AutoReport v5
By Marcelo Dantas

Changelog

v5.0 - First release.

Introduction

AutoReport is an application used to generate reports either on HTML or PDF formats, which can contain images, texts and charts from external SQLite databases or CSV files.
Every element on a report is configurable and can be modified as desired to fit the requirements of the report's audience.
The reports are created from an object oriented definition file, which describes the images, texts and charts added to the report.
Image objects can be of any format used by HTML pages. Text objects can be added directly to the definition file or sourced from text files or external commands.
Chart objects use the Google Charts library and their data sets can be added directly, sourced from SQL queries or external CSV files.

Here is the general format (sections) of a report definition file:

```
{report}
        {defaults}
                {text}
                {image}
                {chart}
                {variable}
                {page}
        {pages}
                [...]
```

Report Section

The report section describes the format of the report pages as well as their contents.
Components of the report section are:
offline - *boolean* - Defines if AutoReport will use a local copy of Google Charts or an online copy. This allows AutoReport to be used on environments with no Internet access. Default: false (use online)
language - *text* - Defines the language which will be used by the Google Charts library and AutoReport in general to select language and translation files. The options are 'en', 'de', 'es', 'fr' and 'pt_br'. If not defined, the default is 'en'.
title - *text* - Defines the title which will be used for the report html page. If not defined, 'Report v5.x' is used.
stylesheet - *text* - Defines the location of the stylesheet file to be used by the report generator to format the report pages. If not defined, the default.css file will be used.
defaults - *object* - Defines the default attributes which will be added to each object.
pages - *table* - Defines each page of the generated report.

Defaults Section

The default section defines default attributes of each object which will be used to create pages, as well as the page itself.
Attributes defined here will be added automatically to the respective object every time one is created. For example: if it is defined here that the text object will have the color red, every text object defined, unless stated otherwise, will have the color red.
The objects which can be defined here are ‘text’, ‘image’, ‘chart’, ‘variable’ and ‘page’.

Page Section

The page section brings objects together to define the final report. The page level is a JSON table which contains each ‘page’ object, which is made from instances of ‘text’, ‘image’, ‘chart’ and ‘variable’. For each entry on the page section a new page will be generated on the report.

Text Object

Text objects define text boxes which can be added to the report pages. These objects are used for any text that is displayed on the reports.
Each text object can have the following attributes:

object - *text* - Always “text”.
style - *object* - Defines the format and position of the text object. Follows the syntax of HTML CSS style sheets.
text - *text* - Text content of the text object, if added directly to the report definition file.
src - *text* - Source of the text content. Can be ‘sql’, ‘file’ or ‘cmd’. If defined, its result will override the ‘text’ attribute.
sql - *text* - If ‘src’ is ‘sql’, this will define the SQL query used to set the ‘text’ attribute. Ignored otherwise.
file - *text* - If ‘src’ is ‘file’, this will define the text file used to set the ‘text’ attribute. Ignored otherwise.
cmd - *text* - If ‘src’ is ‘cmd’, this will define an external command to be executed, which output will set the ‘text’ attribute. Ignored otherwise.

Example of a text object definition:

```
{
        "object": "text",
        "style": {
                "top": 25,
                "right": 40,
                "width": 700,
                "height": 200,
                "color": "white",
                "font-size": 36,
                "text-align": "right"
        },
        "text": "Monthly Report - {monthText}/{year}"
}
```

Image Object

Image objects are used to illustrate the report and can use the same image formats used on HTML pages.
Each image object can have the following attributes:

object - *text* - Always “image”.
style - *object* - Defines the format and position of the image object. Follows the syntax of HTML CSS style sheets.
src - *text* - Defines the source of the image file.
width - *integer* - Defines the width of the image.
height - *integer* - Defines the height of the image.

Example of an image object definition:

```
{
        "object": "image",
        "style": {
                "top": 0,
                "left": 0
        },
        "src": "{relPath}html_files/images/header.jpg",
        "width": 1100,
        "height": 100
}
```

Chart Object

Chart objects use the Google Charts library to implement charts which are used to represent data collections on the reports. Charts can be generated from SQL queries, CSV files or entered manually to the report definition as json data points.
Each chart object can have the following attributes:

object - *text* - Always “chart”.
style - *object* - Defines the format and position of the chart object. Follows the syntax of HTML CSS style sheets.
title - *text* - Defines the title of the chart
type - *text* - Defines the type of chart to be generated. Can be one of the following:

area
bar
bubble
calendar
column
line
pie
sankey
scatter
table

src - *text* - Defines the source of the data points to be used when creating the chart. It can be ‘sql’, ‘csv’ or ‘last’. If the source is ‘last’ the chart will use the same data set as the one last added to the report. This is useful when creating table charts immediately following a graphical one.
sql - *text* - If ‘src’ is ‘sql’, this will define the SQL query used to define the chart data points. Ignored otherwise.
csv - *text* - If ‘src’ is ‘csv’, this will define the CSV text file used to define the chart data points. Ignored otherwise.
db - *text* - Defines the database file to be used for the SQL query. Must be present when ‘src’ is set to ‘sql’.
data - *table* - Defines the data points of the chart. Its content is overridden by the ‘src’ entry.
centered - *boolean* - Defines if the chart is to be centered on the page.
legend - *text* - Defines the location of the chart legend. Can be ‘left’, ‘right’, ‘top’, ‘bottom’.
divisor - *integer* - Defines a number to which the data points will be divided.
decimals - *integer* - Defines how many decimal places each divided number will have.
noLabel - *boolean* - Defines if the chart labels are to be omitted.
topN - *integer* - Defines how many data points to be added to the chart.
fontSize - *integer* - Defines the size of the chart labels text.
titleSize - *integer* - Defines the size of the chart title text.
titleColor - *text* - Defines the color of the chart title in #rrggbb hexadecimal format.

