ProviewR
OPEN SOURCE PROCESS CONTROL

Release Notes V5.8

2021 04 16

Table of Contents

Upgrading to ProviewR V5.8.0

This document describes new functions i ProviewR V5.8.0, and how to upgrade a project from V5.7.0 to V5.8.0.

New functions

Graphs from scripts in rt_xtt

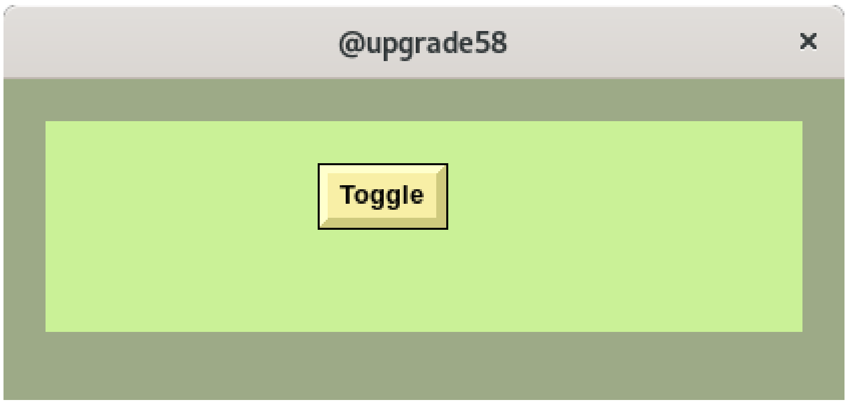
Instead of drawing a graph in the Ge editor, it is now possible to to write a ge script instead. The script can call Ge script functions to draw graphical elements in the graph, but also xtt script functions. This makes it possible to get information from the database and adapt the graph to the current state.

The script graph is opened as an ordinary graph but the graph name is replaced by the script name preceded by a '@', eg 'open graph @myscript'. Scripts are by default read from $pwr_exe or $pwrp_exe, and for any other location the path should be added, eg 'open graph @"$pwrp_login/myscript"'.

Example

A simple script that draws a rectangle and a push button

```
main()
  int id;
  float x1;
  float y1;
  float x2;
  float y2;
  float width;
  float height;

  SetDraw(0);
  SetBackgroundColor(eDrawType_Color66);

  # Draw rectangle
  x1 = 1;
  y1 = 1;
  width = 18;
  height = 5;
  id = CreateRectangle(x1, y1, width, height);
  SetObjectFillColor(id, eDrawType_Color74);
  SetObjectFill(id, 1);
  SetObjectBorder(id, 0);

  # Draw pushbutton
  x1 = 7.5;
  y1 = 2;
  x2 = 10.5;
  y2 = 3.5;
  id = CreateObject("pwr_buttontogglecenter", x1, y1, x2, y2);
  SetObjectAttribute(id, "Text", "Toggle");
  SetObjectAttribute(id, "ToggleDig.Attribute", "H1-
Dv1.ActualValue##Boolean");

  # Set graph size
  SetGraphAttribute("x1", 20.0);
  SetGraphAttribute("y1", 7.0);

  SetDraw(1);
endmain
```



New Ge script functions

BuildGraph()

Build the current graph, i e copy the graph to $pwrp_exe.

ClearAll()

Remote all objects in the graph.

CreateArc()

Create an Arc object.

CreateAxis()

Create an Axis object.

CreateAxisArc()

Create an AxisArc object.

CreateBar()

Create a bar object.

CreateBarArc()

Create a BarArc object.

CreateDsTrend()

Create a DsTrend object.

CreateDsTrendCurve()

Create a DsTrendCurve object.

CreateFastCurve()

Create a DsFastCurve object.

CreateImage()

Create an Image object.

CreateObject()

Create a subgraph object.

CreatePie()

Create a Pie object.

CreatePolyLine()

Create a PolyLine object.

CreateRectangle()

Create a rectangle object.

CreateRectRounded()

Create a rounded rectangle object.

CreateSevHist()

Create a SevHist curve object.

CreateText()

Create a text object.

CreateTrend()

Create a trend object.

CreateXYCurve()

Create an XYCurve object.

DashInsert()

Insert an object into a dashboard cell.

