter" description="This is the padding etc in the inner table cells.">
  <CSS value="padding:20px;background-color:white;text-align:center;vertical-align:top"/>
</classStyle>
<classStyle name="ch"><CSS value="background-color:white"></CSS>
</classStyle>
</classStyles>
```

Create a local class

In Report Studio, you can create a new local class and assign it to any objects that will appear on the report multiple times. For example, if you have a layout table with 20 cells in it, you can create a local class that does such things as set the padding, background color, horizontal alignment, and vertical alignment and apply it to every cell. This way, if you need to make future modifications to the table cells (for example, reduce the padding), you only need to update the local class.
To create a new local class, select **Classes** from the **View** menu in Report Studio. As shown in Figure 18, drag a **Class** object from the **Toolbox** to the **Local Classes pane** and specify the properties for the class.

**Figure 18. Classes View showing the Toolbox, the Local Classes pane, the Global Class Extensions pane, and the Properties pane**

In our example, we have created a new local class called **Interior_Gutter** and set the following properties:

- **Label** - Interior_Gutter
- **Description** – The text “This is the padding etc. in the inner table cells”
- **Padding** - 20px 20px 20px 20px
- **Background Color** - White
- **Horizontal Alignment** - Left
- **Vertical Alignment** - Top

To apply a local class, exit the Classes View and return to **Page View** by selecting **Report Pages** from the **View** menu. Select an object (e.g. a specific Table Cell), scroll down to its **Classes** property, click the ellipsis button, select the class that you just created, and click the green arrow (Figure 19).
Figure 19. Classes dialog

Note that you can apply more than one class to an object. They can be local classes or global class extensions or a combination of both. To do this, in the **Selected classes pane**, use the up and down arrow buttons to specify their order. Classes are applied from top to bottom. The styling properties from all classes are merged together but if multiple classes specify the same property, the property from the last class in the Selected classes pane will override the properties from the previous classes.

**Modify global class extensions**

You currently can’t create a new global class extension in Report Studio, but you can modify an existing one by selecting it and changing its properties. As shown in Figure 20, we have selected the global class extension called **Chart** and changed the **Background Color** property to **White**. A pencil symbol now appears beside its icon to indicate that the class has been modified.

Figure 20. A modified global class extension

**Create layout component references**

When we design a report template, we often create a library of layout component references that can be accessed by the template.

A layout component reference is an object in IBM Cognos Report Studio’s Toolbox (Figure 21) that is used to create standard, repeated layout objects. They are commonly used for headers, footers,
and complex titles within a report template. You can specify the layout component reference once and reuse it many times in many different reports. In fact, using layout component reference objects has a similar effect to using classes for styling.

**Figure 21. Layout component reference object in Toolbox**

![Layout component reference object in Toolbox](image)

**Build library of layout component references**

A best practice is to place all potential layout component references in a single IBM Cognos Report Studio report that is named `Layout Component References`. This report, as shown in Figure 22, acts as a central library and is structured in a table with three columns:

- **Name** – The name of the layout component reference
- **Object** – the actual contents (objects) of the layout component reference
- **Description** – a brief description and the names of any overrides

**Figure 22. Layout component reference report in Report Studio**

<table>
<thead>
<tr>
<th>Name</th>
<th>Object</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td><code>&lt;Product Name Placeholder&gt; &lt;Report Title Placeholder&gt;</code></td>
<td>Override Objects: ProductName, ReportTitle, Logo</td>
</tr>
<tr>
<td>ChartTitle</td>
<td><code>Chart Title: ChartTitleText</code></td>
<td>This object should be used for all chart titles instead of the chart title property</td>
</tr>
</tbody>
</table>

Figure 22 shows two layout component references - **Header** and **ChartTitle**. Figure 23 provides a closer look at the layout component reference named Header. It contains an outer Table, an inner Table, and three Text Items.

**Tip:** All of the font styling for the Text Items has been specified on the Table Cells that contain them, not on the Text Items themselves. In this way, when we override the Text Items later (which deletes them), we will preserve the text formatting that we require after the Text Item is gone. For example, regarding the Product Name label, the properties that we set on the containing Table Cell are:

- **Foreground Color:** #676767
In the example above, the **name** property for the outer Table has been set to **Header** (Figure 24). This is how you designate an object as a layout component reference. If an object (such as the Header table) contains other objects (such as ProductName, ReportTitle, and Logo), then the child objects can be overridden.

**Tip:** It is often useful to wrap a layout component reference in a Table or Block, and assign this wrapper the Name.

Be aware that if you create any local classes or modify any global class extensions in a layout component reference’s source report, they will be local only to the layout component reference’s report and will be ignored by the parent report.

