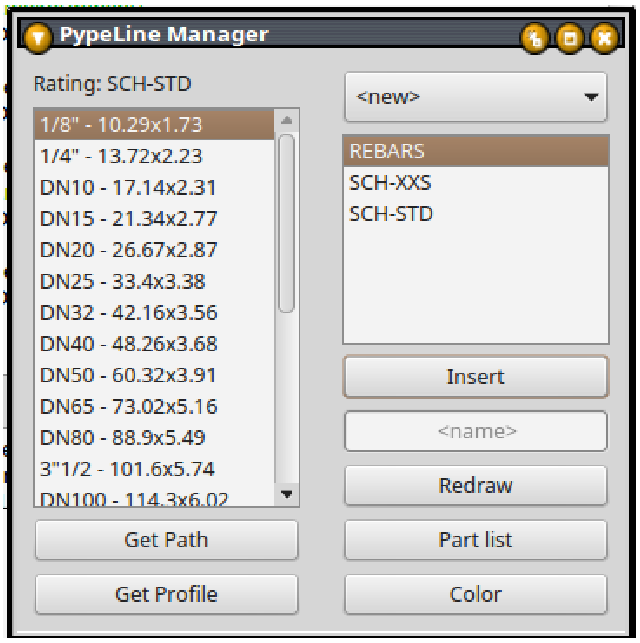
PYPELINE AND PYPEBRANCH

This tutorial is intended to describe in deeper detail the "meta-objects" defined in flamingo pipe-toolbar.
Pypeline has been already used in previous tutorials: just to recall the main points, it defines a collection of pype-parts that can be also layed down on a common path and collect into a Group.
The python class defines some methods to redraw, select profiles, select routes, change color etc. which can be executed from a common dialog.
The main point is that pype parts are actually independent from the path of the PypeLine, so they can be moved, modified or even deleted. They will be re-created only when the PypeLine will be redrawn.


PypeLine Manager

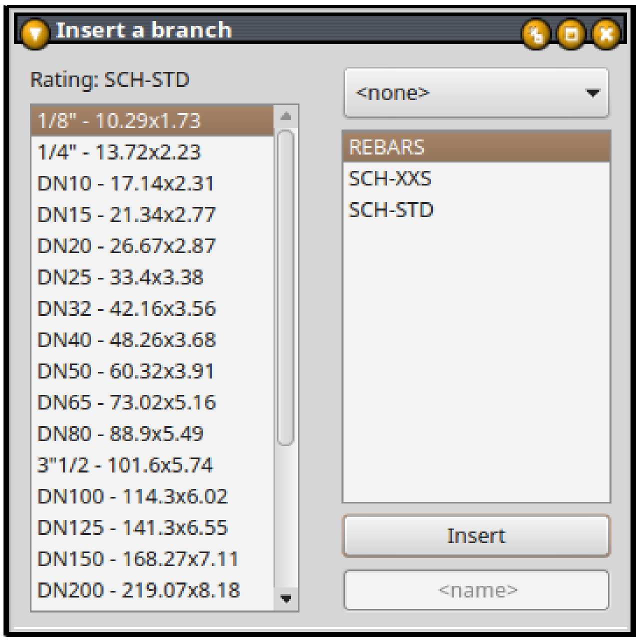
Rating: SCH-STD

<new>

1/8" - 10.29x1.73
1/4" - 13.72x2.23
DN10 - 17.14x2.31
DN15 - 21.34x2.77
DN20 - 26.67x2.87
DN25 - 33.4x3.38
DN32 - 42.16x3.56
DN40 - 48.26x3.68
DN50 - 60.32x3.91
DN65 - 73.02x5.16
DN80 - 88.9x5.49
3"1/2 - 101.6x5.74
DN100 - 114.3x6.02

REBARS
SCH-XXS
SCH-STD

This may be annoying for long and complex routes of pyping. For this reason a new meta-object (i.e. collection of parts) is now available: the PypeBranch.
The Insert... dialog is similar to the others, with the two lists of sizes and ratings of pipe, the combo for the PypeLines, a line-edit for the name and an [Insert] pushbutton.
The difference with PypeLine is that this object has a mandatory Base attribute and that must be a DWire (or a Sketch) with a continuous path.
The pype parts (only tubes and curves) are automatically updated as soon as the Base is changed but they can not be modified individually after the object is created.