GetGraphName()

Get name of the current graph.

OpenGraph()

Open a graph.

PolyLineAdd()

Add a segment to a polyline.

RotateSelected()

Rotate selected objects.

SaveGraph()

Save the current graph.

SetBackgroundColor()

Set background color.

SetSelectTextBold()

Set bold text on selected objects.

SetSelectTextFont()

Set font on selected objects.

Other script functions

tstlog_open()

Open a test log file.

tstlog_close()

Close a test log file.

tstlog_log()

Print a message to a test log file.

tstlog_vlog()

Print a formated message to a test log file.

getmsg()

Get text for status variable.

tzset()

Set time zone.

sort()

Sort a string array in alphabetical order, or a float or integer array in numerical order.

sin()

Sine function.

cos()

Cosine function.

ODD()

Check if a value is odd.

EVEN()

Check is a value is even.

MAX()

Return the largest of two values.

MIN()

Return the smallest of two values.

Script arrays with dynamic size

A dynamic array is declared without size, eg

```
float darray[];
```

It’s created with zero size and can be incremented by a call to arraypush(). The function arraysize() will return the current size of the array.

arrayclear()

Clear the content of a dynamic array and set size to zero.

arraypush()

Add an element to the back of a dynamic array.

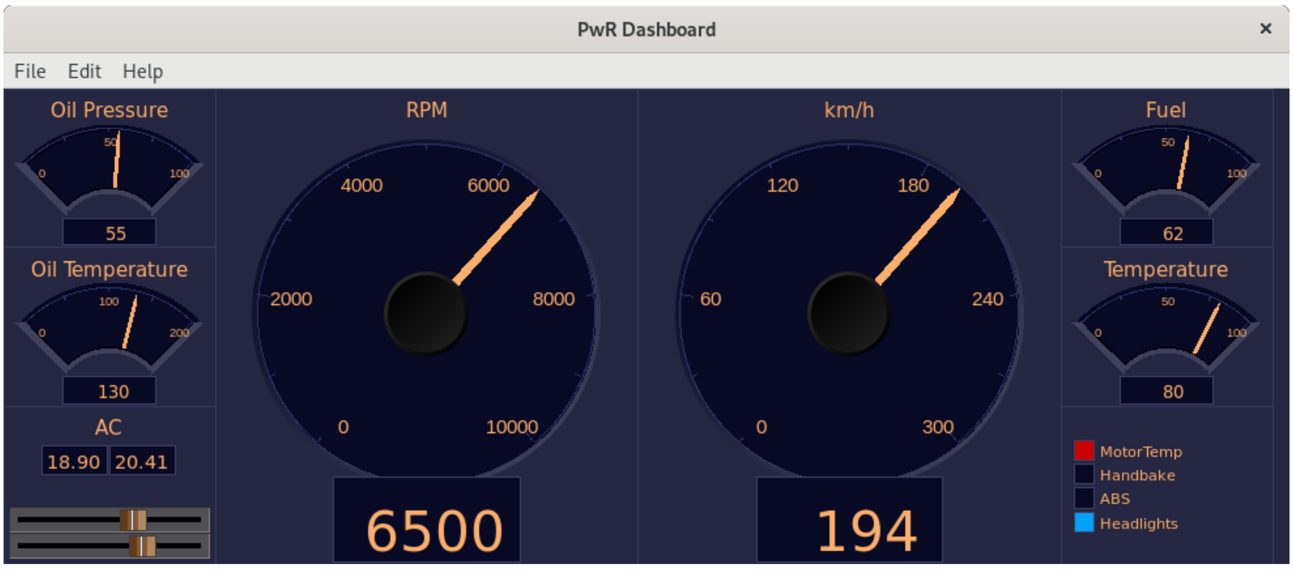
arraysize()

Returns the current size of a dynamic array.

Python3

The Python interface is ported to Python3. It's cannot be executed with Python2 any more.

Dashboard



The dashboard is a simplified graph with a very limited number of building blocks. The building blocks are called dashcells that can display values in the shape of indicators, value fields, bars, trends, gauges etc.

The dashboard can be created in the Ge editor, or in the operator environment.

The dashboard is divided in rows and columns where the cells are positioned. A cell initially has the height of one row and the width of one column, but can be expanded over several rows and columns.