**How to insert a layout component reference**

To add a layout component reference to a report template, open the report template in Report Studio and drag a layout component reference object from the Toolbox (see Figure 21 above) onto the report where you would like it to be. In the resulting **Component Reference** dialog (Figure 25), select the **Another report** radio button and click the ellipsis button. Navigate to your layout component reference library report, select it, and click **OK**. The Component Reference dialog will now show a list of layout component references and their overrides in the **Available components to reference** pane. In our example we will select **Header** and click **OK**.
As shown in Figure 26, the layout component reference has been added to our template. Ensure that its **Embed** property is set to **Reference** so any changes made to the layout component reference’s source are reflected in every report that refers to it. If the Embed property is set to **Copy**, a copy of the external object as it stands at the time of insertion is stored in the report.

Now it is time to override the placeholders. Select the layout component reference we just added and click on the ellipsis button for the **Overrides** property. Select all of the objects that you would like to override and click **OK**. As shown in Figure 27, the objects selected for override are ProductName, ReportTitle and Logo.
In Figure 28 we see that drop zones have been added to each of the three overrides in the layout component reference. Now we will drag in two Text Items and one Image object from the Toolbox:

- The first new Text Item will become the product’s name and inherit the font from our corporate style guide that we require because it has been set on the Table Cell container. We just need to specify the exact text string we need.
- The second new Text Item will become the report’s name and inherit the corporate font we require. Again we just need to specify the required text string.
- The new Image object will display the company logo. We just need to select it and specify the URL property.

Tip: You can override a child object with any other object, not just an object of the same type. In the example above, the child object was a Text Item placeholder for a logo, but we replaced it with an Image. Another common strategy is to use a Text Item placeholder and then override it with a Chart object.
How to reload a modified layout component reference

If you modify the source of a layout component reference, all of the referenced objects in your existing reports will automatically pick up the changes when you open or run them. This is one of the strengths of this technique. However, if you are iteratively experimenting with a layout component reference and you want to preview the changes in a report that is already open and using the layout component reference, then perform the following three steps:

1. Select Layout Component Cache from the Tools menu.
2. Click the Reload Components button.
3. Click Close.

Chapter 5 - Modify Table Styles

Report authors can apply a table style to quickly, easily, and consistently format tables, lists, crosstabs, and repeater tables. This technique can be applied to both new and existing reports. The actual table styles are specified in a file called TableStyles.xml. This chapter will provide examples of how to modify many aspects of TableStyles.xml.

It is important to be aware of the following when modifying TableStyles.xml.

- Any changes that you make to this file are lost if you reinstall or upgrade IBM Cognos BI software.
- If you apply a table style and then later make any changes to the list or crosstab (e.g. add a new column, add a total, etc.) then you need to re-apply the table style again in order for the new pieces to pick it up.

You may still need to make small manual changes to your lists, crosstabs, etc. that are not covered by the applied table style. For example, in a list report, you may want to manually left-align your string columns and right-align your number columns.

A new feature in IBM Cognos BI version 10.2 allows new items that are inserted in existing lists or crosstabs to inherit the applied table style for its type.

For more information about table styles, please see the online IBM Cognos Report Studio User Guide specific to your IBM Cognos BI version.

How to apply a table style

As shown in Figure 29 below, to access the table styles you can right-click any cell in a list and select Style and then select Apply Table Style.
Figure 29. The Style context menu to apply a table style

![Image of Style context menu]

The resulting **Apply Table Style** dialog shown in Figure 30 below displays a list of table styles that you can apply along with a preview of the selected style.

Figure 30. Apply Table Style dialog

![Image of Apply Table Style dialog]

**Collisions with the global style sheet**

When making edits to TableStyles.xml, if you do not specify a property in TableStyles.xml but the same property is specified in the global style sheet as described earlier in Chapter 3, then the property from the global stylesheet will take precedence and be displayed. This is common with borders and images. Therefore, if you are editing TableStyles.xml and you are not getting the results that you are expecting when you run your reports, you will need to check that the global style sheet is not causing the problem. See Appendix B in the attached Appendices.zip file for more information.

If the same property is specified in both the global style sheet and TableStyles.xml with a different value in each, then the property value in TableStyles.xml will take precedence and be displayed.

Additionally, you can apply a table style to a list or crosstab and then overwrite any specific properties in Report Studio by selecting the object and changing its properties. This lowest level of properties always takes precedence in a collision.

**How to modify TableStyles.xml for lists**

Figure 31 shows an example of a newly-created list with the **Default** table style applied to it.
