EE-206

Systèmes de mesure

Lesson Outline

- LabVIEW: what & why?
- Getting started
- Controls vs indicators

LabVIEW Trivia

- What does LabVIEW stand for?

Laboratory **V**irtual **I**nstrument **E**ngineering **W**orkbench.

- What is LabVIEW?

LabVIEW is a **graphical programming language**, typically used for data acquisition, instrument control, and industrial automation.

Introduced in 1986, it is supported by several OS (mostly Windows, Unix and Linux) and it can be installed on PCs as well as industrial controllers.

Graphical Program

MATLAB, HTML, Java are all traditional textual languages: the code consists of a sequence of operations/instructions.

LabVIEW is a graphical (G) programming language:

- no text but a sort of block diagram
- each block correspond to an operation/instrument
- each block has different options/configurations
- the connection defines the order of the different stages and the flow of the data

Advantages

LabVIEW yields also signal processing functionalities, but it is mostly used for:

1. remote instrument control
2. measurement data acquisition
3. automatic control routines

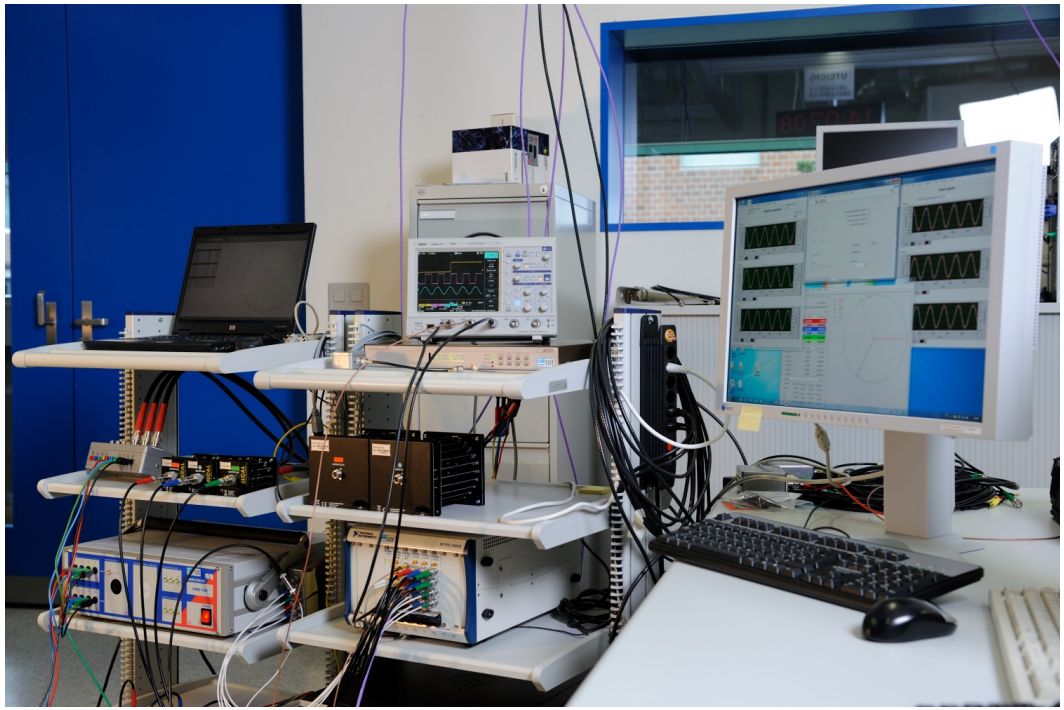
NB: a program in LabVIEW is called **V**irtual **I**nstrument (**VI**) as it behaves as an instrument with controls and outputs (either numeric results or graphs).

Examples

- Coordination and interface between different instruments

➔ **P**CI e**X**tensions for **I**nstrumentation (**PXI**)

- waveform generator
- voltage amplifier
- current amplifier
- shunt & dividers
- GPS synchronization
- data acquisition
- graph and result logs



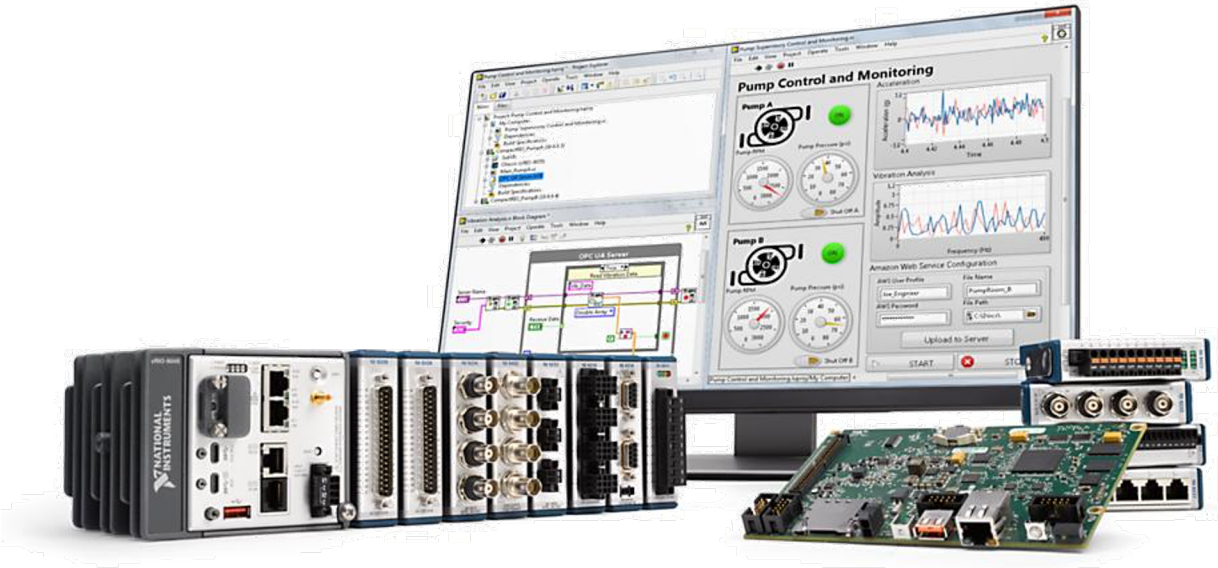
Examples

- Real-time stand-alone measurement unit

➔ **c**ompact **R**econfigurable **IO** modules (**cRIO**)

- real-time controller
- network interface
- pluggable modules
- FPGA board for

(fast, deterministic)



Examples

- Plug & play acquisition board

➔ **c**ompact **D**ata **A**c**Q**uisition platform (cDAQ)

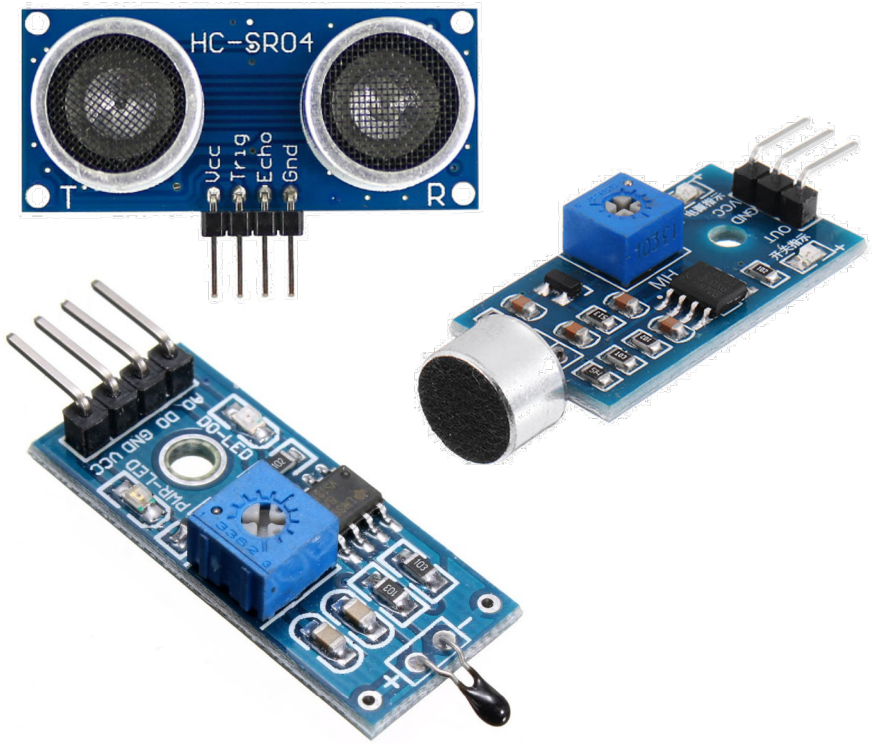
- extension of the PC
- pluggable modules
- data acquisition
- data transfer
- programmable trigger
- control outputs

Examples

- Interface with third-party sensors (Arduino style)

➔ e.g. ultrasound, thermistor, microphone

- integrated libraries
- dedicated functions
- app examples
- self-test routines
- automatic scaling



Getting started

Let's launch Labview..