The cell can contain several elements. A bar cell for example, can contain three elements. Each element is connected to an analog attribute that are displayed with a bar. A trend cell can contain two elements and show two curves.

There are three major cell types, analog, digital and object.

Analog dashboard cells

Bar

A bar cell displays an analog value with a bar and a value. I can contain three elements.

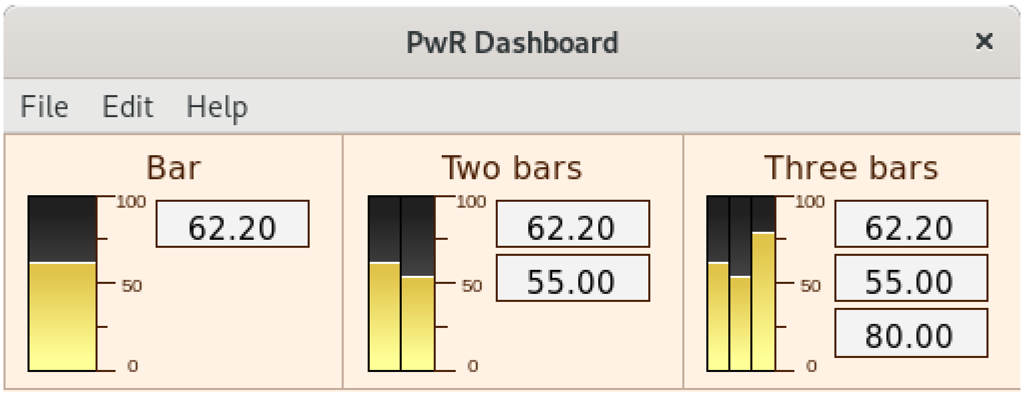
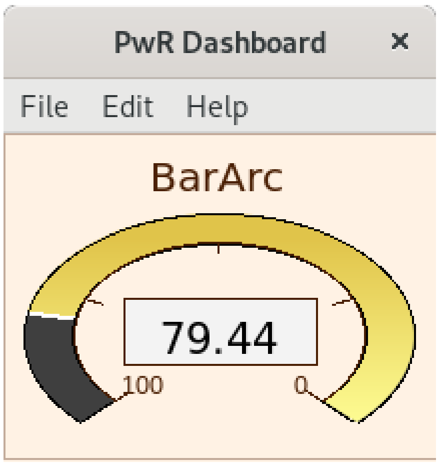
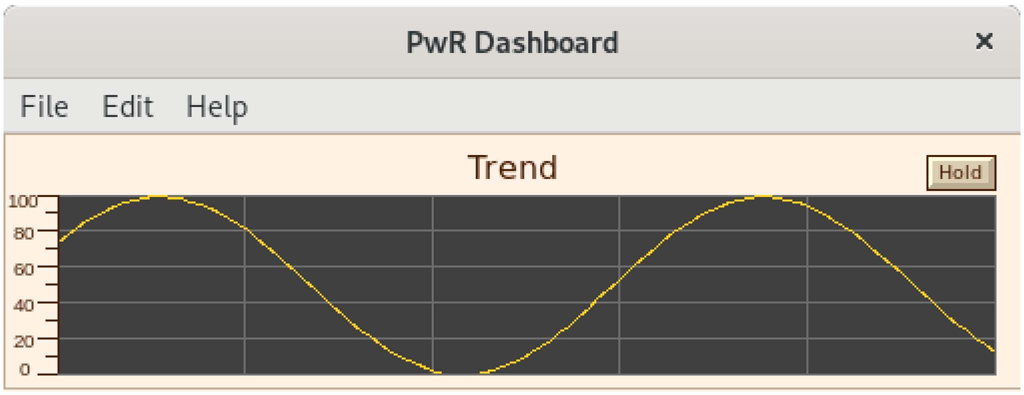


