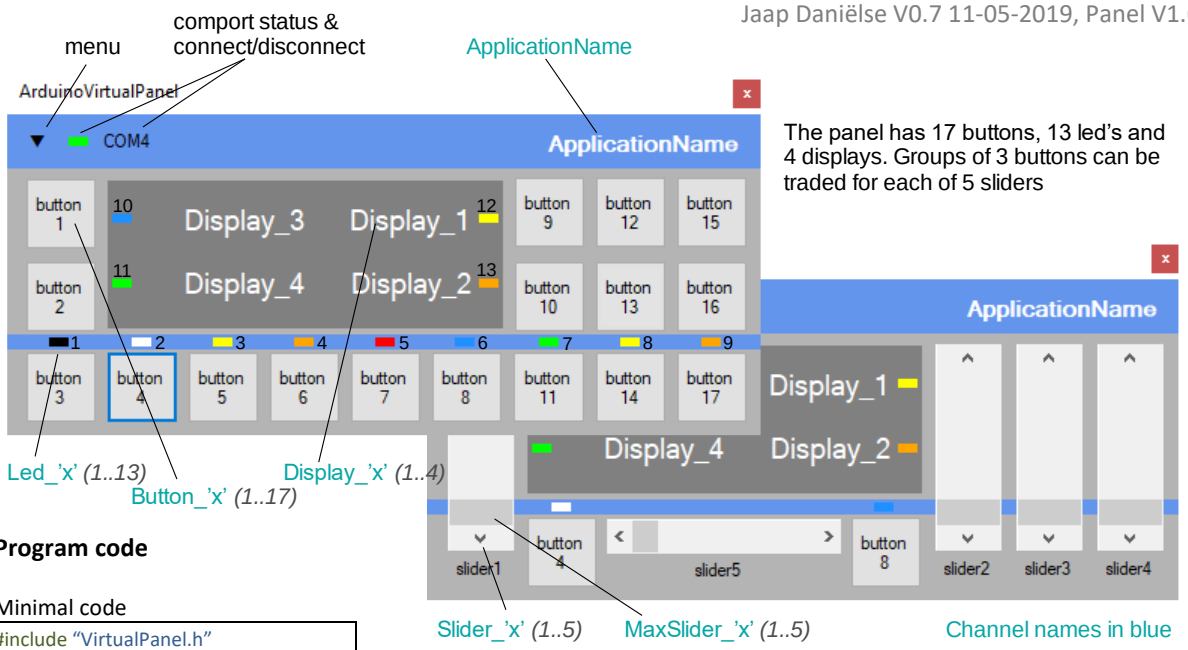
# Virtual Panel

**Quick Reference**

# Arduino Experiment Control Panel

Jaap Daniëlse V0.7 11-05-2019, Panel V1.0




The panel has 17 buttons, 13 led's and 4 displays. Groups of 3 buttons can be traded for each of 5 sliders

## Program code

Minimal code

```
#include "VirtualPanel.h"

void setup ()
{   Panel.Init();        }

void loop ()
{   Panel.Receive();   }

// callbackroutine
void PanelCalback (int event, int type)
{ switch (event)
  { case channel:
      ...
      break;
}}
```

```
Panel.Send (channel, variable*);
Panel.Sendf (channel, formatted string**, format variables ...);
```

* types depending on channel.
** see printf formatting

## Main Panel channels/events

Send() or Sendf() functions.
Receive via PanelCallback() event

ApplicationName Send

| | |
|---|---|
| string | application name text |
| color | text color |

PanelConnected Receive

| | |
|---|---|
| void | On connect. |

Reset Send

| | |
|---|---|
| void | resets panel |

DynamicDisplay Send

| | |
|---|---|
| bool | activate/ deactivate |
| int16 | delay mS (100-2000)* |

Receive

| | |
|---|---|
| void | on delay freq. |

*Default 250mS

UnixTime Send

| | |
|---|---|
| void | request |

Receive

| | |
|---|---|
| uint32 | (local) unix time code |

Beep Send

| | |
|---|---|
| void | 500 Hz 400 mS |
| Int16 | Freq. Hz 400 mS |
| uint32* | Frequency, Duration |