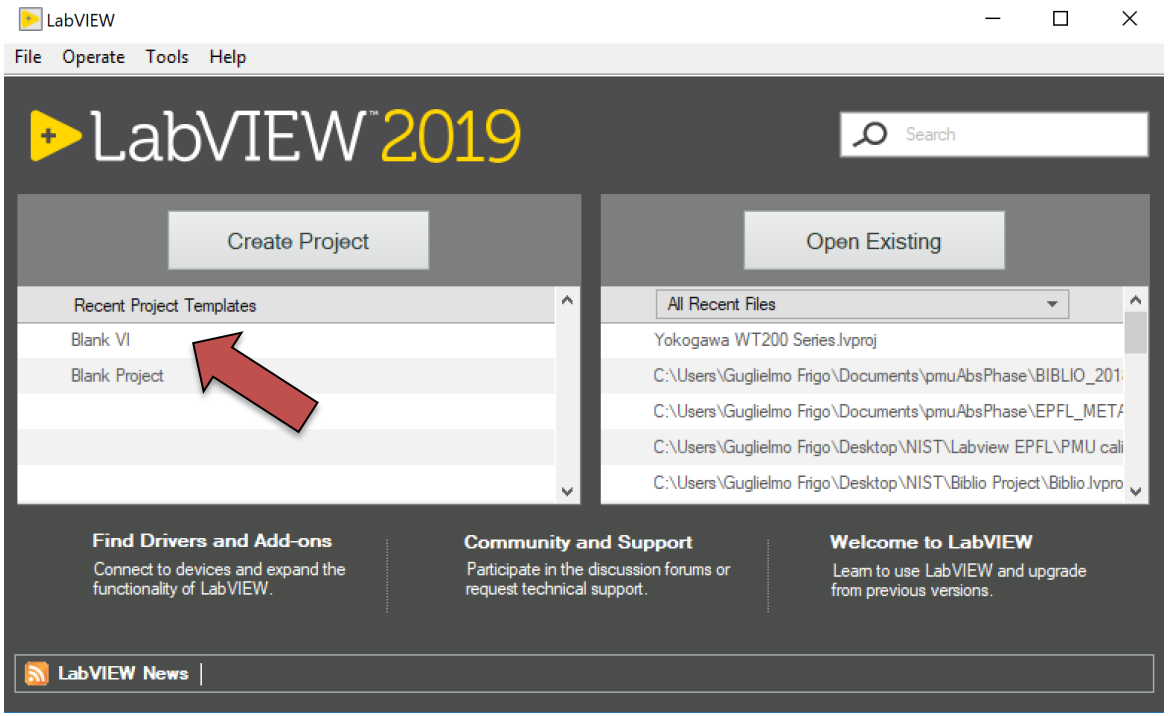
Create a file:

- VI → single file (.vi)
- Project → folder

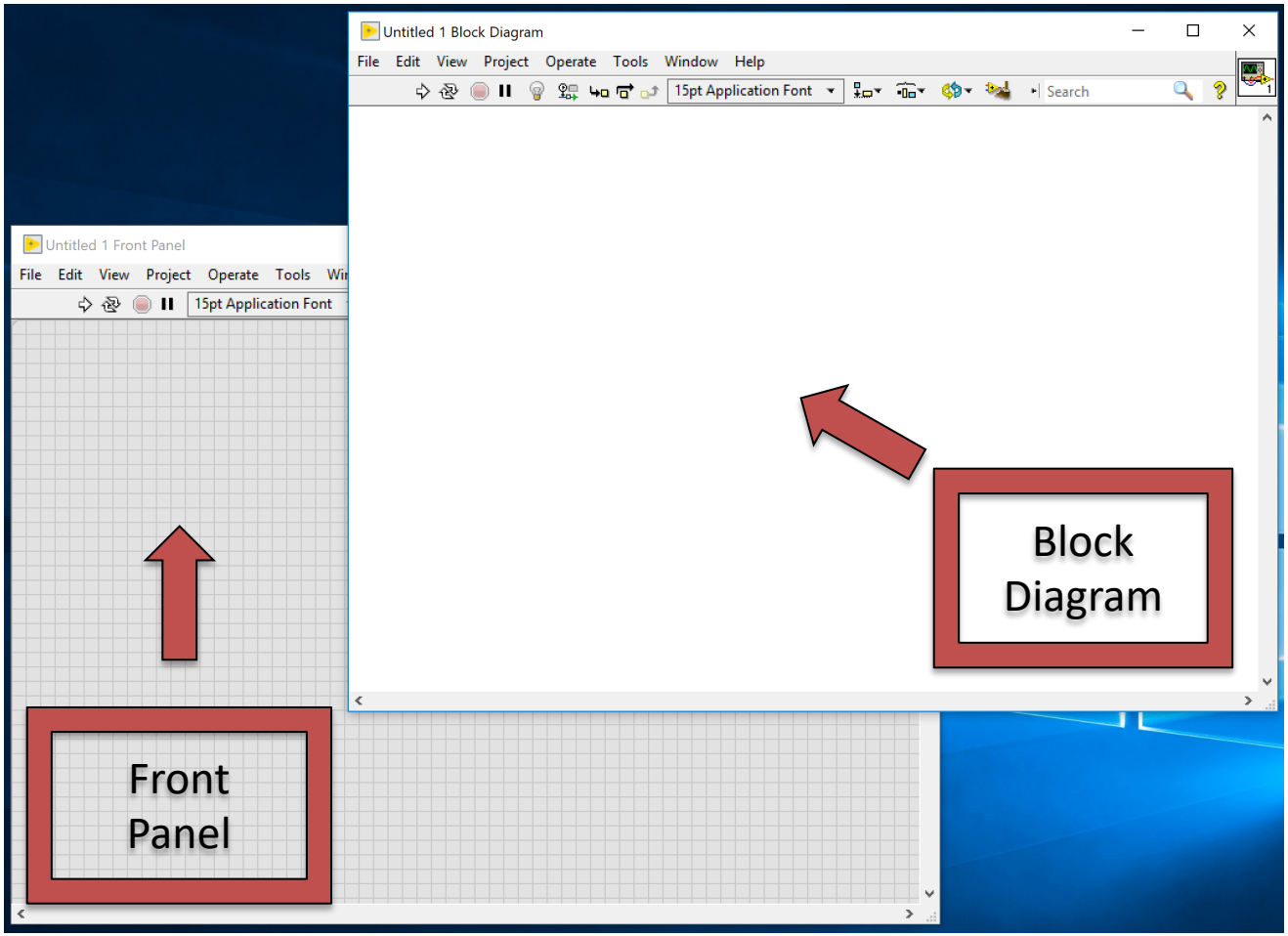
with multiple files (.lvproj)

Open a file:

- Recent files' history



Blank VI


Untitled 1 Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
Search
Untitled 1 Front Panel
File Edit View Project Operate Tools
15pt Application Font
Block Diagram
Front Panel

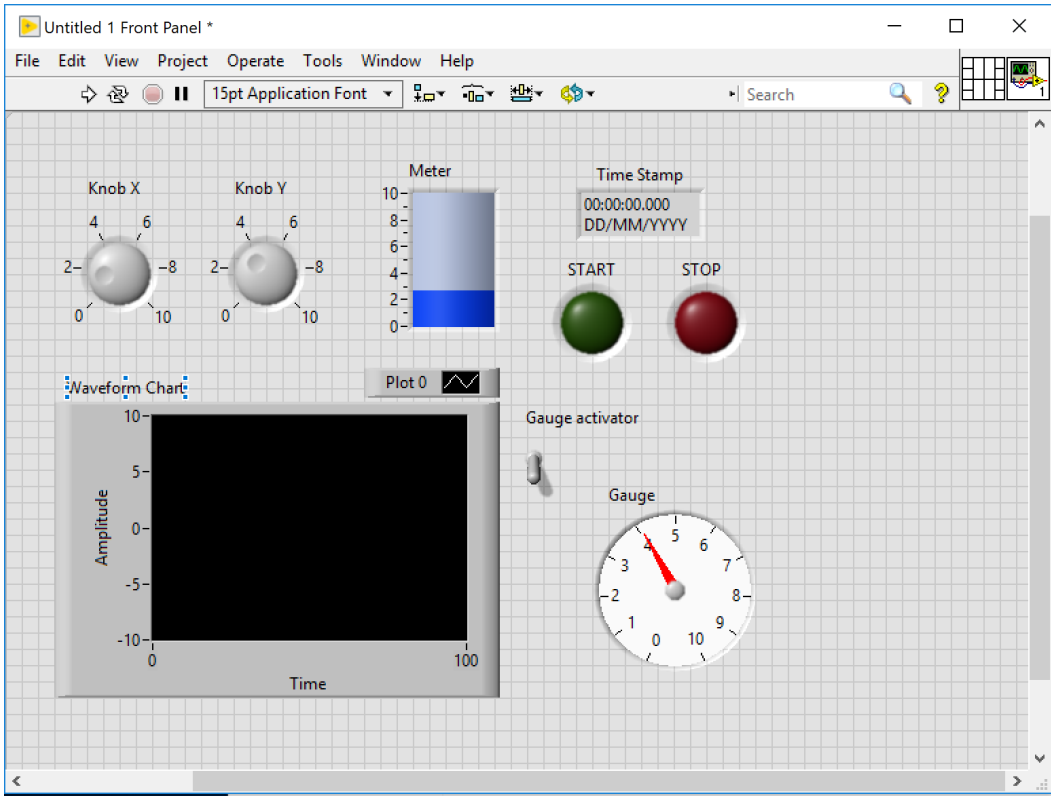
Front Panel

The Front Panel is the user interface of your VI

Once populated with:

