Rhinopiping - User Manual

Introduction

RhinoPiping is a rhino® plugin to create parametric piping networks in contextual 3D.

Plugin installation

⚠️ RhinoPiping is currently only compatible with Rhino 5.

Overview

RhinoPiping exists in two versions, **RhinoPiping LT** and **RhinoPiping Pro**. <u>These two versions are not compatible</u>, so when installing one, the other should be uninstalled beforehand. In both cases, two installers are provided: one for the plugin itself, and one for the provided basic catalog. The two of them need to be installed.

Rhinopiping LT

This is the free limited version. No hidden costs, no activation. The plugin installer can be downloaded at: https://downloads.tomkod.com/ProductFiles/RhinoPiping-LT-1.2.8.zip

Rhinopiping Pro

This is the full featured version, it needs a product key. Shop The plugin installer can be downloaded at: https://downloads.tomkod.com/ProductFiles/RhinoPiping-Pro-1.2.8-WithZoo.zip

Generalities

Command names

RhinoPiping command names follow these general rules:

1. The RhinoPiping commands begin with "RP_" (e.g. *RP_CreatePipeline*)
2. The Catalog Configuration commands begin with "RP_CC_" (e.g. *RP_CC_CreateItem*)
3. The Additional Attribute commands begin with "AA_" (e.g. *AA_Report*)

Since Rhino3D has an auto-completion feature, it can be really useful to remember these.

Capabilities

RhinoPiping lets you flawlessly design a piping system made of any kind of rigid pipes as long as its section is a circle. Our engine, mainly based on connection circles can handle any number of components, each of them having up to eight connection circles.

This system also means that RhinoPiping could be used to design any other system using circular section elements, such as some electrical wiring or sometimes HVAC systems.

Connection circles

When a connection circle is green, it means it is not connected. While adding a piping element to your pipeline, the closest disconnected connection circle is highlighted in yellow, while the selected pipeline is highlighted in white.


When two connection circles are connected, they become blue-colored. In the example below, the butterfly valve is connected to the pipe.
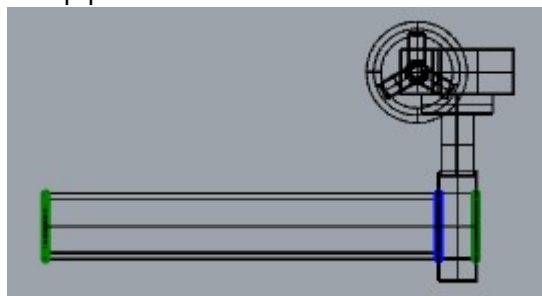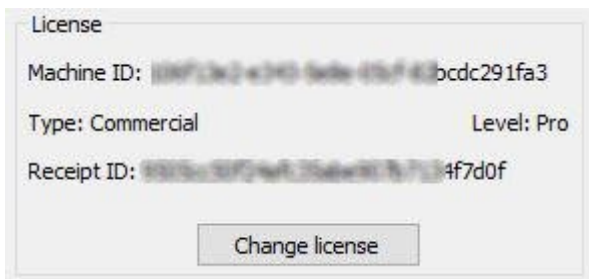

Options

RhinoPiping adds a tab in the rhino options where you can select:

- The nominal size display. Each connection circle has a nominal size and can be only connected with a piping element of the same nominal size. RhinoPiping handles two different display modes. One is named "DN", it is the International standard notation in mm, and the other is named "NPS", which is the Anglo-Saxon equivalent. (For example DN: 32 is equivalent to NPS: 1 ¼)
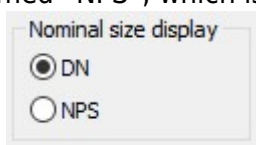
- The license information and activation. This part has been removed after RhinoPiping-1.0.9.

- The RP_Iso3D options, which are specific to the command *RP_Iso3D*
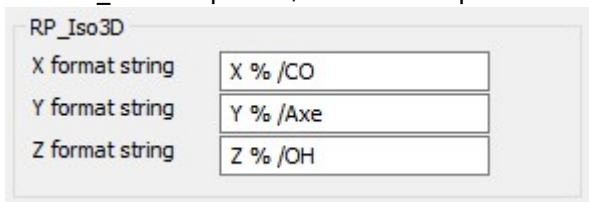
Layers

Every piping element created with RhinoPiping must be located in the layer named "Piping network", or its sub-layers. If the piping layer "Piping network" does not exist, it will be created.

