

WEB DASHBOARDS

COMPLETE REFERENCE

This is the reference guide for InfoCaptor “Web” Dashboards.

Contents

Introduction	5
Purpose	5
Web Dashboards	6
Install Web Dashboards Server Components	7
Web Dashboards Architecture	11
report_engine.php	12
dash/dashboard_definition.php	13
dbconfig/dashboard_database_connection.php	13
report_engine.css	14
Other Directories and files	15
license	15
chart_templates	15
Convert Desktop Dashboard to Web Dashboard	16
Desktop Dashboard to Raw PHP Definition	16
Structure of Database Configuration File	24
Structure of Dashboard Definition File	26
General Dashboard properties	26
Utility Functions	28
Dashboard Parameters	29
['type']	30
['data_type']	30
['id']	30
['name']	30
['raw_type']	30
['connection']	30
['width']	30
['height']	30
['x_position']	31
['y_position']	31
['format']	31
['font']	31
['font_color']	32
['background_color']	32
['default_value']	32
['enable_all']	32
['all_label']	33
['all_value']	33

['display_members']	33
['control_type'].....	33
['default_null_value']	33
['static_values']	34
['static_display_values']	34
['sql']	35
More Dashboard and Parameter Properties	36
\$d_options['top_button_x_location'] :	36
\$d_options['top_button_y_location'].....	36
\$d_options['top_button_width'].....	36
\$d_options['top_button_height']	36
\$d_options['use_go_checkobx_x_location']	36
\$plet['list']	36
\$plet_group.....	37
Table and Chart Definitions	38
['id']	38
['name'].....	38
['type']	38
['connection'].....	38
['width']	38
['height'].....	38
['x_position']	39
['y_position']	39
['max_rows'].....	39
['column_names']	39
['number_of_columns'].....	39
['show_pager'].....	39
['column_display_names'].....	39
['column_alignment']	39
['column_class']	39
['column_width']	39
['template'].....	39
['background_color']	42
['enable_column_sort']	42
['dynamic_script'].....	42
['total_script'].....	42
['sql']	42
['chart_margin'].....	44
['chart_size']	44
['legend'].....	45
Gauge Specific Settings.....	46
['x_center'] , ['y_center']	46
['radius'].....	46
['range_radius']	46
['tick_radius']	46
['number_radius']	46
['major_ticks'].....	46
['minor_ticks'].....	46
['ticks_range_angle'].....	47
['tick_numbers'].....	47
['range'].....	48
['pointer']['radius']	48
['pointer']['points']	48
['text_x'] , ['text_y'].....	49
Chart Settings.....	50

CHART BORDER	50
CHART GRID	52
CHART GUIDE.....	54
CHART LABEL.....	56
SERIES	58
SERIES COLOR	59
AXIS CATEGORY.....	59
AXIS CATEGORY LABEL.....	62
AXIS TICKS.....	63
AXIS VALUE.....	65
AXIS VALUE LABEL.....	67
FILTER.....	68
DRAW	72
LEGEND	82

Introduction

Purpose

Explains the process to convert Desktop based Dashboards to Web based Dashboards.

Terms

Desktop Dashboard Designer: This is the application that helps you create dashboards on your desktop

Desktop Dashboard definition file: This definition is stored in the “.icv” file

Web Dashboard: This is the web version of the same Desktop Dashboard definition. The .icv file is converted into a set of php files for presenting on the web. The web dashboard uses a Flash based charting engine. The desktop dashboard uses a Java based charting engine. So there is an obvious difference in the look and feel between the desktop and web versions.

The web version inherits all the connections and queries and portlet properties so it can render it in the browser.

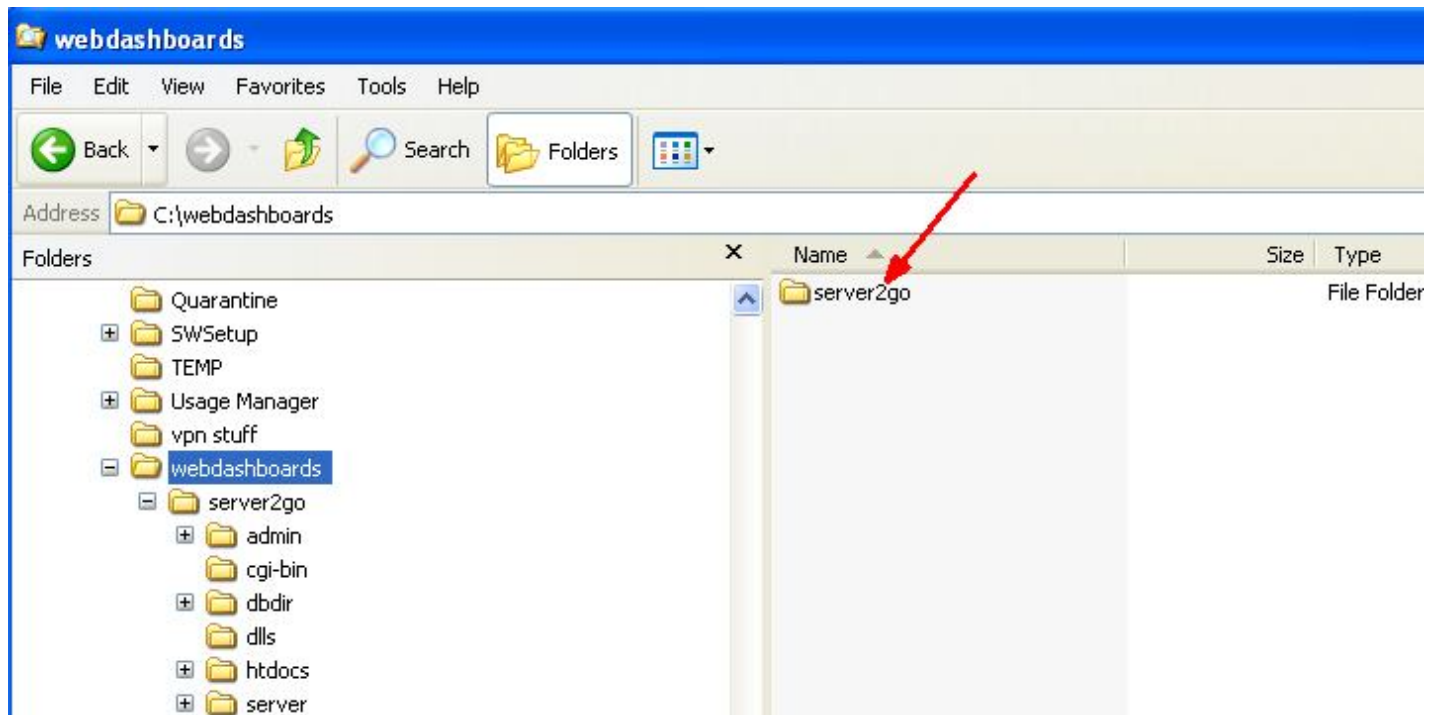
Web Dashboards

Benefits of Web dashboard

- Easy and global access: The web dashboards can be accessed using simple URLs so you can access them from any PC or location.
- Integrate with your Portal: You can integrate it with your portal and add your own content
- Integrate with your application such as ERP, MRP, CRM etc; Provide your dashboard link as part of your ERP application so it provides a seamless experience.
- Single source of truth: Web dashboards can present information from multiple data sources on the single page so you can consolidate and have “single source of truth”
- Easy to maintain and update: Updating the dashboard or adding new dashboards is done using a single node entry on the web server so it is easy to maintain and update

Install Web Dashboards Server Components

1. Download the server ZIP file from <https://s3.amazonaws.com/infocaptor/webdashboards.zip>
2. Unzip the webdashboards.zip into any directory.



3. Copy the server2go directory into the InfoCaptor Desktop parent directory



- 4.
5. That is all needed to get started for publishing to web.
6. To view the sample dashboard follow the steps
 - a. Run the server2go.exe that is located in the Infocaptor/server2go/ directory



b.

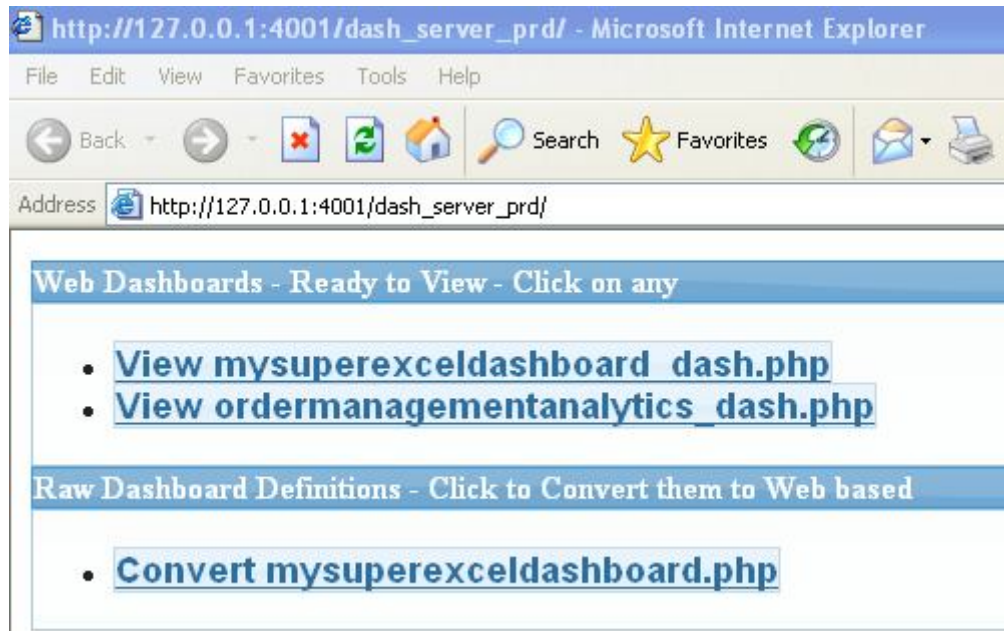
c. Start the local webserver by double clicking the **server2go.exe**

d. It will take few seconds to start the webserver



e. It will take few seconds to start the webserver

f. Once started, it launches the default web page for you to get started



g.

h. Click on the “View ordermanagementanalytics_dash.php” or “View mysuperexceldashboard_dash.php”

i. It should show a dashboard like below



j.

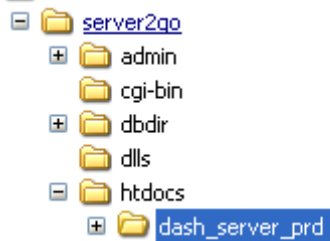
k. Click on the ‘Go’ button to refresh the dashboard.

l. The rest of the guide explains steps to arrive at this web dashboard.

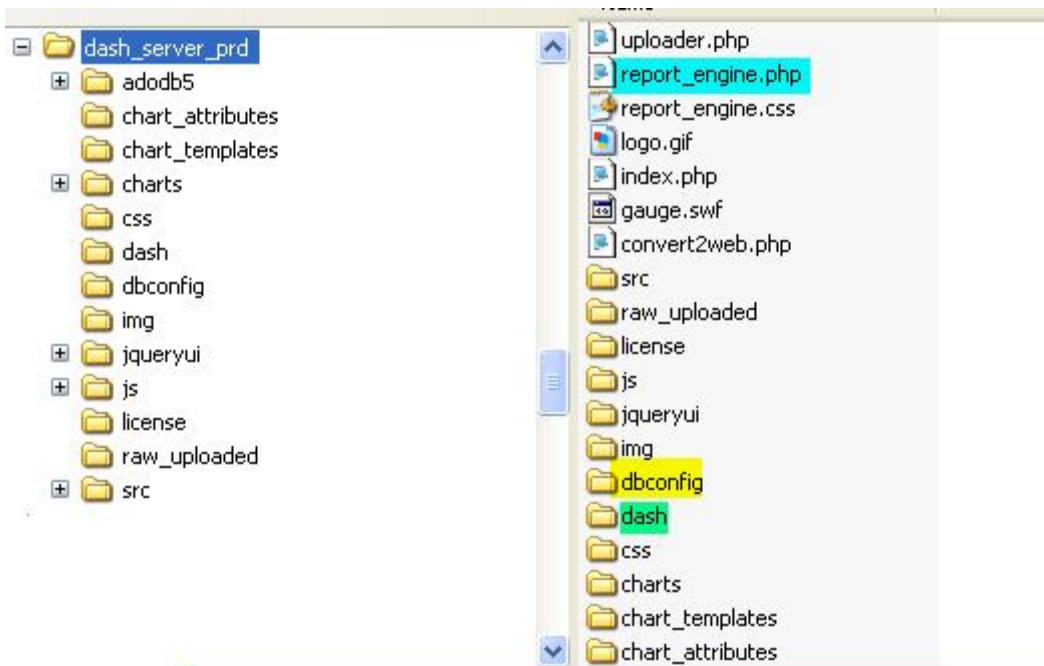
NOTE: server2go is a **third party web server** and is provided only for testing purpose. It is based on the Apache web server. When you are ready to finally publish it on your own webserver (Apache or IIS), simply copy the **dash_server_prd** directory to your webserver. We will cover the details later in this document.

Web Dashboards Architecture

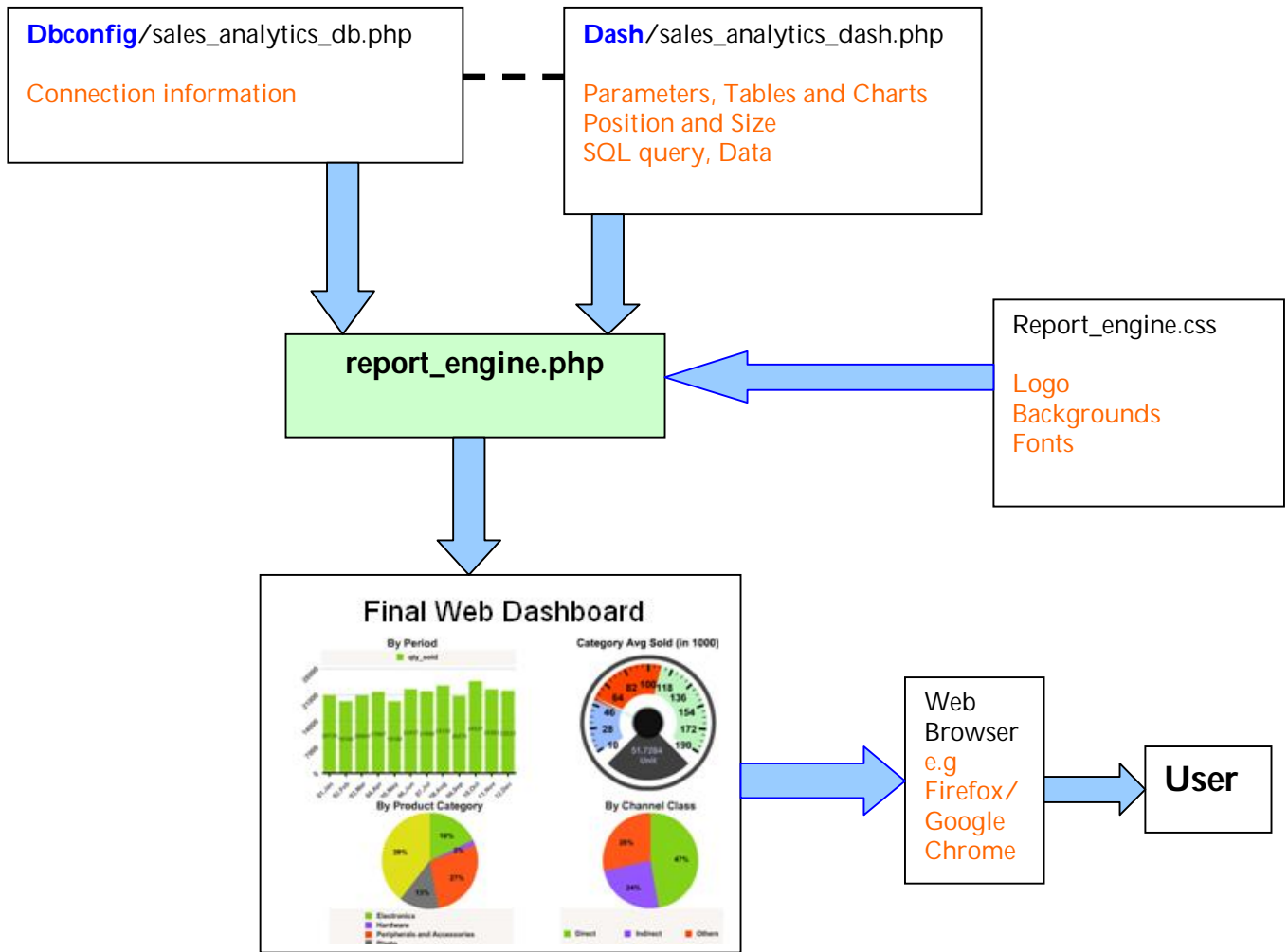
Inside the server2go directory, there is a **htdocs** directory and within that there is a **dash_server_prd** directory. The directory “htdocs” is the root of your webserver for displaying any web objects such as html files etc.



Within the dash_server_prd directory, we have the dashboard engine and other related files and directory for dashboard definitions.



The directories and files are explained in more details in the following section.