Figure 31. List with the Default table style applied

![Default table style applied](image)

However, we would like to apply our own corporate styling to our lists, as shown in Figure 32 where borders are removed and a grey scale color scheme is used instead of the default blue one.

Figure 32. List with corporate style guide applied

![Corporate style guide applied](image)

To facilitate this, we will modify the existing Modern table style for lists within the file called TableStyles.xml. Subsequently, every time we apply the Modern table style to a list in a new or existing report, the list will pick up the corporate style guide specification. The steps involved in the process of modifying TableStyles.xml are:

1. On your IBM Cognos BI server, find the file <c10_install>c10\webcontent\pat\res\TableStyles.xml.
2. Copy this file and save it as TableStyles_BU.xml.
3. Open the original TableStyles.xml in an XML editor.
4. We are going to modify the existing table style called Modern. Search the XML file for the text string `<tableStyle name="Modern" indentLevel="1">`.
5. Scroll down to the `<list>` tag.
6. Make your style modifications to each className's `<style>` tags (see Appendix A in the attached Appendices.zip file).
7. Save the file.
8. Clear your browser’s cache.

If you are making iterative changes to TableStyles.xml and would like to verify their effects while you work, you only need to re-open any reports and apply the table style again in order for the changes to take effect. You do not need to restart the IBM Cognos BI service.

Create new table style

Alternatively, rather than modifying the existing Modern table style and losing the original Modern specification, we could create our own new style and add it to the Apply Table Style dialog. The steps involved in this process are:

1. On your IBM Cognos BI server, find the file <c10_install>c10\webcontent\pat\res\TableStyles.xml.
2. Copy this file and save it as **TableStyles_BU.xml**.
3. Open the original TableStyles.xml in an XML editor. We are going to clone the existing table style named Modern.
4. Search the XML file for the text string `<tableStyle name="Modern" indentLevel="1">`.
5. Copy everything between the `<tableStyle>` and `</tableStyle>` tags and paste it anywhere within the file. The order of the styles in this file will dictate the order of the styles in the Report Studio dialog. Normally one would paste custom styles at the end.
6. Rename your new table style. In this example the new style will be called MyNewStyle and `<tableStyle>` tag should read `<tableStyle name="MyNewStyle" indentLevel="1">`.
7. Scroll down to the `<list>` tag.
8. Make your style modifications to each class name's `<style>` tags (see Appendix A in the attached Appendices.zip file).
9. Save the file.
10. On your IBM Cognos BI server, find the file `<c10_install>/webcontent/pat/res/reportstudio_en.xml`.
11. In this file, search for `IDS_TS_Modern` to locate the correct section and then create a new entry for the new style you just created in that same section. For example, `<string id="IDS_TS_MyNewStyle" type="List Item Label">MyNewStyle</string>`.
12. Save the file.
13. For other languages, repeat steps 7 -9 on the necessary reportstudio_xx.xml (where `xx` is the language code) files.
14. Clear your browser's cache, re-open Report Studio and continue authoring your reports. The new style should now be available.

### How to modify TableStyles.xml for crosstabs

Figure 33 shows an example of a newly-created crosstab with the **Default** table style applied to it.

**Figure 33. Crosstab with the Default table style**

![Default table style](image)

However, we would like to apply our own corporate styling to our crosstabs, as shown in Figure 35 where certain borders are removed and a grey scale color scheme is used instead of the default blue one.

**Figure 34. Crosstab with corporate style guide applied**

![Corporate style guide](image)

To facilitate this, we will modify the existing **Modern** table style for crosstabs within TableStyles.xml. Subsequently, every time we apply the Modern table style to a crosstab in a
new or existing report, the crosstab will pick up the corporate style guide specification. The steps involved in the process of modifying TableStyles.xml are:

1. On your IBM Cognos BI server, find the file `<c10_install>c10\webcontent\pat\res \TableStyles.xml`.
2. Copy this file and save it as `TableStyles_BU.xml`.
3. Open the original TableStyles.xml in an XML editor.
4. We are going to modify the existing table style called Modern. Search the XML file for the text string `<tableStyle name="Modern" indentLevel="1">`.
5. Scroll down to the `<crosstab>` tag.
6. Make your style modifications to each class name's `<style>` tags (see Appendix A in the attached Appendices.zip file).
7. Save the file.
8. Clear your browser's cache.

If you are making iterative changes to TableStyles.xml and would like to verify their effects while you work, you only need to re-open any reports and apply the table style again in order for the changes to take effect.