Fig Bar cells with one, two and three elements

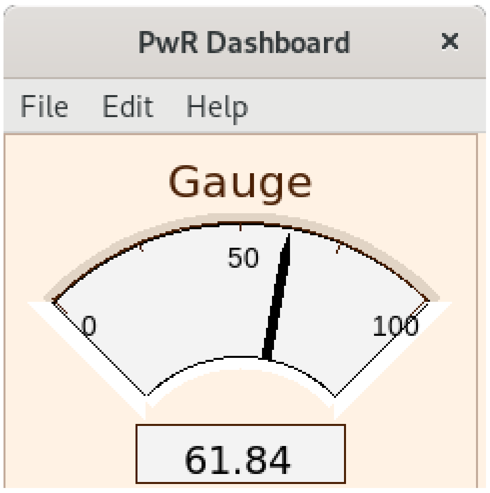
BarArc



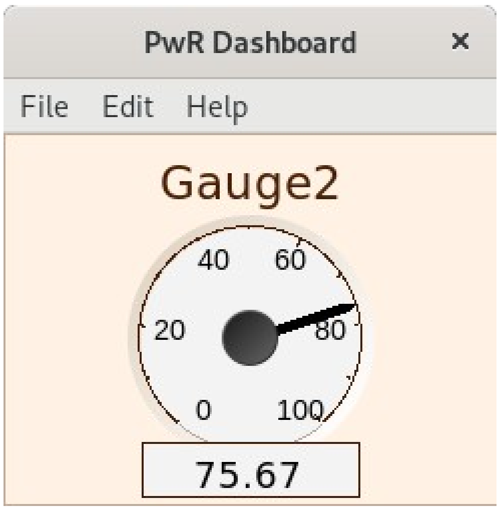
Trend



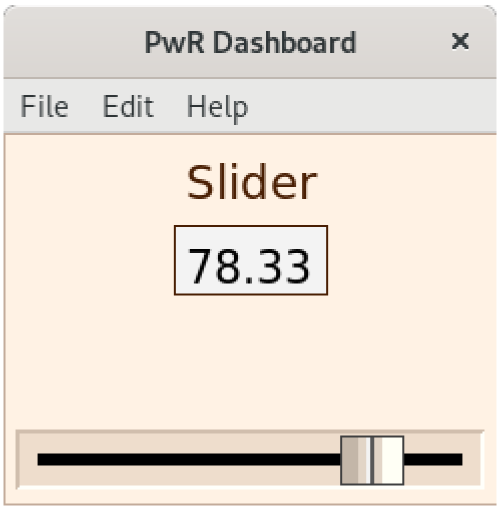
Gauge



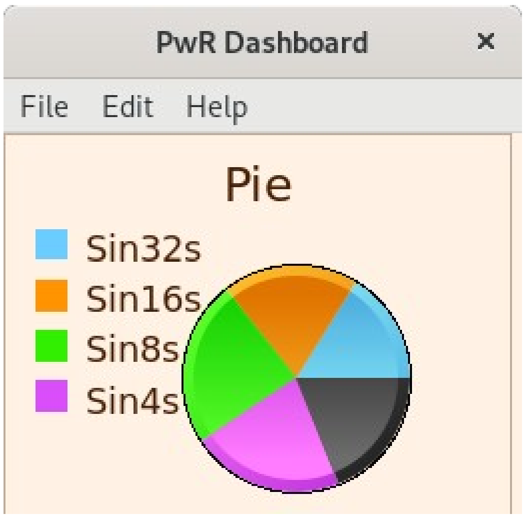
Gauge2



Slider