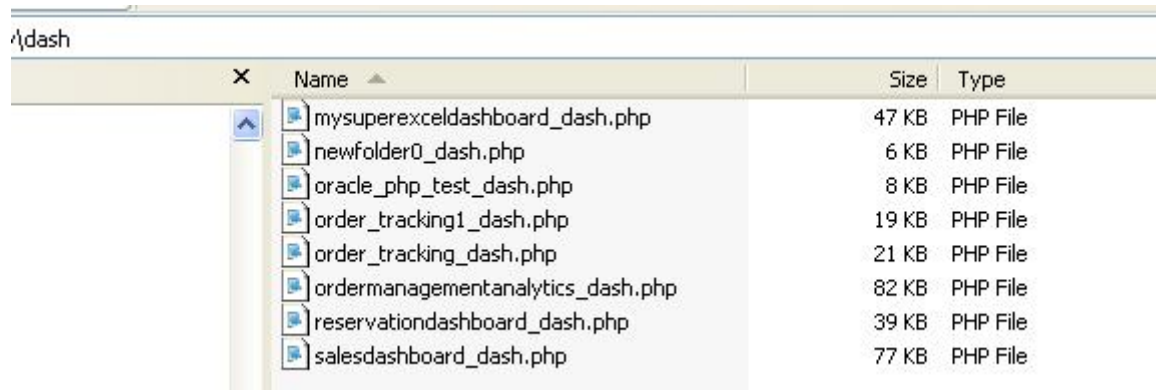
report_engine.php

At the heart of the web dashboard architecture lies the report_engine.php file. This is the main engine that renders the final dashboard on the web browser. To the report_engine.php you pass the name of the dashboard and it will parse the dashboard definition and present it on the browser.

NOTE: Do not try to edit or modify this file(report_engine.php) as it is encrypted.

dash/dashboard_definition.php

Each dashboard is defined and stored in the **dash** subdirectory as a PHP file. You can easily open this PHP file in any text editor and view and modify the properties and code. For e.g if you had a sales dashboard the file name would be something like sales_dashboard_dash.php. Every dashboard definition is given a prefix of “**_dash**” before **the .php** extension.

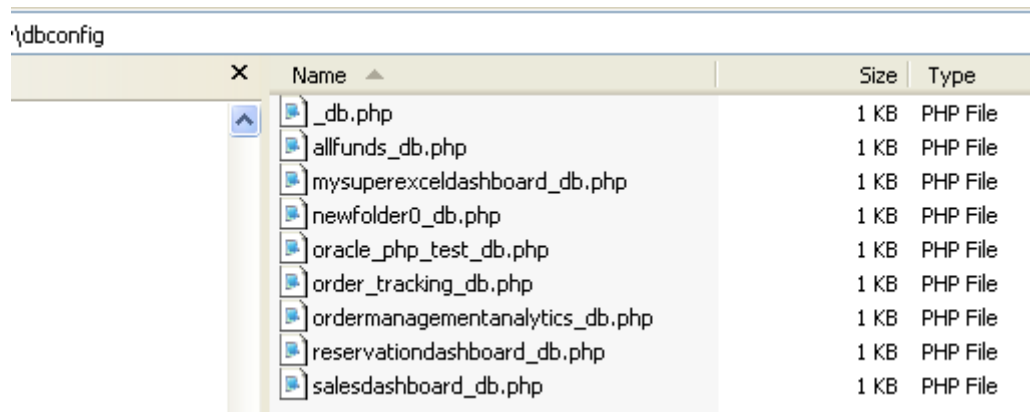


Name	Size	Type
mysuperexceldashboard_dash.php	47 KB	PHP File
newfolder0_dash.php	6 KB	PHP File
oracle_php_test_dash.php	8 KB	PHP File
order_tracking1_dash.php	19 KB	PHP File
order_tracking_dash.php	21 KB	PHP File
ordermanagementanalytics_dash.php	82 KB	PHP File
reservationdashboard_dash.php	39 KB	PHP File
salesdashboard_dash.php	77 KB	PHP File

As shown in the screenshot, each .php file is a unique and independent dashboard. Later we will explain the structure and content of this _dash.php files.

dbconfig/dashboard_database_connection.php

Each dashboard definition in the “dash” directory has a corresponding database configuration file.



Name	Size	Type
_db.php	1 KB	PHP File
allfunds_db.php	1 KB	PHP File
mysuperexceldashboard_db.php	1 KB	PHP File
newfolder0_db.php	1 KB	PHP File
oracle_php_test_db.php	1 KB	PHP File
order_tracking_db.php	1 KB	PHP File
ordermanagementanalytics_db.php	1 KB	PHP File
reservationdashboard_db.php	1 KB	PHP File
salesdashboard_db.php	1 KB	PHP File

Each database config file is given a prefix of “_db” before the .php extension. You can easily modify the contents of this file using any text editor.

Example: The salesdashboard_db.php file is directly linked to the salesdashboard_dash.php. This link information is stored in the salesdashboard_dash.php file.



report_engine.css

Any cascading style changes goes in this file. You can change the default background colors, add logos or pictures to all of your dashboards by editing this file.

NOTE: Need knowledge of HTML and CSS.

Other Directories and files

Under the **dash_server_prd** directory there are few important directories and files and following section describes them

license



The license directory contains the license.php file and if this file or the directory are not present then the dashboards will not work.

Even for evaluation or trial period, you need the license.php file. For your copy of the evaluation license.php you can register at the following URL

http://www.infocaptor.com/service/web_dashboard_demo.php

chart_templates

This directory “chart_templates” contains a file named **current_template.php**. This file is used during the conversion process from “raw” dashboard definition to the final dash php file in the **dash** directory

You can edit this file in any PHP editor and update the template settings. See Chart settings section to update the appropriate chart XML settings.

Convert Desktop Dashboard to Web Dashboard

Assumption: You have already designed and developed your desktop dashboard and is now ready to be published on the web. If you need instructions on how to develop the desktop dashboard then [follow this tutorial](#)

(https://s3.amazonaws.com/infocaptor/Dashboard_Tutorial.pdf)

If you wish to skip the dashboard development and just test the web portion and remainder of the section then you can download the pre-built dashboard here https://s3.amazonaws.com/infocaptor/order_dashboard.zip

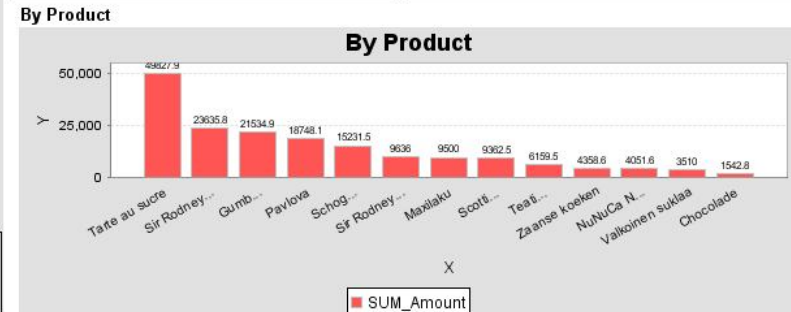
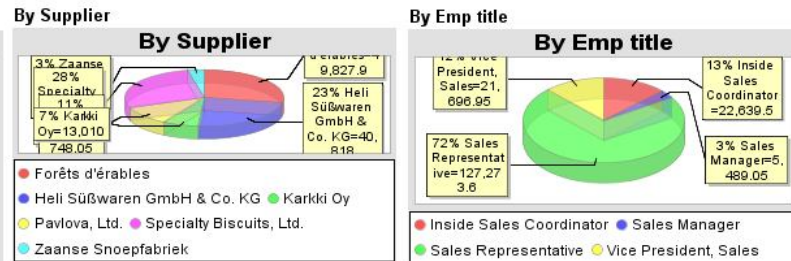
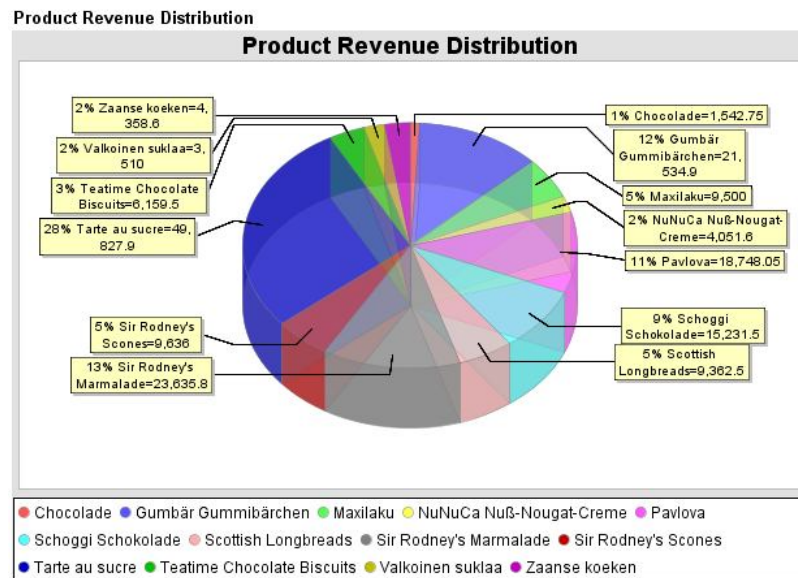
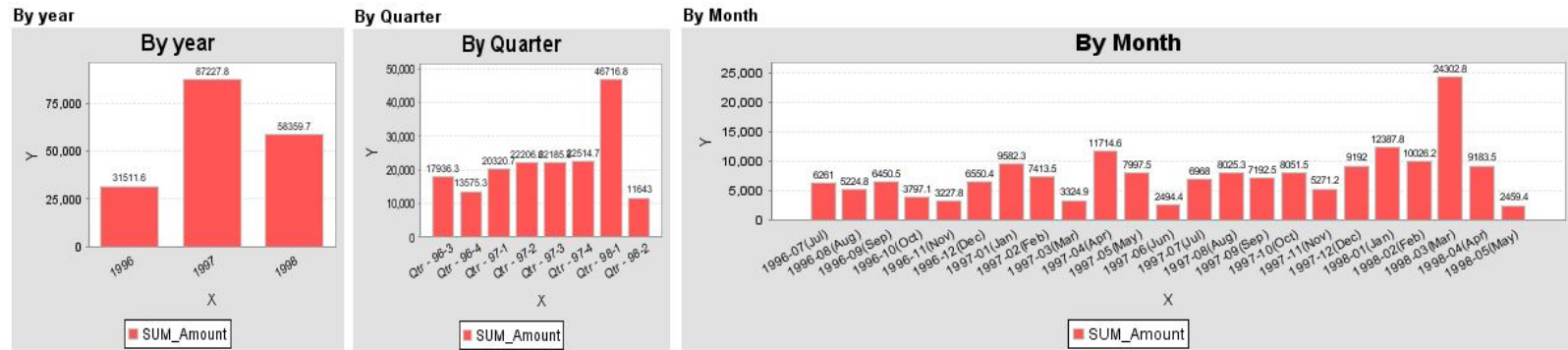
Desktop Dashboard to Raw PHP Definition

The first step is to launch your Desktop Dashboard using infocaptor and then publish the definition to a raw PHP file. This raw PHP file is created in the “**raw_uploaded**” directory. The reason it is called as raw_uploaded because the dashboard definition contains raw information about the Desktop dashboard definition. It still needs to be converted to a proper format so that the dashboard engine (report_engine.php) can parse and understand and finally render the dashboard in the browser.

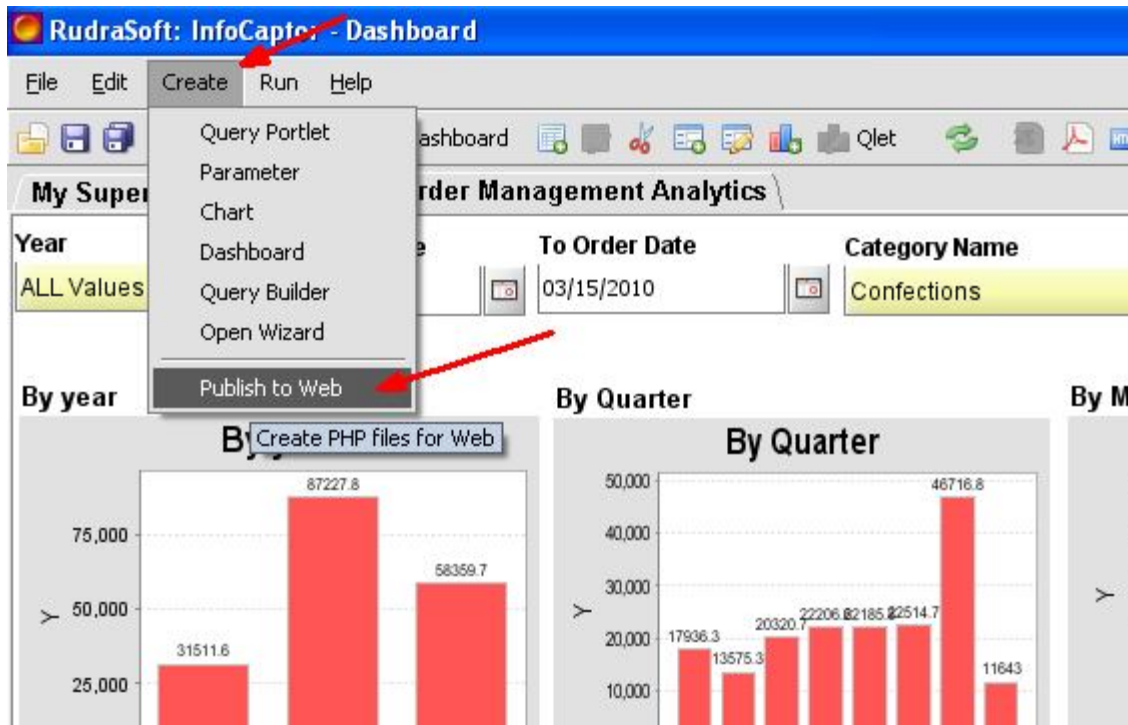
Convert Desktop Dashboard to Raw PHP definition Steps

1. Consider the following “Order Management Analytics” dashboard

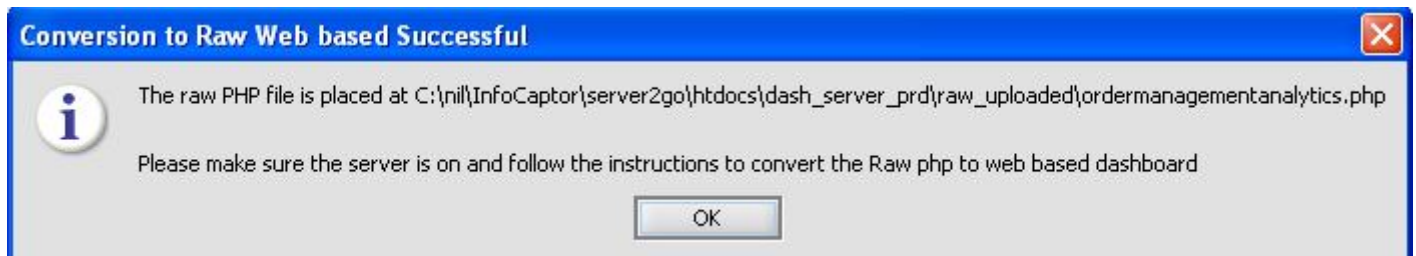
Year: ALL Values | From Order Date: 03/15/1996 | To Order Date: 03/15/2010 | Category Name: Confections



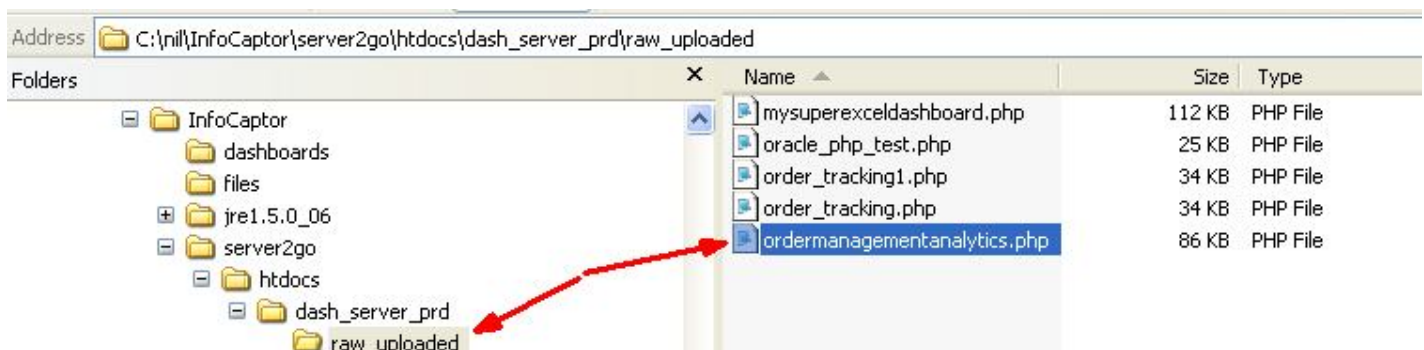
- Download the completed dashboard and data file so you can follow the instructions https://s3.amazonaws.com/infocaptor/order_dashboard.zip
- Open the dashboard in InfoCaptor
- Navigate to the menu Create → Publish to Web



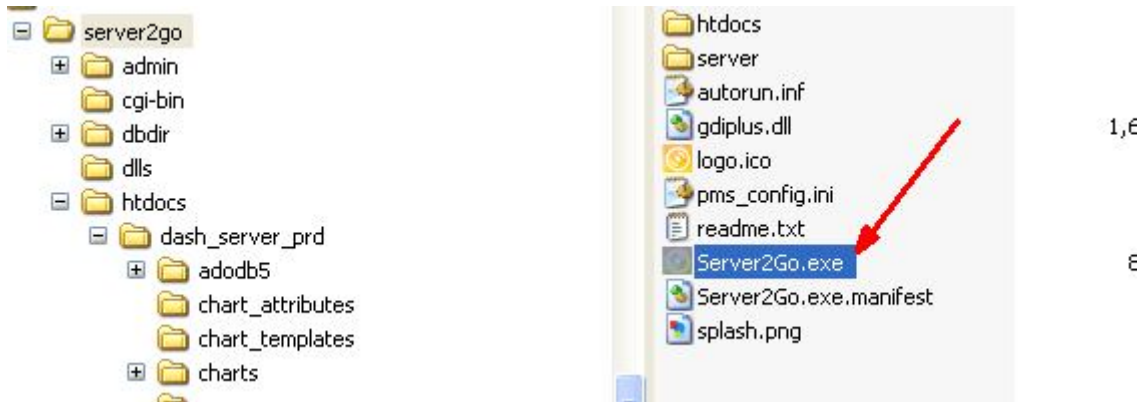
5. It provides a confirmation that the file is created in the raw_uploaded directory



