Embedding Custom Images Inside Tables in Web Templates

**Applies to:**
SAP Business Intelligence (BI).

**Summary**
This article describes how to use a table data provider in BI Web template with the table API to render an image dynamically. It also details out special care to be taken for managing styles.

**Author(s):** Hitesh Joshi

**Company:** Larsen & Toubro Infotech Ltd

**Created on:** 03 February 2007

**Author Bio**
Hitesh Joshi has more than two years of BW experience and is currently working as BI Consultant in Larsen & Toubro Infotech Ltd. During this period he has extensively worked on production support as well as development projects as a BI Consultant.
# Table of Contents

Introduction ...................................................................................................................................... 3  
Scenario........................................................................................................................................... 3  
BW Query output with custom image............................................................................................... 3  
  LED GIF Images in MIME repository ........................................................................................... 4  
Table API......................................................................................................................................... 6  
BW Query ........................................................................................................................................ 9  
Related Content............................................................................................................................. 21  
Disclaimer and Liability Notice....................................................................................................... 22
Introduction

The standard BW template provided by SAP in 3.0B and 3.5 is capable of showing data in tabular and graphical ways. If a BW query shows some comparison (ex. Actual v/s Plan) it is easier to show the variance in form of an indicator such as traffic signal or LED image for the exception defined on the query.

Scenario

Recently my customer asked me to produce a report that plotted values for the table records displayed on the page with some image indicator on the variance. As the user observed through the records in the table, they didn’t like the entire contents of exception cell getting coloured, instead they wanted some color image, which fits in the current theme and also highlights the variance deviation. Since it was a very specific requirement, I decided to use MODIFY_CLASS and traffic gif icons to accomplish this task. In this article, I will explain how I used a sample BW template with the table API to render an icon dynamically.

BW Query output with custom image.

The following screen shows BW query output with my class, rendered dynamically.

<table>
<thead>
<tr>
<th>Sales Office</th>
<th>Revenue (actu als)</th>
<th>Revenue (Plan)</th>
<th>Act/Plan Variance</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>▼ STEEL</td>
<td>14,948,708 INR</td>
<td>14,898,655 INR</td>
<td>0.34%</td>
<td>🟢</td>
</tr>
<tr>
<td>▲ NORTH</td>
<td>1,235,741 INR</td>
<td>1,204,554 INR</td>
<td>2.57%</td>
<td>🟢</td>
</tr>
<tr>
<td>▼ SOUTH</td>
<td>6,141,385 INR</td>
<td>6,100,777 INR</td>
<td>0.67%</td>
<td>🟢</td>
</tr>
<tr>
<td>▼ EAST</td>
<td>1,595,125 INR</td>
<td>1,635,555 INR</td>
<td>-2.29%</td>
<td>🟢</td>
</tr>
<tr>
<td>▼ West</td>
<td>1,523,204 INR</td>
<td>1,567,174 INR</td>
<td>-2.81%</td>
<td>🟢</td>
</tr>
<tr>
<td>▼ INTL</td>
<td>4,452,254 INR</td>
<td>4,393,584 INR</td>
<td>1.34%</td>
<td>🟢</td>
</tr>
</tbody>
</table>

The web page displays BW Query output in tabular form with an embedded LED image. The table shows data for two key figures for given characteristics combination. Using the navigation block, the user can expand down on the hierarchy or can scroll the table records; the LED image is dynamically rendered based on the key figures visible in the table.

Let’s first understand the challenges.

The first challenge was to get the proper images suitable for this task. I was looking at some sample images for this purpose; I found some good LED style GIFs in MIME repository.
LED GIF Images in MIME repository

The second challenge was to render these gif’s in table control dynamically. I looked for a mechanism to help me display this image in the table view. A good option is to use Web Design API for tables. Please refer to the SAP documentation on the Table API – manipulating Cell Content. The table API Method DATA_CELL can be used to produce the additional row and render the image.

Method DATA_CELL can help get the required image information and related values. Let’s see the components involved in this sample.

There are three key figures. The Revenue (Actuals) and Revenue (Plan) key figures are displayed as currency amount, and the Variance key figure depicts the variance between Actuals and Plan as percentage. The Status renders as an image against the values of variance. Let’s see the typical HTML / XML coding required displaying this image on a web page.