* use _Sound helper function:
long _Sound( int freq, int dur)

Button_'x' (1..17) Send

| | |
|---|---|
| any | button text |
| color[1] | button color |
| size[1] | text size |

Receive

| | |
|---|---|
| Void | on button click |

Slider_'x' (1..5) Send

| | |
|---|---|
| bool | visible/invisible |
| string | slider label text |
| int16 | set (initial) value |

Receive

| | |
|---|---|
| int16 | value on slider action |

MaxSlider_'x' (1..5) Send

| | |
|---|---|
| int16 | maximum value* |

* positive only

[1] See: *Special strings*

Led_'x' (1..13) Send

| | |
|---|---|
| bool | visible/invisible |
| color[1] | led color |

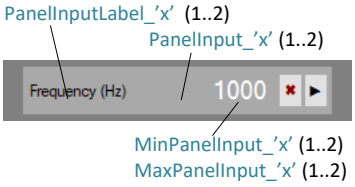
Display_'x' (1..4) Send

| | |
|---|---|
| any | display as text |
| color[1] | display color |
| size[1] | text size |

Receive

| | |
|---|---|
| void | on double click |

*Panel Input*




PanelInput_'x' (1..2) Send

| | |
|---|---|
| bool | static/volatile |
| any | value |

Receive

| | |
|---|---|
| any * | value |

*Type same as sent type

MinPanelInput_'x' (1..2) Send

| | |
|---|---|
| num* | value |

MaxPanelInput_'x' (1..2) Send

| | |
|---|---|
| num* | value |

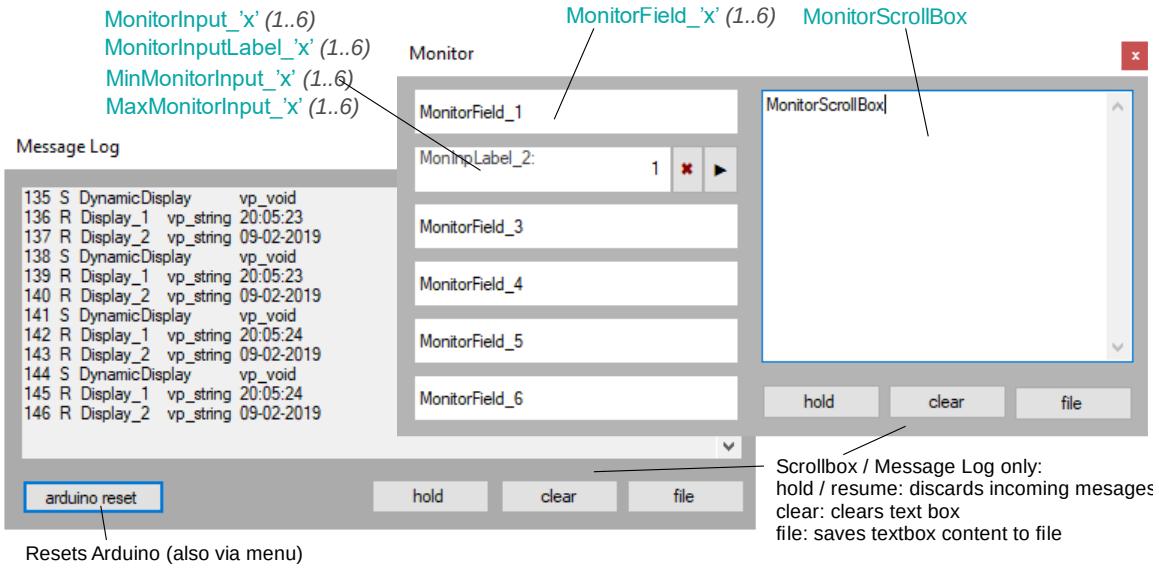
*Int16 int32, float32
When string min/max length.