annotationSize - *integer* - Defines the font size of the data point annotation text.
annotationColor - *text* - Defines the color of the data point annotation text in the #rrggbb hexadecimal format.
alwaysOutside - *boolean* - Defines if the data point annotation values should always be written outside the columns.
slantedText - *boolean* - Defines if the labels for the X axis are shown horizontal or slanted.
labelAsAlpha - *boolean* - Defines if a numeric label field should be treated as strings.
logScale - *boolean* - Defines if the Y scale is linear or logarithmic.
is3D - *boolean* - Defines if the pie chart is to be drawn in 3D perspective mode.
isStacked - *boolean* - Defines if the column chart is to be drawn with parallel or stacked data points.
xMin - *integer* - Defines the minimum value of the X axis.
xMax - *integer* - Defines the maximum value of the X axis.
showTextEvery - *integer* - Defines how many data points to write labels on.
backgroundColor - *text* - Defines the chart background color in the #rrggbb hexadecimal format.
revDNS - *boolean* - Defines if IP Address fields should be resolved to their reverse DNS names. This option slows down the chart generation considerably when turned on.
fields - *object* - Comma separated list of field types to override the assumed type of data points. E.G. treat a number result as a string.
options - *object* - Defines additional Google Charts library options which will be added to the chart, overriding its default configuration.

Example of a chart object definition:

```
            {
                    "object": "chart",
                    "style": {
                            "top": 140,
                            "left": 0,
                            "width": 1100,
                            "height": 680
                    },
                    "title": "Events per Device",
                    "type": "pie",
                    "legend": "right",
                    "others": false,
                    "src": "sql",
                    "sql": "select deviceName as 'Device Name',count(*) as 'Number of
Events' from attacks group by deviceName order by count(*) desc limit {topN}"
            }
```

Variable Object

Variable objects define internal named variables which can be used on the configuration files to replace text values. Named variables appear on text enclosed on { } (curly braces) characters. These {variables} are replaced by their value upon processing of the text by the report generator.
All the command line parameters passed to AutoReport are automatically defined as named variables.
Each variable object can have the following attributes:

object - *text* - Always “variable”.
Name - text - Defines the name of the variable to be created.
src - *text* - Source of the variable value. Can be ‘sql’ or ‘cmd’. If defined, its result will override the ‘value’ attribute.
sql - *text* - If ‘src’ is ‘sql’, this will define the SQL query used to set the ‘value’ attribute. Ignored otherwise.
cmd - *text* - If ‘src’ is ‘cmd’, this will define an external command to be executed, which output will set the ‘value’ attribute. Ignored otherwise.
value - *text* - Value of the variable. Its content is overridden by the ‘src’ entry.

The variables {version} and {pageNumber} are always defined and can be used by the report definition file.

Index Object

AutoReport implements a special object of the type ‘Index’, which is used to automatically generate a table of contents for the report.
This object will scan the entire file and generate a TOC. Currently there is no control over the size of the page list, so to use this object a report must have less than 100 pages.
The only attribute available to the index object is ‘style’ to allow positioning it within the page.

Pre Requisites

AutoReport is designed to run from Linux, however it can also be executed from Windows PowerShell or from inside Windows WSL (recommended).
When used on Linux, AutoReport cannot generate the PDF output, as this uses Headless Chrome to generate the PDF, and it is not available on the Linux distribution of AutoReport.
To generate PDFs use Windows WSL instead.
On Linux AutoReport requires php-cli, php-curl and php-sqlite3.

Installation

Just unpack the AutoReport package onto a folder. All files required to execute AutoReport are contained in the distribution package.

Command Line

Autoreport is executed with the following syntax:

```
./report.php -- -file="<report.json>" [-parm[=value]...]
```

or

```
php -f report.php -- -file="<report.json>" [-parm[=value]...]
```

-file - The report definition file to be used on the report generation, this is the only mandatory parameter for AutoReport.

-parm - Defines a boolean named variable which will be set to true.
-parm=value - Defines a named variable which will be set to the respective value.

All these named variables become available to the report generator the same way as the ones defined on the report definition files.

Here's one example of using AutoReport:

```
./report.php -- -file="json_files/MonthlyReport.json" -id="CID"
-month=”09" -monthText="September" -monthDays=30 -year="2021"
-longName="Customer Name"
```

In this example the MonthlyReport file is used to generate the report with all the other parameters being passed to it as named variables.