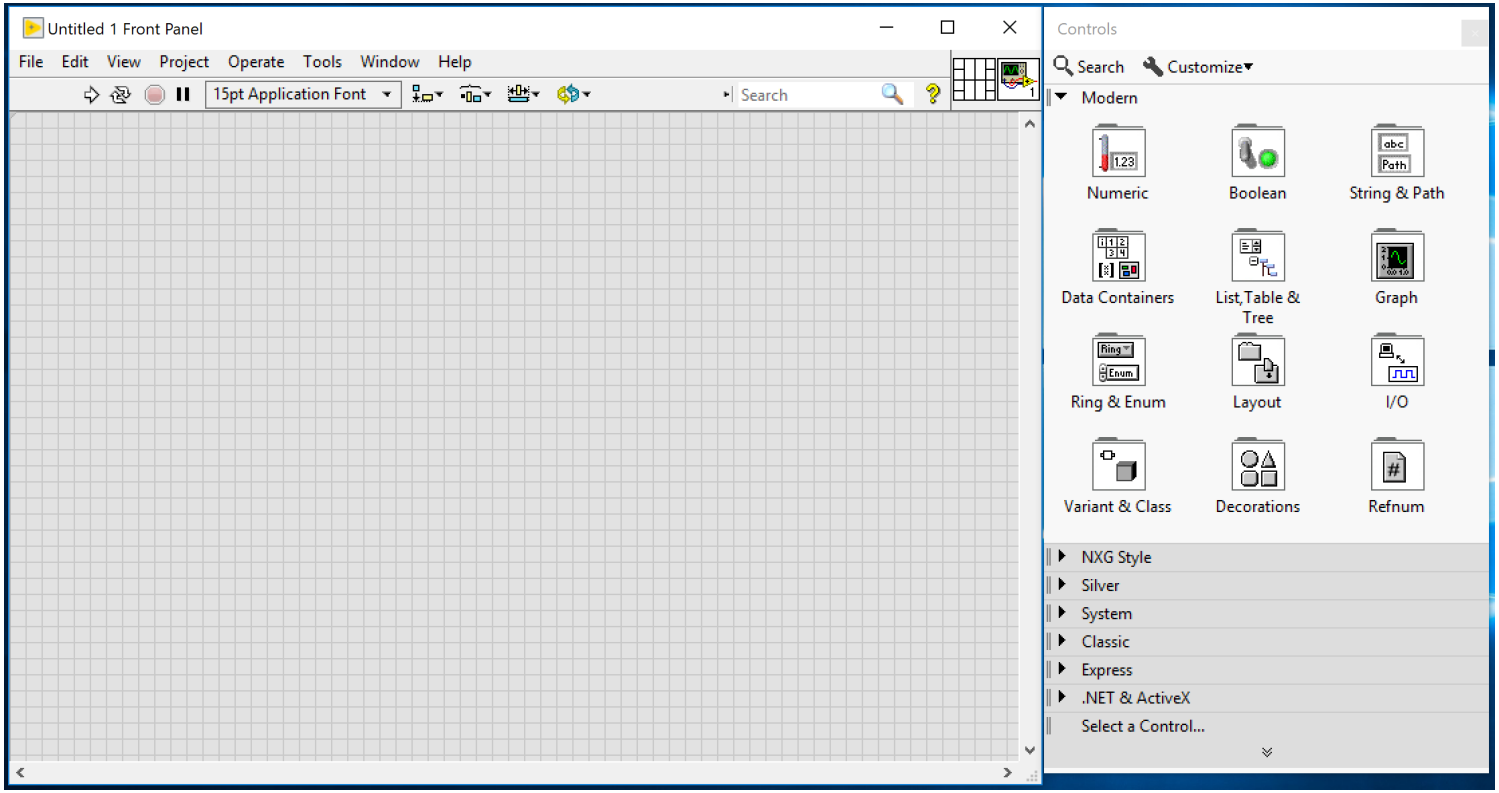
- controls
- indicators
- graphs

it will appear as the front panel of a real instrument

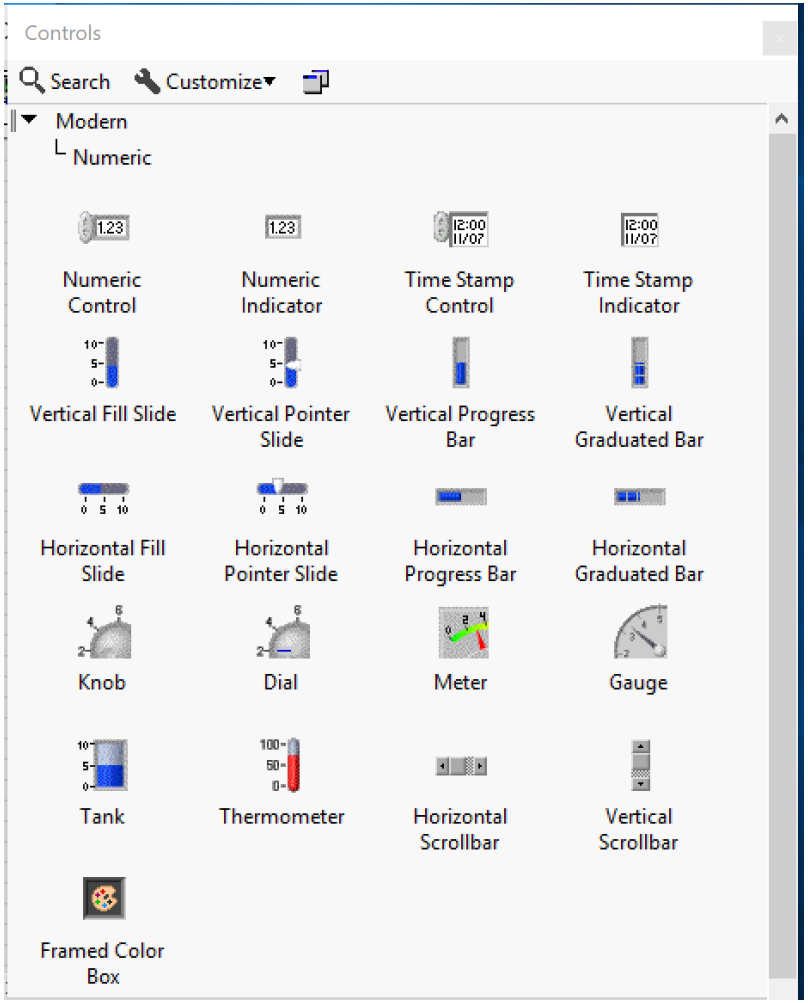


Controls Palette

In order to introduce new controls and switches we can use View → Controls Palette