**Chapter 6 - Modify chart templates**

When report authors create a new chart in Report Studio, it picks up the default style from ChartTemplates.xml. This chapter provides guidance on how to modify many aspects of ChartTemplates.xml for **clustered column charts** (standard bar charts). The technique presented here can be used with all of the other Cognos chart types.

This technique is optimized for new charts. Existing charts will not pick up any modifications that you make to ChartTemplates.xml because this file is only accessed at the point when a new chart is created. However, you can rebuild an existing chart using the modified ChartTemplates.xml file and then delete the original chart.

This is a useful technique if we only have one corporate style for all charts. If we have multiple styles such as a "blue style" and a "green style", then we would need to:

- Create multiple versions of ChartTemplates.xml on the IBM Cognos BI server and selectively activate the required one when authoring reports, or,
- Employ multiple IBM Cognos BI servers for authoring, with each server having a different version of ChartTemplates.xml

There are 179 different chart types in IBM Cognos 10 BI! You will need to modify the style of each one that you intend to use in your reports. Fortunately most only use a small subset of the available charts.

It is important to be aware of the following when modifying ChartTemplates.xml.

- Any changes that you make to ChartTemplates.xml are lost if you reinstall or upgrade IBM Cognos BI software. You will need to manually restore them via a backup copy of the file.
• If you modify ChartTemplates.xml, all of your subsequent new charts will follow this style - you have lost the previous default style. IBM Cognos Report Studio 10 only supports one template per chart type.
• This technique can only be applied to new charts. To update existing reports, report authors will need to manually adjust the properties of their charts.

For more information about creating charts, please see Chapter 6 in the IBM Cognos Report Studio User Guide specific to your version.

How many chart types are there?

In IBM Cognos Report Studio, open the Tools menu and select Options. As shown in Figure 35, on the Advanced tab, ensure that the Use legacy chart authoring check box is not checked. In this way, you will only be able to use the new Cognos 10 charts.

Figure 35. Options dialog

In Report Studio 10.2, there are 179 default charts, divided into 14 categories:

• Column (24)
• Line (14)
• Pie, Donut (16)
• Bar (24)
• Area (24)
• Point (4)
• Combination (8)
• Scatter, Bubble (11)
• Bullet (4)
• Gauge (19)
• Pareto (4)
• Progressive (4)
• Microchart (10)
• Advanced (13)

As shown in Figure 36, if you drag in a Chart object from the Toolbox, the Insert Chart dialog will open (Figure 37), allowing you to choose which chart to create.
Figure 36. Chart object in Toolbox

Figure 37. Insert Chart dialog

Tip: You can use the tooltip and the icon for each chart type in the Insert Chart dialog to help you determine which element in ChartTemplates.xml refers to which chart type.

How to modify a chart’s style in ChartTemplates.xml

As shown in the code snippet below from the global Cognos style sheet, there are ten styles that affect charts.

```
Chart styles
------------
ct : chart title
cb : chart body
cn : chart note
cs : chart subtitle
cf : chart footer
lx : chart legend title
lg : chart legend
at : chart axis title
al : chart axis labels
ch : chart
```

The default values for these ten chart styles are specified in the global Cognos style sheet as follows:

```
/* Chart Styles */
/* -------------- */
```
Many, but not all, of the charting styles shown earlier are called in ChartTemplates.xml and are ready to be overwritten by us. They are lg, lx, at, al, and ch. Our starting point will be the default Cognos specification for a clustered column chart. This section of code can be found in the ChartTemplates.xml file by searching for the following opening element.

```
<chartTemplate name="v2_column_rectangle_clustered">
```

Figure 38 shows a completely default clustered column chart that has just been created in Report Studio.
Figure 38. Default clustered column chart in Report Studio

The default bar palette shown in Figure 39, contains 21 different colors with a linear gradient.

Figure 39. Default bar palette

The axis lines are black and the gridlines employ color banding (Figure 40).

Figure 40. Gridlines dialog

Figure 41 shows a modified clustered column chart in Report Studio that follows our corporate style guide, as specified in Chapter 1 where fonts and colors where defined.
Figure 41. Modified clustered column chart

We can modify ChartTemplates.xml so that all new clustered column charts follow our corporate style guide as soon as they are created. In essence we are going to pre-load their properties. The steps involved in modifying ChartTemplates.xml are:

1. On your IBM Cognos BI server, find the file `c10_install\c10\webcontent\pat\res\ChartTemplates.xml`.
2. Copy this file and save it as `ChartTemplates_BU.xml`.
3. Open the original ChartTemplates.xml in an XML editor.
4. We are going to modify the existing style for clustered column charts. Search the XML file for the text string `<!--V2 charts-->`.
