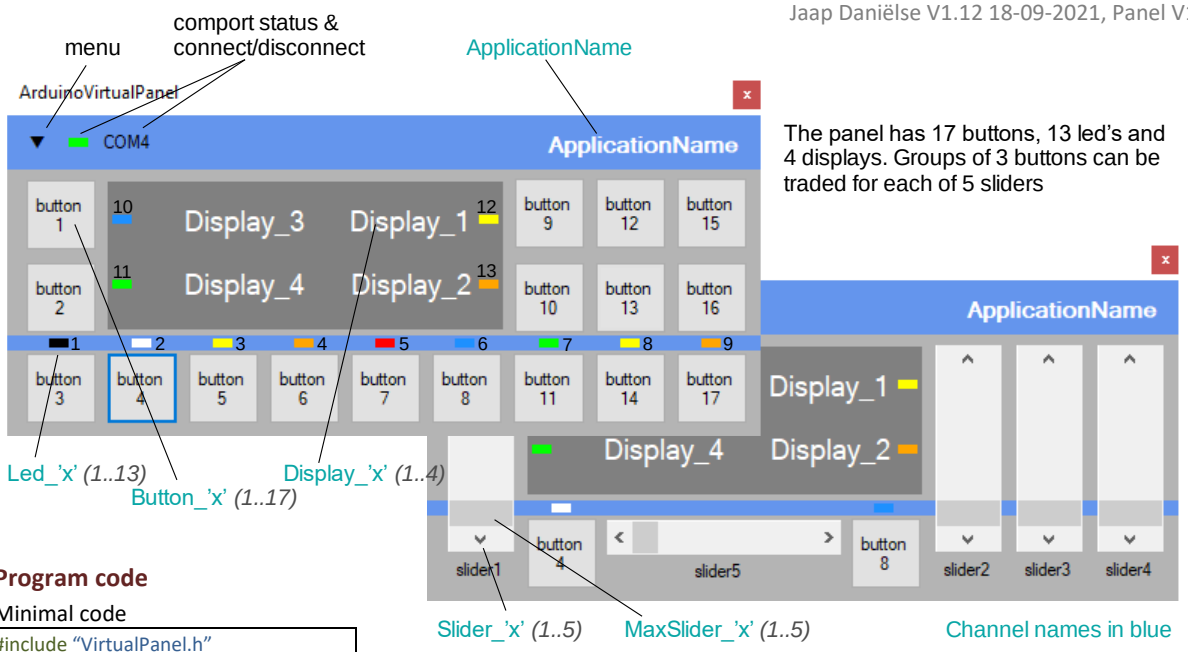
# Virtual Panel

**Quick Reference**

# Arduino Experiment Control Panel

Jaap Daniëlse V1.12 18-09-2021, Panel V1.3.x




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.begin();    }

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

// callbackroutine
void PanelCalback (vp_channel event)
{ 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

| | |
|---|---|
| char* | appl. name text |
| color[1] | text color |

### PanelConnected — receive

| | |
|---|---|
| void | on connect |

### Reset — send

| | |
|---|---|
| void | reset 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 | def: (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 text color |
| size[1] | text size |
| repeat[1] | button (no)repeat |

receive

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

### Slider_'x' (1..5) — send

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

receive

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

### MaxSlider_'x' (1..5) — send

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

* positive only

### PanelColor — send

| | |
|---|---|
| color[1] | panel color |

### Led_'x' (1..13) — send

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

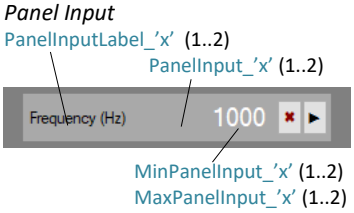
### Display_'x' (1..4) — send

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

Receive

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

### *Panel Input*




### PanelInput_'x' (1..2) — send

| | |
|---|---|
| bool | true:static/false:volatile |
| any | set value |

receive

| | |
|---|---|
| any* | value |
| void | discard |

*Type same as sent type

### MinPanelInput_'x' (1..2) — send

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

### MaxPanelInput_'x' (1..2) — send

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

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

### PanelInputLabel_'x' (1..2) — send

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

[1] See: *Special strings*

OpenFile_'x' (1..4) send

| char* | file path string* |
|---|---|

Receive

| int32 | line count if open |
|---|---|
| void | if file not open |

*- dir. path only, sets dialog path.
- filename or wildcard + ext. opens or creates file via dialog.
- ext. sets dialog file filter.
- /f forces open/create w/o dialog if specified dir. / dialog dir. valid.