The HTML tag <OBJECT> is used here to display the table. This OBJECT tag has typical XML structure.

```
<object>
  <param name="OWNER" value="SAP_BW"/>
  <param name="CMD" value="GET_ITEM"/>
  <param name="NAME" value="TABLE_1"/>
  <param name="ITEM_CLASS" value="CL_RS_RWITEM_GRID"/>
  <param name="DATAPROVIDER" value="DATAPROVIDER_1"/
  <param name="WIDTH" value="700"/>
  <param name="CAPTION" value="Fixed Cost at Cost Center"/>
  <param name="BORDER_STYLE" value="NO_BORDER"/>
  <param name="ALT_STYLES" value=""/>
  <param name="MODIFY_CLASS" value="ZCL_F2F_BW_E2E_GRID"/>
  ITEM: TABLE_1
</object>
```

The HTML tag <OBJECT> is used here to display the table. This OBJECT tag has typical XML structure.

Among the various PARAM tags, one consists of query data in typical XML format. This XML structure consists of OWNER and CMD tags along with their attributes and values. The node name <ITEM_CLASS>
defines the base class as CL_RSR_WWW_ITEM_GRID. <DATAPROVIDER> contains the name of the dataprovider created in the template.

As you can see, this table control is typically designed to display values derived from the query in grid format. In short, this coding needs to be made in the template or table API that will render the web page to show the table along with image.

The most important parameter for us is MODIFY_CLASS

\[<\text{param name}="\text{MODIFY\_CLASS}" \text{value}="\text{ZCL\_E2E\_BW\_E2E\_GRID}"/>\]

I have set value of MODIFY_CLASS to ZCL_E2E_BW_E2E_GRID. This ABAP class or API has following details.
### Table API

The Property settings for the API class are as follows. Super class is `ZCL_RSR_WWW_MODIFY_TABLE`.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interfaces</th>
<th>Friends</th>
<th>Attributes</th>
<th>Methods</th>
<th>Events</th>
<th>Types</th>
<th>Aliases</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class Interface</td>
<td><code>ZCL_E2E_BW_E2E_GRID</code></td>
<td></td>
<td></td>
<td>Implemented / Active</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**General Data**

- Released internally
- Fixed point arithmetic
- Unicode checks active
- Shared Memory-Enabled

- Message Class
- Program status
- Category: General object type
- Package: $TMP
- Original Language

- Created: DEVELOPER 08/03/2006
- Last change: DEVELOPER 08/03/2006
The inherited attributes, are listed as follows

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Level</th>
<th>Vis.</th>
<th>Re.</th>
<th>Typing</th>
<th>Associated Type</th>
<th>Description</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>N_R_DATA_SET</td>
<td>Instan.</td>
<td>Pub.</td>
<td></td>
<td>Type</td>
<td>CL_RSR_DATA_SET</td>
<td>Data Definition</td>
<td></td>
</tr>
<tr>
<td>N_GENERATE_LINKS</td>
<td>Instan.</td>
<td>Pub.</td>
<td></td>
<td>Type</td>
<td>RS_BOOL</td>
<td>Generate Navigation Links</td>
<td></td>
</tr>
<tr>
<td>N_STATELESS</td>
<td>Instan.</td>
<td>Pub.</td>
<td></td>
<td>Type</td>
<td>RS_BOOL</td>
<td>Page Displayed in State</td>
<td></td>
</tr>
<tr>
<td>N_DATA_PROVIDER_NAME</td>
<td>Instan.</td>
<td>Pub.</td>
<td></td>
<td>Type</td>
<td>STRING</td>
<td>Name of Data Provider, f</td>
<td></td>
</tr>
<tr>
<td>N_JAVASCRIPT</td>
<td>Instan.</td>
<td>Pub.</td>
<td></td>
<td>Type</td>
<td>RS_BOOL</td>
<td>Java Script Active</td>
<td></td>
</tr>
<tr>
<td>N_ALT_STYLES</td>
<td>Instan.</td>
<td>Pub.</td>
<td></td>
<td>Type</td>
<td>RS_BOOL</td>
<td>Alternating Row Format</td>
<td></td>
</tr>
<tr>
<td>N_ITEM_NAME</td>
<td>Instan.</td>
<td>Pub.</td>
<td></td>
<td>Type</td>
<td>STRING</td>
<td>Name of Item</td>
<td></td>
</tr>
<tr>
<td>N_NO_DATA</td>
<td>Instan.</td>
<td>Pub.</td>
<td></td>
<td>Type</td>
<td>RS_BOOL</td>
<td>No Data Found</td>
<td></td>
</tr>
<tr>
<td>N_NO_AUTHORIZATION</td>
<td>Instan.</td>
<td>Pub.</td>
<td></td>
<td>Type</td>
<td>RS_BOOL</td>
<td>No Authorization</td>
<td></td>
</tr>
<tr>
<td>N_R_CREATION_PARAME</td>
<td>Instan.</td>
<td>Pub.</td>
<td></td>
<td>Type</td>
<td>CL_RSR_PARAMETER</td>
<td>Item Creation Parameter</td>
<td></td>
</tr>
</tbody>
</table>
Among the various methods, I have overridden the DATA_CELL method to embed the image.