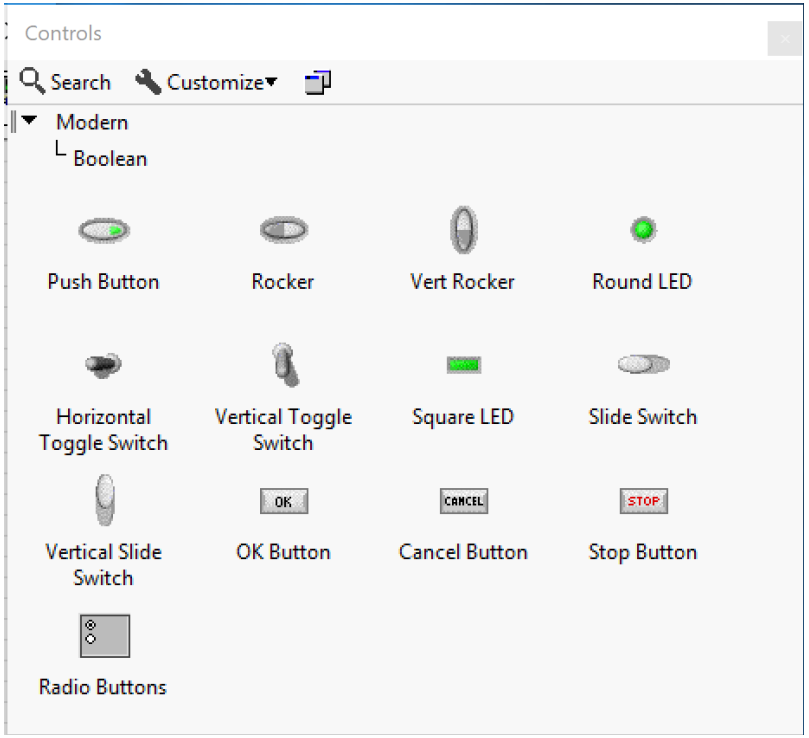


Controls types



Numeric

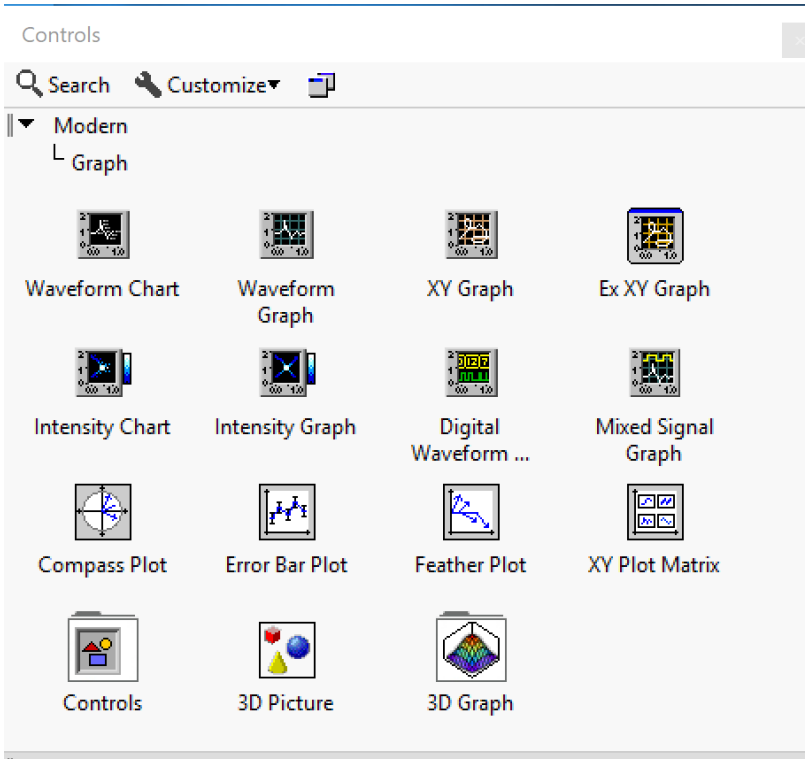
Boolean

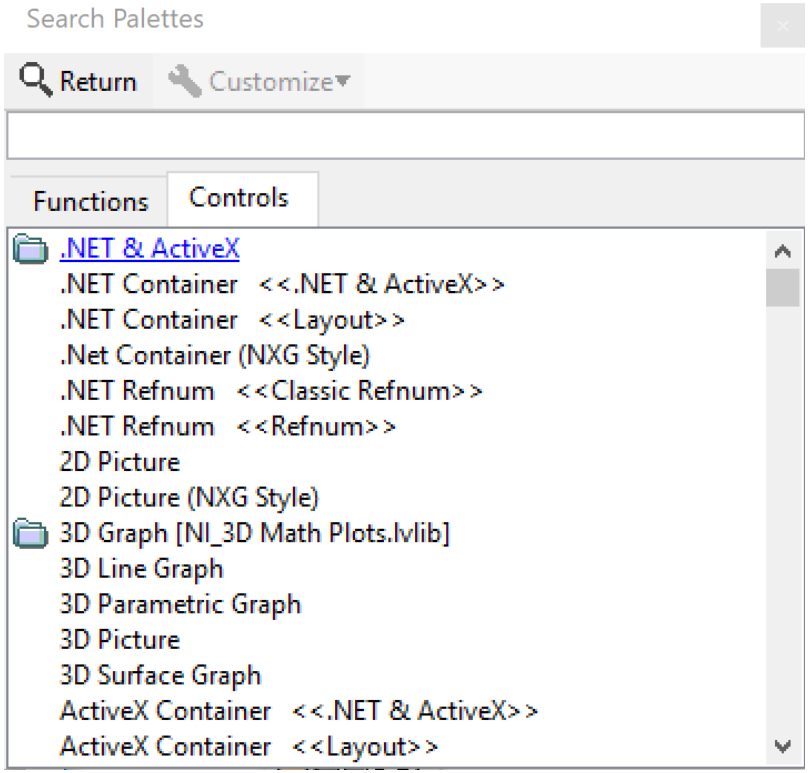


Controls types

Graph

Search into the library

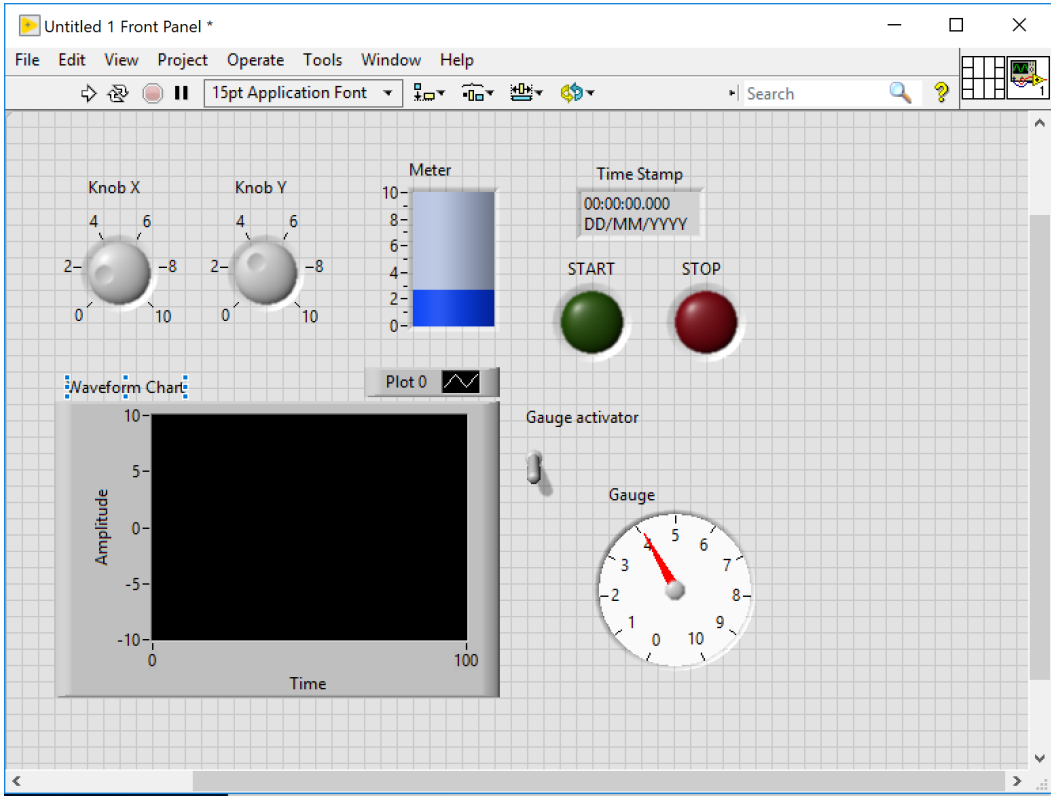




Example

Just by dragging and dropping on the front panel, we get…

- manual controls
- digital controls
- switches/leds
- gauge/tank meter



Controls vs Indicators

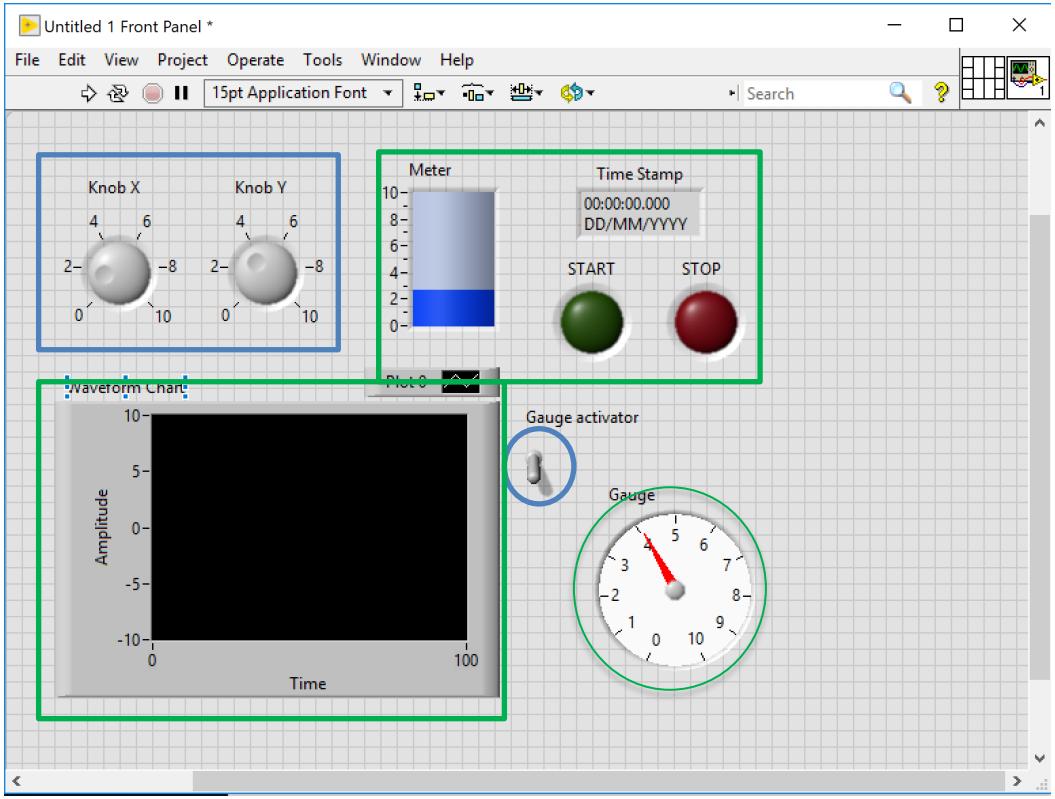
The same difference between function inputs and outputs:

- **Controls (inputs)**

the user can set specific parameters or trigger specific operations

- **Indicators (output)**

the user can visualize the results or the process state through numbers or graphical tools