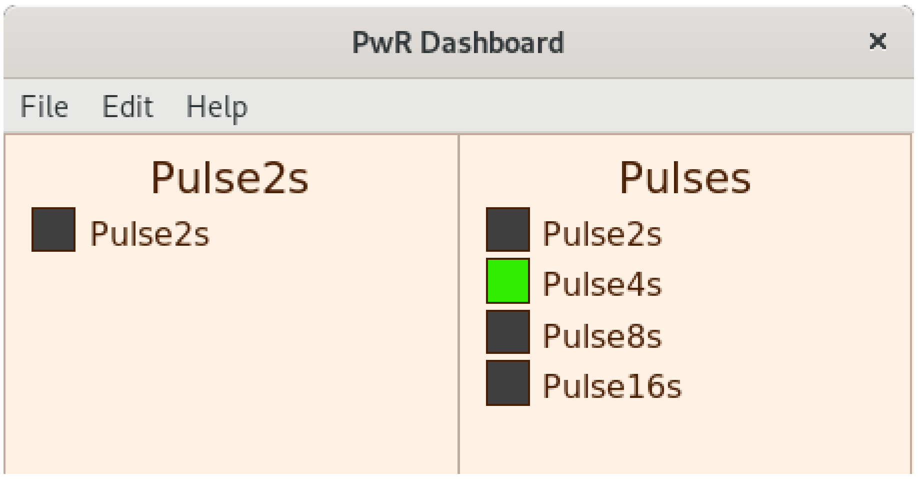


Pie

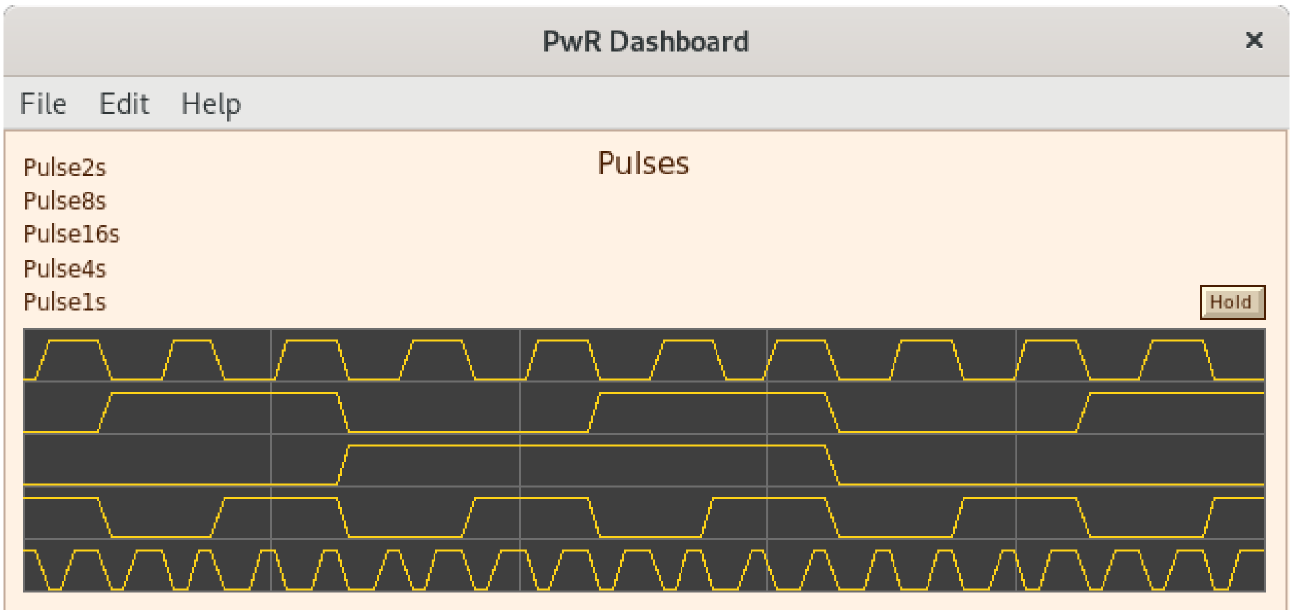


Digital dashboard cells

Indicator



DigitalTrend



Object dash cells

So far object cells are created for DsTrend, DsTrendCurve, SevHist and Sim_SignalGenerator. Two examples are displayed here.