6. The file ordermanagementanalytics.php is now created in the raw_uploaded directory as shown in the screenshot below



7. Start the local webserver by double clicking the **server2go.exe**



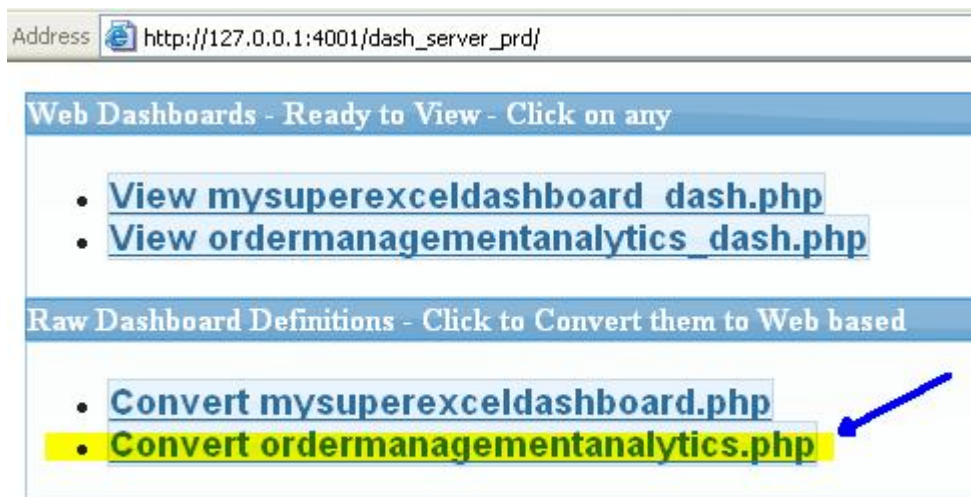
8. Start the local webserver by double clicking the **server2go.exe**

9. It will take few seconds to start the webserver



10. It will take few seconds to start the webserver

11. Once started, it launches the default web page for you to get started



12. Scroll towards the bottom of this page and you will see the conversion section
13. The page lists all the files that are available in the raw_uploaded directory, so all you need to do is click on the link to convert the definition.
14. Once you click on “Convert ordermanagementanalytics.php”, it shows the following status



Conversion complete..

Total Parameters = 4

Total Portlets = 7

Click below link to view the dashboard

dash/ordermanagementanalytics_dash.php

15. Conversion is a one time process and is needed whenever you change the dashboard definition in InfoCaptor desktop designer. As you see above, it says “conversion complete” and it also tells you how many parameters and portlets are available on the dashboard.
16. Click on the link pointed by the red arrow to view the dashboard
17. If you get the below message, which means you are using the trial period or evaluating the software. Please click on the “Get Trial Key” link.



error#2. Registration or Demo Expired

[Get Trial Key](#)

Please register.

Where should we email your Trial Key?

1. **First Name** :

2. **Last Name** :

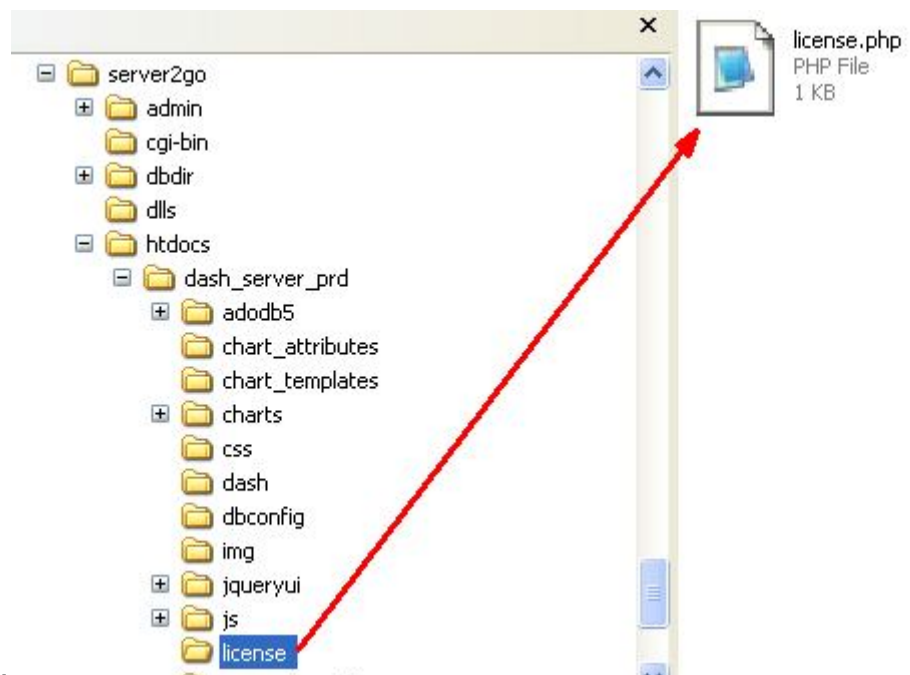
3. **Email** :

18.

19. Enter your name and email address and a license file will be emailed to your address.

20. Alternatively you can download the license file and copy the contents and paste it in the license.php file which is located in the license subdirectory.

NOTE: A valid license.php file is needed to view the dashboards. During the evaluation period you get a temporary license file that you can place it in this directory (dash_server_prd / license)



21.

22. Once you have a valid license file, you can click the dashboard link by going to the base URL (http://127.0.0.1:4001/dash_server_prd/)

23. Click on the below link

4. **View Dashboards** : Your web dashboard is ready to view refresh this page to see the latest converted dashboards

Web Dashboards - Ready to View - Click on any

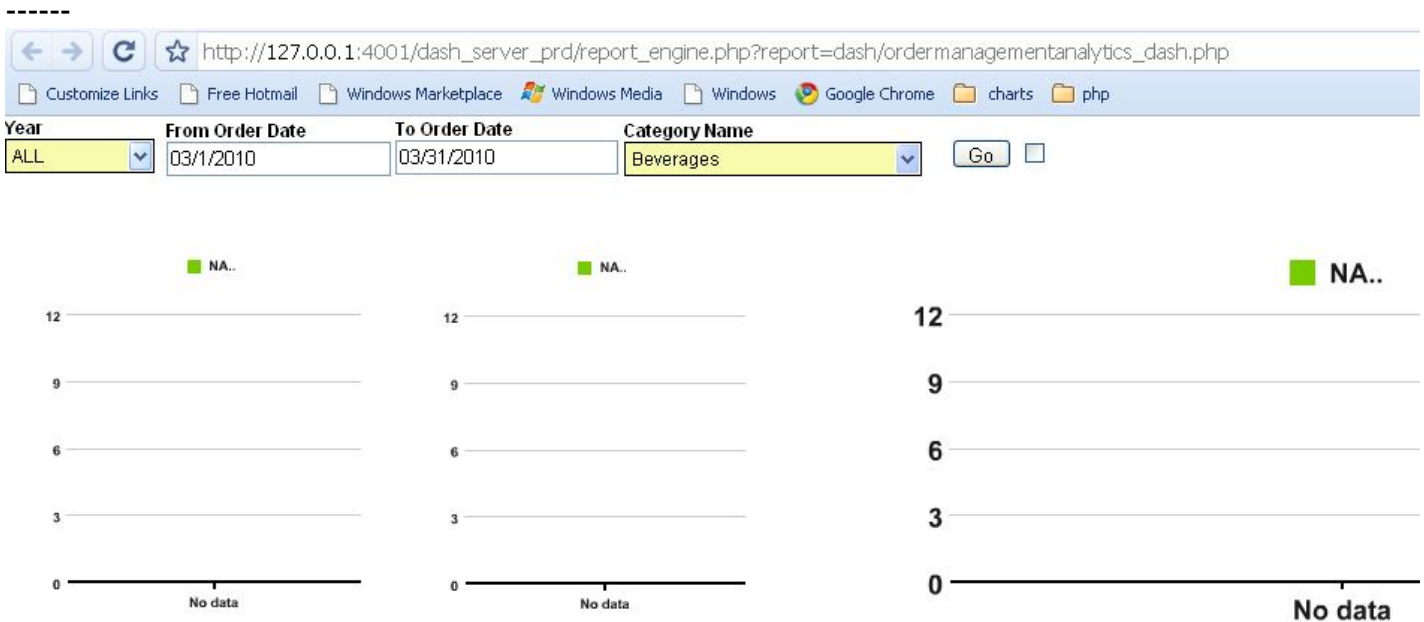
- [View ordermanagementanalytics_dash.php](#)

Raw Dashboard Definitions - Click to Convert them to Web based

- [Convert ordermanagementanalytics.php](#)

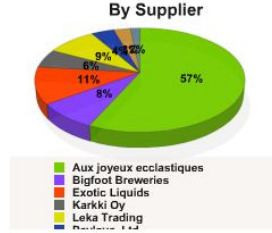
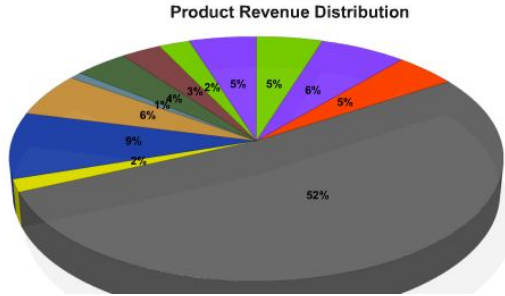
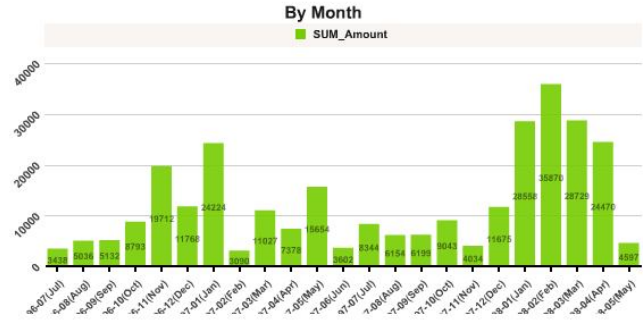
24.

25. Your dashboard now shows up but may not show any data or charts yet. Just adjust the parameter values and you will see the charts



In the above dashboard, the data is available only from 1996 through 1998, so we change the From Order Date parameter value to 03/1/1996 and **then click on the “Go” button**

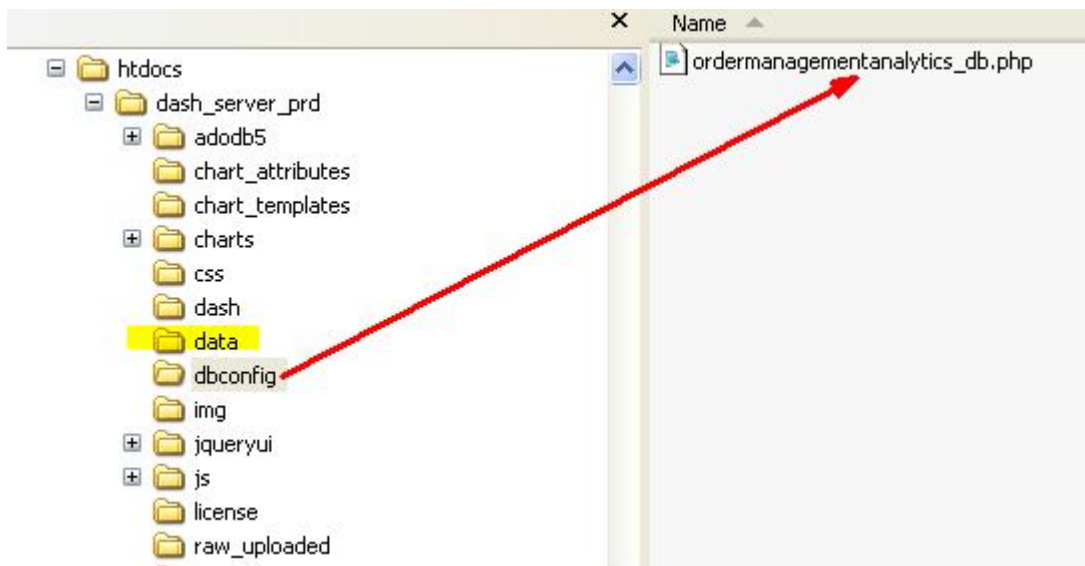
Year: ALL |
 From Order Date: 1/1/1996 |
 To Order Date: 03/31/2010 |
 Category Name: Beverages |



We have now completed the cycle from Desktop to Web dashboard

Structure of Database Configuration File

All the database configuration file including connections to Excel and Access files are stored under the dbconfig directory.



You can edit the _db.php file in any text editor

```
<?php
$db_array['order_raw.xls']['type']='excel';
$db_array['order_raw.xls']['host']='';
$db_array['order_raw.xls']['port']='';
$db_array['order_raw.xls']['name']='C:/Documents and Settings/My Documents/order_raw.xls';
$db_array['order_raw.xls']['user']='';
$db_array['order_raw.xls']['pass']='';

?>
```

The name variable stores the path to the Excel file. So if you needed to make the dashboard directory portable, you can copy your order_raw.xls file into a directory under dash_server_prd. In this case, we already have a directory named “data” so you can copy the file in the dash_server_prd/data directory.

After copying the file, change the path to this directory.

```
<?php
$db_array['order_raw.xls']['type']='excel';
$db_array['order_raw.xls']['host']='';
$db_array['order_raw.xls']['port']='';
$db_array['order_raw.xls']['name']='data/order_raw.xls';
$db_array['order_raw.xls']['user']='';
$db_array['order_raw.xls']['pass']='';

?>
```


Sample connection settings for MySQL database

```
<?php
$db_array['localmysql']['type']='mysql';
$db_array['localmysql']['host']='localhost';
$db_array['localmysql']['port']='3306';
$db_array['localmysql']['name']='test';
$db_array['localmysql']['user']='root';
$db_array['localmysql']['pass']='';
?>
```

If your dashboard uses multiple connections to different databases then you will see multiple entries within the same dashboard config file.

Excel Connections

While using Excel as primary database, there could be locking issue of the Excel Files. For e.g if you are designing dashboard in InfoCaptor desktop and you try to publish the dashboard to web, the dashboard will get converted successfully but may not render and give a connection error.

To avoid this problem, during conversion, two more properties are added for Excel connection types.

NOTE: These properties are not implemented in the current release

['use_local_excel_cache'] = 'Y'

The property indicates to create a local copy of the excel file before connecting.

If you decide to share it with multiple users then you may hit the connection issue as two simultaneous queries hitting the same Excel file may time out or throw an error.

To avoid this problem, there is one more property provided

['multi_user_excel_cache']='N'

By default it is set to No but when you decide to open the dashboard access to more than one user then you may need to set this to 'Y'.

The drawback is that you will have to purge the temp directory as for every user session it creates a copy of the Excel file.

Structure of Dashboard Definition File

Each Dashboard definition is a [PHP](#) file. You can easily open any PHP file in any text editor. The structure of the php file is easy to understand and hence with little introduction and knowledge you can change the values or content of the dashboard.

Each dashboard definition is a collection of dashboard parameters, table portlet and Chart Portlets. Within the dashboard definition file, there are numerous properties of each objects such as the name of the object, display properties and the SQL query behind each object.

All the properties and SQL code is stored as PHP variables so that you can manipulate them by writing your own functions or PHP code. Basic knowledge of PHP scripting is required only if you want to change the converted dashboard from Desktop to web.

NOTE: It is possible to define a new dashboard by directly typing the PHP code in a text editor. You can use the existing dashboard php file, copy/clone the file, rename it and start editing the contents. The main benefit of the GUI Dashboard designer is to easily define the dashboard without worrying of the pixel location of the objects and manually sizing them within PHP.

General Structure of PHP dashboard definition file

```
<?PHP
```

```
....general and dashboard related properties....
```

```
....some utility functions....
```

```
.....dashboard parameters... $plet variables....
```

```
.....dashboard portlets ....$plet variables...
```

```
.....$plet variables with type='Table'
```

```
.....$plet variables with type='Chart..'
```

```
?>
```

General Dashboard properties

```
<?php
$debug='0';
$dashboard_report='My Super Excel Dashboard';
$dashboard_id='mysuperexceldashboard';
$button_text='Go';
$database_config='dbconfig/mysuperexceldashboard_db.php';
$d_options['use_plet_positions']= true; //this will place ex
$d_options['grid_height_offset']=-80;//this is the offset ac
$d_options['top_button_offset']=0;//Push all the objects do
$d_options['top_button_display']=true;//Display the Go butto
```

- \$debug : Not used currently
- \$dashboard_report : Name or title of the dashboard
- \$dashboard_id :- Internal ID of the dashboard. Used in drill options
- \$button_text : Besides the end of all parameters, there is a 'Go' button. And besides the 'Go' button there is a checkbox.