FileOpenDialogTitle_'x' (1..4) send

| char* | set dialog title |
|---|---|

ReadLineFile_'x' (1..4) send

| void | read next line |
|---|---|
| int32 | set next read line nr. |

Receive

| char* | line read * |
|---|---|
| void | end of file |

* Truncates to 60 chars.

WriteLineFile_'x' (1..4) send

| char* | write next line |
|---|---|
| int32 | set next write line nr. |

ClearFile_'x' (1..4) send

| void | clear open file |
|---|---|

DeleteFile_'x' (1..4) send

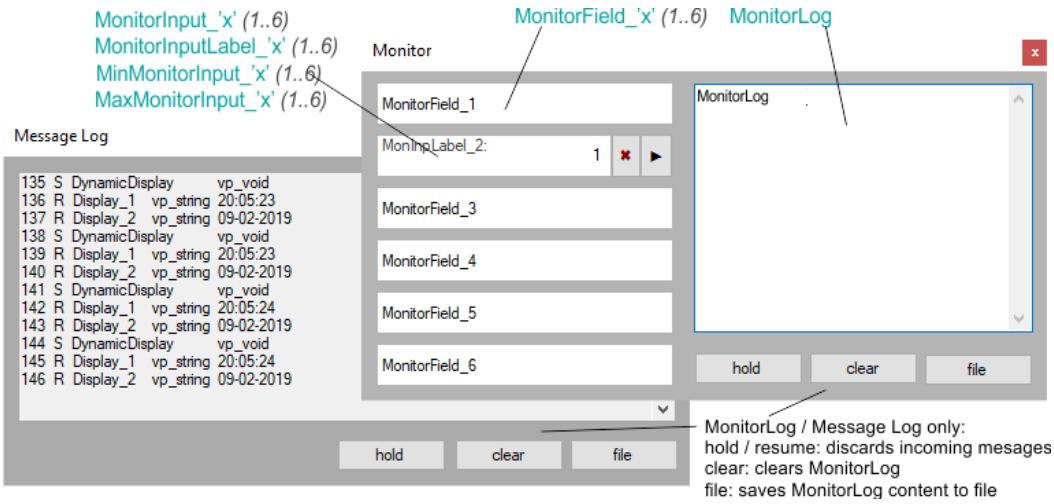
| void | delete open file |
|---|---|

## Message Log Panel

Records panel incoming (R) and panel 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 send

| bool | win. visible/invisible |
|---|---|

MonitorField_'x' (1..6) send

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

MonitorInput_'x' (1..6) send

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

receive

| any* | value |
|---|---|
| void | discard |

*Type same as sent type

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.

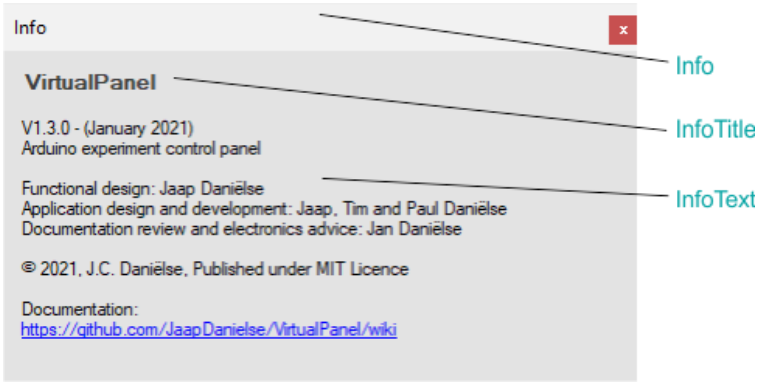
MonitorLog send

| any | display as text |
|---|---|
| $CLEAR[1] | clear Log |

[1]*See special strings*

## Info Panel

Application dependent help panel.