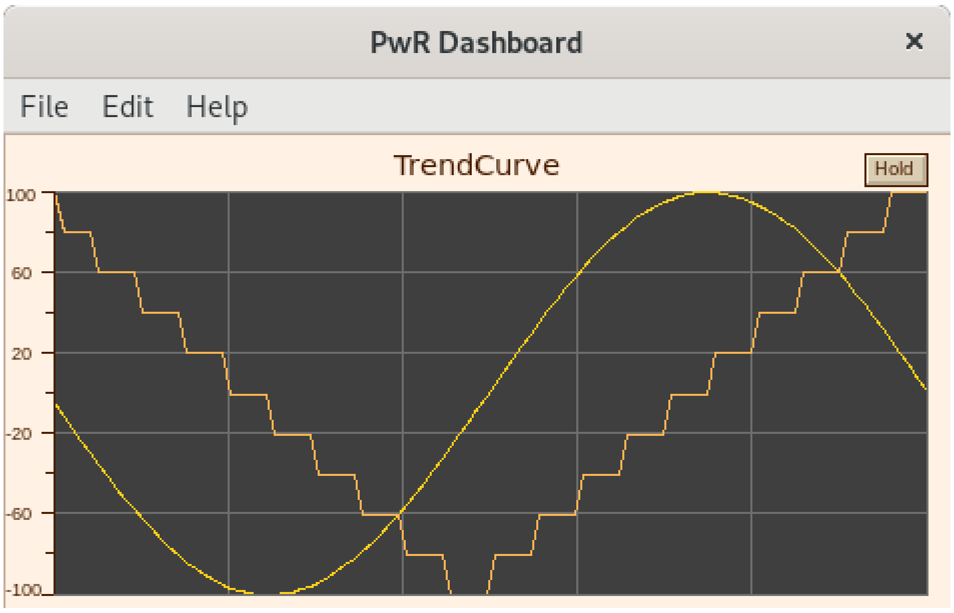
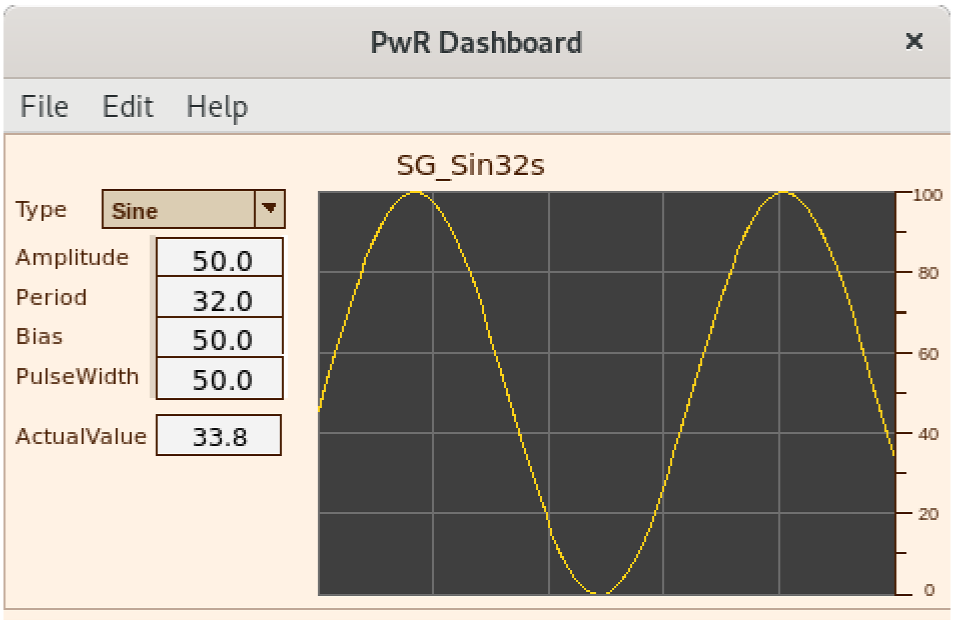


Fig Object cell for DsTrendCurve



Type: Sine
Amplitude: 50.0
Period: 32.0
Bias: 50.0
PulseWidth: 50.0
ActualValue: 33.8

Fig Object cell for Sim_SignalGenerator

Create a dashbord in the Ge editor

A dashboard is created in the editor by setting Dashboard in GraphAttributes to 1. Dashcells can now be created from Other/DashCell in the palette. Select a color theme from File/Colortheme/Select as the cells are draw in color theme colors.

Open the Object Attributes for the cell and select a type in Dash.Type. Connect each element to an attribute in the database and fill in other settings in the Object Attributes.

The size of the dashboard are set in DashRows and DashColumns in GraphAttributes.

When the dashboard is saved, a pwd-file is created on $pwrp_pop, and when it’s built, it’s copied to $pwrp_exe from where it can be opened by rt_xtt.

In rt_xtt the dashboard is opened from Functions/Dashboard/Open or with the command ‘open dashboard ‘name’’.