Now let's look at the details of the BW query.
BW Query

My BW query design is as follows.

As you can see, various free characteristics and two key figures are used in the design. I have placed key figures in columns and only one characteristic in rows.
‘Actual Costs’ Restricted key figure is defined as shown below

<table>
<thead>
<tr>
<th>Description</th>
<th>Actual Costs</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Costs Summary Cube</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calculated Key Figure</td>
<td>Value type</td>
</tr>
<tr>
<td>Variance</td>
<td>Actual</td>
</tr>
<tr>
<td>Restricted Key Figures</td>
<td>Key Figure</td>
</tr>
<tr>
<td>Actual Costs</td>
<td>Amount of cost</td>
</tr>
<tr>
<td>Plan Costs</td>
<td></td>
</tr>
<tr>
<td>Key Figure(A) 1</td>
<td></td>
</tr>
<tr>
<td>Key Figure(N) 3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dimensions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost Center</td>
<td></td>
</tr>
<tr>
<td>Cost Element</td>
<td></td>
</tr>
<tr>
<td>Valuation</td>
<td></td>
</tr>
<tr>
<td>Timo</td>
<td></td>
</tr>
<tr>
<td>Unit</td>
<td></td>
</tr>
<tr>
<td>Data Package</td>
<td></td>
</tr>
</tbody>
</table>


‘Plan Costs’ Restricted key figure is defined as shown below
‘Variance’ is calculated as shown below

![Edit Formula Window]

<table>
<thead>
<tr>
<th>Description</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variance</td>
<td>'Actual Costs' % 'Plan Costs'</td>
</tr>
</tbody>
</table>

**Operands**

- Structure Elements
  - Actual Costs
  - Plan Costs
  - Status

- Formula Variable

**Functions**

- Basic Functions
  - High

- Percentage Functions

- Mathematical Functions

- Data Functions

- Trigonometric Functions

- Boolean Operators
Status is defined same as variance
Exception “Variance Tolerance” is defined on status column as shown below.

Cell content is available in the form of an HTML code string in the `C_CELL_CONTENT` parameter of the `DATA_CELL` method. HTML content strings relate to cell value content.

Typical HTML string in the `C_CELL_CONTENT` parameter of `DATA_CELL` is as follows.

```html
String: <c_cell_content>
Type: STRING
Current Length (in Characters): 118
Content (fr. Offset): 0 - 118

<a href="#"><img src="/sap/bw/Mime/IEx/Icons/imo_s_s_1edg.gif" border="0" alt="Good" width=15 height=15 ></a>&nbsp;
```

This HTML will focus on the cell inner part that is the cell value only.
Sales Office | Revenue (actials)
---|---
STEEL | 14,948.708 INR
NORTH | 1,236.741 INR
SOUTH | 6,141.395 INR
EAST | 1,595.125 INR
West | 1,523.204 INR
INTL | 4,452.254 INR

To render this cell in the HTML table, this string is enclosed in `<TD>` and `</TD>` html tag as shown below.

```
<td nowrap class="SAPBEXstdData"><a title="Context Menu for 14,948.708 INR" href="Javascript:;onclick="return false;" onContextMenu="SAPBWM(event,5,3,'','','','',3,1);return false;">14,948.708 INR</a></td>
```

This outer HTML is not available to edit in the API method call. But we can append further HTML strings to C_CELL_CONTENT to get more cells or rows in table.

See the DATA_CELL method code.

```
method DATA_CELL.
*CALL METHOD SUPER->DATA_CELL
* EXPORTING
*  I_X =
*  I_Y =
*  I_VALUE =
*  I_DISPLAY_VALUE =
*  I_NUMERICAL_SCALE =
*  I_NUMERICAL_PRECISION =
*  I_CURRENCY =
*  I_UNIT =
*  I_ALERTLEVEL =
*  I_IS_SUM =
* CHANGING
*  C_CELL_ID =
*  C_CELL_CONTENT =
*  C_CELL_STYLE =
*  C_CELL_TD_EXTEND =
*  .
data: l_r_url_red type string,
       l_r_url_green type string,
       l_r_url_yellow type string,
       l_r_strg_ALT_red type string,
       l_r_strg_ALT_green type string,
```
l_r_strg_ALT_yellow       type string,
l_r_tmp_var type string.

* Set text for alternative if no pic
l_r_strg_ALT_red = 'Critical'.
l_r_strg_ALT_green = 'Good'.
l_r_strg_ALT_yellow = 'Moderate'.