RhinoPiping protection

Some modifications of a pipeline created by RhinoPiping with a rhino3D command are forbidden to protect the piping data, for example:

1. If you try to scale a piping element, you will get this message:
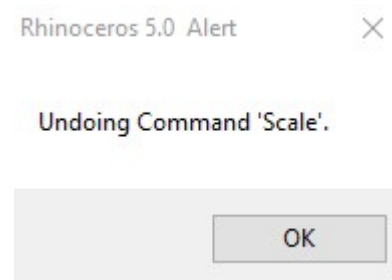
1. If you try to delete/move/rotate (etc.) a piping element alone when its connection circles are connected, it will also be forbidden, and you will get this message:
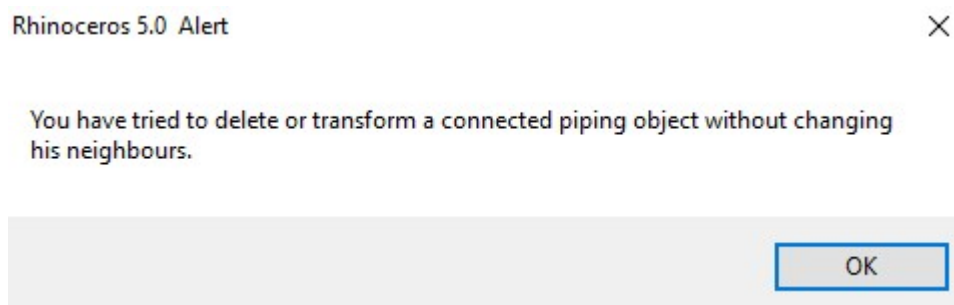
These modifications would have been alright if the piping element hadn't been connected (Try using the command "RP_Disconnect" if needed), or if your selection contains all the connected piping elements.

⚠️ If you open a file with RhinoPiping elements with RhinoPiping uninstalled, it will work but you mustn't modify any element belonging to the "Piping network" layer. (If you do, it can result in file corruptions, and make RhinoPiping crash later)

Additional attributes integration

RhinoPiping is deeply integrated with a free rhino plugin named **AdditionalAttribute**. As its name suggests, *AdditionalAttribute* adds a special tab in rhino3D properties for attributes.

These attributes can be defined in RhinoPiping's catalog (specific to a pipe class for example), calculated, read-only (or not), and a lot more. Special internal attributes will also be displayed, like the nominal diameter of the selected element, its type, and family, etc…
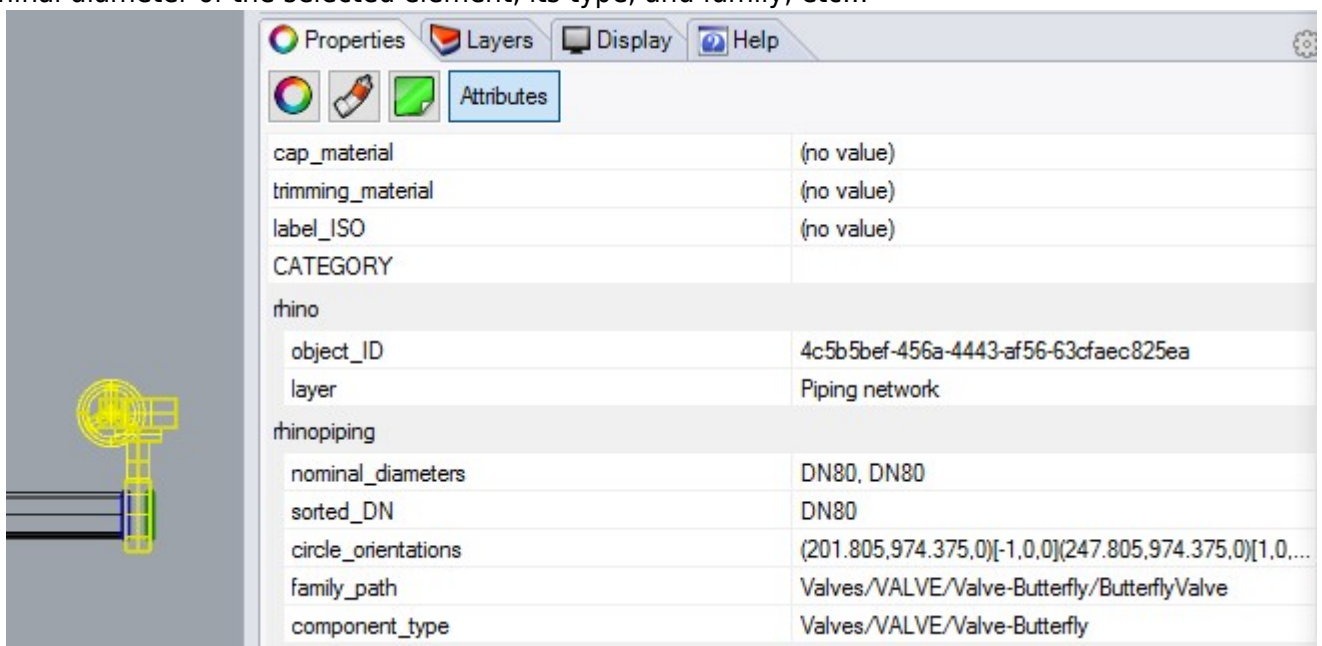
How To 's