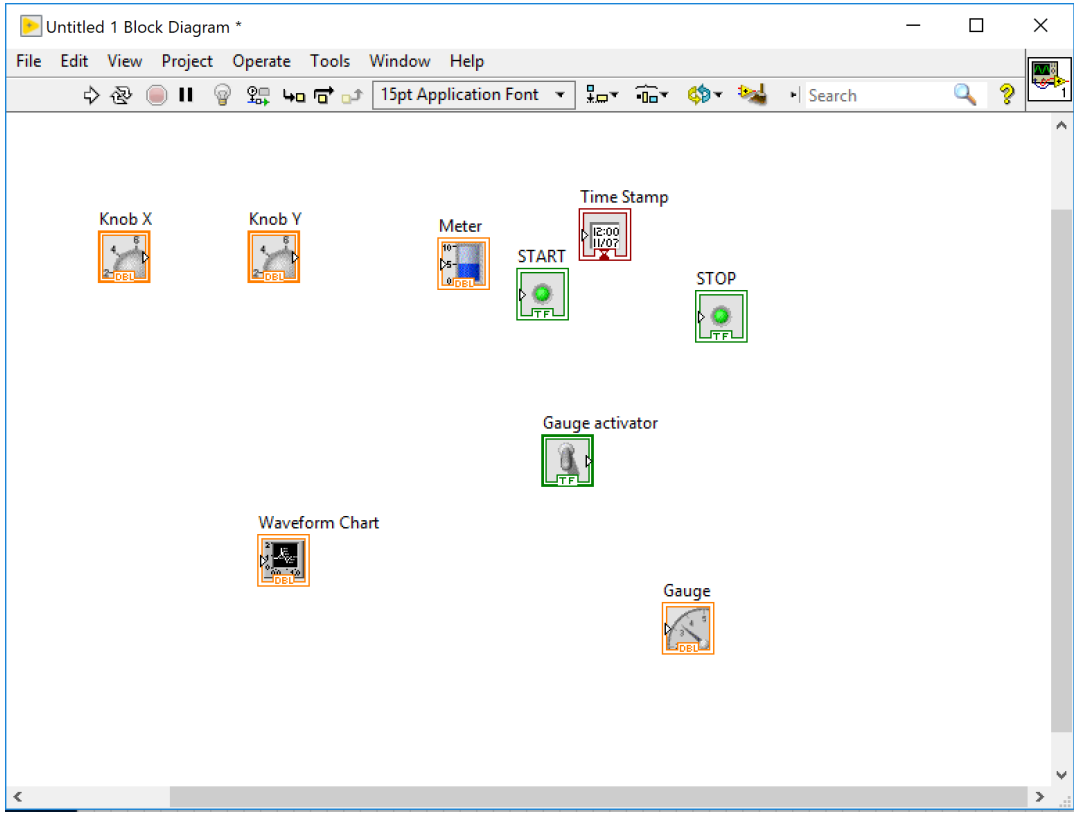


Block Diagram

Every time you drag something on the front panel you have its counterpart on the block diagram, where you can:

- connect them
- move them

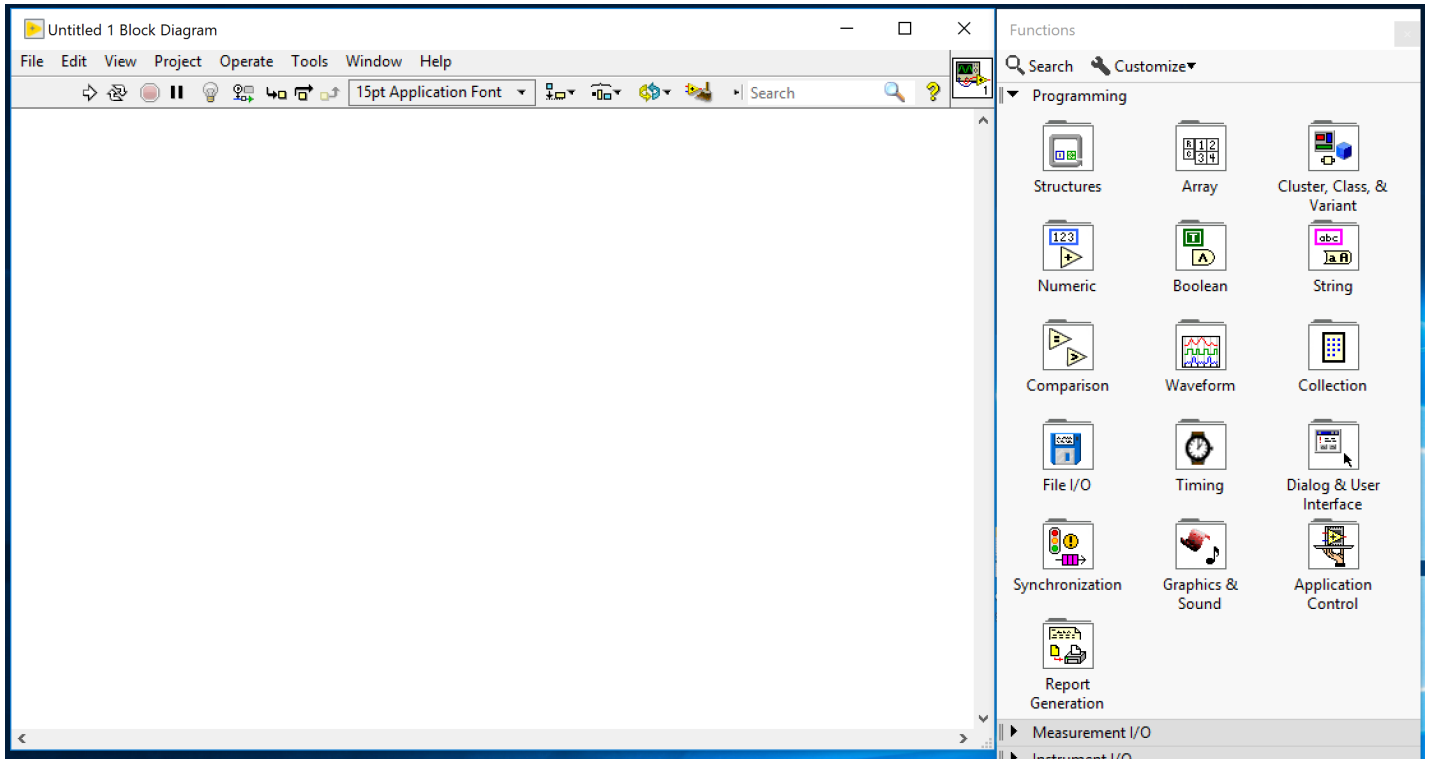
NB#1: NO ZOOM
NB#2: if you delete here it will disappear also in the front panel!



Functions Palette

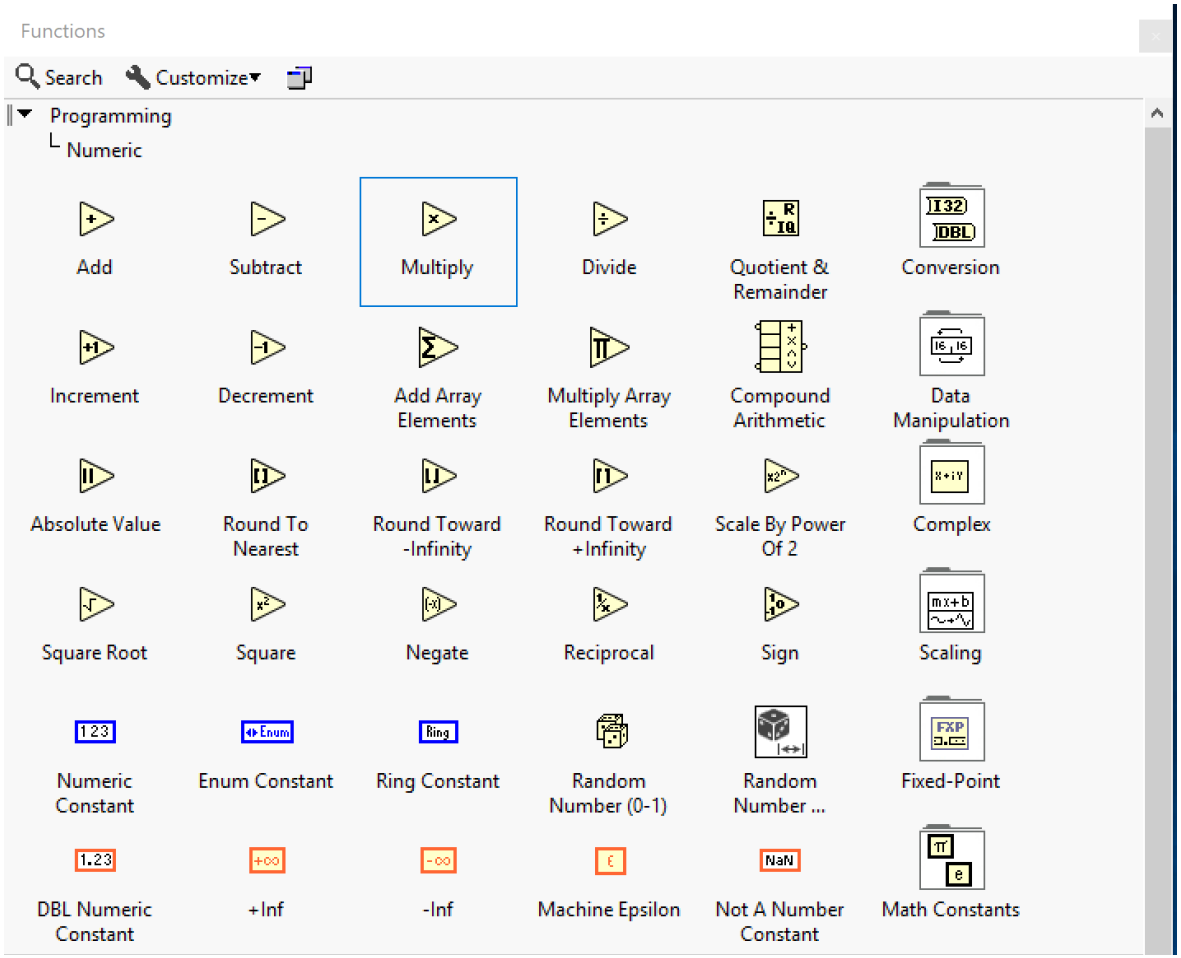
Controls and indicators can be connected in several ways