5. Scroll down to `<chartTemplate name="v2_column_rectangle_clustered">`.
6. Make your style modifications (see Appendix C in the attached Appendices.zip file).
7. Save the file.
8. Clear your browser’s cache.
10. Drag in a new chart of the type that you just modified the style for in order for the changes to take effect. You do not need to re-start the IBM Cognos BI service.

When you modify ChartTemplates.xml and create a new chart in Report Studio that uses the modified style, you will see all of the modifications instantiated in the properties of each object that you select. Figure 42 shows the modifications made to the Axis Labels.

Figure 42. Axis Labels properties

Tip: If you need help in determining the correct property name or value that you are looking to modify, you can make the changes that you need manually in a chart in Report Studio and then inspect the XML to see the syntax for how it was implemented. You can also scroll through ChartTemplates.xml and explore the syntax.
Chapter 7 - Modify RAVE Charts

IBM Rapidly Adaptive Visualization Engine (RAVE) visualizations are a quick and easy way to create visual look and feel templates for your charts. These templates can be created to follow your corporate style guide to the letter. Once created your report authors can use the same RAVE visualization for all of your charts in the same class, promoting consistency.

This technique of using modified RAVE visualizations works best with new charts. For existing charts, you can rebuild your current charts with a new RAVE chart beside it for reference, and then delete the original chart. At this time, you can’t convert an existing chart into a RAVE chart.

Be aware that if you have an existing library of modified RAVE visualizations and you upgrade the IBM Cognos BI software, you will need to create a deployment prior to the upgrade in order to preserving your RAVE library. For more information about deployments, please see the Administration and Security Guide specific to your version of IBM Cognos BI.

Also note the following about RAVE visualizations:

- RAVE visualizations do not currently support conditional styles or conditional palettes, so if you are planning to highlight data that crosses a specified threshold in your charts, you will need to use regular IBM Cognos 10 BI charts instead.
- RAVE visualizations do not currently support the No Data Contents property in Report Studio. If a RAVE chart does not return any data it can only display an empty chart.

Download the IBM Cognos Visualization Customizer utility and samples

The first step in this process is to download and unzip the free Visualization Customizer utility from AnalyticsZone. A link to AnalyticsZone is found in the Resources section of this document. This utility has a prerequisite of 32-bit Java and is currently only supported on Microsoft Windows operating systems.

The second step is download the example RAVE visualizations from AnalyticsZone. The RAVE visualizations are ZIP files (do not unzip them!) and we can download them individually or we can download all of them in one ZIP and then extract them to see the individual ZIPs. In the next section we will open them in the Visualization Customizer utility and make modifications to match our corporate style guide.

Each individual sample visualization ZIP files is called a visualization bundle. It contains all of the files and images that IBM Cognos Report Studio requires.

For more information about the IBM Cognos Visualization Customizer utility, please see the Getting Started guide found on the download page for Visualization Customizer on AnalyticsZone.

How to modify an existing RAVE visualization

We can open any of the sample visualizations in the IBM Cognos Visualization Customizer utility and modify their appearance to match our corporate style guide. In this section, we will modify the default clustered column chart sample that is shown in Figure 43.
Figure 43. Clustered column chart with default styling

Figure 44 shows how our final modified clustered column chart will look with a blue color scheme.

Figure 44. Modified clustered column chart with a blue color scheme

Launch the IBM Cognos Visualization Customizer utility

We can launch the IBM Cognos Visualization Customizer utility by double-clicking the file VisCustomizer.jar (or equivalent as the filename may change). From the File menu, click Open. Browse to the sample file that we downloaded from AnalyticsZone called vis.sample.clusteredcolumn.default_10_2_1_1.zip and click OK. Open the File menu, select Save As, and save the new visualization with the final name that we would like it to have (for example vis.clusteredcolumn.corporate.zip).

In the IBM Cognos Visualization Customizer tool, we will see the JSON code in an editor panel on the left, and a preview of the chart on the right. Full details on how to use this utility can be found in the Getting Started guide found on the download page for Visualization Customizer on AnalyticsZone.

Modify Palette

The first thing that we will modify is the bar palette. In the JSON code, scroll down to the keyword "palette": and we will see the array of aesthetic values that make up the bar colors for the chart. We will modify the existing palette to include only five colors in various shades of blue by making
the following changes. Note that the final element in any array, ","#bfe2ef", does not have a comma after it.

```
"palette": [
  "#1b6a87",
  "#1183ad",
  "#53abcc",
  "#84c7e0",
  "#bfe2ef"
]
```

Click the **Draw Chart** button as shown in Figure 45 to verify the syntax of the JSON code and preview the results. Figure 46 shows the new color palette applied. Open the **File** menu, click **Save As**, and save your work with the same name (there is no Save option, just Save As).