PanelInputLabel_'x' (1..2) Send

| | |
|---|---|
| any | Input label text |

**Message Log Panel**
Records incoming (R) and outgoing (S) messages.

**Monitor panel**
Provides a log panel and additional displays and inputs




**Message Log**

Format:
146 R Display_2 vp_string Test
{MessageNumber}{Send/Receive}
{channel}{VarType}{Value}

**Monitor channels / events**

Monitor

| | |
|---|---|
| bool | visible/invisible |

**Special strings**

**Color strings**
For: ApplicationName, Display, Led, Button.

| | |
|---|---|
| $DELETE* | |
| $OFF** | |
| $BLACK | |
| $GRAY | |
| $PURPLE | |
| $PINK | |
| $BLUE | |
| $GREEN | |
| $YELLOW | |
| $ORANGE | |
| $RED | |
| $WHITE | |

* draw only
** Led only

**Graph Type strings**
Set graph type. Rolling values are added right and move to left. Static waits until all values have been sent then displays.

| | |
|---|---|
| $ROLING* | Set rolling graph |
| $STATIC | Set static graph |

* default

MonitorField_'x' (1..6)

| | |
|---|---|
| any | display as text |

MonitorLogPanel

| | |
|---|---|
| Any | display as text |

MonitorInput_'x' (1..6) Send

| | |
|---|---|
| bool | static/volatile |
| any* | value |

Receive

| | |
|---|---|
| any* | value |

*Type same as sent type

**Pen size strings Draw**
GraphPen, GraphValue

| | |
|---|---|
| $1PX* | 1 pixel |
| $2PX | 2 pixels |
| $3PX | 3 pixels |
| $4PX | 4 pixels |

* default

**Text attributes/size strings**

| | |
|---|---|
| $SMALL | fontsize small |
| $NORMAL* | fontsize normal |
| $BIG | fontsize big |
| $BOLD | bold text |

*Default. Resets bold and big

**Unicode characters**
Using Send() or Sendf() to send a string, Unicode characters can be used. Simply copy and paste into the string.

**Helper function Sound**
long _Sound( int freq, int dur)
Combines two int16_t (frequency Hz, duration mS) into one uint32_t.

MonitorInputLabel_'x' (1..2) Send

| | |
|---|---|
| any | Input label text |

MinMonitorInput_'x' (1..6) Send
MaxMonitorInput_'x' (1..6) Send

| | |
|---|---|
| num* | value |

*Int16 int32, float32
When string min/max length.

**Helper functions Draw**
uint16_t _Point(byte x, byte y)
combines 2 bytes into uint16_t (x,y) for a point.
Wen sent to GraphDrawLine consecutive points are connected in a line.

uint32_t _Line(byte Fx, Fy, Tx, Ty)
Combines four bytes into uint32_t (x from, y from, x to, y to)

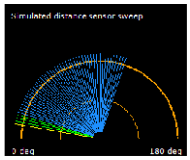
uint16_t _VPoint(byte x, byte y)
uint32_t _VLine(byte Fx, Fy, Tx, Ty)
Same as _Point and _Line but transform y values from value (0-255) to coordinate (0-220).

**Helper function Float string**
char * _FString(floatNumber, length, decimals);

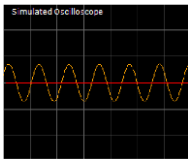
**Panel Delay function**
bool Panel.Delay(int16_t milliseconds, bool receive)
Allows to check for incoming messages during delay. If receive is true. Panel receive is called. If an incoming message was detected true is returned.

## Graph Panel

Supports simple graphical display functions (rolling graph, static graph, free draw) including 4 extra buttons and 5 labels with color bars to associate with a graph.

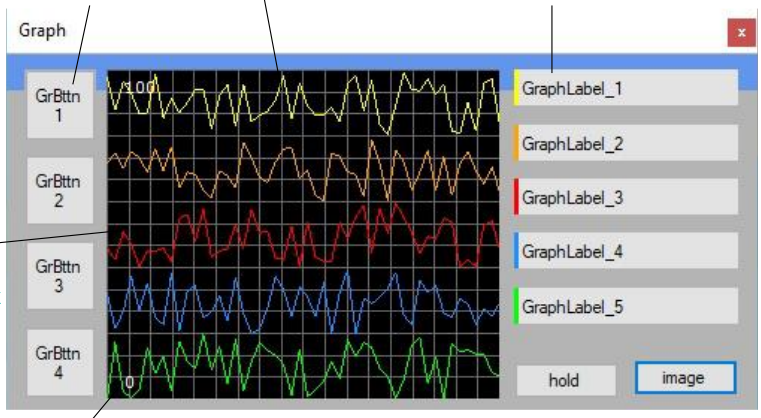


Graph Draw



Static Graph

GraphGrid
GraphDrawLine
GraphDrawPixel
GraphText
GraphClick
GraphRightClick
GraphDoubleClick