View → Functions Palette



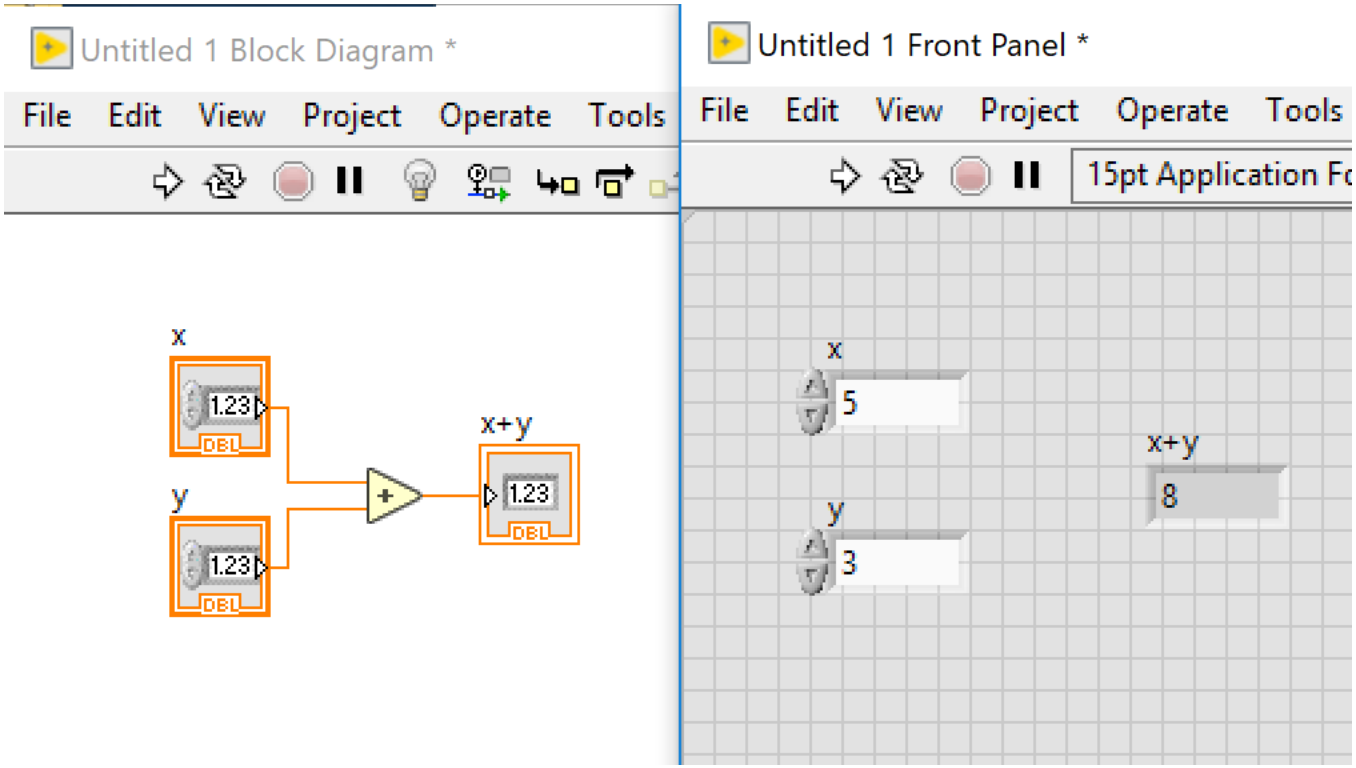
Numeric

NB: In the block diagram it is also possible to create constants: values that are not modifiable by the user (controls) or by the execution of the program (indicators).

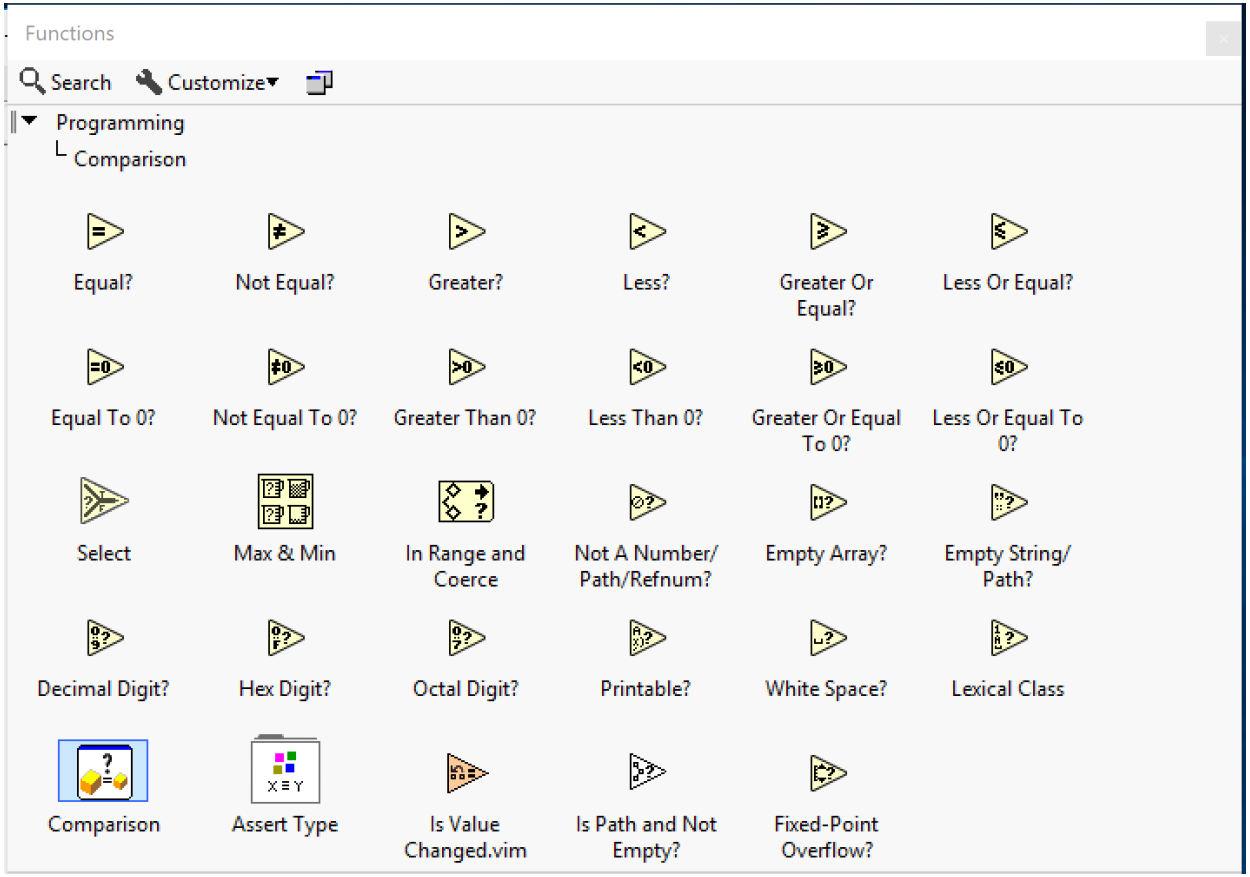


Numeric example

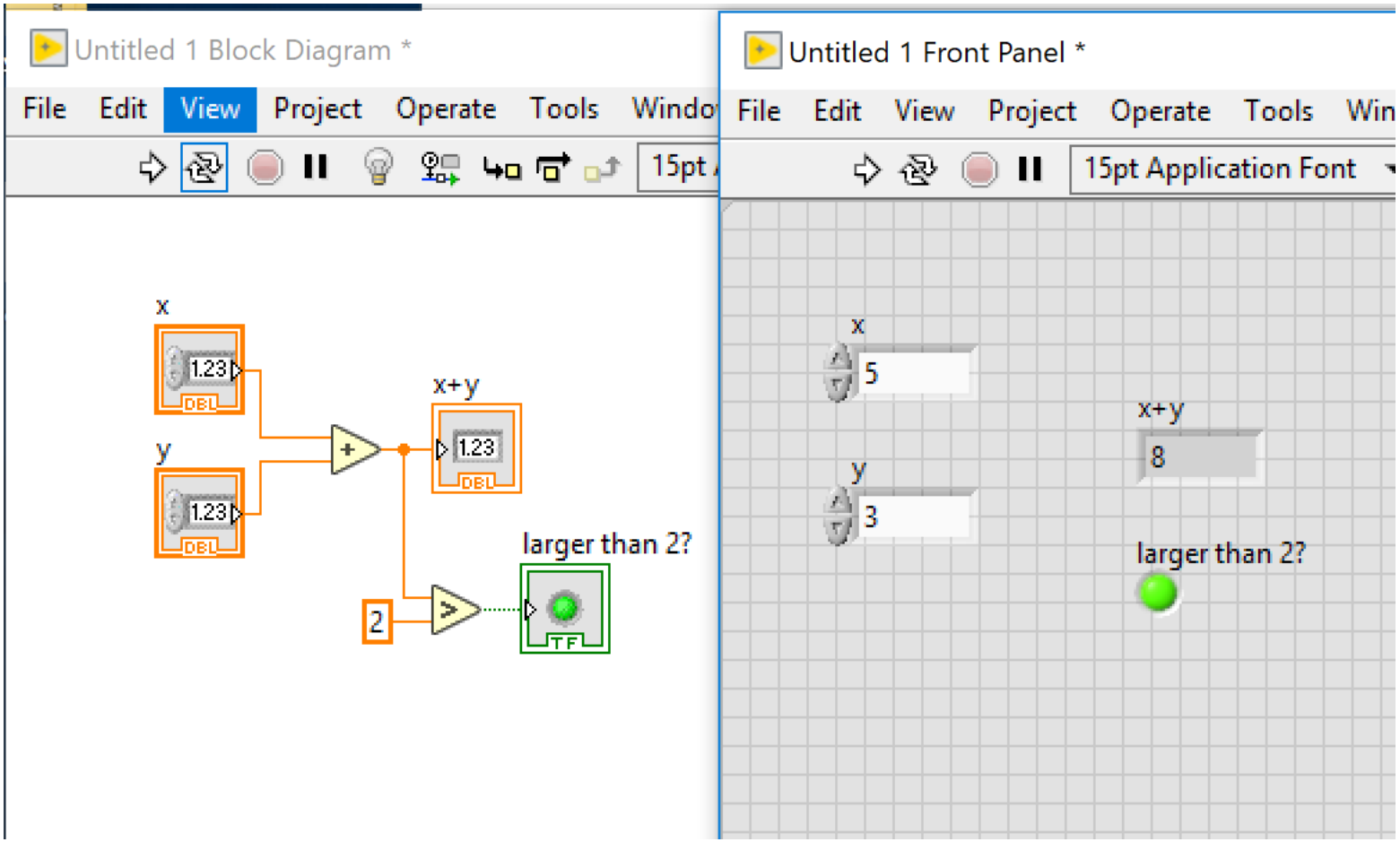
- The sum of two numbers requires two control, a function and one indicator