Insert a branch

Rating: SCH-STD

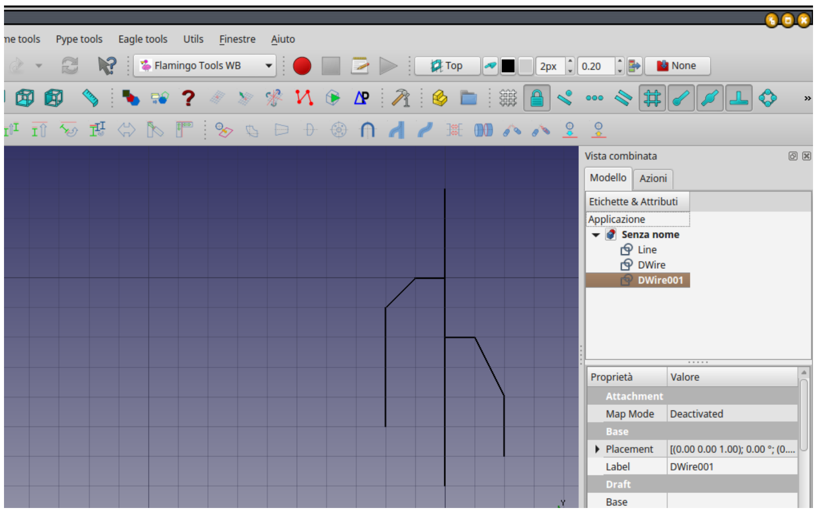
<none>

1/8" - 10.29x1.73
1/4" - 13.72x2.23
DN10 - 17.14x2.31
DN15 - 21.34x2.77
DN20 - 26.67x2.87
DN25 - 33.4x3.38
DN32 - 42.16x3.56
DN40 - 48.26x3.68
DN50 - 60.32x3.91
DN65 - 73.02x5.16
DN80 - 88.9x5.49
3"1/2 - 101.6x5.74
DN100 - 114.3x6.02
DN125 - 141.3x6.55
DN150 - 168.27x7.11
DN200 - 219.07x8.18

REBARS
SCH-XXS
SCH-STD

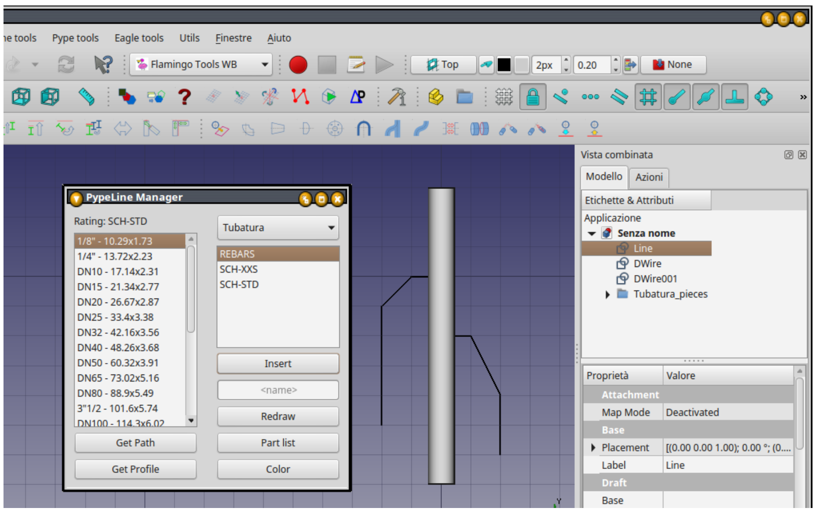
So in some way PypeBranches are easier to handle but less flexible than PypeLines. So they are perfect to be inserted in a PypeLine to organize it.

Let's see an example.