**Figure 45. Draw Chart button**

![Figure 45. Draw Chart button](image)

**Figure 46. Color palette modifications applied**

![Figure 46. Color palette modifications applied](image)

**Modify Data**

The current JSON specification preview only shows three bars in the chart. We will modify the sample data to include five bars so that we can better preview all of our five new colors. This will also make our thumbnails look more representative of the specification. The first step is to add two more items to the series. Scroll up to the top of the JSON code to the `data` element and find the text "id":"series". Make the following modifications and then click the **Draw Chart** button to see a preview.
As shown in Figure 47, the series modifications are applied to the legend - there are now five items instead of three.

**Figure 47. Legend modifications applied**

![Legend modifications applied](image)

To make it easier to scan the chart and the JSON code, we will give more meaningful names to our three Categories. Right now they are all set to the value `abc`. Find the text `"id":"categories"` and make the following modifications:

```
"id":"categories",
"label":"Categories",
"categories":
[
  "AAA",
  "BBB",
  "CCC"
]
```

The results of our category modifications can be seen along the X-axis, as shown in Figure 48.
Figure 48. Category modifications applied

Now we need to add two more sets of data values (e.g. bars) to each of the three Categories. We do this within the "rows": element of the JSON code. I have highlighted the additions below in bold text:

```
"rows":
[ [
  0,
  0,
  15
]
[ 0,
  1,
  17
]
[ 0,
  2,
  16
]
[ 0,
  3,
  10
]
[ 0,
  4,
  8
]
[ 1,
  0,
  24
]
[ 1,
  1,
  22
]
[ 1,
  2,
  26
]
```
Figure 49 shows the six new rows (bars) that we added to the chart.

**Figure 49. Row modifications applied**

![Bar Chart](image)

**Modify Legend**

Our legend appears too close to the chart for our liking so this is something we will modify. We also need to slightly reduce the legend's font size. It is very important to verify that the background of
the legend is transparent in case we specify a background color for the chart or page and we want it to show through. Find the element "legends": and make the following modifications, highlighted in bold, to the JSON code. Click the **Draw Chart** button to see a preview.

```
"legends":
  [{
    "boundsStyle":
      {
        "fill":"transparent",
        "padding":
          {
            "left":"20%"
          },
        "valign":"start"
      },
    "scale":
      {
        "reverse":false,
        "local":false
      },
    "aesthetics":
      [
        {
          "aesthetic":
            {
              "$ref":"_PALETTE_"
            }
        }
      ],
    "labelStyle":
      {
        "fill":"#808080",
        "font":
          {
            "family":"Arial Unicode MS, Andale WT, Tahoma, Arial, MS UI Gothic, Gulim, SimSun, PMingLiU, Raghu8, sans-serif",
            "size":"8pt"
          },
        "location":
          {
            "width":"20%"
          },
        "title":
          [
            {
              "$ref":"series"
            }
          ],
        "titleStyle":
          {
            "align":"start",
            "fill":"#808080",
            "font":
              {
                "family":"Arial Unicode MS, Andale WT, Tahoma, Arial, MS UI Gothic, Gulim, SimSun, PMingLiU, Raghu8, sans-serif",
                "size":"8pt",
                "weight":"bold"
              },
            "padding":
              {
                "bottom":3
              }
          }
  }
```
Here are some specific things to note about the modifications:

- The `legendPosition` has been set to `right` but it could also be `left`, `top` or `bottom`.
- The `location` element specifies that the chart will take up 80% of the assigned width and the legend will take up 20%.
- The `padding` element under `titleStyle` indicates that the legend title will have 3 pixels of padding below it before the legend values starts.
- `"fill":"#808080"` specifies the font color of the legend's title and the legend's labels.
- RAVE will automatically truncate the legend labels according to how much space has been assigned to the chart.

As shown in Figure 50, we can see the results of our modifications to the legend.

**Figure 50. Legend modifications applied**

### Modify Overall Chart Properties

It is a good idea to specify a default `size` for our charts in our corporate style guide. We need to optimize their width so that they are not too small to read and we can differentiate between bars and labels or so large that there is too much empty space between the elements. We can only set the `width` and `height` of a visualization in pixels (e.g. 600 by 300). At this time, we can’t set it using percentages (e.g. 75% by 50%).