Comparison


Functions
Search
Customize
Programming
Comparison
Equal?
Not Equal?
Greater?
Less?
Greater Or Equal?
Less Or Equal?
Equal To 0?
Not Equal To 0?
Greater Than 0?
Less Than 0?
Greater Or Equal To 0?
Less Or Equal To 0?
Select
Max & Min
In Range and Coerce
Not A Number/ Path/Refnum?
Empty Array?
Empty String/ Path?
Decimal Digit?
Hex Digit?
Octal Digit?
Printable?
White Space?
Lexical Class
Comparison
Assert Type
Is Value Changed.vim
Is Path and Not Empty?
Fixed-Point Overflow?

Comparison example



Tools Palette

Since LabVIEW is a graphical programming language it is important to use the mouse and the cursor..

View → Tools Palette

Automatic selection

Operating

Positioning

Labeling

Wiring

Shortcut

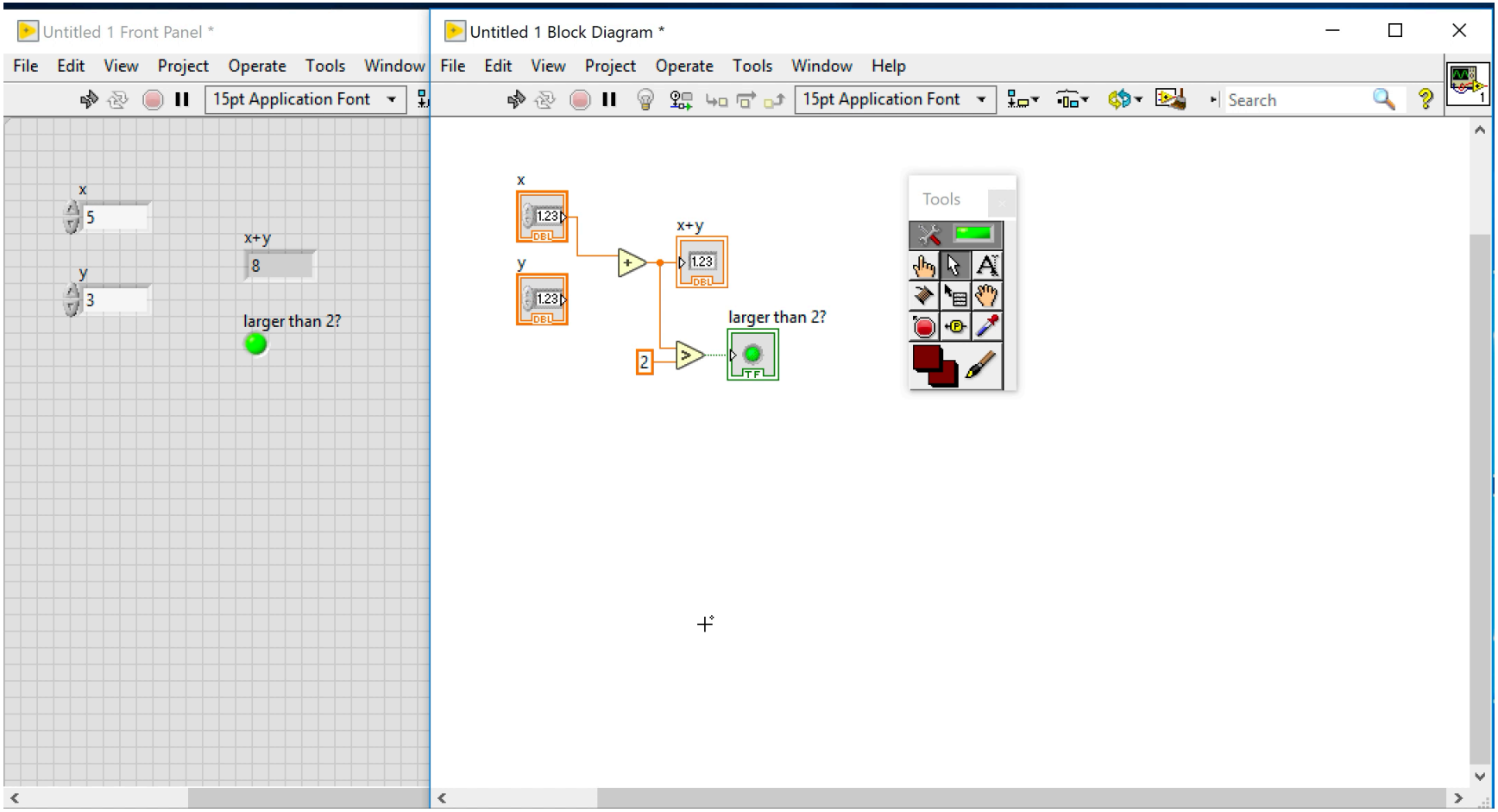
Scrolling

Setting a breakpoint

Probing

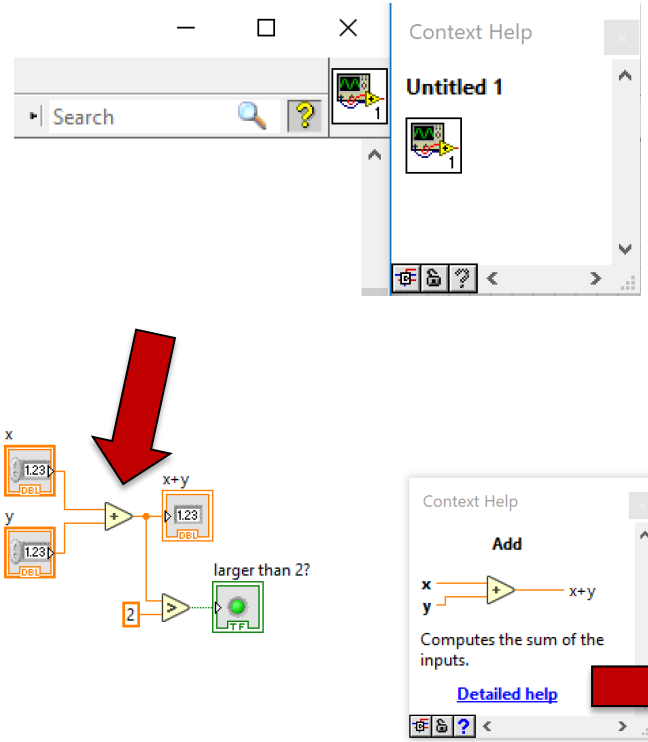
Picking a color

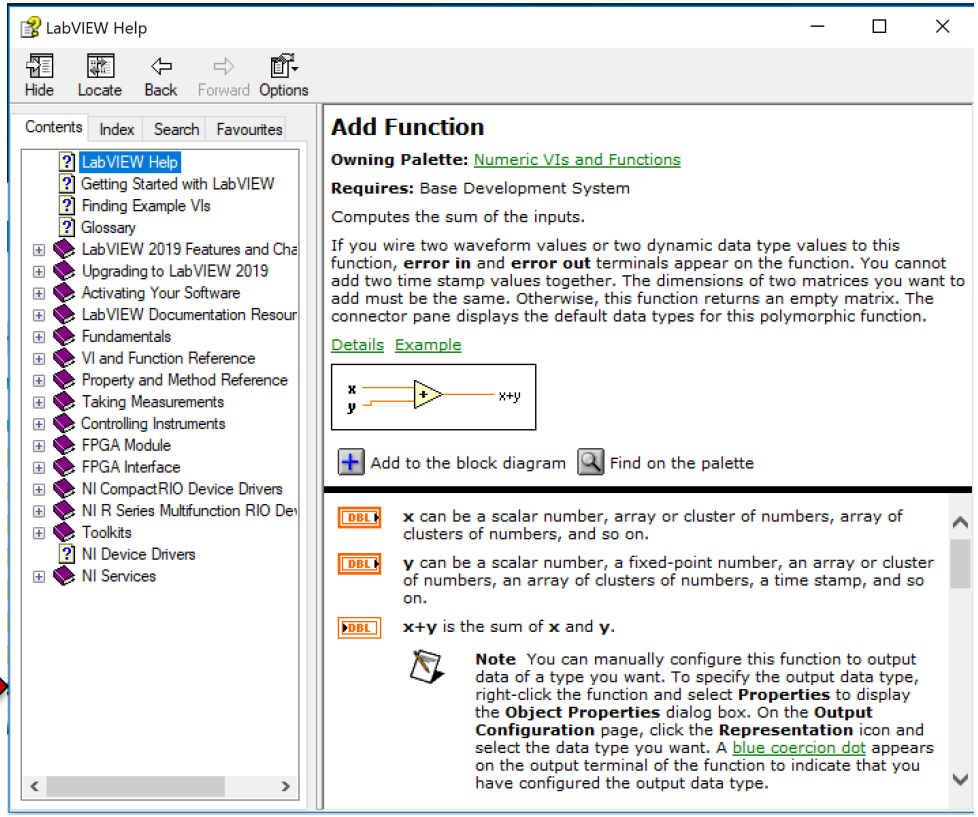
Tools example



Help

If you don’t know what something is, you just ask for help…





Execution modes

When the code is ready, we can run it!