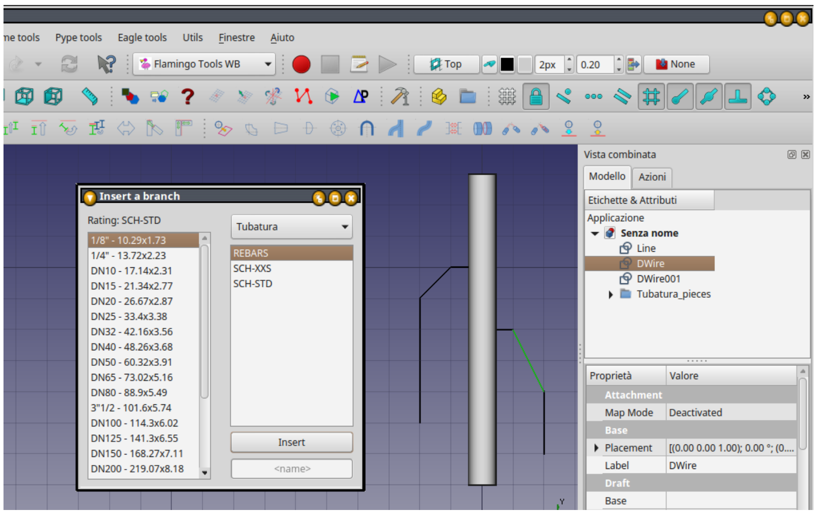
Proprietà	Valore
Attachment	
Map Mode	Deactivated
Base	
Placement	[(0.00 0.00 1.00); 0.00 °; (0....
Label	DWire001
Draft	
Base	

To start with, just sketch down a route with one Line (the main header) and two DWires (the branches)



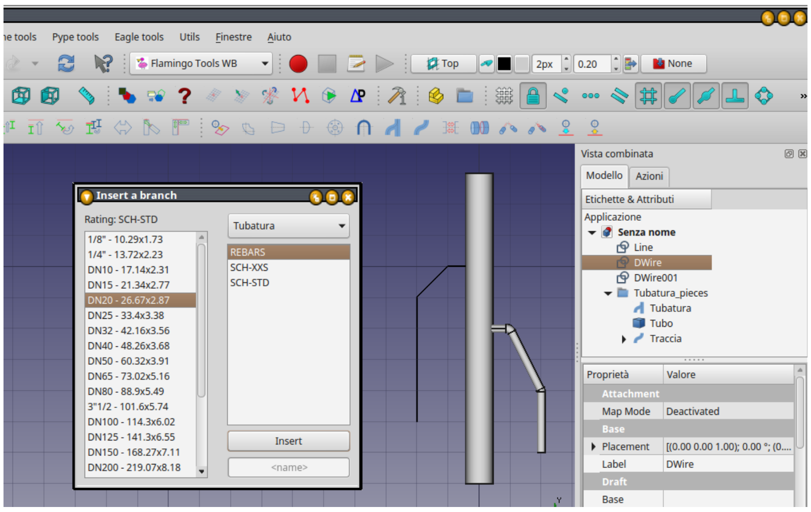
Proprietà	Valore
Attachment	
Map Mode	Deactivated
Base	
Placement	[(0.00 0.00 1.00); 0.00 °; (0....
Label	Line
Draft	
Base	

Select the center Line and over it insert one PypeLine DN80. The PypeLine's group will be added in the model's tree.



Proprietà	Valore
Attachment	
Map Mode	Deactivated
Base	
Placement	[(0.00 0.00 1.00); 0.00 °; (0....
Label	DWire
Draft	
Base	

Now insert the branches by selecting in the viewport the DWire, in the dialog the PypeLine to attach (optionally) and finally the size and the rating (which can be different from the PypeLine).