Tip: The size of a visualization can be changed in Report Studio via the RAVE chart's properties. Other than this, for the versions applicable in this document, there are not many properties that can be modified in Report Studio – they are all specified within the JSON code itself. However, in IBM Cognos BI 10.2.2, new visualizations are available that support several out-of-the-box modifiable properties in Report Studio. And beyond this, visualizations can be authored or modified to expose properties as required using the Property Model in the JSON code.

It is also very important to verify that the background of the chart and legend is set to transparent, in case we specify a background color for the page and we want it to also appear as the background color for the chart and legend. Alternatively, we could also assign a specific color to the chart’s background. Both the chart and the legend have a `fill` element in the JSON code. We need to be careful with transparencies. For example, if we have a transparent chart or legend with black text everywhere and we change the background to black, all of the text will disappear into the background.

The `padding` element refers to the space around the outside of the chart.

Scroll down to the bottom of the JSON code to the `size` element, make the following modifications, highlighted in bold, where we are simply changing the width and height to 600 and 400 respectively, and then click the **Draw Chart** button to see a preview.

```
"size":
{
  "width":600,
  "height":400
},
"style":
{
  "padding":"2%",
  "fill":"transparent"
}
```

As shown in Figure 51, we can see the results of the final changes to the chart. We are now finished making our modifications and the chart complies with our corporate style guide.

**Figure 51. Chart modifications applied**

![Chart modifications applied](image)

Modify String Resources
Now that we are finished modifying the JSON specification, we can proceed with finalizing the visualization bundle. In the IBM Cognos Visualization Customizer utility, click on the **Chart Type Definition** tab. Full details on how to use this tab and what it is doing behind the scenes can be found in the *Getting Started* guide found on the download page for Visualization Customizer on AnalyticsZone.

At the top of the screen, note the **Bundle Name** field. It will contain the original sample's name that we are modifying. In Figure 52, the bundle name is `vis.sample.clusteredcolumn.default_10_2_1_1`. Before proceeding with the steps below, we must manually change this text to our new filename that we have been saving our visualization bundle to - in this case the new bundle name would be `vis.clusteredcolumn.corporate`.

**Figure 52. Chart Type Definition tab**

Click on the **Generate Viz Bundle Definition** button to update the chart specification based on the JSON grammar that we modified in the previous sections. The utility will also update all of the supporting string resource files.

Click on the **Validate Bundle** button to verify the syntax and structure.

Click on the **String Resources** sub-tab to view the language file. The default strings have been generated in English. These strings are used in IBM Cognos Report Studio to display:
• The Bundle Name (vis.clusteredcolumn.corporate in Figure 51 above) which is used in IBM Cognos Administration on the Library tab and in the Visualization Gallery as the ID.
• The Bundle Tags (Column Corporate in Figure 51 above) which are tags that allow the user to refine their search for a specific chart in the Visualization Gallery.

**Update Thumbnails**

In the IBM Cognos Visualization Customizer utility, click on the Thumbnails tab and click the Generate Thumbnails button (Figure 53).

**Figure 53. Thumbnails tab**

![Thumbnails tab](image)

These thumbnail images are used in IBM Cognos Report Studio in the Visualization Gallery dialog (Figure 54).
We are now finished modifying the sample clustered column chart to match our corporate style guide. Open the **File** menu, click **Save As**, and save your work with the same name.

### A few words about treemaps

When using treemaps, for the **positioning** element, use "layout":"squarify" instead of "layout":"pivotBySize". Squarify will place the biggest square at the top left and the smallest square at the top right.

```
"positioning":
{
  "layout":"squarify",
  "leafOnly":false,
}
```

Sample visualizations that place labels over the palette, which is common for treemaps, employ a dark shadow on the labels. This syntax is highlighted in bold below. In this way, the text is both light and dark, thus viewable and accessible on all of the palette background colors.

```
"leaf":
{
  "content":
  [
    {
      "$ref":"category1"
    }
  ],
  "style":
  {
    "fill":
    {
      "a":0.85,
      "h":0,
      "s":0,
    }
  }
```

---

**Figure 54. Report Studio Visualization Gallery dialog**
The elements that can be set for the shadow are:

- \texttt{dx} - the horizontal offset of the shadow
- \texttt{dy} - the vertical offset of the shadow
- \texttt{stdDev} - the amount of blurring applied to the shadow
- \texttt{color} - the color of the shadow (default is black)

Figure 55 provides an example of white text with a black shadow overlaid on a multi-colored treemap palette. This approach makes it much easier to read the label, regardless of the palette color below it.