By default, when you change any parameter value by selecting a different value from the drop down, it automatically triggers the dashboard refresh. Sometimes this is not a desired feature when you have more than 3 or 4 dashboard parameters. In this case, the users can simply select the check box and then the auto refresh will be disable for that session. Now once the checkbox is checked, users will need to manually click on the 'Go' button to refresh the dashboard. You can change what appears on the 'Go' button such 'Search' or 'Refresh'

- \$database_config='dbconfig/mysuperexceldashboard_db.php' : This is the path to the database configuration file for this dashboard definition.
- \$d_options['use_plet_positions']= true; : This is not used in the current version. By default the behavior is true.
- \$d_options['grid_height_offset']= -80; : This is the offset added to the actual height for accomodating the caption,pager and column header height. You may change this value but there is no need to modify this.
- \$d_options['top_button_offset']=0; : Push all the objects down by this pixel height. This is useful to adjust the relative position of all objects. For e.g if you want to show a banner on the top of the dashboard page and then want all objects to start after certain pixel height then you specify the height in this variable. To change the banner or logo, update the logo.gif file in the dash_server_prd directory and the corresponding entry in the report_engine.css file.
- \$d_options['top_button_display']=true : If you don't want the go button set it to false.

Utility Functions

```
$percent = 'percentof';  
function percentof($value,$percent)  
{  
    return ($value*$percent)/100;  
}
```

```
$get_position = 'getPosition';  
/*  
 * Total height /total width of the canvas  
 * offset = deduct this from the actual height or width  
 * per = percentage of the remaining height or width and based on this return the relative position  
 */  
function getPosition($total_height,$offset,$per)  
{  
    return $total_height - (($total_height-$offset)*$per)/100;  
}
```

NOTE: Do not modify the above functions. If you want to add some custom logic into your dashboard then you can create your own functions and use them within the dashboard object variables.

Dashboard Parameters

Each dashboard parameter is defined within the \$plet variable.



For the “Region” Dashboard parameter, let us take a detail look into the dashboard definition and go through the properties

```

35
36
37 /***** region_p34 *****/
38 $plet['region_p34']['type']='dynamic_lov' ;
39 $plet['region_p34']['data_type']='CHAR' ;
40 $plet['region_p34']['id']='region_p34' ;
41 $plet['region_p34']['name']='Region' ;
42 $plet['region_p34']['raw_type']='PLET' ;
43 $plet['region_p34']['connection']='Excel Demo' ;
44 $plet['region_p34']['width']=213 ;
45 $plet['region_p34']['height']=50 - 27 ;
46 $plet['region_p34']['x_position']=134 ;
47 $plet['region_p34']['y_position']=14 ;
48 $plet['region_p34']['format']='Author,Description etc' ;
49 $plet['region_p34']['font']='font-family:Arial;font-name:Arial;font-style:plain;font-size:12px' ;
50 $plet['region_p34']['font_color']='#000099' ;
51 $plet['region_p34']['background_color']='#ccccff' ;
52 $plet['region_p34']['static_values']='' ;
53 $plet['region_p34']['default_value']='' ;
54 $plet['region_p34']['static_values']='' ;
55 $plet['region_p34']['enable_all']='Y' ;
56 $plet['region_p34']['all_label']='ALL' ;
57 $plet['region_p34']['all_value']='%' ;
58 $plet['region_p34']['display_members']='1' ; //value decides whether it is a multilist combo or single selection ;
59 $plet['region_p34']['control_type']='LIST' ; //In future, other values could be radio,slider,checkbox etc ;
60 $plet['region_p34']['default_null_value']='%' ; // ;
61 $plet['region_p34']['static_display_values']= $plet['region_p34']['static_values' ] ;
62 $plet['region_p34']['sql']= <<<EOD
63     select
64     distinct country_region
65     FROM [detail_data$]
66
67 EOD;

```

Each parameter is a multi-dimensional array of properties.

```

/***** region_p34 *****/
$plet['region_p34']['type']=

```

The first index in the array represents the parameter unique identifier. In this case “region_p34” is the internal identifier for the “Region” parameter. The second index represents the property name. In this case ‘type’ is one of the property name for the parameter with internal identifier “regions_p34”.

Plet properties

['type'] : Determines the type of the parameter. Valid values are 'dynamic_lov', 'static_lov', 'date', 'text'.

- 'dynamic_lov' : This is for dynamically generated list of values through SQL code. There should be a valid 'sql' code for this type
- 'static_lov' : This is a static list of values. The list is a comma separated array defined in the \$plet['region_p34']['static_values'] variable. If you know that the list of values is always going to be the same and never or rarely change then you can increase the performance of the dashboard by replacing the dynamic lov to static one and providing a list of values. See 'static_values' property for more details on how to convert a dynamic lov to static lov.
- 'date' : This is a date type parameter.
- 'text' : This is a free text input field.

['data_type'] : Valid values are 'CHAR' and 'DATE'

['id'] : Internal identifier of the parameter. This is the same as the first index of the \$plet array

['name'] : Display name of the parameter

['raw_type'] : This is the original parameter type inherited from the Desktop dashboard designer. Not all parameter types are supported yet in the web dashboard. In future all the raw_type parameter value will have a corresponding web value type assigned during the conversion process.

['connection'] : This is the connection index for the \$db_array defined in the database config file.

Dashboard Definition points to the Db config file

```
$database_config='dbconfig/mysuperexceldashboard_db.php';
```

```
$plet['region_p34']['connection']='Excel_Demo';
```

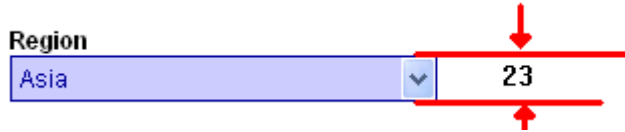
```

1 <?php
2 $db_array['Excel_Demo']['type']='excel';
3 $db_array['Excel_Demo']['host']='';
4 $db_array['Excel_Demo']['port']='';
5 $db_array['Excel_Demo']['name']='C:\nil\InfoCaptor\dashboards\test_data.xls';
6 $db_array['Excel_Demo']['user']='';
7 $db_array['Excel_Demo']['pass']='';
8
9 ?>

```

['width'] : The width in pixel for the parameter. E.g \$plet['region_p34']['width']=213 ;

['height'] : The height in pixel for the parameter : e.g \$plet['region_p34']['height']=50 – 27; ie. 23
(NOTE: It is subtracting 50 – 27 during the conversion process. The total height 50 represents the complete frame height in the Desktop designer and it includes the height for the text label for the parameter. In the web version the rendering engine draws the parameter and label separately and hence we need to discount for the label height from the total height.



['x_position'] : The relative x position of the parameter with respect to the browser canvas.

['y_position'] : The relative y position of the parameter with respect to the browser canvas.

['format'] : This is the date format for the Date Parameter e.g :
\$plet['fromorderdate_p36']['format']='dd-M-yy' ;

NOTE: The format defined will dictate how the values will be displayed when you select the date parameter value.

From Order Date	To Order Date
03/1/1996	03/31/2010

March 2010						
Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

And this format affects the SQL query that uses this parameter for filtering. So take extra **precaution** when you decide to change the date format. If you do change, then make sure the format is correctly changed in all the SQL queries for the portlets (tables and charts) that use this parameter.

['font'] : This value affects the font for values in the drop down list

Region

['font_color'] : This affects the color of the font in the drop down list

['background_color'] : This affects the background color of the drop down list

['default_value'] : This value is defaulted when you first launch the dashboard e.g.

```
$plet['region_p34']['default_value']='Asia' ;
```

For date type parameter you can provide a default value to the 'From and to pair so that the From can default to the first day of the month and the To date can default to Today's date or last day of the month.

This can be achieved by using the direct PHP Date function.

For eg. To provide the first day of the month as default value for the 'From Date'

From Date

```
$plet['fromorderdate_p36']['default_value']=date('m/1/Y') ;
```

To provide the last day of the month for

To Date

```
$plet['toorderdate_p38']['default_value']=date('m/t/Y') ;
```

Check <http://php.net/manual/en/function.date.php> for more date options.

NOTE: You can also hard code a fixed value in the default property like

```
$plet['fromorderdate_p36']['default_value']= '1/1/1996' ;
```

['enable_all'] : Valid values are 'Y', 'N', 'INCLUDE_ALL_AS_LIKE', 'INCLUDE_ALL_AS_CHAR', 'INCLUDE_ALL_AS_NUM'.

- ➔ INCLUDE_ALL_AS_CHAR will pass the 'All' value as a comma separated values and all the values are treated as string enclosed in quotes.. Avoid using this approach. The preferred approach is to use the INCLUDE_ALL_AS_LIKE option.

- ➔ INCLUDE_ALL_AS_NUM will pass the 'All' value as a comma separated values and all the values are treated as number without quotes.
- ➔ INCLUDE_ALL_AS_LIKE will pass the '%' value when the 'ALL' value is selected from the drop down.
- ➔ 'N' : Disable the 'ALL' Values
- ➔ 'Y' If you provide just 'Y' then you will need to provide your own mechanism to handle the 'ALL' value.

['all_label'] : The default label is 'ALL' but you can change this to 'All Values' etc

['all_value'] : The general 'ALL' value to be passed is '%' but you can pass any string value. This value will be passed to the SQL query in the portlet when the user selects the 'ALL' option in the drop down.

['display_members']: Not used in this version

['control_type'] : Not used in this version

['default_null_value'] : There is no possibility of passing empty or null values in the case of drop down select parameters. But in the case of date parameters, users can deliberately empty the text field. In the case of 'text' parameter type (see 'type' property), users can choose to not type any value. In such case you can provide a default value. Preferred value is the '%' and make sure that you have a like condition in the SQL.

['static_values'] : list of string values that you want to be part of the drop down list

['static_display_values'] : list of corresponding string values that you want to display to the user. There is a one to one mapping for each value in the **'static_values'** list

For e.g to convert a dynamic value to static value follow the steps below

→ Go to the Dashboard

→ Right click on the page and view the “HTML source code” / “View Page Source”

→

→ Locate the html code for the “Region” parameter

```

215
216
217 <!--no group-->
218
219 <div style="position:absolute; top:14px; left:134px; width: 213px; height: 23px;">
220 <label for="region_p34">Region</label>
221
222 <SELECT NAME="region_p34" id="region_p34" onchange="selectSubmitParameters('
family:Arial;font-name:Arial;font-style:plain;font-size:12px ; color:#000099 ; backg
223 <option value="">ALL
224 <option value="Americas">Americas
225 <option value="Asia">Asia
226 <option value="Europe">Europe
227 <option value="Middle East">Middle East
228 <option value="Oceania">Oceania
229 </select></div>
230
231

```

- The static list of values are "%","Americas","Asia","Europe","Middle East","Oceania"
- The static list of **DISPLAY** values are "ALL","Americas","Asia","Europe","Middle East","Oceania"
- So in the PHP code for the parameter “Region” we change the static properties

→ From

```
59 $plet['region_p34']['static_values']= ;
60 $plet['region_p34']['static_display_values']= $plet['region_p34']['static_values'] ;
```

→ To

```
59 $plet['region_p34']['static_values']= array("%","Americas","Asia","Europe","Middle East","Oceania");
60 $plet['region_p34']['static_display_values']= array("ALL","USA","Asia","Europe","Middle East","Oceania") ;
```

```
$plet['region_p34']['static_values']= array("%","Americas","Asia","Europe","Middle East","Oceania");
```

```
$plet['region_p34']['static_display_values']= array("ALL","USA","Asia","Europe","Middle East","Oceania") ;
```

NOTE: The display value is ‘ALL’ and the corresponding value is ‘%’. Similarly the second display value is ‘USA’ and the passing value is ‘Americas’ that matches the column value in the database.

['sql'] : This is the SQL query that brings back the result set from the database.

```
61 $plet['region_p34']['sql']= <<<EOD
62     select
63     distinct country_region
64     FROM [detail_data$]
65
66 EOD;
```

The sql query and lot of other string values are defined using the PHP heredoc syntax. The syntax is “<<<EOD

Type your string or SQL code

EOD;

Note: There should be an immediate carriage return after the “<<<EOD” and there should be no white space or any other character before the “EOD;” You may use any delimiter and not limited to ‘EOD’. Anything that you define between the “<<<EOD and EOD;” is considered a complete string. With this format for defining string, you can provide single quotes, double quotes or even refer other PHP variables in the String. **VERY Important : There should be a new line character right after the first EOD and there should not be any space before the last EOD. If not followed PHP will error out.**

More Dashboard and Parameter Properties

```

664
665
666 $d_options['top_button_x_location']=633;//X position where the Go button will appear
667 $d_options['top_button_y_location']=$d_options['top_button_offset'] +15;//Y Position of the Go button
668 $d_options['top_button_width']=40;//Width of the Go button
669 $d_options['top_button_height']=20;//Height of the Go Button
670 $d_options['use_go_checkobx_x_location']=633+$d_options['top_button_width']+5 ;//X- position of the Checkbox. The che
671 $plet['list'] = array ('fromorderdate_p36' , 'toorderdate_p38' , 'categoryname_p34' , 'year_p32');
672
673
674 $plet_group['list'] =array('g1');
675 $plet_group['g1']['label']='Select Values';
676 $plet_group['g1']['members'] = array ('fromorderdate_p36' , 'toorderdate_p38' , 'categoryname_p34' , 'year_p32');
677
678
679 $plet_group['g1']['width']=700;
680
681
682 /*****Parameter End*****/
683 $plet['relative_position']=true;//
684
685
686

```

Right after all the parameter definitions end, there is a list of few more properties.

The conversion process updates these values after the parameter definitions are written to the file so they appear right after all the plet variables are defined.

\$d_options['top_button_x_location'] :

X position where the Go button will appear

\$d_options['top_button_y_location'] : \$d_options['top_button_offset'] +15. Y Position of the Go button

\$d_options['top_button_width'] : Width of the Go button

\$d_options['top_button_height'] :Height of the Go Button

\$d_options['use_go_checkobx_x_location'] : X- position of the Checkbox. The checkbox is used to disable AutoRefresh when you change the drop down value e.g.
633+\$d_options['top_button_width']+5 ;

\$plet['list'] : All the parameters that are defined are put in this array. Any plet that is not in this list will not be displayed on the dashboard even though the parameter is defined independently.

e.g : **\$plet['list'] = array ('region_p34' , 'year_p33');**

\$plet_group : This is not used in the current version but in order to show the parameters a default one group is automatically created during the conversion from desktop to web definition. Just make sure that the values in \$plet['list'] and \$plet_group['g1']['members'] are the same.

```
$plet_group['list'] =array('g1');
```

```
$plet_group['g1']['label']='Select Values';
```

```
$plet_group['g1']['members'] = array ('region_p34' , 'year_p33');
```

```
$plet_group['g1']['width']=700;
```

```
-----End of Parameter properties-----
```

Table and Chart Definitions

Each table or chart is defined as a \$qlet variable. The 'type' variable indicates whether it is a table or chart or gauge.

List of properties

['id'] : This is the internal identifier of the portlet (table/chart/gauge)

['name'] : Display name of the portlet.

['type'] : Valid values are

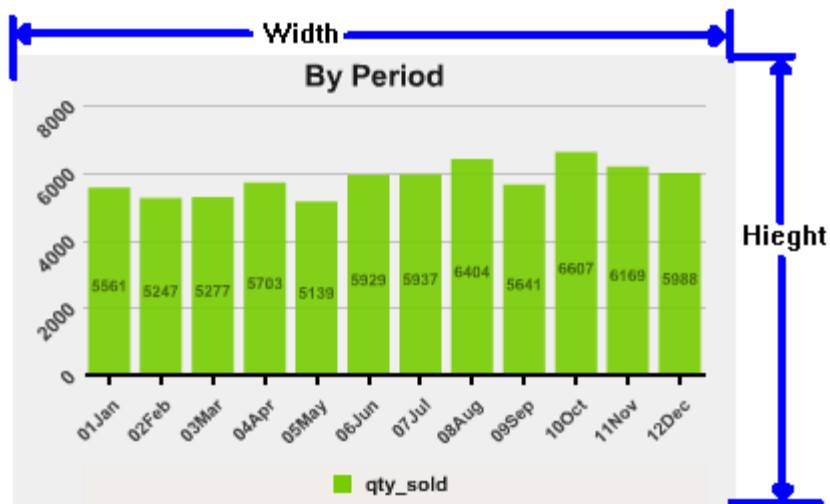
- table
- chart_column
- chart_pie
- chart_3d pie
- chart_3d column
- chart_stacked column
- chart_stacked 3d column
- chart_line
- chart_donut
- chart_area
- chart_stacked area
- gauge_dial
- gauge_meter*
- gauge_thermometer*
- chart_bubble
- chart_scatter
- chart_polar
- chart_bullet
- gauge_horizontal*
- gauge_vertical*

(* marked types will be implemented in future)

['connection'] : See Plet properties → Connection

['width'] : See Plet properties → Width

['height'] : See Plet properties → height



['x_position'] : See Plet properties → X_position

['y_position'] : See Plet properties → y_position

['max_rows'] : Limit the number of rows. Applicable for only charts and gauges. For table portlets you can specify the row limitation using the pagination template property. See Template property below.

['column_names'] : List of all column names that the SQL query will return

['number_of_columns'] : Total number of columns

['show_pager'] : This is applicable only for type=table. This displays the pagination bar at the bottom of every table portlet

['column_display_names'] : Display names of all the columns

['column_alignment'] : Column alignment or justification (right, left, center)

['column_class'] : This is inherited from the desktop dashboard. It indicates whether the column is string, number or date type.