Create a dashboard in rt_xtt

By selecting an attribute in the navigator, and activating Functions/Dashboard/Insert (Ctrl+D) in the menu, the attribute is added to a dashboard. By default it’s added to the ‘PwR Dashboard’, but if there is a dashboard that is in edit mode, it will be inserted in this dashboard instead. It’s possible to add new attributes to the dashboard until all rows and columns are filled.

For an analog attribute a bar cell is created. This can be changed by entering edit mode in the dashboard from File/Edit (Ctrl+E) in the dashboard menu. Open the cell attributes (double click) and select another type in Dash.Type. Leave the edit mode with Ctrl+E.

When entering edit mode, the Ge editor is opened in restricted mode. Some functionality in edit mode is

- Add. Create an empty cell.
- Delete. Delete cell.
- Copy. Copy a cell.
- Paste. Paste previously copied cells.

- Connect. Connect selected attribute in the navigator to cell.
- Merge. Merge cells. The selected cells will be merged into the one that was first selected.
- From GraphAttributes, the scan time and dashboard size can be modified.

Create a user object dashcell

A dashboard cell for a user object (or any other object) can be created with a Ge script. The script is named dash_'classname'.ge_com and placed on $pwrp_exe. The script should contain the function dash_objectgraph() with the arguments below. All arguments are configurable from the cell attributes editor, except the cell identity.

g	integer	DashCell identity.
elements	integer	Number of elements. 1 for object dashcells.
title	string	Title. By default the object name
time_range	float	Time range than can be used if the cell contains a trend.
direction	int	Direction for bars, trends etc
object	string	Instance object for the cell.
text	string	Configured text.
min_value	float	Min value for axis, trends, bars, sliders etc
max_value	float	Max value for axis, trends, bars, sliders etc

The example below is a script for an Iv object, $pwrp_exe/dash_iv.ge_com.

```
function int dash_objectgraph(int g, int elements, string title,\
 float time_range, int direction, string object, string text, \
 float min_value, float max_value)
  float x1;
  float y1;
  float x2;
  float y2;
  float twidth;
  int id;
  float gx1;
  float gx2;
  float gy1;
  float gy2;
  float gw;
  float gh;
  string attr;

  MeasureObject(g, gx1, gy1, gx2, gy2);
  gw = gx2 - gx1;
  gh = gy2 - gy1;

# Draw header text
  GetTextExtent(title, 3, eFont_LucidaSans, 0, twidth);
  x2 = (gx1 + gx2)/2 - twidth/2;
  y2 = gy1 + 1;
  id = CreateText(title, x2, y2, 3, eFont_LucidaSans, 0, \
    eDrawType_CustomColor5);
  DashInsertObject(g, id);

# Draw value field
  x1 = gx1 + 1.0;
  y1 = gy1 + 1.5;
  x2 = gx2 - 1.0;
  y2 = y1 + (x2 - x1) / 2.5;
  id = CreateObject("pwrct_valuelargecenter", x1, y1, x2, y2);
  attr = object + ".ActualValue##Int32";
  SetObjectAttribute(id, "Value.Attribute", attr);
  SetObjectAttribute(id, "Value.Format", "%d");
  DashInsertObject(g, id);

endfunction
```

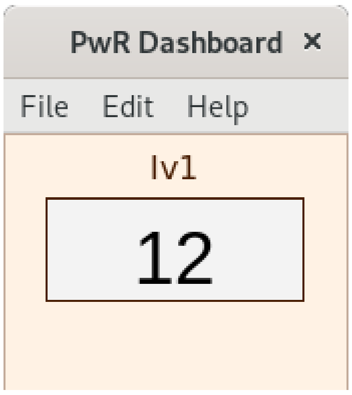


Fig The user object dashcell

Start runtime as another node

The -n option to rt_ini makes it possible to start a runtime environment that is configured for another node on the current node. The Qcom bus id though has to be the same.

```
> rt_ini -n pwrtest01a
```

will start the node pwrtest01a on the current node.

It is also possible to redirect qcom connections to other nodes with the alias statement in pwrp_alias.dat.

```
alias pwrtest01b steel-arrow 10.255.0.98
```

will redirect the connection to pwrtest01b to the node steel-arrow with ip address 10.255.0.98.