**Figure 55. Treemap text shadow applied**

You should take note of the behaviour of the legend and color assignments for treemaps. The legend will show the palette in the correct order with the Categories arranged alphabetically instead of by size. Therefore, in a treemap, the largest square at the top left will most likely not be in the first color of the palette. It will be assigned the color that corresponds to its alphabetical order. In the example above, AAA is the first color in the legend and is the first square, but BBB is the third smallest square. This is done so that if we drill down or interact with the treemap, each Category maintains the same color regardless of size (e.g. Canada will always be Red).
Import the modified visualization bundle

Now that we have modified and saved the clustered column visualization bundle, how do we make it available to our report authors in our IBM Cognos BI environment? Full details on this process can be found in the Getting Started guide found on the download page for Visualization Customizer on AnalyticsZone.

First, open IBM Cognos Connection in your browser and select **IBM Cognos Administration** from the **Launch** menu (Figure 56).

**Figure 56. IBM Cognos Connection Launch menu**

The **Library** tab is used to import and display the environment’s visualization bundles. To see and manage content on the Library tab, you need to be a member of the **Library Administrators** or **System Administrators** role.

Visualizations must be imported into the Library before they are available to use in Report Studio. As shown in Figure 57, click the **Import Visualization** toolbar button.

**Figure 57. Import Visualization toolbar button on Library tab**

In the resulting dialog (Figure 58), click the **Browse** link and navigate to the modified visualization bundle called **vis.clusteredcolumn.corporate.zip**. Select this file and click **Open**. If you have re-importing existing visualization bundle, select the **Replace existing entries** checkbox. Click the **Import** button and wait for the text **(Succeeded)** to appear. Click **Close**.
Figure 58. A successful visualization import dialog

![Success dialog](image)

Figure 59 shows the Library which now contains three visualizations.

Figure 59. Library tab with three visualizations

![Library tab](image)

How to use the modified visualization bundle in IBM Cognos Report Studio

Now that we have published our modified corporate visualization bundle, how do we use it in Report Studio?

Open Report Studio and create a new blank report. From the Toolbox, drag a **Visualization** object (Figure 60) onto the report. The Visualization Gallery dialog, shown in Figure 54 above, will open.
In the Visualization Gallery dialog, select the Clustered – Column chart that we modified and click OK. The report will now contain an object for the chart as shown in Figure 61.

If we select the chart object, we are shown a small list of properties that we can manually modify (Figure 62). As can be seen there are only minimal overrides that we can make via the properties including Data Format, Width, Height, and Name.

If we select the Data Container (Figure 63), we are shown another small list of available properties (Figure 64) that includes properties such as Query, Tooltips, Name, and Maximum Hotspots.
Figure 63. Data Container is selected

![Data Container](image)

Figure 64. Data Container properties

![Data Container Properties](image)

Now we can see that most of the chart specification is inherited from the JSON grammar in the visualization bundle. This is an easy and efficient way to ensure that all of our clustered column charts follow our corporate style guide.

To finish our chart, we need to drag in some data from the Package (data source) into the Categories and Series placeholders.

If you make further changes to an existing visualization bundle that is in uses in a report, you can update the existing charts by re-opening the report in Report Studio and selecting Update Visualizations from the Tools menu.

The before and after JSON code for our example can be found in Appendix D in the attached Appendices.zip file.

Chapter 8 - Add Corporate Palettes

This chapter describes how to add new corporate palettes into the Report Studio palette dialog as shown in Figure 65. These chart palettes are specified in the XML file `<c10_install>\webcontent \pat\res\ReportPresets.xml`. 
There are 17 chart palettes provided by Cognos each having its own element in the ReportPresets.xml. Each chart palette is further divided into 4 sub-palettes:

- `<v2_solidPalette>` is used by bar, column, pie, gauge, and pareto charts.
- `<v2_linePalette>` is used by line charts.
- `<v2_pointPalette>` is used by scatter and bubble charts.
- `<v2_progressivePalette>` is used by progressive charts.

This process has been fully documented in the developerWorks article titled Create Custom Palettes in IBM Cognos 10 Report Studio so it won't be repeated here.

An alternative approach is to define the palette for each chart type directly in ChartTemplates.xml as described in Chapter 6.
## Downloads

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Appendices</td>
<td>814KB</td>
</tr>
</tbody>
</table>
Resources

- W3C Web Content Accessibility Guidelines
- AnalyticsZone
About the author

Steve Macko

Steve Macko is an Advisory Designer for the IBM Samples Team. He has been with IBM Cognos for over 18 years. Steve began his Cognos career as a usability/user research specialist and Manager. Three years ago he became a full-time report author, living and breathing in Report Studio, and applying his usability and user experience design skills to all of his authoring projects.

© Copyright IBM Corporation 2015
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)