## Info channels/ events

Info send

| bool | win. visible/invisible |
|---|---|
| $CLEAR[1] | Resets to default. |

InfoTitle send

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

*Also clears InfoText

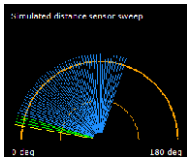
InfoText send

| char* | Info text* |
|---|---|

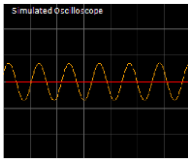
* max 60 char per send.
Can be repeated for larger text

## Graph Panel

Graphic display functions (rolling/static graph, draw) panel, including additional labels and buttons.



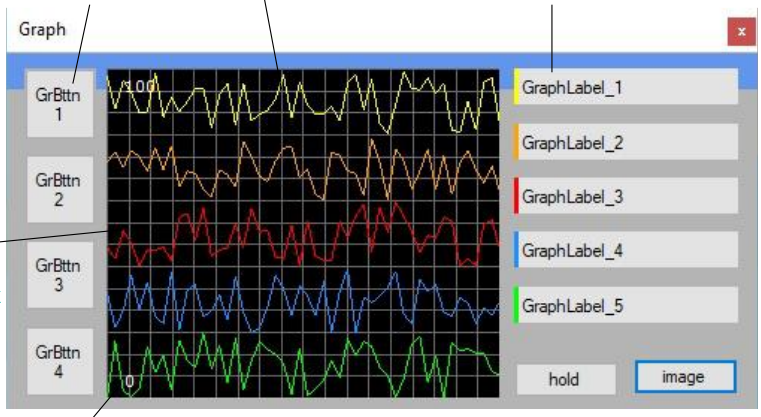
Graph Draw



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 | win. visible/invisible |
| $CLEAR[1] | clear graph* |

*Not values

GraphGrid send

| | |
|---|---|
| int16 | vert. grid count |

GraphDrawLine send

| | |
|---|---|
| void | line start |
| uint16[2] | line point (x,y) |
| uint32[2] | line segment (x,y,x',y') |
| color[1] | line color |
| width[1] | line width |

GraphDrawPixel send

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

GraphDrawCircle send

| | |
|---|---|
| params[2] | circle parameters |
| color[1] | circle color |
| width[1] | circle width |

GraphCaption_'x' (1..2) send

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

Graph Panel 255(x) X 220(y)
Actual 263(x) for GraphValue

GraphText send

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

GraphValue_'x' (1..5) send

| | |
|---|---|
| byte | graph value (0-255) |
| color[1] | graph color |
| width[1] | line width string |
| type[1] | rolling/static |
| $CLEAR[1] | clear sent values |

GraphValueCount_'x' (1..5) send

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

*Default value 50.

[1]See: *Special strings*
[2] See: *Helper functions Draw _Point, _Line _Circle*

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** | click position |

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

GraphLabel_'x' (1..5) send

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

* $OFF (color bar invisible)

GraphInput_'x' (1..5) send

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

receive

| | |
|---|---|
| any* | value |
| void | discard |

*Type same as sent type

GraphInputLabel_'x' (1..5) send

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

MinGraphInput_'x' (1..5) send
MaxGraphInput_'x' (1..5) send

| | |
|---|---|
| num* | min/max value |

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

## Data types and Panel Variables

**Data types**

| | |
|---|---|
| vp_type::vp_void | void |
| vp_type::vp_boolean | bool |
| vp_type::vp_string | char* |
| vp_type::vp_byte | byte |
| vp_type::vp_int | int16 |
| vp_type::vp_uint | uint16 |
| vp_type::vp_long | int32 |
| vp_type::vp_ulong | uint32 |
| vp_type::vp_float | float |