RolingGraph

hold / resume: discards incoming mesages
image: saves graph pane content to file

### Graph channels/events

Graph Send

| | |
|---|---|
| bool | visible/invisible |
| string | $CLEAR |

GraphGrid Send

| | |
|---|---|
| int16 | vert. gridcount |

GraphDrawLine Send

| | |
|---|---|
| void | Line start |
| uint16[2] | point 2 x byte (x,y) |
| uint32[2] | Line 4 x byte (Fx,Fy,Tx,Ty) |
| color[1] | line color |
| width[1] | line width string |

GraphDrawPixel Send

| | |
|---|---|
| color[1] | pixel color |
| uint16[2] | point 2 x byte (x,y) |

GraphCaption_'x' (1..2) Send

| | |
|---|---|
| any | Caption text |

GraphText Send

| | |
|---|---|
| color[1] | text color |
| uint16[2] | point 2 x byte (x,y) |
| string | text |

GraphValue_'x' (1..5) Send

| | |
|---|---|
| byte | point 2 x byte (x,y) |
| color[1] | Graph color |
| width[1] | line width string |
| type[1] | rolling/static |
| $CLEAR | clear sent values |

GraphValueCount_'x' (1..5) Send

| | |
|---|---|
| int16 | hor. value count |

[1]See: *Special strings*

[2] Helper functions:
uint16_t _Point(byte x, byte y)
Uint32_t _Line(byte Fx, Fy, Tx, Ty)

GraphLabel_'x' (1..5) Send

| | |
|---|---|
| bool | visible/invisible |
| any | label text |
| color[1] | color bar color* |

* $OFF (color bar invisible)

GraphButton_'x' (1..4) Send

| | |
|---|---|
| any | button text |
| color[1] | button color |
| size[1] | text size |

Receive

| | |
|---|---|
| void | on button click |

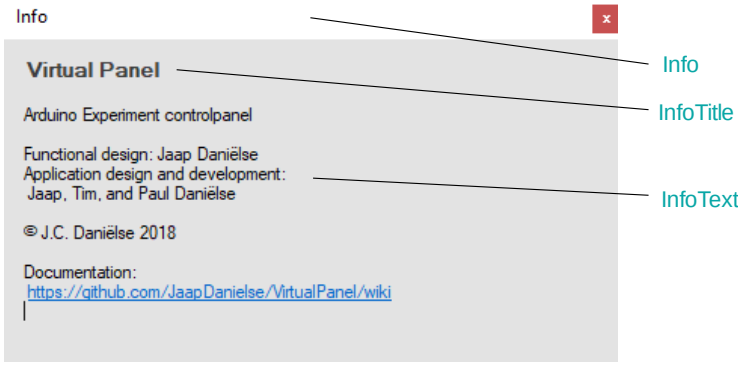
GraphClick Receive
GraphRightClick Receive
GraphDoubleClick* Receive

| | |
|---|---|
| uint16** | point 2 x byte (x,y) |

* occurs together with GraphClick
**uint 2 x byte (X,Y)
(same as DrawPoint and DrawLine

## Info Panel

Application dependent help panel.




### Info channels/ events

Info Send

| | |
|---|---|
| bool | visible/invisible |
| string | $CLEAR |

>bool false/true (window visible)

InfoTitle Send

| | |
|---|---|
| any* | title text |

*clears InfoText

InfoText Send

| | |
|---|---|
| string* | Info text |
| $CLEAR | Clears info text |

* max 60 char per send.
Can be repeated for larger text

## Miscellaneous

**Sendf() / Printf formatting**

Limited list.