Insert a branch

Rating: SCH-STD

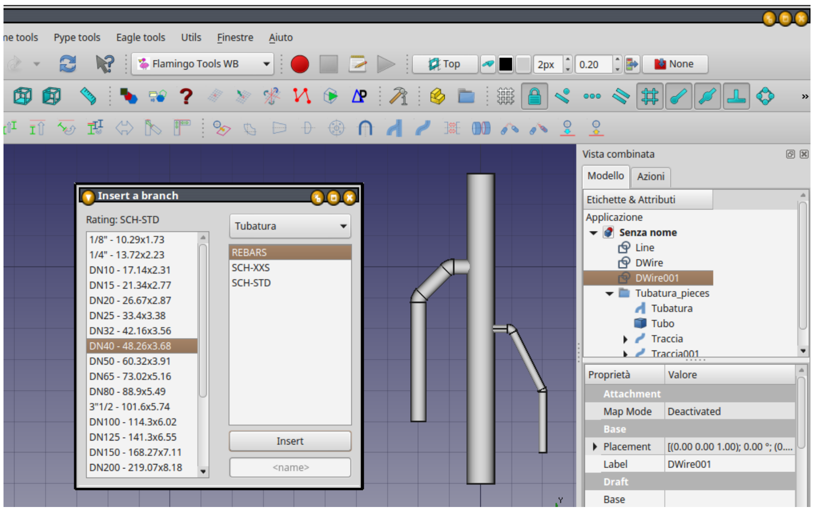
1/8" - 10.29x1.73	Tubatura
1/4" - 13.72x2.23	REBARS
DN10 - 17.14x2.31	SCH-XXS
DN15 - 21.34x2.77	SCH-STD
DN20 - 26.67x2.87	
DN25 - 33.4x3.38	
DN32 - 42.16x3.56	
DN40 - 48.26x3.68	
DN50 - 60.32x3.91	
DN65 - 73.02x5.16	
DN80 - 88.9x5.49	
3"1/2 - 101.6x5.74	
DN100 - 114.3x6.02	
DN125 - 141.3x6.55	Insert
DN150 - 168.27x7.11	
DN200 - 219.07x8.18	<name>

Applicazione
- Senza nome
  - Line
  - DWire
  - DWire001
  - Tubatura_pieces
    - Tubatura
    - Tubo
    - Traccia

Proprietà	Valore
Attachment	
Map Mode	Deactivated
Base	
Placement	[(0.00 0.00 1.00); 0.00 °; (0....
Label	DWire
Draft	
Base	

Now insert the first branch DN20.

Notice in the model tree the new branch with all its components under it.



Insert a branch

Rating: SCH-STD

1/8" - 10.29x1.73	Tubatura
1/4" - 13.72x2.23	REBARS
DN10 - 17.14x2.31	SCH-XXS
DN15 - 21.34x2.77	SCH-STD
DN20 - 26.67x2.87	
DN25 - 33.4x3.38	
DN32 - 42.16x3.56	
DN40 - 48.26x3.68	
DN50 - 60.32x3.91	
DN65 - 73.02x5.16	
DN80 - 88.9x5.49	
3"1/2 - 101.6x5.74	
DN100 - 114.3x6.02	
DN125 - 141.3x6.55	Insert
DN150 - 168.27x7.11	
DN200 - 219.07x8.18	<name>

Applicazione
- Senza nome
  - Line
  - DWire
  - DWire001
  - Tubatura_pieces
    - Tubatura
    - Tubo
    - Traccia
    - Traccia001

Proprietà	Valore
Attachment	
Map Mode	Deactivated
Base	
Placement	[(0.00 0.00 1.00); 0.00 °; (0....
Label	DWire001
Draft	
Base	

Similarly you add the second branch, of a different size for instance (DN40).