How to add a Nominal Diameter in the allowed DN list

Every DN (Nominal Diameter) used in a piping class, an elbow class or a piping component must beforehand be declared in the file nominal-sizes.xml. If this rule is not followed, Rhinopiping won't be able to start when launching Rhinoceros.

This file's path is located in RhinoPiping "catalogs" folder. With windows 7, the default full path is: "C:\ProgramData\Navinn\catalogs\nominal-sizes.xml"

This file has two roles:

Limit the allowed DN to a predefined list Manage the equivalence between the DN (Nominal Diameter) and the NPS (Nominal Pipe Size – American standard) The values must absolutely be ascending. For example, to add the NPS 6 1/2, we must add it between NPS 6 and NPS 7:

```xml
1. <?xml version="1.0"?>
2. <nominal-sizes>
3.  <item NPS="1/8" DN="6"/>
4.  <item NPS="1/4" DN="8"/>
5.  <item NPS="3/8" DN="10"/>
6.  <item NPS="1/2" DN="15"/>
7.  <item NPS="5/8" DN="16"/>
8.  <item NPS="3/4" DN="20"/>
9.  <item NPS="7/8" DN="22"/>
10. <item NPS="1" DN="25"/>
11. <item NPS="1 1/8" DN="28"/>
12. <item NPS="1 1/4" DN="32"/>
13. <item NPS="1 3/8" DN="35"/>
14. <item NPS="1 1/2" DN="40"/>
15. <item NPS="1 5/8" DN="42"/>
16. <item NPS="1 3/4" DN="45"/>
17. <item NPS="1 7/8" DN="48"/>
18. <item NPS="2" DN="50"/>
19. <item NPS="2 1/2" DN="65"/>
20. <item NPS="3" DN="80"/>
21. <item NPS="3 1/2" DN="90"/>
22. <item NPS="4" DN="100"/>
23. <item NPS="4 1/2" DN="110"/>
24. <item NPS="5" DN="125"/>
25. <item NPS="6" DN="150"/>
26. <item NPS="6 1/2" DN="165"/>
27. <item NPS="7" DN="175"/>
28. <item NPS="8" DN="200"/>
29. <item NPS="9" DN="225"/>
30. <item NPS="10" DN="250"/>
31. <item NPS="12" DN="300"/>
32. <item NPS="14" DN="350"/>
33. <item NPS="16" DN="400"/>
34. <item NPS="18" DN="450"/>
35. <item NPS="20" DN="500"/>
36. <item NPS="22" DN="550"/>
37. <item NPS="24" DN="600"/>
38. <item NPS="26" DN="650"/>
39. <item NPS="28" DN="700"/>
40. <item NPS="30" DN="750"/>
41. <item NPS="32" DN="800"/>
42. <item NPS="34" DN="850"/>
43. <item NPS="36" DN="900"/>
44. <item NPS="38" DN="950"/>
```

```
45.  <item NPS="40" DN="1000"/>
46.  <item NPS="42" DN="1050"/>
47.  <item NPS="44" DN="1100"/>
48.  <item NPS="46" DN="1150"/>
49.  <item NPS="48" DN="1200"/>
50.  <item NPS="56" DN="1400"/>
51. </nominal-sizes>
```