Event data type received in:

| | |
|---|---|
| Panel.vpr_type | vpr_type |

**Panel variables**
(Event data received)

| | |
|---|---|
| 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 |

vpr_void DynamicDisplay *(timer)*,
Button, GraphButton *(click)*,
ReadLineFile *(eof)*,
Display *(double click)*, PanelInput,
MonitorInput, GraphInput *(discard)*
vpr_bool OpenFile, WriteLineFile
vpr_string ReadLineFile *(line read)*
vpr_int Slider *(slider value)*
vpr_long UnixTime *(timecode)*
OpenFile *(linecount)*
any type: PanelInput, MonitorInput ,
GraphInput *(send)*

*Code example:*

```
if (Panel.vpr_type==vp_type::vp_int)
  MyInt = Panel.vpr_int;
```

## Special strings

### Color strings

For: ApplicationName, Display, Led, Button, GraphButton, GraphValue, GraphDrawLine, GraphDrawPixel, GraphDrawCircle.

| | |
|---|---|
| $DEL(ETE)* | |
| $OFF** | |
| $BLACK | |
| $GRAY | |
| $PURPLE | |
| $PINK | |
| $BLUE | |
| $GREEN | |
| $YELLOW | |
| $ORANGE | |
| $RED | |
| $BROWN | |
| $WHITE | |

* draw only ** Led only

### Graph Type strings

Set graph type for: GraphValue. 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

### Pen size strings Draw

Size for: GraphDrawPixel, GraphDrawLine, GraphDrawCircle, GraphValue.

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

* default

### Text attributes/size strings

For: Display, Button, GraphButton.

| | |
|---|---|
| $SMALL | font size small |
| $NORMAL* | font size normal |
| $BIG | font size big |
| $BOLD | bold text |
| $xPT** | point size |

*Default. Resets bold and big

** Buttons x = 6, 8, 10, 12,14, 16PT
Displays x = 10, 12, 14, 16, 18PT

### Clear Function

MonitorLog, Info, Graph, GraphValue.

| | |
|---|---|
| $CLEAR | clear/reset entity |

### Button repeat Function

Button.

| | |
|---|---|
| $REPEAT | set button rep. |
| $NOREPEAT* | set button click |

*Default

## (Helper) Functions

### 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.

### Panel Synchronous request

bool PanelSyncRequest(event)
Request event and waits for answer. Only for ReadLineFile_x and UnixTime events.
*Concurrent use blocked!*
On success true : PanelSrqStatus = vpsrq_Success else false : vpsrq_Timeout / vpsrq_InvalidChannel / vpsrq_ConcurrencyErr.

### Helper function Sound

uint32_t _Sound( int freq, int dur)
Combines two int16_t (frequency Hz, duration mS) into one uint32_t.

### Helper functions Draw

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

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

*_Circle()*
char * _Circle(byte x, byte y, byte rad, int angle, int arc)
Center (x,y) *rad* (radius), start *angle*, *arc* angle. Omitting *angle* and *arc* draws a full circle.

*_VPoint() /_VLine() /_VCircle()*
uint16_t _VPoint(byte x, byte y)
uint32_t _VLine(byte Fx, Fy, Tx, Ty)
char * _VCircle(byte x, byte y, byte rad, int angle, int arc)
Same as _Point, _Line and _Circle but transforms y values from value (0-255) to coordinate (0-220).

### Sendf() / Printf formatting

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

*specifiers (limited list)*

| | | |
|---|---|---|
| %c | ascii char | byte |
| %d | signed dec. | int16 |
| %ld | signed dec. | int32 |
| %u | unsigned dec. | uint16 |
| %lu | unsigned dec. | uint32 |
| %o | unsigned octal | any |
| %x | uns. hex lc/uc | any |
| %s | string | char[] |
| %f* | float | float |

*Not AVR supported. see: sendf() float

*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

### Helper function Float string

char * _FString(floatNumber, length, decimals);

### sendf() float

*Float not supported on AVR (Uno, Nano, Mega … )*
Use _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.

### Unicode characters

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

### 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. (*not Due*)
*Example:*
Panel.sendf (Display_1, F("Value %d"), 10);