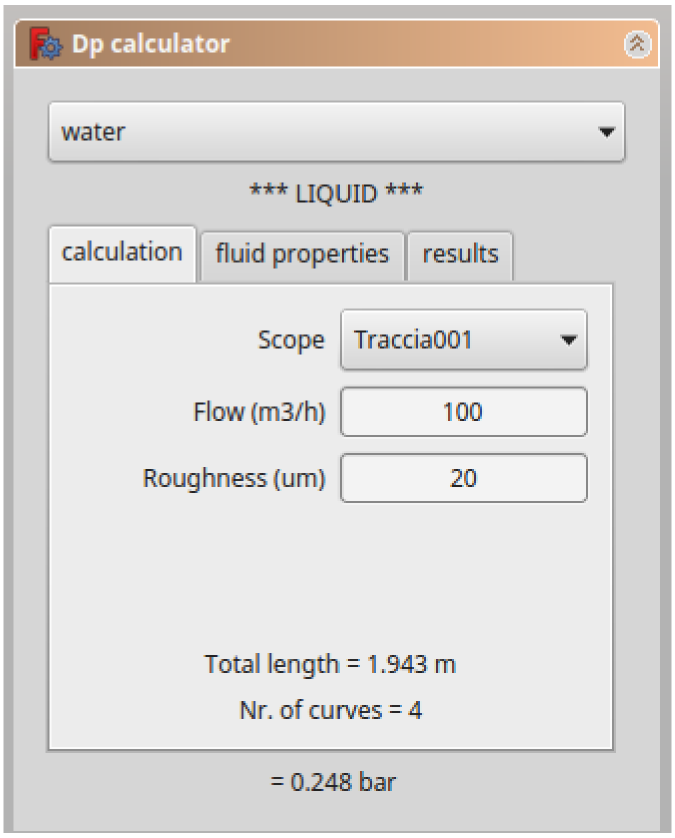
In this way is possible to organize better the pipes of a pype-line: for example if you move "DWire001" or change the length of its segments, the pipes of the branch will update automatically.

So now, if you want to move back and forth or rotate the branches respect to the main pipe header, it's necessary only to change the placement of the Base (i.e. the DWire or the Sketch). Then all the pipes and curves are redrawn automatically following it.

PRESSURE LOSS CALCULATOR

This new tool in the "Utils" menu allows to calculate the pressure losses over the elements (curves and tubes) selected in the viewport or directly by selecting one PypeBranch from the drop-down box, as the ones created in the example above for instance.

The dialog, depicted here at right, is divided in three tabs: one for the bare calculation, one for setting the fluid properties and the last to review in detail the results of calculation.
The common widgets, always visible outside the tabs, are the combo-box for selection of the fluid + one label that indicate its state (at the top) and one label with the result of calculation (at the bottom).



water

*** LIQUID ***

Scope: Traccia001

Flow (m3/h): 100

Roughness (um): 20

Total length = 1.943 m

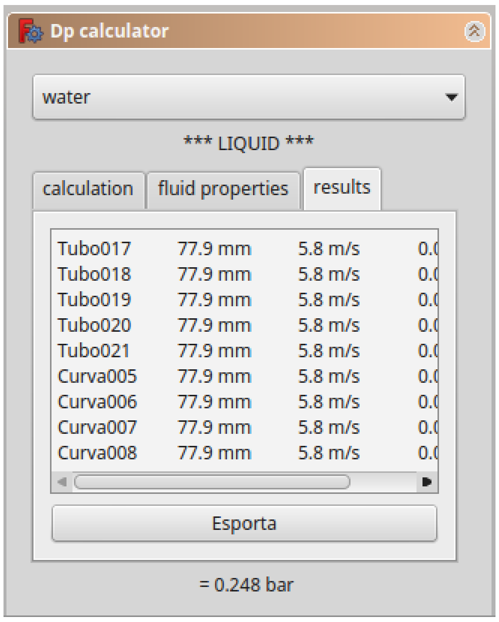
Nr. of curves = 4

= 0.248 bar

In the **calculation** tab you select the scope of calculation from the combo-box: it's possible to select one-by-one the elements in the viewport or directly one complete PypeBranch among those present in the active document.
Then you insert the flowrate and the pipe roughness, press OK and tah-dah: the result is shown at the bottom of the dialog. Also a brief summary of the pipe route is printed at the bottom of the tab.

The calculation is performed using the "fluids" python module provided by Caleb Bell (see at the end of this paragraph) over each pipe and elbow elements + each "pype" element which has defined a non-zero and positive flow factor "Kv" among its properties; this latter is to take into account other concentrated pressure losses or pype-elements (valves, orifices..) that may be created in future.
Also, the calculation is performed with the assumption that fluid can be considered uncompressible, even if the gas/vapour state is selected (that's true if the pressure loss is negligible compared with the pressure of the gas).