*%[flags][width][length]specifier*

*specifiers*

| | |
|---|---|
| %d | signed decimal |
| %ld | unsigned int32 |
| %u | unsigned decimal |
| %o | unsigned octal |
| %x | unsigned hex |
| %c | character |
| %s | string |

*flags*

| | |
|---|---|
| - | left justify |
| + | force sign |
| 0 | pad zero's |

*Examples:*

```
Panel.Sendf (Display_1, "Test %d",
10) // output: Test 10
Panel.Sendf(Display_1, "Test
%03d", 10) // output: Test 010
Panel.Sendf(Display_1, "Test
%+d", 10) // output: Test +10
```

Float not supported on AVR (Uno, Nano, Mega … ) Then use:

```
char outstr[10];
dtostrf(floatNumber,  length,
decimals, outstr);
```

and Panel.sendf using "%s"

or the _FString() helper function.

```
char * _FString(floatNumber,
length, decimals); again with
Panel.sendf using "%s"
```

Example:

```
Panel.Sendf(Display_1, "Value %s"
, _FString(FloatValue, 5, 2);
```

Prints FloatValue using 5 chars, 3 of which are a '.' and 2 decimals.

**F() Macro**

In both Send() and Sendf() the F() macro for strings is allowed. This will force the string to be placed in program memory.

*Example:*

```
Panel.Sendf
 (Display_1, F("Value %d"), 10);
```

**Panel Variables**

Retrieve incoming event data.

| | |
|---|---|
| Panel.vpr_void | void |
| Panel.vpr_bool | bool |
| Panel.vpr_string | char[ ]*** |
| Panel.vpr_byte | byte |
| Panel.vpr_int* | int16_t |
| Panel.vpr_uint** | unint16_t |
| Panel.vpr_long | int32_t |
| Panel.vpr_ulong | unit32_t |
| Panel.vpr_float | float32_t |

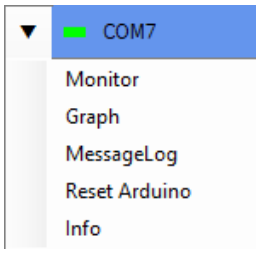
* Slider_1(-5)

** GraphClick, GraphRightClick, GraphDoubleClick

*** Max 35 char.

**Menu**

Drop down from main panel.




| | |
|---|---|
| Monitor | Open/close monitor window* |
| Graph | Open/close Graph window* |
| Message Log | Open/close Msg.Log window |
| Reset Arduino | Reset Arduino (not all processor types) |
| Info | Open/close Info window * |

* Can also be opened using channel.

**Button Special Symbol strings**

| | |
|---|---|
| $ONOFF | ○● |
| $LEFT | ◀ |
| $RIGHT | ▶ |
| $UP | ▲ |
| $DOWN | ▼ |
| $DOT | ● |
| $LTURN | ↺ |
| $RTURN | ↻ |
| $RUN | ▶ |
| $PAUSE | ⏸ |
| $STOP | ■ |
| $SET | ✱ |

## Code snippets

**Button**

```
Panel.Send(Button_1, "on\noff");  //init
…
case Button_1:  // Button_1 case in event switch
   // Button_1 code
 break;
```

**Slider**

```
Panel.Send(Slider_1, "level");  //set label
Panel.Send(MaxSlider_1, 255);  //set max value
Panel.Send(Slider_1, 127);  //set (initial) value
…
case Slider_1: // Slider_1 case in event switch
  MySliderValue = Panel.vpr_int;    // copy value
  // Slider_1 code
 break;
```

**Input**

```
case Display_1: // Display_1 double clicked
  Panel.Send(PanelInputLabel_1, "Inp. value:"); //set labe
  Panel.Send(MinPanelInput_1, 0);  //set min. value
  Panel.Send(MaxPanelInput_1, 100);  //set max. value
  Panel.Send(PanelInput_1, 42);  //set current value
break;

case PanelInput_1: //PanelInput_1 case in event switch
  MyInputValue = Panel.vpr_int;    // copy value
  // PanelInput_1 code
break;
```

**Graph**

```
Panel.Send(GraphGrid, 10);  //set grid nbr vert sections
Panel.Send(GraphValueCount_1, 100);  //set nbr of value
Panel.Send(GraphValue_1, "$RED");  //set color red
…
Panel.Send(GraphValue_1,  Value);  //send value
// graph default "rolling"
```