['column_width'] : Column width in pixels

['template'] : The table portlet is displayed using the jQuery Grid plugin and this template stores the settings for the jqGrid plugin. In case of chart or gauge type, it contains the chart XML

Example table template

```
$qlet['byproductcategory_q33']['template'] = <<<EOT
```

```
jQuery("#byproductcategory_q33").jqGrid({
  #url#,
  datatype: 'xml',
```

```

mtype: 'POST',
#column_names#,
#column_model#,
#pager_str#,
rowNum:100,
rowList:[100,200,300],
sortname: "",
sortorder: "desc",
rowHeight:10,
scrollOffset:0,
viewrecords: true,
    height:#height#,
    width:{$qlet['byproductcategory_q33']['width']},
imgpath: 'themes/basic/images',
#edit_url#,
caption: '{$qlet['byproductcategory_q33']['name']}',
    footerrow : false,
    userDataOnFooter : false,
    viewsortcols: false,
    altRows:true,
    ExpandColClick: true,
    altclass: 'other_row',
    mtype:'POST',
    loadError : function(xhr,st,err)
    {
        //jQuery("#status").html("Type: "+st+"; Response: "+ xhr.status + " "+xhr.statusText);
    },
    loadComplete: function()
    {
    }
}
}).navGrid('#pager_byproductcategory_q33',{view:true,edit:true,add:true,del:true});

```

EOT;

To restrict the number of rows you can enable the pagination bar by setting show_pager= true; and then update the rowNum and rowList properties accordingly

Example Chart Template

```

$qlet['byperiod_g41']['template']= <<<EOD
    <chart_guide horizontal='true' vertical='true' thickness='1' color='000000' alpha='35' type='dashed' radius='3'
fill_alpha='75' line_alpha='0' background_color='FF4400' text_h_alpha='90' text_v_alpha='90' prefix_h=' ' suffix_h=' '
suffix_v=' ' />
    <chart_label color='000000' alpha='50' size='8' position='middle' as_percentage='true'/>

    <chart_grid_h thickness='1' type='solid' />

    <series bar_gap='0' set_gap='20' transfer='false' />

    <axis_category alpha='75' orientation='diagonal_up' size='9'/>

    <axis_value
    prefix="
                                suffix="

    decimals="
    decimal_char="

```



```

separator=""
show_min='true'
size='10'
alpha='75'
orientation='diagonal_up'
/>

```

```

<legend layout='horizontal'
fill_color='F0EBE6'
bullet='square'
font='arial'
bold='true'
size='10'

width='{$qlet['byperiod_g41']['legend']['width']}'
height='{$qlet['byperiod_g41']['legend']['height']}'
x='{$qlet['byperiod_g41']['legend']['top_x']}'
y='{$qlet['byperiod_g41']['legend']['top_y']}'

alpha='90'
/>

```

```

<chart_rect x='{$qlet['byperiod_g41']['chart_position']['top_x']}'
y='{$qlet['byperiod_g41']['chart_position']['top_y']}'
width='{$qlet['byperiod_g41']['chart_size']['width']}'
height='{$qlet['byperiod_g41']['chart_size']['height']}'
positive_color='000000' positive_alpha='0' negative_color='ff0000' negative_alpha='0'
/>
<draw>
<text size='15' x='0' y='0' width='{$qlet['byperiod_g41']['width']}' height='20' h_align='center'
v_align='top'>{$qlet['byperiod_g41']['name']}</text>
</draw>

```

```

<chart_pref line_thickness='3'
point_shape='square'
point_size='7'
fill_shape='false'
connect='false'
tip='none'

```

```

point_size='7'
point_shape='square'
trend_thickness='3'
trend_alpha='20'
line_thickness='0'

```

```

rotation_x='30'
rotation_y='30'
drag='true'

```

```

select='true'

```

```

zero_line='true'
/>

```

```

<chart_type>#chart_type#</chart_type>

```

```

#chart_data#

```

```

#link_data#

```

EOD;

Example Gauge Template

```
$qlet['categoryavgsoldin1000_g58']['template']= <<<EOD
```

```
<text size='15' x='0' y='0' width='{$qlet['categoryavgsoldin1000_g58']['width']}' height='{$title_height}' align='center'
>{$qlet['categoryavgsoldin1000_g58']['name']}</text>
```

```
    <circle x='#x_center#' y='#y_center#' radius='#radius#' fill_color='FFFFFF' fill_alpha='100'
line_thickness='6' line_color='333333' line_alpha='90' />
```

```
    #range_code#
```

```
    <circle x='#x_center#' y='#y_center#' radius='40' fill_color='FFFFFF' fill_alpha='255' line_thickness='0'
line_alpha='50' />
```

```
    #radial_ticks#
```

```
    #radial_numbers#
```

```
    #pointer_code#
```

```
    <circle x='#x_center#' y='#y_center#' radius='80' start='135' end='225' fill_color='111111' fill_alpha='80' />
```

```
    <circle x='#x_center#' y='#y_center#' radius='20' fill_color='111111' fill_alpha='100' line_thickness='5'
line_alpha='50' />
```

```
    <text x='{$qlet['categoryavgsoldin1000_g58']['text_x']}' y='{$qlet['categoryavgsoldin1000_g58']['text_y']}'
width='100' size='12' color='ccccff' alpha='70' align='center'>#pointer_value#
Unit</text>
```

EOD;

['background_color'] : Applicable for chart types. You can provide any valid hex number.
 ['background_color']= 'FFFFFF' ;

['enable_column_sort'] : To be implemented in future

['dynamic_script'] : Not used

['total_script'] : Not used

['sql'] : SQL query that pulls information from the database. The sql has references to the parameters defined in the dashboard.

Example

```
$qlet['byperiod_g41']['sql']= <<<EOD
```

```
select calendar_month_name, sum(quantity_sold) as qty_sold
```

```
from [detail_data$]
```

```
where fiscal_year = PVAL['year_p33']
```

```
and country_region like 'PVAL['region_p34']'
```

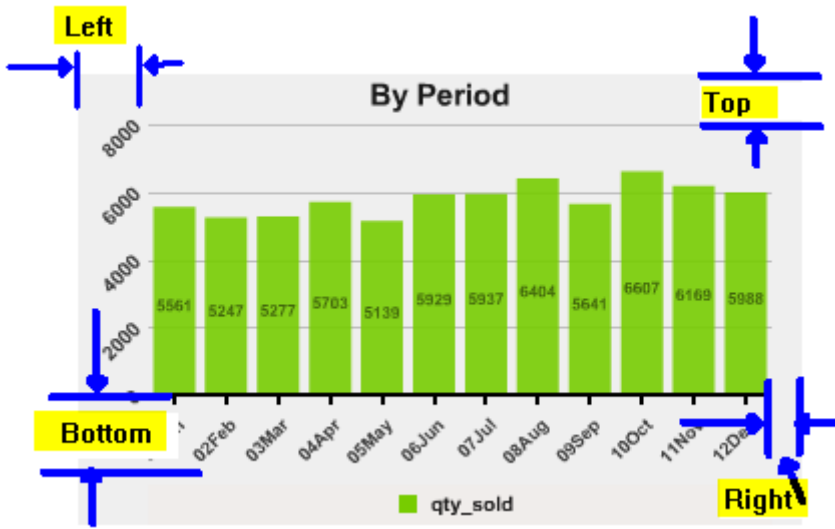
```
group by calendar_month_name
```

```
EOD;
```

If a portlet is dependent on any dashboard parameter then the SQL should have a where condition referencing the parameter by the portlet internal identifier. In the above yellow highlighted “PVAL” indicates that this is a reference to the parameter value with identifier “year_p33” and “region_p34”. At run time, when the user selects a value for any of the parameter drop down, the PVAL is substituted with the value and then the entire SQL is passed to the database for query.

['chart_margin'] : Example values

```
array(left =>'35',right=>'15',top=>'25',bottom=>'45');
```



['chart_size'] : This is the actual size where the chart is rendered.

Example

```
$qlet['byperiod_g41']['chart_size']=
  array(
    width => $qlet['byperiod_g41']['width']
      - $qlet['byperiod_g41']['chart_margin']['left']
      - $qlet['byperiod_g41']['chart_margin']['right']

    ,height => $qlet['byperiod_g41']['height']
      - $qlet['byperiod_g41']['chart_margin']['top']
      - $qlet['byperiod_g41']['chart_margin']['bottom']
      -20
  );
```

By default, the above chart_size formula is provided in the dashboard definition. The actual chart size is computed using the actual portlet size and then discounting the margin values.

For the width, the left and right margin values are subtracted from the portlet width.

For the height, the top, bottom and a fixed value of 20 is subtracted from the portlet height. The value 20 corresponds for the legend height (e.g qty_sold)

['legend'] : Stores the legend width, height, and position. Pie charts do not use the settings in the legend box. Pie chart have

```
$qlet['byperiod_g41']['legend']=  
  array(  
    width=> $qlet['byperiod_g41']['width']  
      - $qlet['byperiod_g41']['chart_margin']['left']  
      - $qlet['byperiod_g41']['chart_margin']['right']  
  
    ,height => 20  
  
    ,top_x=> $qlet['byperiod_g41']['chart_margin']['left']  
  
    ,top_y => $qlet['byperiod_g41']['height'] - 20  
  
    ,label_max_length =>10  
  );
```

Width : Determines the width of the legend box.

Height : 20 is the default height of the legend box

Gauge Specific Settings

For Dial Gauges

['x_center'] , **['y_center']** : The center of the circle is located half way across the width and same is the case with the height.

Example

```
$qlet['categoryavgsoldin1000_g58']['x_center']= $qlet['categoryavgsoldin1000_g58']['width']/2 ;
```

```
$qlet['categoryavgsoldin1000_g58']['y_center']= ($qlet['categoryavgsoldin1000_g58']['height']-
$title_height)/2 +$title_height ;
```

['radius'] : This is the minimum of the the width and height of the portlet size.

['range_radius'] : Radius of the colorful ranges

['tick_radius'] : radius of the number markers

['number_radius'] : radius for the actual numeric digits

['major_ticks'] : Determines how many major ticks and its properties

```
array(
    color    => '000000' ,
    length   => 10,
    thickness => 3,
    number_of_parts => 10);
```

//0 is for auto- engine will compute the increment automatically by dividing the range into 10 equal parts,

['minor_ticks'] : Determines the number of minor ticks in between two major ticks and its properties.

```
array(
    color    => '000000' ,
    length   => 5,
    thickness => 1,
    number_of_parts => 4);
```

//this will divide the major tick into equal parts. 0 will indicate to divide the major ticks into 10 equal parts

['ticks_range_angle'] : ticks_range_angle is the span of the dial that will use to display the ticks and numbers.

```
array(start_angle => -120, end_angle =>120);
```

The 12 o'clock position is the zero angle position. Anything clockwise from 12'0clock position is positive angle and anything anti-clockwise is -ve angle. For complete circle use -180 to +180.

['tick_numbers'] : Font and color of the tick numbers

```
array(font_size => 14, color =>'000000');
```

['range'] : The gauge is divided into multiple sections. Typically it has 3 sections but you may have as many of them as needed. Each section is defined by a numeric range and has its own color. The range is defined between start and end value. The following attributes define the properties of the section such as start,end value and color etc

```
$qlet['categoryavgsoldin1000_g58']['range']=
  array(
    array( start_value => 0,
          end_value => 40,
          attributes => "
            <circle x='#x_center#' y='#y_center#'
              radius='#radius#' fill_color='008400' fill_alpha='100'
              line_thickness='1' line_color='008400' line_alpha='90'
            />"
          ),
    array( start_value => 40,
          end_value => 60,
          attributes => "
            <circle x='#x_center#' y='#y_center#'
              radius='#radius#' fill_color='ffff00' fill_alpha='100'
              line_thickness='1' line_color='ffff00' line_alpha='90'
            />"
          ),
    array( start_value => 60,
          end_value => 100,
          attributes => "
            <circle x='#x_center#' y='#y_center#'
              radius='#radius#' fill_color='e70000' fill_alpha='100'
              line_thickness='1' line_color='e70000' line_alpha='90'
            />"
          )
  );
```

['pointer']['radius'] : The radius of the pointer.

```
= $qlet['categoryavgsoldin1000_g58']['radius'] - 5; //
```

['pointer']['points'] : These points define the shape of the pointer. The definition is a set of points that form a polygon. You can create any shape for the pointer. The pointer is displayed relative to the gauge center. So design the points with (0,0) origin.

for e.g if you need a 2 pixel thick line, then you just need a 4 (x,y) points relative to the center of the gauge

```
$qlet['categoryavgsoldin1000_g58']['pointer']['points']=
  array(
    array ( x => 3.5, y => 0),
    array ( x => 0, y => -$qlet['categoryavgsoldin1000_g58']['pointer']['radius']),
    array ( x => -0, y => -$qlet['categoryavgsoldin1000_g58']['pointer']['radius']),
    array ( x => -3.5 , y=> 0),
  );
```

```
$qlet['categoryavgsoldin1000_g58']['pointer']['attributes']=<<<EOD
<polygon fill_color='DEffff' fill_alpha='90' line_alpha='0'>
EOD;
```


['pointer']['value'] : Display property of the Gauge value

```
array(  
    decimals=>2 , decimal_str => '.' , thousand_separator => ','  
);
```

['text_x'] , ['text_y'] : Position of the text box that displays the pointer value.

['text_width'] : This is the width of the text box that contains the pointer value.

Chart Settings

This is the reference section to all the chart settings. You can customize your charts using the various chart attributes found in this section. For customizing the chart, you should update the template section of each chart in the dashboard definition php file

For e.g, to modify the properties of the first bar chart which is “By Year” we locate the template section for that chart and update it accordingly.

```
$qllet['byyear_g45']['template'] = <<<EOD
<chart_guide horizontal='true' vertical='true' thickness='1' color
<chart_label color='000000' alpha='50' size='8' position='middle'

<chart_grid_h thickness='1' type='solid' />

<series bar_gap='0' set_gap='20' transfer='false' />

<axis_category alpha='75' orientation='diagonal_up' size='9' />

<axis_value
    prefix=''
    suffix=''
    decimals=''
    decimal_char=''
    separator=''
    show_min='true'
    size='10'
    alpha='75'
    orientation='diagonal_up'
/>
```

CHART BORDER

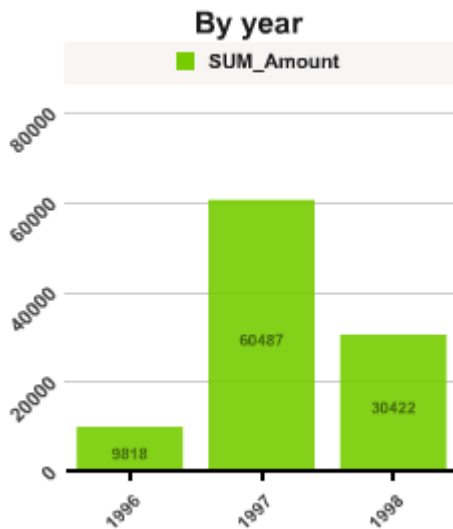
```
<chart_border top_thickness='int'
    bottom_thickness='int'
    left_thickness='int'
    right_thickness='int'
    color='string'
/>
```

Description

chart_border sets the chart's border attributes.

- **top_thickness:** The thickness of the border's top line. Valid values are zero and above. Zero makes this line invisible. The default is zero. In polar charts, this parameter is ignored.

- **bottom_thickness:** The thickness of the border's bottom line. Valid values are zero and above. Zero makes this line invisible. The default is zero for pie and bar charts, and 2 for all other charts. In polar charts, this parameter is used for the chart's outer border.
- **left_thickness:** The thickness of the border's left line. Valid values are zero and above. Zero makes this line invisible. The default is 2 for bar charts, and 0 for all other charts. In polar charts, this parameter is used for the value-axis.
- **right_thickness:** The thickness of the border's right line. Valid values are zero and above. Zero makes this line invisible. The default is zero. In polar charts, this parameter is ignored.
- **color:** The border color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).



Once we add the below border code to the above chart template

```
<!-- show red bottom and left border lines -->
<chart_border top_thickness='0'
  bottom_thickness='5'
  left_thickness='5'
  right_thickness='0'
  color='FF0000'
/>
```

The template now looks like ..

```
$qlet['byyear_g45']['template']= <<<EOD
```

```
<!-- show red bottom and left border lines -->
<chart_border top_thickness='0'
              bottom_thickness='5'
              left_thickness='5'
              right_thickness='0'
              color='FF0000'
              />
```

```
<chart_guide horizontal='true' vertical='true' thickness='1' color='000000' alp
<chart_label color='000000' alpha='50' size='8' position='middle' as_percentag
```

```
<chart_grid_h thickness='1' type='solid' />
```

```
<series bar_gap='0' set_gap='20' transfer='false' />
```

```
<axis_category alpha='75' orientation='diagonal_up' size='9' />
```

```
<axis_value
```

NOTE: You can attributes anywhere within the template, there is no sequence needed.

After applying the above border customizations, the chart looks like below

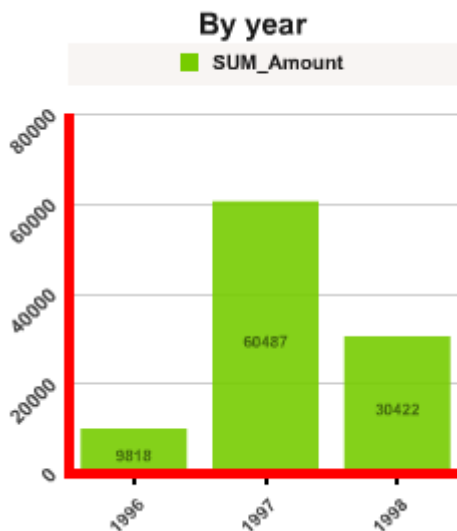


CHART GRID

`chart_grid_h` sets the chart's horizontal grid attributes.

- **thickness:** The thickness of the horizontal grid lines. Valid values are zero and above. Zero makes the horizontal lines invisible. The default is 1.

- **color:** The horizontal grid color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **alpha:** The horizontal grid transparency value. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 20.
- **type:** The horizontal grid line type. Valid values are **solid**, **dotted**, and **dashed**. The default is solid.