To be said, the first time you invoke the dialog, it takes few seconds to import that module due to the large chemicals database included in it... but it's worth!



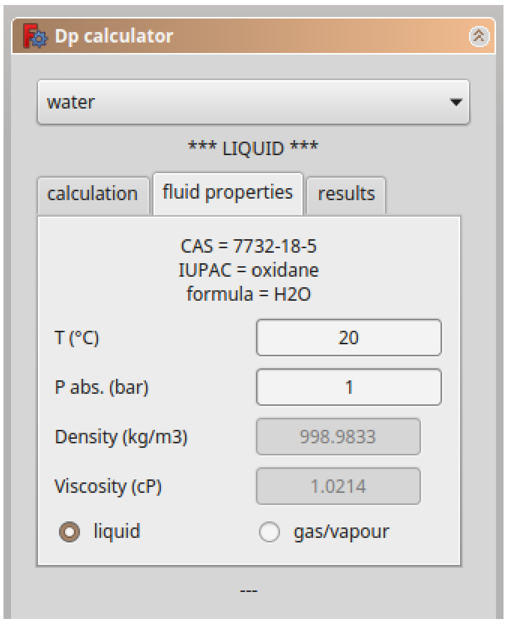
*** LIQUID ***

Tubo017	77.9 mm	5.8 m/s	0.(
Tubo018	77.9 mm	5.8 m/s	0.(
Tubo019	77.9 mm	5.8 m/s	0.(
Tubo020	77.9 mm	5.8 m/s	0.(
Tubo021	77.9 mm	5.8 m/s	0.(
Curva005	77.9 mm	5.8 m/s	0.(
Curva006	77.9 mm	5.8 m/s	0.(
Curva007	77.9 mm	5.8 m/s	0.(
Curva008	77.9 mm	5.8 m/s	0.(

= 0.248 bar

After that you can see the pressure losses related to each element in the **results** tab.
In this tab a push-button also allows to export the data in a .csv file, which you can edit afterwards as a spreadsheet.

In the **fluid properties** tab you can change the condition of fluid in the pipe.

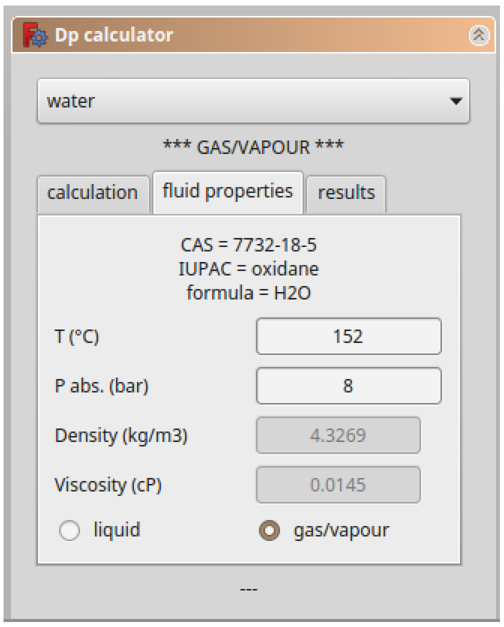


*** LIQUID ***

CAS = 7732-18-5
IUPAC = oxidane
formula = H2O

T (°C): 20
P abs. (bar): 1
Density (kg/m3): 998.9833
Viscosity (cP): 1.0214
[✓ liquid] gas/vapour

For instance, with water you may select a different temperature, pressure and even its state: the Density and Viscosity will change consistently.