* set picture either RED, GREEN or YELLOW
l_r_url_red         = '/sap/bw/Mime/BEx/Icons/imo_s_s_ledr.gif'.
l_r_url_green       = '/sap/bw/Mime/BEx/Icons/imo_s_s_ledg.gif'.
l_r_url_yellow      = '/sap/bw/Mime/BEx/Icons/imo_s_s_ledy.gif'.

* Check alert level by I_ALERTLEVEL

case I_ALERTLEVEL.
* green
  When 1.
    l_r_tmp_var = ' '.
    concatenate '<a href="#"> <img src="'
                  l_r_url_green
      " border="0" alt="'
                  l_r_strg_ALT_green
      " width=15 height=15 ></a>&nbsp;'
    l_r_tmp_var
    into l_r_tmp_var.
    c_cell_content = l_r_tmp_var.
    C_CELL_STYLE = 'zstatus'.
  when 2.
    l_r_tmp_var = ' '.
    concatenate '<a href="#"> <img src="'
                  l_r_url_green
      " border="0" alt="'
                  l_r_strg_ALT_green
      " width=15 height=15 ></a>&nbsp;'
    l_r_tmp_var
    into l_r_tmp_var.
    c_cell_content = l_r_tmp_var.
    C_CELL_STYLE = 'zstatus'.
  when 3.
l_r_tmp_var = ''
concatenate '<a href="#"> <img src="l_r_url_green" border="0" alt="l_r_strg_ALT_green" width=15 height=15 ></a>&nbsp;' l_r_tmp_var into l_r_tmp_var.
c_cell_content = l_r_tmp_var.
C_CELLSTYLE = 'zstatus'.

yellow

when 4.
concatenate '<a href="#"> <img src="l_r_url_yellow" border="0" alt="l_r_strg_ALT_yellow" width=15 height=15 ></a>&nbsp;' l_r_tmp_var into l_r_tmp_var.
c_cell_content = l_r_tmp_var.
C_CELLSTYLE = 'zstatus'.

when 5.
concatenate '<a href="#"> <img src="l_r_url_yellow" border="0" alt="l_r_strg_ALT_yellow" width=15 height=15 ></a>&nbsp;' l_r_tmp_var into l_r_tmp_var.
c_cell_content = l_r_tmp_var.
C_CELLSTYLE = 'zstatus'.

when 6.
l_r_tmp_var = ''
concatenate '<a href="#"> <img src="l_r_url_yellow" border="0" alt="l_r_strg_ALT_yellow" width=15 height=15 "></a>&nbsp;' l_r_tmp_var into l_r_tmp_var.
c_cell_content = l_r_tmp_var.
C_CELL_STYLE = 'zstatus'.
*red
when 7.
  l_r_tmp_var = '.
  concatenate '<a href="#" > <img src="'
    l_r_url_red
    '" border="0" alt="'
    l_r_strg_ALT_red
    '" width=15 height=15 ></a>&nbsp;' 
  l_r_tmp_var
  into l_r_tmp_var.
  c_cell_content = l_r_tmp_var.
  C_CELL_STYLE = 'zstatus'.
when 8.
  l_r_tmp_var = '.
  concatenate '<a href="#" > <img src="'
    l_r_url_red
    '" border="0" alt="'
    l_r_strg_ALT_red
    '" width=15 height=15 ></a>&nbsp;' 
  l_r_tmp_var
  into l_r_tmp_var.
  c_cell_content = l_r_tmp_var.
  C_CELL_STYLE = 'zstatus'.
when 9.
  l_r_tmp_var = '.
  concatenate '<a href="#" > <img src="'
    l_r_url_red
    '" border="0" alt="'
    l_r_strg_ALT_red
    '" width=15 height=15 ></a>&nbsp;' 
  l_r_tmp_var
  into l_r_tmp_var.
  c_cell_content = l_r_tmp_var.
  C_CELL_STYLE = 'zstatus'.
Endcase.
endmethod.
The exceptions are defined for status column and are passed in `_ALERTLEVEL` field to the `DATA_CELL` method. Case statement captures this alert value and accordingly set the contents of cell to green, yellow or red icons.

To override the default style imposed by `c_cell_content` and to keep the existing style of the table, a "zstatus.css" file was created and kept in MIME repository.

Contents of `zstatus.css` file looks like this

```css
.CELL_STYLE = 'zstatus'.
```

The statement

```
C_CELL_STYLE = 'zstatus'.
```

Sets the style of the status cell to the table style.

The results are amazing. I have further added conditional items to the template to view the results of conditions. This works perfectly, as required. In this example we have focus to embed the custom image at
the end of the table. We can develop solution to embed custom image in additional column of the table as well.
Related Content

Please refer to the SAP documentation on the

- Web Design API for tables
- Table API – manipulating Cell Content.
- Storage of Stylesheets and Symbols
Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.