Example

```
<chart_grid_h thickness='2' color='FF0000' alpha='15' type='dashed' />
```

NOTE: The chart by default has a grid property set as below

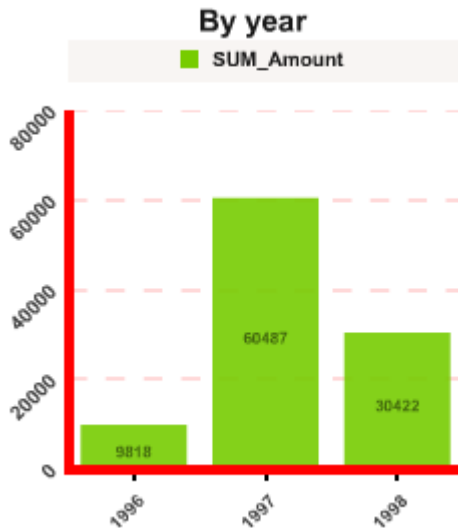
```
<chart_guide horizontal='true' vertical='true' thick
<chart_label color='000000' alpha='50' size='8' po
```

```
<chart_grid_h thickness='1' type='solid' />
```

We change it to below

```
<chart_grid_h thickness='2' color='FF0000' alpha='15' type='dashed' />
```

Result



Similarly

[chart_grid_v](#) sets the chart's vertical grid attributes.

- **thickness:** The thickness of the vertical grid lines. Valid values are zero and above. Zero makes the vertical lines invisible. The default is zero.

- **color:** The vertical grid color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **alpha:** The vertical grid transparency value. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 20.
- **type:** The vertical grid line type. Valid values are **solid**, **dotted**, and **dashed**. The default is solid.

CHART GUIDE

chart_guide sets one or two guide lines to connect the cursor position with the axes and simplify reading their values. The guides are the two red lines that move with the cursor when the mouse is over this chart:

By default a standard Chart Guide is provided through the below code in the template

```
<chart_guide horizontal='true' vertical='true' thickness='1' color='000000' alpha='35' type='dashed' radius='3' fill_alpha='75' line_s
```

```
<chart_guide horizontal='true' vertical='true' thickness='1' color='000000' alpha='35'
type='dashed' radius='3' fill_alpha='75' line_alpha='0' background_color='FF4400'
text_h_alpha='90' text_v_alpha='90' prefix_h=' ' suffix_h=' ' suffix_v=' ' />
```

Format

```
<chart_guide horizontal='boolean'
  vertical='boolean'
  thickness='number'
  color='string'
  alpha='number'
  type='string'
  snap_h='boolean'
  snap_v='boolean'
  connect='boolean'

  radius='number'
  fill_color='string'
  fill_alpha='number'
  line_color='string'
  line_alpha='number'
  line_thickness='number'

  text_h_alpha='number'
  text_v_alpha='number'
  prefix_h='string'
  suffix_h='string'
  prefix_v='string'
  suffix_v='string'
  decimals='int'
  decimal_char='string'
  separator='string'
  font='string'
  bold='boolean'
  size='number'
  text_color='string'
  background_color='string'
/>
```

Guide lines are supported by all **chart types** except for 3d, pie, and polar charts.

- **horizontal:** A boolean that determines whether the horizontal guide is visible or not. Valid values are **true** or **false**. The default is **false** (no horizontal guide).
- **vertical:** A boolean that determines whether the vertical guide is visible or not. Valid values are **true** or **false**. The default is **false** (no vertical guide).
- **thickness:** The thickness of the guide lines. Valid values are 1 and above. The default is 1.
- **color:** The guide's line color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **alpha:** The guide's line transparency value. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 20.
- **type:** The guide's line type. Valid values are **solid**, **dotted**, and **dashed**. The default is solid.
- **snap_h:** A boolean that determines whether the horizontal guide snaps to data points or not. Valid values are **true** or **false**. The default is **false** (no snap).
- **snap_v:** A boolean that determines whether the vertical guide snaps to data points or not. Valid values are **true** or **false**. The default is **false** (no snap).
- **connect:** By default, guides appear when the cursor is inside **chart_rect**. This attribute is a boolean that determines whether the guides appear when the cursor is outside of **chart_rect**. This can be used to connect the guides of different charts in a **composite chart**. Valid values are **true** or **false**. The default is **false** (show guides only when the cursor is inside **chart_rect**).
- **radius:** Guides can optionally have a circle that highlights data points when the cursor gets close to them. The *radius* attribute sets the radius of this circle in pixels. The default is 4.
- **fill_color:** The color inside the data circle. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **fill_alpha:** The transparency value inside the data circle. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 0. To hide the circle, set both *fill_alpha* and *line_alpha* to zero.
- **line_color:** The data circle's border color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **line_alpha:** The data circle's border transparency value. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 0. To hide the circle, set both *fill_alpha* and *line_alpha* to zero.
- **line_thickness:** The data circle's border thickness. The default is 0 pixels (no border).
- **text_h_alpha:** Guides can optionally have labels that display the axes values. This determines the labels' transparency for the horizontal guide. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 50. To hide the label for the horizontal guide, set this attribute to zero.

- **text_v_alpha:** Guides can optionally have labels that display the axes values. This determines the labels' transparency for the vertical guide. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 50. To hide the label for the vertical guide, set this attribute to zero.
- **prefix_h:** The characters to add before the guide's horizontal label (example: \$10). The default is nothing.
- **suffix_h:** The characters to add after the guide's horizontal label (example: 10%). The default is nothing.
- **prefix_v:** This characters to add before the guide's vertical label (example: \$10). The default is nothing.
- **suffix_v:** The characters to add after the guide's vertical label (example: 10%). The default is nothing.
- **decimals:** The number of decimal places to the right of the decimal point in the guide's number label (example: 10.45). It does not affect the string label. The default is zero (no decimals).
- **decimal_char:** The character to use at the left of a decimal fraction in the guide's number label (example: 10.5). It does not affect the string label. The default is '.' (dot or full stop).
- **separator:** The character to place between every group of thousands in the guide's number label (example: 1,00,000). It does not affect the string label. The default is nothing.
- **font:** The font used in the guide's labels. The default is Arial.
- **bold:** A boolean that indicates whether the font is bold or not in the guide's labels. The default is true.
- **size:** The font's size in the guide's labels. The default is 12.
- **text_color:** The font's color in the guide's labels. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **background_color:** This determines the labels' background color to make them visible over the graph. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. When omitted, guide labels have no background. The default is omitted (no background).

CHART LABEL

chart_label sets the attributes of the labels that appear over the graphs.

```
<chart_label prefix='string'
  suffix='string'
  decimals='int'
  decimal_char='string'
  separator='string'
  position='string'
  hide_zero='boolean'
  as_percentage='boolean'
  font='string'
  bold='boolean'
  size='int'
  color='string'
  alpha='int'
```



```
shadow='string'  
bevel='string'  
glow='string'  
blur='string'  
</>
```

- **prefix:** The characters to add before numbers (example: \$10). The default is nothing.
- **suffix:** The characters to add after numbers (example: 10%). The default is nothing.
- **decimals:** The number of decimal places to the right of the decimal point (example: 10.45). The default is zero (no decimals).
- **decimal_char:** The character to use at the left of a decimal fraction (example: 1.5). The default is '.' (dot or full stop).
- **separator:** The character to place between every group of thousands (example: 1,00,000). The default is nothing.
- **position:** The position where to place the labels. Each **chart type** has a different set of position values (see **Label Position** below).
- **hide_zero:** This determines whether to hide value labels equal to zero. The default is false (shows zero labels).
- **as_percentage:** This is relevant in pie charts only. It determines whether to display the values as raw data, or as percentages of the whole pie. The default is false.
- **font:** The font used for the chart labels. The default is Arial.
- **bold:** A boolean that indicates whether the font is bold or not. The default is true.
- **size:** The font's size. The default font size is calculated based on the background size.
- **color:** The font's color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **alpha:** This affects the labels' transparency only when the embedded font is used. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 100.
- **shadow:** The ID of a shadow **filter** to apply to chart labels. This is omitted by default (no shadow).
- **bevel:** The ID of a bevel **filter** to apply to chart labels. This is omitted by default (no bevel).
- **glow:** The ID of a glow **filter** to apply to chart labels. This is omitted by default (no glow).
- **blur:** The ID of a blur **filter** to apply to chart labels. This is omitted by default (no blur).

Label Position

Each **chart type** has different values for the 'position' key. The valid values are:

Line chart: center, above, below, left, right, hide

Column chart: top, bottom, middle, middle-up, middle-down, outside, hide

Stacked Column chart: top, bottom, middle, middle-up, middle-down, hide

Floating Column chart: inside, outside, hide

3D Column chart: over, middle, hide

Image Column chart: top, bottom, middle, middle-up, middle-down, outside, hide

Stacked 3D Column chart: middle, hide

Parallel 3D Column chart: over, middle, hide

Pie charts: inside, outside, circular, hide

3D Pie charts: inside, outside, circular, hide

Image Pie charts: inside, outside, circular, hide

Donut charts: inside, hide

Bar chart: left, center, right, outside, hide

Stacked Bar chart: left, center, right, hide

Floating Bar chart: inside, outside, hide

Area chart: center, above, below, left, right, hide

Stacked Area chart: center, above, below, left, right, hide

3D Area chart: above, hide

3D Stacked Area chart: above, hide

Candlestick chart: Candlestick chart has no chart labels. Use **tooltip** instead.

Scatter chart: center, above, below, left, right, hide

Polar chart: center, above, below, left, right, hide

Bubble chart: center, hide

Mixed chart: If one position key belonging to the above chart types isn't sufficient to show all the labels on a mixed chart, add more keys separated by underscores. For example, a column and line mixed chart can have the position key "top_above".

SERIES

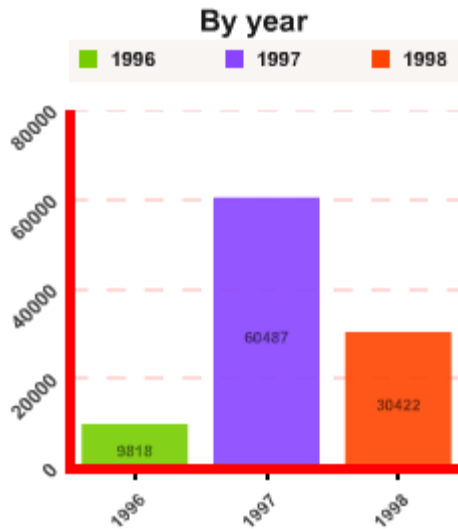
series determines the look of the series graphs of some chart types

- **bar_gap:** The gap between columns or bars in the same set. Valid values are from 0 to 100. Standard column and bar charts (not 3d) support negative bar_gap values, which makes the columns or bars overlap. The default is 0. This applies to column and bar charts only.
- **set_gap:** The gap between sets of columns or bars. Valid values are from 0 to 100. The default is 20. This applies to column and bar charts only.
- **transfer:** A boolean that transfers the series colors to categories. This is used if the chart has only one series (one row of data) and you want each category value to have different color. This doesn't apply to line, area, scatter, and mixed charts. The default value is false.

Example: By setting the series code in our template to

```
<series bar_gap='0' set_gap='20' transfer='true' />
```

Where transfer =true now results in the following colors in the chart



SERIES COLOR

series_color sets the colors to use for the chart series

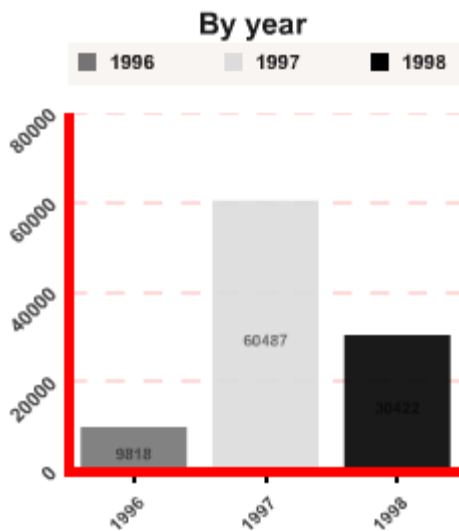
```
<series_color>
  <color>string</color>
  [ <color>string</color> ]
</series_color>
```

series_color contains any number of colors. Each color is a string of triple hexadecimal values representing the red, green, and blue components of a color.

If **series_color** is omitted, then ten default colors are used (77CC00, 8844FF, FF4400, 666666, DDDD00, 2244AA, C89341, 668899, 4C6b41, 844648). If the number of defined colors is less than needed, then the defined colors are repeated.

Example

```
<!-- set series color to gray scale -->
  <series_color>
    <color>747474</color>
    <color>DCDCDC</color>
    <color>000000</color>
  </series_color>
```



AXIS CATEGORY

`axis_category` sets the label attributes for the category-axis.

```
<axis_category skip='number'
  font='string'
  bold='boolean'
  size='number'
  color='string'
  alpha='number'
  orientation='string'
```

```
  shadow='string'
  bevel='string'
  glow='string'
  blur='string'
```

```
<!-- area, stacked area, line charts -->
  margin='boolean'
```

```
<!-- scatter and bubble charts -->
  min='number'
  max='number'
  steps='number'
  mode='string'
  prefix='string'
  suffix='string'
  decimals='number'
  decimal_char='string'
  separator='string'
```

/>

- **skip:** If this axis holds too many labels, the skip key allows skipping (hiding) some labels. A zero value doesn't hide any labels. If the skip value is 1, then the first label is displayed, the following label is skipped, and so on. If the skip value is 2, then the first label is displayed, the following 2 are skipped, and so on. The default is zero.
- **font:** The font used in the category-axis. The default is Arial.
- **bold:** A boolean that indicates whether the font is bold or not. The default is true.
- **size:** The font's size. The default font size is calculated based on the chart size.
- **color:** The font's color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **alpha:** This affects the labels' transparency only when the embedded font is used. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 90. To hide all labels in this axis, set the alpha to 0.
- **orientation:** This affects the labels' orientation only when the embedded font is used. Valid values are **horizontal**, **diagonal_up**, **diagonal_down**, **vertical_up**, and **vertical_down**. Polar charts also accept the value **circular**. The default value is horizontal.
- **shadow:** The ID of a shadow **filter** to apply to the category labels. This is omitted by default (no shadow).
- **bevel:** The ID of a bevel **filter** to apply to the category labels. This is omitted by default (no bevel).
- **glow:** The ID of a glow **filter** to apply to the category labels. This is omitted by default (no glow).
- **blur:** The ID of a blur **filter** to apply to the category labels. This is omitted by default (no blur).
- **margin:** This applies to area, stacked area, and line charts only. It is a boolean that indicates whether to leave a margin on the left and right of the graph, or bump it against the left and right chart borders. The default is true (leave a margin). In mixed charts, there's always a margin to align area and line charts with column charts.
- **min:** This applies to scatter and bubble charts only, when the category-axis is calculated like the value-axis. This determines the minimum value to start this axis with. The default is calculated from the chart's data.
- **max:** This applies to scatter and bubble charts only, when the category-axis is calculated like the value-axis. This determines the maximum value to end this axis with. The default is calculated from the chart's data.
- **steps:** This applies to scatter and bubble charts only, when the category-axis is calculated like the value-axis. This determines the number of steps between the minimum and maximum values. If the minimum value is negative, and the maximum value is positive, then 'steps' is the number of steps between zero and the larger of max and absolute min. The default is 4.
- **mode:** This applies to scatter and bubble charts only, when the category-axis is calculated like the value-axis. This determines the way the above min and max values are adjusted. Valid values are:

- **all:** Min and max are adjusted to show all the chart values. This is the default value.
- **trim:** Min and max are not adjusted, even if a chart value falls outside of this range and becomes partially or totally invisible.
- **prefix:** This applies to scatter and bubble charts only, when the category-axis is calculated like the value-axis. This determines the characters to add before the labels (example: \$10). The default is nothing.
- **suffix:** This applies to scatter and bubble charts only, when the category-axis is calculated like the value-axis. This determines the characters to add after the labels (example: 10%). The default is nothing.
- **decimals:** This applies to scatter and bubble charts only, when the category-axis is calculated like the value-axis. This determines the number of decimal places to the right of the decimal point (example: 10.45). The default is zero (no decimals).
- **decimal_char:** This applies to scatter and bubble charts only, when the category-axis is calculated like the value-axis. This determines the character to use at the left of a decimal fraction (example: 1.5). The default is '.' (dot or full stop).
- **separator:** This applies to scatter and bubble charts only, when the category-axis is calculated like the value-axis. This determines the character to place between every group of thousands (example: 1,00,000). The default is nothing.

AXIS CATEGORY LABEL

`axis_category_label` is an array that overrides the numeric, default `axis_category` labels in scatter and bubble charts

The first value in `axis_category_label` overrides the first axis label, the second value in `axis_category_label` overrides the second axis label, and so on.

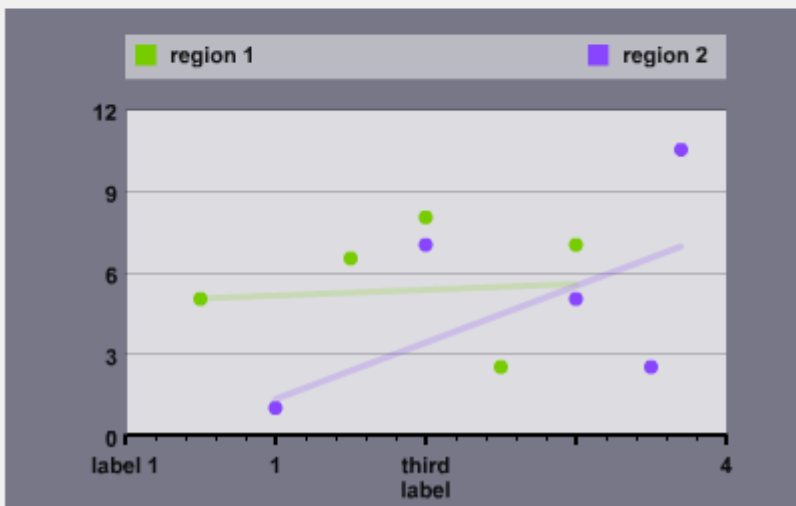
- To override a value label, place the new text in the corresponding position.
- To keep the original value label, place a **null** in the corresponding position.
- To hide a value label, place an empty string in the corresponding position.
- To add a multi-line label, break the lines with `\r`. Example: "Show this in\rtwo lines".
- If the `axis_category_label` holds less values than the number of the axis labels, then the unrepresented labels remain unchanged.