Run the code once

Run the code iteratively

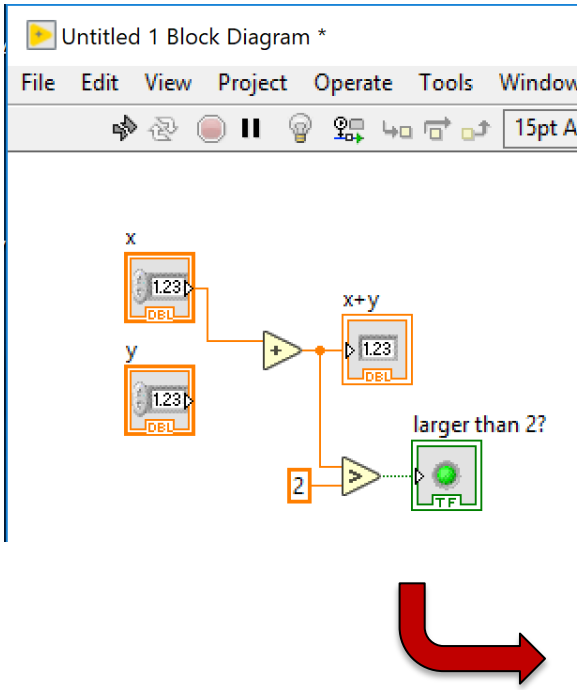
Abort the execution

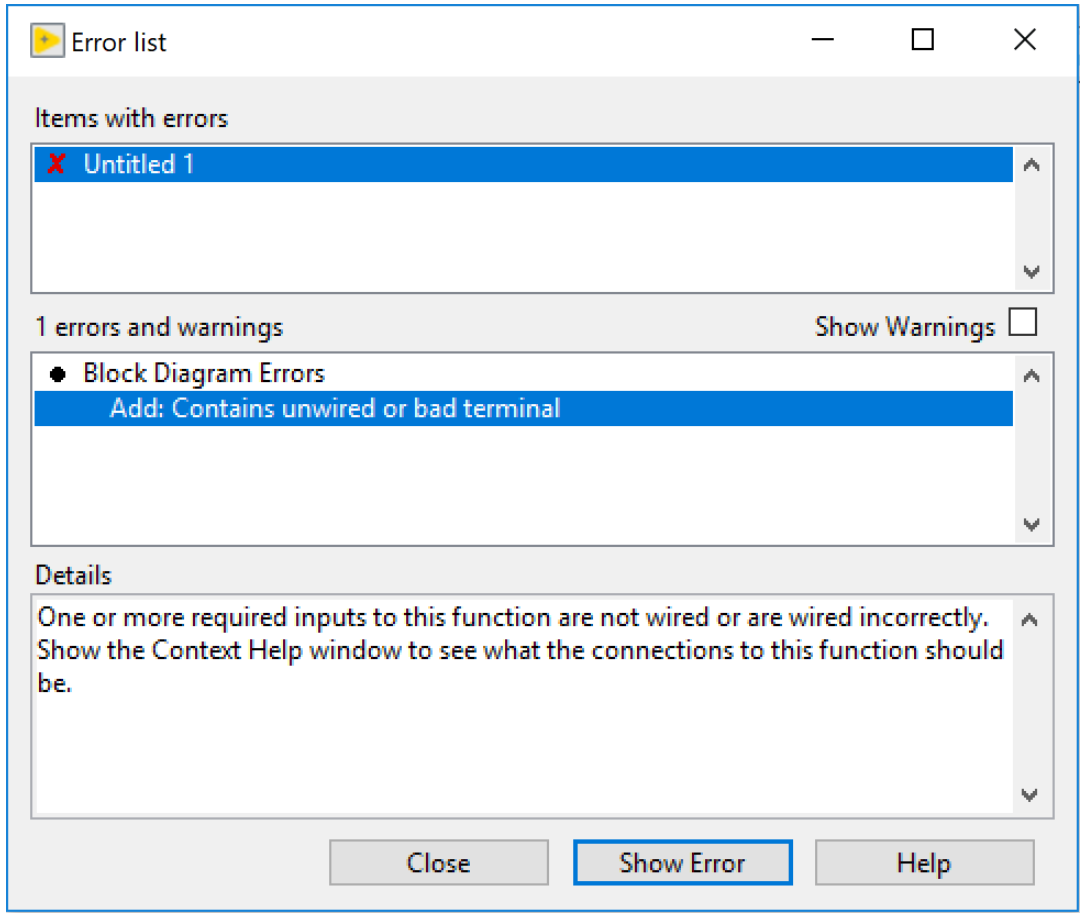
Pause the execution

Run in DEBUG mode

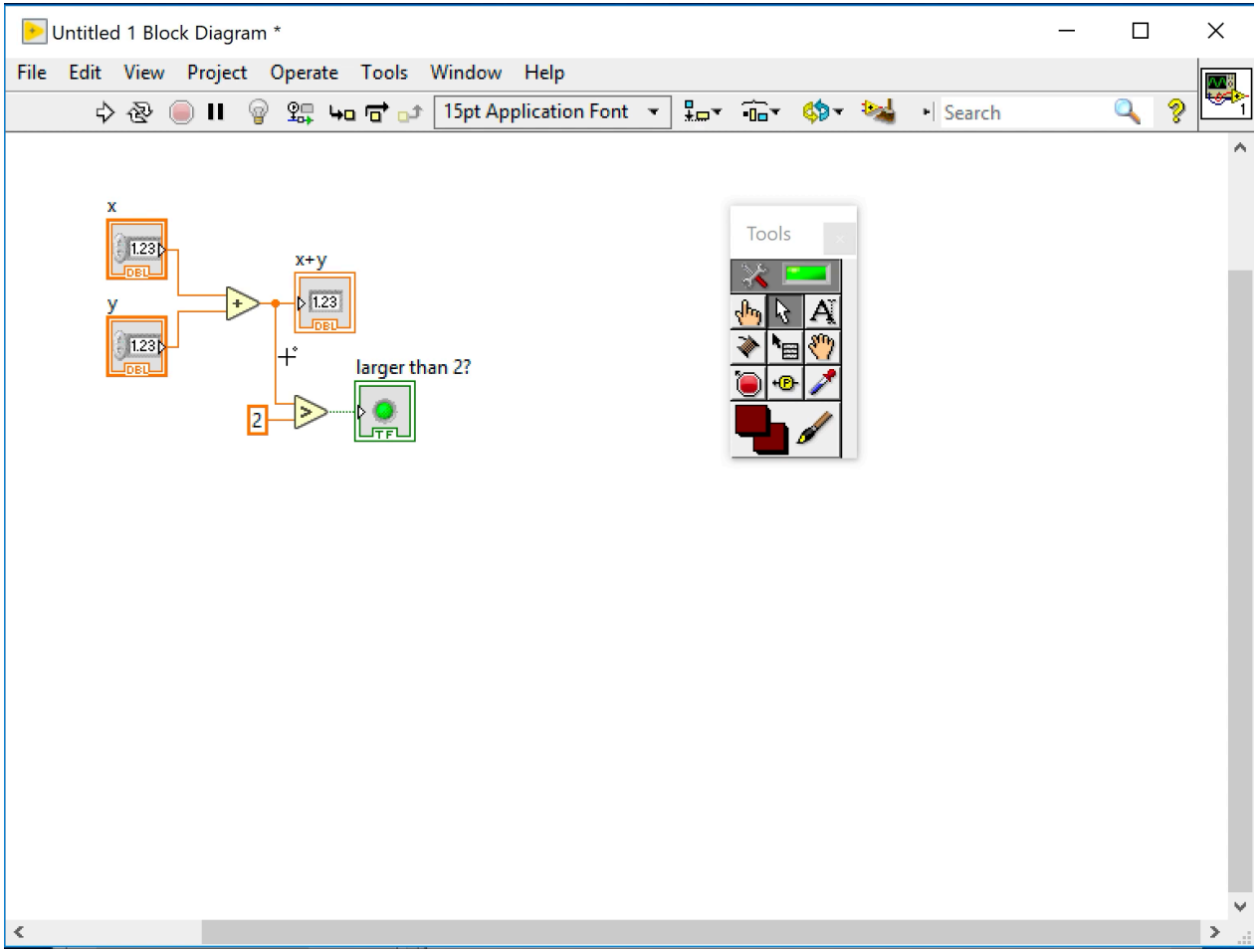
If there is something wrong in the code, e.g. a block disconnected, the arrow is broken and the code cannot be run.

Broken arrow





Debug example



Recap: was everything clear?

- What's the difference between block diagram and front panel?
- What's the difference between a control, a constant and an indicator?
- What are the main tools in LabVIEW?