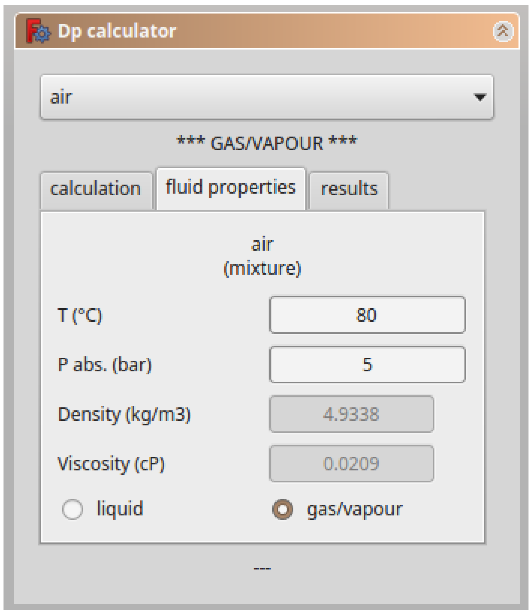


*** GAS/VAPOUR ***

CAS = 7732-18-5
IUPAC = oxidane
formula = H2O

T (°C): 152
P abs. (bar): 8
Density (kg/m3): 4.3269
Viscosity (cP): 0.0145
liquid [✓ gas/vapour]

When the state is changed between liquid and gas/vapour remember that also the units of flowrate will switch from *m3/h* to *kg/hr*.



Dp calculator

air

*** GAS/VAPOUR ***

air
(mixture)

T (°C): 80

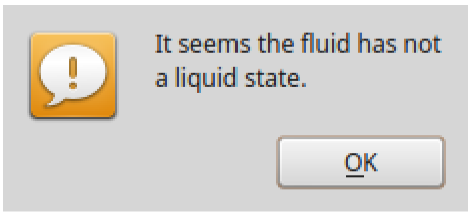
P abs. (bar): 5

Density (kg/m3): 4.9338

Viscosity (cP): 0.0209

liquid [✓ gas/vapour]

Every time the properties are changed, the program makes a check in its database. When the type of fluid is changed (for example to "air") it try to guess the solid state at the pressure and temperature specified.
If there is no data for the fluid at that conditions (for example, liquid air at atmospheric pressure and 20°C) a warning is issued.



It seems the fluid has not a liquid state.

In any case, if the fluid you are looking for is not listed, it's possible to choose <custom fluid> and insert manually the Density and Viscosity values.

A special thank to Mr. Caleb Bell who wrote the python libraries used for this feature.

https://github.com/CalebBell

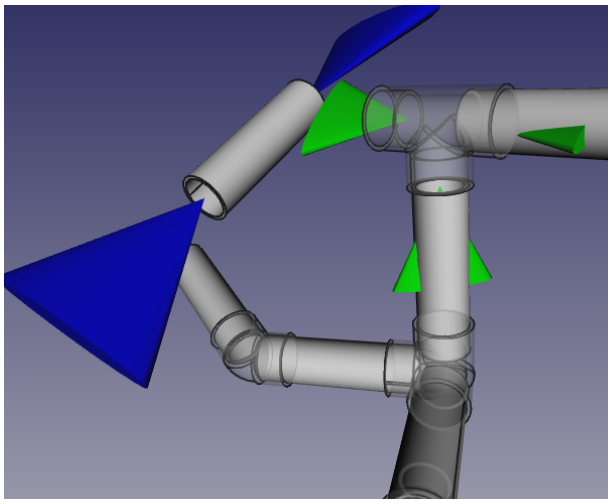
*Remember, before using this tool, to install the python modules **fluids** and **thermo** in your system. That shall be done separately with*

```
$ (sudo) pip (or pip3) install thermo

$ (sudo) pip (or pip3) install fluids
```

QUICK PYPEs JOIN

Also I'd like to thank Mr. Krenzler (https://github.com/rkrenzler) who started a new workbench to create pipe fittings in FreeCAD.
This workbench has the ability to create piping parts that are compatible with those used in flamingo; therefore any tool that apply in flamingo can be used also for the parts of OSE-piping-workbench.
To test this, in flamingo you can find a new tool to join quickly piping parts created with both workbenches. In the example shown at side, the OSE parts are linked with flamingo's pypes using this tool:



1) Select the target object
2) select the target port (green arrows)
3) select the part to move
4) select the port to join (blue arrows)
5) return to point 2) and select a new part and ports or press 'ESC'

BTW: in this folder there is a little cheat-sheet that explains how to create new FeaturePython that can be compatible with the tools of flamingo.