Display a DsTrend object in a graph

New Ge component to view a DsTrend curve in a Ge graph. The component is found under Analog/DsTrend in the palette.

The object can is connected to one or two DsTrend objects in the database and can display two curves.

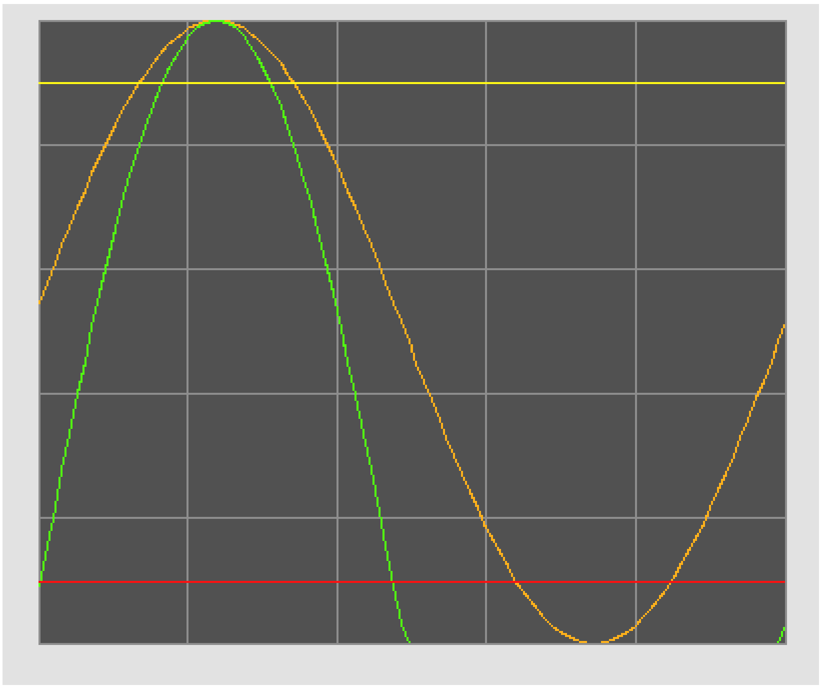


Fig DsTrend curve displayed in a Ge graph

Display a DsTrendCurve object in a graph

New Ge component to view a DsTrendCurve curve in a Ge graph. The component is found under Analog/DsTrendCurve in the palette.

The object can is connected to a DsTrendCurve object in the database and can display two curves.

Display a history curve in a graph

New Ge component to view a SevHist curve in a Ge graph. The component is found under Analog/SevHist in the palette.

The object can is connected to one or two SevHist object in the database and can display two curves.

In case the curve should be viewed from a sev server, where the SevHist object might not be present, the attributes name and history server node can be supplied instead.

The curve can be displayed as a snapshot or cyclically updated.

New Types and Classes

PulseTrainM

Plc function object with pulse train outputs with different cycle times: 128 s, 64 s, 32 s, 16 s, 8 s, 4s 2 s, 1 s, 500 ms, 250 ms, 125 ms, 62.5 ms, 31.2 ms and 15.6 ms. In comparison with PulseTrain the pulse times in PulseTrainM are a multiple of each other and synchronized.

DtoI

Plc function object to convert a digital signal to integer.

LightATv

Absolute time value that is not IO copied and only contains the basic methods. When used in a plc program considerations about thread safety and execution order has to be taken.

LightDTv

Delta time value that is not IO copied and only contains the basic methods. When used in a plc program considerations about thread safety and execution order has to be taken.

Upgrade procedure

The upgrading has to be done from any V5.7. If the project has a lower version, the upgrade has to be performed stepwise following the schema

V2.1 -> V2.7b -> V3.3 -> V3.4b -> V4.0.0 -> V4.1.3 ->V4.2.0->V4.5.0->V4.6.0->V4.7.0->V4.8.6->(V5.0.0)->V5.1.0->V5.2.0->V5.3->V5.4->V5.5->V5.6->V5.7->V5.8

Enter the administrator and change the version of the project to V5.8.0. Save and close the administrator.

Enter the directory volume and save.

I you have any class volumes, enter the class editor and build the volume.

Enter the configurator for each root volume and activate 'Function/Update Classes' and build.