Example

```

<chart>
  <chart_type>scatter</chart_type>
  <!-- override the first label -->
  <!-- keep the default second label by using a null -->
  <!-- override the third label with a multi-line text -->
  <!-- hide the fourth label with an empty string-->
  <!-- keep the default fifth label by not representing it -->
  <axis_category_label>
    <string>label 1</string>
    <null/>
    <string>third\rlabel</string>
    <string></string>
  </axis_category_label>
</chart>

```



AXIS TICKS

`axis_ticks` sets the tick marks on the chart axes.

- **value_ticks:** A boolean that indicates whether the ticks on the value axis are visible or not. The default is false.
- **category_ticks:** A boolean that indicates whether the ticks on the category axis are visible or not. The default is true.
- **position:** A string that determines where on the axis to display the ticks. Valid values are *outside*, *inside*, and *centered*. The default is *outside*.
- **major_thickness:** The thickness of major ticks. Major ticks are those that appear next to the axis labels. The default is 2 pixels.

- **major_color:** The color of major ticks. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **minor_thickness:** The thickness of minor ticks. Minor ticks are those that appear between major ticks. The default is 1 pixel.
- **minor_color:** The color of minor ticks. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **minor_count:** This applies to the value axis only. It sets the number of minor ticks between every 2 major ticks. The default is 4. The category axis displays minor ticks only in place of its skipped labels.

Example

```

<!-- show centered ticks -->
<!-- show 3 minor ticks between every 2 major ticks -->
<!-- make minor ticks red -->
<axis_ticks value_ticks='true'
            category_ticks='true'
            position='centered'
            major_thickness='2'
            major_color='FF00FF'
            minor_thickness='1'
            minor_color='ff0000'
            minor_count='3'
            />

```

```
EOD;
```

```
$qlet['byyear_g45']['template'] = <<<EOD
```

```

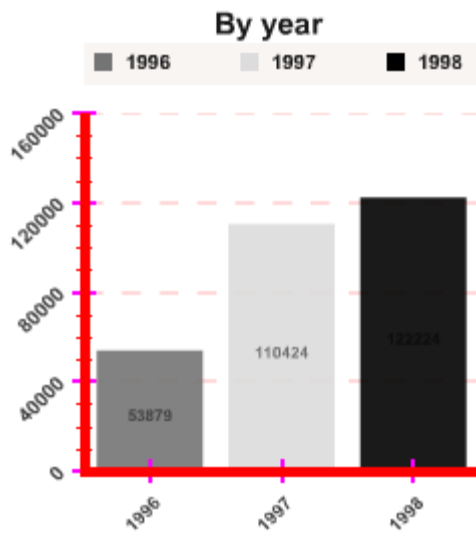
<!-- show centered ticks -->
  <!-- show 3 minor ticks between every 2 major ticks -->
  <!-- make minor ticks red -->
  <axis_ticks value_ticks='true'
              category_ticks='true'
              position='centered'
              major_thickness='2'
              major_color='FF00FF'
              minor_thickness='1'
              minor_color='ff0000'
              minor_count='3'
              />

```

```

<!-- show red bottom and left border lines -->
<chart_border top_thickness='0'
              bottom_thickness='5'
              left_thickness='5'

```

AXIS VALUE

axis_value sets the label attributes for the value-axis.

```
<axis_value min='float'
  max='float'
  mode='string'
  steps='int'

  prefix='string'
  suffix='string'
  decimals='int'
  decimal_char='string'
  separator='string'
  show_min='boolean'
  font='string'
  bold='boolean'
  size='int'
  color='string'
  background_color='string'
  alpha='int'
  orientation='string'

  shadow='string'
  bevel='string'
  glow='string'
  blur='string'
/>
```

- **min:** The minimum value to start the value-axis with. The default is zero in most charts.
- **max:** The maximum value to end the value-axis with. The default is calculated from the chart's data.
- **mode:** Determines the way the above *min* and *max* values are adjusted. Valid values are:
 - **all:** Min and max are adjusted to show all the chart values, including those outside of the visible **scrolling** area. This is the default value.
 - **stretch:** Min and max are repeatedly adjusted to show only the chart values within the visible **scrolling** area. This option is available to charts that can be **scrolled** (even when scrolling is not applied to them).
 - **trim:** Min and max are not adjusted, even if a chart value falls outside of this range and becomes partially or totally invisible. This option is available to charts that can be **scrolled** (even when scrolling is not applied to them), and to the scatter and bubble charts.
- **steps:** The number of steps between the minimum and maximum values. If the minimum value is negative and the maximum value is positive, then *steps* is the number of steps between zero and the larger of max and absolute min. The default is 4.
- **prefix:** The characters to add before the value numbers (example: \$10). The default is nothing.
- **suffix:** The characters to add after the value numbers (example: 10%). The default is nothing.
- **decimals:** The number of decimal places to the right of the decimal point (example: 10.45). The default is zero (no decimals).

- **decimal_char:** The character to use at the left of a decimal fraction (example: 1.5). The default is '.' (dot or full stop).
- **separator:** The character to place between every group of thousands (example: 1,00,000). The default is nothing.
- **show_min:** A boolean that indicates whether show or hide the first label in the value-axis. Hiding this first label might be necessary if it overlaps with the first label in the category axis. The default is true (show the first label).
- **font:** The font used in the value-axis. The default is Arial.
- **bold:** A boolean value that indicates whether the font is bold or not. The default is true.
- **size:** The font's size. The default font size is calculated based on the chart size.
- **color:** The font's color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **background_color:** This applies to *polar* charts only. It determines the labels' background color to make them visible over the graphs. When omitted, *axis_value* labels have no background. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is omitted (no background).
- **alpha:** This affects the labels' transparency, only when the embedded font is used. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 90. To hide all labels in this axis, set the alpha to 0.
- **orientation:** This affects the labels' orientation, only when the embedded font is used. Valid values are *horizontal*, *diagonal_up*, *diagonal_down*, *vertical_up*, and *vertical_down*. The default value is *horizontal*.
- **shadow:** The ID of a shadow **filter** to apply to the value labels. This is omitted by default (no shadow).
- **bevel:** The ID of a bevel **filter** to apply to the value labels. This is omitted by default (no bevel).
- **glow:** The ID of a glow **filter** to apply to the value labels. This is omitted by default (no glow).
- **blur:** The ID of a blur **filter** to apply to the value labels. This is omitted by default (no blur).

Two Value-Axes

In some cases, it might be necessary to display two value-axes to represent data using two different scales. See **axis_value_label** for more information.

AXIS VALUE LABEL

axis_value_label is an array that overrides the default **axis_value** labels to display custom text instead. This can also be used to reverse the axis_value.

```
<axis_value_label>
  <string>...</string>
  [ <string>...</string> ]
</axis_value_label>
```

The first value in **axis_value_label** overrides the first axis label, the second value in **axis_value_label** overrides the second axis label, and so on.

- To override a value label, place the new text in the corresponding position.
- To keep the original value label, place a **null** in the corresponding position.
- To hide a value label, place an empty string in the corresponding position.
- To add a multi-line label, break the lines with **\r**. Example: "Show this in\rtwo lines".
- If the **axis_value_label** holds less values than the number of the axis labels, then the unrepresented labels remain unchanged.

```
<!-- override the first label -->
<!-- keep the default second label by using a null -->
<!-- override the third label with a multi-line text -->
<!-- hide the forth label with an empty string-->
<!-- keep the default fifth label by not representing it -->
<axis_value_label>
  <string>label 1</string>
  <null/>
  <string>third\rlabel</string>
  <string></string>
</axis_value_label>
```

FILTER

filter defines any number of filters to enhance the look of or highlight different graphic elements.

<filter>

```
<!-- shadow -->
<shadow id='string'
  distance='int'
  angle='int'
  color='string'
  alpha='int'
  blurX='int'
  blurY='int'
  strength='int'
  quality='int'
  inner='boolean'
  knockout='boolean'
  hideObject='boolean'
 />
```

```
<!-- bevel -->
<bevel id='string'
  distance='int'
  angle='int'
  highlightColor='string'
  highlightAlpha='int'
  shadowColor='string'
  shadowAlpha='int'
  blurX='int'
  blurY='int'
  strength='int'
  quality='int'
  type='string'
  knockout='boolean'
 />
```

```
<!-- glow -->
<glow id='string'
  color='string'
  alpha='int'
  blurX='int'
  blurY='int'
  strength='int'
  quality='int'
  inner='boolean'
  knockout='boolean'
 />
```

```
<!-- blur -->
<blur id='string'
  blurX='int'
  blurY='int'
  quality='int'
 />
```

</filter>

After defining filters, one or more can be applied to different chart elements. For example, you can define one shadow and two bevel filters. You can apply just the shadow to one element, just the first bevel to a second element, and both the shadow and the second bevel to a third element.

The effects created with filters are not visible in printed charts.

Shadow

The *shadow* filter has the following attributes:

- **id:** Any unique ID to identify the filter. A filter is applied to an element by passing it this ID.
- **distance:** The offset distance of the shadow in pixels. The default is 4.
- **angle:** The angle of the shadow. Valid values are 0 to 360. The default is 45.
- **color:** The color of the shadow. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **alpha:** The transparency value of the shadow color. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 20.
- **blurX:** The amount of horizontal blur. Valid values are 0 to 255. The default value is 4. Values that are a power of 2 (such as 2, 4, 8, 16 and 32) are optimized to render more quickly than other values.
- **blurY:** The amount of vertical blur. Valid values are 0 to 255. The default value is 4. Values that are a power of 2 (such as 2, 4, 8, 16 and 32) are optimized to render more quickly than other values.
- **strength:** The strength of the shadow. The higher the value, the stronger the contrast between the shadow and the background. Valid values are 0 to 255. The default is 1.
- **quality:** The number of times to apply the filter. The default value is 1, which is equivalent to low quality. A value of 2 is medium quality, and a value of 3 is high quality.
- **inner:** Indicates whether the shadow is an inner shadow or not. A value of true specifies an inner shadow. The default is false, an outer shadow (around the outer edges of the object).
- **knockout:** Applies a knockout effect (true), which makes the object invisible and reveals the background through it (see the red text below). The default is false (show the object).
- **hideObject:** Indicates whether the object is hidden or not. A value of true indicates that the object itself is not drawn; only the shadow is visible. The default is false (show the object).

Bevel

The *bevel* filter has the following attributes:

- **id:** Any unique ID to identify the filter. A filter is applied to an element by passing it this ID.
- **distance:** The offset distance of the bevel in pixels. The default is 4.

- **angle:** The angle of the shadow. Valid values are 0 to 360. The default is 45.
- **highlightColor:** The highlight color of the bevel. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "FFFFFF" (white).
- **highlightAlpha:** The transparency value of the highlight color. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 20.
- **shadowColor:** The shadow color of the bevel. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **shadowAlpha:** The transparency value of the shadow color. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 20.
- **blurX:** The amount of horizontal blur. Valid values are 0 to 255. The default value is 4. Values that are a power of 2 (such as 2, 4, 8, 16 and 32) are optimized to render more quickly than other values.
- **blurY:** The amount of vertical blur. Valid values are 0 to 255. The default value is 4. Values that are a power of 2 (such as 2, 4, 8, 16 and 32) are optimized to render more quickly than other values.
- **strength:** The strength of the bevel. The higher the value, the stronger the contrast between the bevel and the background. Valid values are 0 to 255. The default is 1.
- **quality:** The number of times to apply the filter. The default value is 1, which is equivalent to low quality. A value of 2 is medium quality, and a value of 3 is high quality.
- **type:** The type of bevel. Valid values are *inner*, *outer*, and *full*. The default value is *inner*.
- **knockout:** Applies a knockout effect (true), which makes the object invisible and reveals the background through it. The default is false (show the object).

Glow

The *glow* filter has the following attributes:

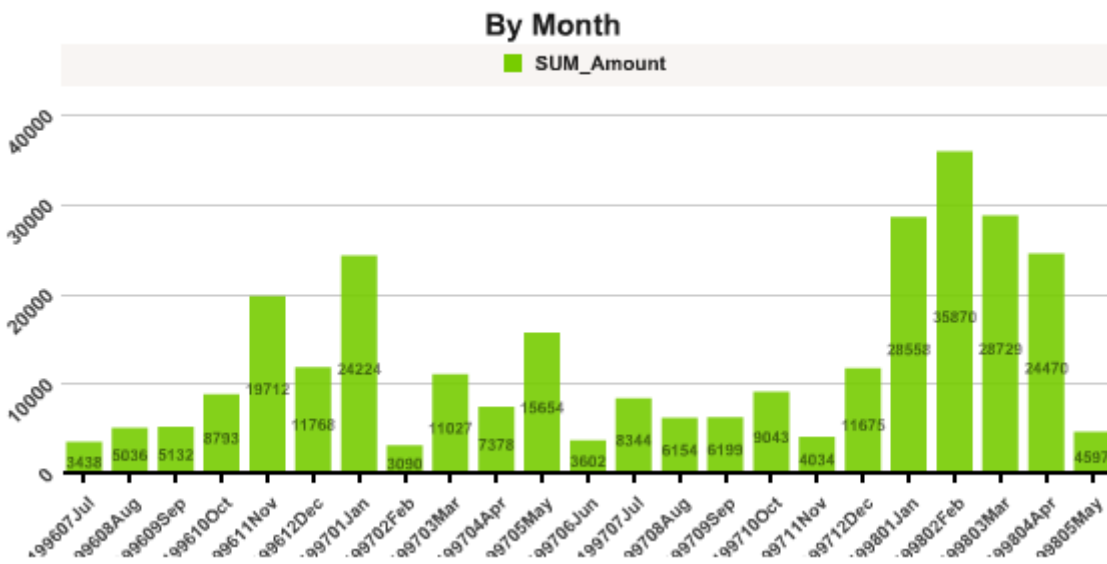
- **id:** Any unique ID to identify the filter. A filter is applied to an element by passing it this ID.
- **color:** The color of the glow. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "88FF00" (green).
- **alpha:** The transparency value of the glow color. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 20.
- **blurX:** The amount of horizontal blur. Valid values are 0 to 255. The default value is 4. Values that are a power of 2 (such as 2, 4, 8, 16 and 32) are optimized to render more quickly than other values.
- **blurY:** The amount of vertical blur. Valid values are 0 to 255. The default value is 4. Values that are a power of 2 (such as 2, 4, 8, 16 and 32) are optimized to render more quickly than other values.

- **strength:** The strength of the glow. The higher the value, the stronger the contrast between the glow and the background. Valid values are 0 to 255. The default is 1.
- **quality:** The number of times to apply the filter. The default value is 1, which is equivalent to low quality. A value of 2 is medium quality, and a value of 3 is high quality.
- **inner:** Indicates whether the glow is an inner glow or not. A value of true specifies an inner glow. The default is false, an outer glow (around the outer edges of the object).
- **knockout:** Applies a knockout effect (true), which makes the object invisible and reveals the background through it (see the red text below). The default is false (show the object).

Blur

The *blur* filter has the following attributes:

- **id:** Any unique ID to identify the filter. A filter is applied to an element by passing it this ID.
- **blurX:** The amount of horizontal blur. Valid values are 0 to 255. The default value is 4. Values that are a power of 2 (such as 2, 4, 8, 16 and 32) are optimized to render more quickly than other values.
- **blurY:** The amount of vertical blur. Valid values are 0 to 255. The default value is 4. Values that are a power of 2 (such as 2, 4, 8, 16 and 32) are optimized to render more quickly than other values.
- **quality:** The number of times to apply the filter. The default value is 1, which is equivalent to low quality. A value of 2 is medium quality, and a value of 3 is high quality.



We apply the following filter to the above chart

```
<!-- define two bevel and one shadow filters -->
<filter>
  <bevel id='bevel1' angle='60' blurX='100' blurY='100' distance='35' highlightColor='0000ff' shadowColor='ff0000' />
  <bevel id='bevel2' angle='45' blurX='3' blurY='3' distance='3' highlightAlpha='50' highlightColor='ffffff' />
  <shadow id='shadow1' distance='5' angle='45' color='000000' alpha='75' blurX='10' blurY='10' />
</filter>

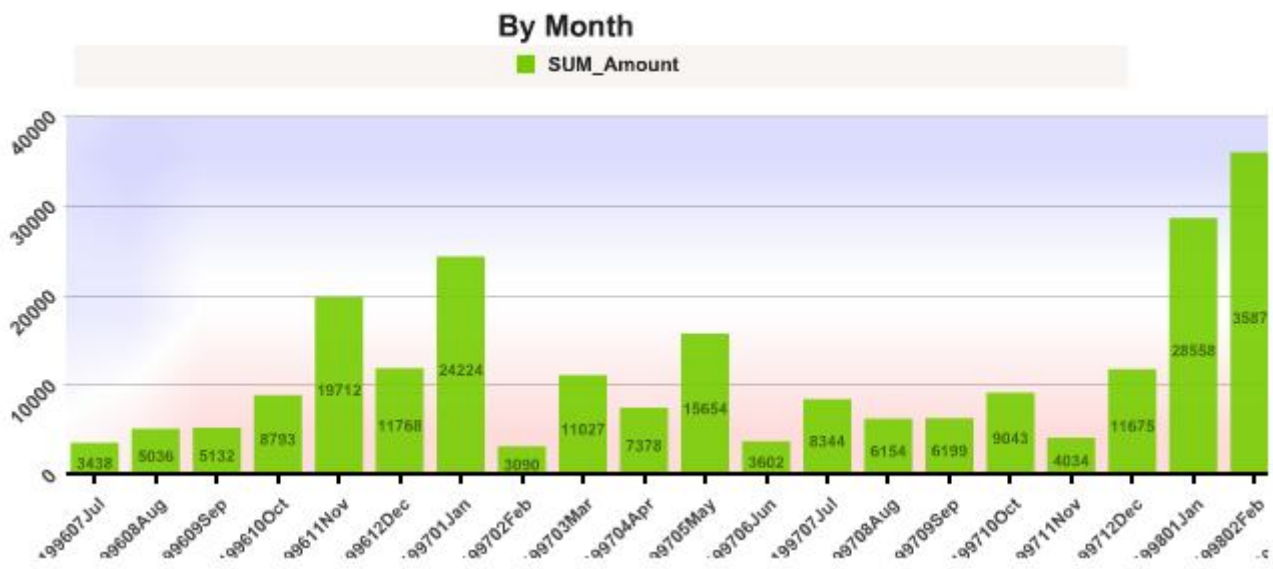
<!-- apply the first bevel to chart_rect -->
<chart_rect x='100' width='701' bevel='bevel1' />
```

```
$qlet['bymonth_g49']['template'] = <<<EOD

<!-- define two bevel and one shadow filters -->
<filter>
  <bevel id='bevel1' angle='60' blurX='100' blurY='100' distance='35' highlightColor='0000ff' shadowColor='ff0000' />
  <bevel id='bevel2' angle='45' blurX='3' blurY='3' distance='3' highlightAlpha='50' highlightColor='ffffff' />
  <shadow id='shadow1' distance='5' angle='45' color='000000' alpha='75' blurX='10' blurY='10' />
</filter>

<!-- apply the first bevel to chart_rect -->
<chart_rect x='100' width='701' bevel='bevel1' />

<chart_guide horizontal='true' vertical='true' thickness='1' color='000000' alpha='35' type='dashed' radius='3' fill_alp
<chart_label color='000000' alpha='50' size='8' position='middle' as_percentage='true' />
```



DRAW

draw holds any number of elements to draw. A *draw* element can be a circle, image (JPEG, unanimated GIF, PNG, or SWF), line, rect, or text.

In the default template, we are using the Draw command to add the title of the chart or gauge.

<draw>

```
<!-- circle -->
<circle layer='string'
  transition='string'
  delay='number'
  duration='number'
  x='number'
  y='number'
  radius='number'
  fill_color='string'
  fill_alpha='number'
  line_color='string'
  line_alpha='number'
  line_thickness='number'
  shadow='string'
  bevel='string'
  glow='string'
  blur='string'
  blend='string'
/>

<!-- image (JPEG, unanimated GIF, PNG, or SWF) -->
<image layer='string'
  transition='string'
  delay='number'
  duration='number'
  url='string'
  x='number'
  y='number'
  width='number'
  height='number'
  rotation='number'
  alpha='number'
  shadow='string'
  bevel='string'
  glow='string'
  blur='string'
  timeout='number'
  retry='number'
  blend='string'
/>

<!-- line -->
<line layer='string'
  transition='string'
  delay='number'
  duration='number'
  x1='number'
  y1='number'
```

```
x2='number'  
y2='number'  
line_color='string'  
line_alpha='number'  
line_thickness='number'  
shadow='string'  
bevel='string'  
glow='string'  
blur='string'  
blend='string'  
</>
```

```
<!-- rect -->
```

```
<rect layer='string'  
  transition='string'  
  delay='number'  
  duration='number'  
  x='number'  
  y='number'  
  width='number'  
  height='number'  
  rotation='number'  
  fill_color='string'  
  fill_alpha='number'  
  line_color='string'  
  line_alpha='number'  
  line_thickness='number'  
  corner_tl='number'  
  corner_tr='number'  
  corner_br='number'  
  corner_bl='number'  
  shadow='string'  
  bevel='string'  
  glow='string'  
  blur='string'  
  blend='string'  
</>
```

```
<!-- text -->
```

```
<text layer='string'  
  transition='string'  
  delay='number'  
  duration='number'  
  x='number'  
  y='number'  
  width='number'  
  height='number'  
  h_align='string'  
  v_align='string'  
  rotation='number'  
  font='string'  
  bold='boolean'  
  size='number'  
  color='string'  
  alpha='number'  
  shadow='string'  
  bevel='string'  
  glow='string'  
  blur='string'  
  blend='string'  
>text to draw</text>
```

```
<!-- button -->
```

```
<button layer='string'
```

```

transition='string'
delay='number'
duration='number'
url_idle='string'
url_over='string'
url_press='string'
x='number'
y='number'
width='number'
height='number'
alpha='number'
shadow='string'
bevel='string'
glow='string'
blur='string'
timeout='number'
retry='number'
blend='string'
/>

```

```
</draw>
```

Circle

Each circle has the following attributes:

- **layer:** The layer to draw the element in. Valid values are **background**, which places the element behind the chart, and **foreground**, which places the element in front of the chart. The default is **foreground**. Elements placed in the same layer are drawn in the same order they're arranged in the **draw** array.
- **transition:** The way to make this element appear. Valid values are **dissolve**, **drop**, **spin**, **scale**, **zoom**, **blink**, **slide_right**, **slide_left**, **slide_up**, **slide_down**, and **none**. The default is **none**, which draws the element immediately without a transition.
- **delay:** The delay in seconds before starting the transition. The default is zero.
- **duration:** The transition's duration in seconds. The default is 1.
- **x:** The horizontal position of the circle's center relative to the upper left corner of the **background** (0, 0). The default is zero.
- **y:** The vertical position of the circle's center relative to the upper left corner of the **background** (0, 0). The default is zero.
- **radius:** The circle's radius. The default is 100.
- **fill_color:** The color inside the circle. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **fill_alpha:** The transparency value inside the circle. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 100.
- **line_color:** The border's color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).

- **line_alpha:** The border's transparency value. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 100.
- **line_thickness:** The border's thickness. The default is 0 pixels.
- **shadow:** The ID of a shadow **filter** to apply to the circle. This is omitted by default (no shadow).
- **bevel:** The ID of a bevel **filter** to apply to the circle. This is omitted by default (no bevel).
- **glow:** The ID of a glow **filter** to apply to the circle. This is omitted by default (no glow).
- **blur:** The ID of a blur **filter** to apply to the circle. This is omitted by default (no blur).
- **blend:** Determines how to mix the graphic's colors with other graphics underneath it. Valid values are:
 - **add**
 - **darken**
 - **difference**
 - **hardlight**
 - **invert**
 - **lighten**
 - **multiply**
 - **normal**
 - **overlay**
 - **screen**
 - **subtract**

Image

Each image has the following attributes:

- **layer:** The layer to draw the element in. Valid values are **background**, which places the element behind the chart, and **foreground**, which places the element in front of the chart. The default is **foreground**. Elements placed in the same layer are drawn in the same order they're arranged in the **draw** array.
- **transition:** The type of the transition. Valid values are **dissolve**, **drop**, **spin**, **scale**, **zoom**, **blink**, **slide_right**, **slide_left**, **slide_up**, **slide_down**, and **none**. The default is **none**, which draws the element immediately without a transition.
- **delay:** The delay in seconds before starting the transition. The default is zero.
- **duration:** The transition's duration in seconds. The default is 1.
- **url:** Relative or absolute URL of a JPEG, unanimated GIF, PNG, or SWF flash animation. This must be in the same sub-domain as the *charts.swf* file (flash security).

- **x**: The horizontal position of the image's top-left corner relative to the upper left corner of the **background** (0, 0). This parameter is omitted by default.
- **y**: The vertical position of the image's top-left corner relative to the upper left corner of the **background** (0, 0). This parameter is omitted by default.
- **width**: The image's width. This parameter is omitted by default.
- **height**: The image's height. This parameter is omitted by default.
- **rotation**: The image's rotation around its upper-left corner (x, y). The default value is zero.
- **alpha**: The image's transparency value. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 100.
- **blend**: Determines how to mix the graphic's colors with other graphics underneath it. Valid values are:
 - **add**
 - **darken**
 - **difference**
 - **hardlight**
 - **invert**
 - **lighten**
 - **multiply**
 - **normal**
 - **overlay**
 - **screen**
 - **subtract**
- **timeout**: The number of seconds to wait before the image loading times out. The default is 30 seconds.
- **retry**: The number of times to retry loading the image before displaying a loading error message. The default is 2 (try twice).

Line

Each line has the following attributes:

- **layer**: The layer to draw the element in. Valid values are **background**, which places the element behind the chart, and **foreground**, which places the element in front of the chart. The default is **foreground**. Elements placed in the same layer are drawn in the same order they're arranged in the **draw** array.
- **transition**: The type of the transition. Valid values are **dissolve**, **drop**, **spin**, **scale**, **zoom**, **blink**, **slide_right**, **slide_left**, **slide_up**, **slide_down**, and **none**. The default is **none**, which draws the element immediately without a transition.
- **delay**: The delay in seconds before starting the transition. The default is zero.

- **duration:** The transition's duration in seconds. The default is 1.
- **x1:** The horizontal position of the line's start point relative to the upper left corner of the **background** (0, 0). The default is zero.
- **y1:** The vertical position of the line's start point relative to the upper left corner of the **background** (0, 0). The default is zero.
- **x2:** The horizontal position of the line's end point relative to the upper left corner of the **background** (0, 0). The default is 100.
- **y2:** The vertical position of the line's end point relative to the upper left corner of the **background** (0, 0). The default is 100.
- **line_color:** The line's color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **line_alpha:** The line's transparency value. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 100.
- **line_thickness:** The line's thickness. The default is 1 pixel.
- **shadow:** The ID of a shadow **filter** to apply to the line. This is omitted by default (no shadow).
- **bevel:** The ID of a bevel **filter** to apply to the line. This is omitted by default (no bevel).
- **glow:** The ID of a glow **filter** to apply to the line. This is omitted by default (no glow).
- **blur:** The ID of a blur **filter** to apply to the line. This is omitted by default (no blur).
- **blend:** Determines how to mix the graphic's colors with other graphics underneath it. Valid values are:
 - **add**
 - **darken**
 - **difference**
 - **hardlight**
 - **invert**
 - **lighten**
 - **multiply**
 - **normal**
 - **overlay**
 - **screen**
 - **subtract**

Rectangle

Each rect has the following attributes:

- **layer:** The layer to draw the element in. Valid values are **background**, which places the element behind the chart, and **foreground**, which places the element in front of the chart. The default is **foreground**. Elements placed in the same layer are drawn in the same order they're arranged in the **draw** array.
- **transition:** The type of the transition. Valid values are **dissolve**, **drop**, **spin**, **scale**, **zoom**, **blink**, **slide_right**, **slide_left**, **slide_up**, **slide_down**, and **none**. The default is **none**, which draws the element immediately without a transition.
- **delay:** The delay in seconds before starting the transition. The default is zero.
- **duration:** The transition's duration in seconds. The default is 1.
- **x:** The horizontal position of the rectangle's upper left corner relative to the upper left corner of the **background** (0, 0). The default is zero.
- **y:** The vertical position of the rectangle's upper left corner relative to the upper left corner of the **background** (0, 0). The default is zero.
- **width:** The rectangle's width. The default is 100.
- **height:** The rectangle's height. The default is 100.
- **rotation:** The rectangle's rotation around its upper left point (x, y). The default value is zero.
- **fill_color:** The color inside the rectangle. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **fill_alpha:** The transparency value inside the rectangle. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 100.
- **line_color:** The border's color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **line_alpha:** The border's transparency value. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 100.
- **line_thickness:** The border's thickness. The default is 0 pixels.
- **corner_tl:** The top-left corner radius in pixels. If this is set to a value above zero, then the top-left corner gets rounded by this radius amount. The default is omitted (square corner).
- **corner_tr:** The top-right corner radius in pixels. If this is set to a value above zero, then the top-right corner gets rounded by this radius amount. The default is omitted (square corner).
- **corner_br:** The bottom-right corner radius in pixels. If this is set to a value above zero, then the bottom-right corner gets rounded by this radius amount. The default is omitted (square corner).

- **corner_bl**: The bottom-left corner radius in pixels. If this is set to a value above zero, then the bottom-left corner gets rounded by this radius amount. The default is omitted (square corner).
- **blend**: Determines how to mix the graphic's colors with other graphics underneath it. Valid values are:
 - **add**
 - **darken**
 - **difference**
 - **hardlight**
 - **invert**
 - **lighten**
 - **multiply**
 - **normal**
 - **overlay**
 - **screen**
 - **subtract**

Text

Each text has the following attributes:

- **layer**: The layer to draw the element in. Valid values are **background**, which places the element behind the chart, and **foreground**, which places the element in front of the chart. The default is **foreground**. Elements placed in the same layer are drawn in the same order they're arranged in the **draw** array.
- **transition**: The type of the transition. Valid values are **dissolve**, **drop**, **spin**, **scale**, **zoom**, **blink**, **slide_right**, **slide_left**, **slide_up**, **slide_down**, and **none**. The default is **none**, which draws the element immediately without a transition.
- **delay**: The delay in seconds before starting the transition. The default is zero.
- **duration**: The transition's duration in seconds. The default is 1.
- **x**: The horizontal position of the rectangle's upper left corner relative to the upper left corner of the **background** (0, 0).
- **y**: The vertical position of the rectangle's upper left corner relative to the upper left corner of the **background** (0, 0).
- **width**: The rectangle's width.
- **height**: The rectangle's height.
- **h_align**: The text's horizontal alignment inside the rectangle. Valid values are *left*, *center*, and *right*. The default is *left*.
- **v_align**: The text's vertical alignment inside the rectangle. Valid values are *top*, *middle*, and *bottom*. The default is *top*.

- **rotation:** The rectangle's rotation. This works only when the embedded font is used . The rectangle rotates around its upper left point (x, y). The default value is zero.
- **font:** The font used to draw the text . The default is Arial.
- **bold:** A boolean value that indicates whether the font is bold or not. The default is true.
- **size:** The font's size. The default font size is calculated based on the **background** size.
- **color:** The font's color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **alpha:** This affects the text's transparency, only when the embedded font is used. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 100.
- **shadow:** The ID of a shadow **filter** to apply to the text. This is omitted by default (no shadow).
- **bevel:** The ID of a bevel **filter** to apply to the text. This is omitted by default (no bevel).
- **glow:** The ID of a glow **filter** to apply to the text. This is omitted by default (no glow).
- **blur:** The ID of a blur **filter** to apply to the text. This is omitted by default (no blur).
- **blend:** Determines how to mix the graphic's colors with other graphics underneath it. Valid values are:
 - **add**
 - **darken**
 - **difference**
 - **hardlight**
 - **invert**
 - **lighten**
 - **multiply**
 - **normal**
 - **overlay**
 - **screen**
 - **subtract**

Multi-line Text

To draw multi-line text, break the text with `\r`. Example: "Show this in\rtwo lines".

LEGEND

legend sets the legend's attributes. The legend is the area that identifies the colors assigned to the graphs.

Legends are available in all chart types except for candlestick, floating, and image charts.

- **transition:** The transition effect used to show the legend. Valid values are **dissolve**, **drop**, **spin**, **scale**, **zoom**, **blink**, **slide_right**, **slide_left**, **slide_up**, **slide_down**, and **none**. The default is **none**, which draws the legend immediately without a transition.
- **delay:** The delay in seconds before starting the transition. The default is zero.
- **duration:** The transition's duration in seconds. The default is 1.
- **x:** The horizontal position of the legend's upper left corner relative to the upper left corner of the **background** (0, 0).
- **y:** The vertical position of the legend's upper left corner relative to the upper left corner of the **background** (0, 0).
- **width:** The legend's width. The default width is calculated to fit within the **background**. If the legend's rectangle is too small for its content, then it expands as necessary.
- **height:** The legend's height. The default width is calculated to fit within the **background**. If the legend's rectangle is too small for its content, then it expands as necessary.
- **toggle:** A boolean that determines whether to enable clicking on legend labels to turn on or off series graphics. Valid values are **true** or **false**. The default is true (enable this feature).
- **toggle_children:** A boolean that determines whether to toggle children charts together with the parent or not (see **composite charts**). Valid values are **true** or **false**. The default is false.
- **layout:** This indicates if the legend is visible, and whether it is horizontal or vertical. Valid values are *horizontal*, *vertical*, and *hide*. The default is *vertical* for pie charts, and *horizontal* for all other charts.
- **margin:** The space between the legend's edge and its content. The default is 5.
- **bullet:** Determines the bullet shape. Valid values are *square*, *circle*, and *line*. The default is *square*.
- **font:** The font used in the legend's labels. The default is *Arial*.
- **bold:** A boolean that indicates whether the font is bold or not. Valid values are **true** or **false**. The default is true.
- **size:** The font's size. This also affects the size of the legend's color squares. The default font size is calculated based on the **background** size.
- **color:** The font's color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **alpha:** This affects the labels' transparency, only when the embedded font is used. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 90.

- **fill_color:** The legend's background color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "FFFFFF" (white).
- **fill_alpha:** The background's transparency. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 30.
- **line_color:** The border's color. This is a string holding triple hexadecimal values representing the red, green, and blue components of a color. The default is "000000" (black).
- **line_alpha:** The border's transparency value. Valid values are 0 (fully transparent) to 100 (fully opaque). The default is 0.
- **line_thickness:** The border's line thickness. Valid values are 0 and above. The default is 1.
- **shadow:** The ID of a shadow **filter** to apply to the legend. This is omitted by default (no shadow).
- **bevel:** The ID of a bevel **filter** to apply to the legend. This is omitted by default (no bevel).
- **glow:** The ID of a glow **filter** to apply to the legend. This is omitted by default (no glow).
- **blur:** The ID of a blur **filter** to apply to the legend. This is omitted by default (no blur).