How to create a pipe class

Every pipe class definition file are declared in the file pipe-classes.xml

This file is located in RhinoPiping's "catalogs\pipe" folder. With windows 7-10 the default full path is: "C:\ProgramData\Navinn\catalogs\pipe\pipe-classes.xml"

In the example below, lines 3-4 are defining the existing attributes for the pipe classes. The "name" option is mandatory, the "type" option is not mandatory, and the "default" option is not mandatory but highly recommended, because when missing it may prevent the command AA_Report to run as it should.

Lines 6-7 are the respective paths of the secondary XML files where the pipe classes are defined. These files should be located in the same folder as for pipe-classes.xml. Using secondary XML files like this is only to help to get a clean structure, it does not correlate with the classes tree in RhinoPiping.

```
1. <?xml version="1.0"?>
2. <pipe-classes>
3.  <attribute name="lineic_mass" type="lineic-mass"/>
4.  <attribute name="CATALOG:Material" default="N/A"/>
5.
6.  <include path="stainless-steel-ANSI-ASME-B36.19M-1985.xml"/>
7.  <include path="steel-ANSI-ASME-B36.10M-1995.xml"/>
8. </pipe-classes>
```

Now let say we need to add a copper piping class and register it in a new file named "Coper.xml". We will declare it like this:

```
1. <?xml version="1.0"?>
2. <pipe-classes>
3.  <attribute name="lineic_mass" type="lineic-mass"/>
4.  <attribute name="CATALOG:Material" default="N/A"/>
5.
6.  <include path="stainless-steel-ANSI-ASME-B36.19M-1985.xml"/>
7.  <include path="steel-ANSI-ASME-B36.10M-1995.xml"/>
8.  <include path="Copper.xml"/>
9. </pipe-classes>
```

Then we will create a "Copper.xml" file in \Navinn\catalogs\pipe where the class will be registered. We could set every data in this file, but to make it easier, RhinoPiping lets you define the values in a CSV file, called from this XML. let see how it works:

```
 1. <?xml version="1.0"?>
 2. <category name="CopperClass" OD-unit="mm" thickness-unit="mm">
 3.  <pipe-class name="CustomClass/Class1">
 4.   <import-csv path="CustomPipeClass\CopperClass1.csv" separator=";">
 5.    <DN></DN>
 6.    <OD></OD>
 7.    <thickness></thickness>
 8.    <attribute:lineic_mass></attribute:lineic_mass>
 9.    <attribute:CATALOG:Material></attribute:CATALOG:Material>
10.   </import-csv>
11.  </pipe-class>
12. </category>
```

Line 2 defines the category name for this class, and the units used for OD (outside diameters) and Thickness. Line 3 defines the class name and its tree position in RhinoPiping. In the example, the tree is set to: CopperClass / CustomClass / Class1. Line 4 lets us define the path for a .csv file where the values will be. With the option separator, it's also possible to choose the separator for this file. The CSV is imported by columns in the same order as the attributes declared in the XML file. Lines 5 to 7 are mandatory internal attributes: DN (nominal diameter): The theoretical nominal diameter of the pipe. (Can be different with the real diameter). It is possible to use the NPS attribute instead. OD (Outside Diameter): The pipe's real outside diameter thickness: The pipe's real thickness.

The lines 8-9 are non-mandatory custom attributes declared in pipe-classes.xml

This is what the CSV file should look like if we want to only have a copper class with DN32 and DN40:

```
32;35;1.5;1.41 kg/m;Copper
40;42;2;1.61 kg/m;Copper
```

If we wish to add another copper piping class, we could just create a new CSV file and update the XML file like this:

```
 1. <?xml version="1.0"?>
 2. <category name="CopperClass" OD-unit="mm" thickness-unit="mm">
 3.  <pipe-class name="CustomClass/Class1">
 4.   <import-csv path="CustomPipeClass\CopperClass1.csv" separator=";">
 5.    <DN></DN>
 6.    <OD></OD>
 7.    <thickness></thickness>
 8.    <attribute:lineic_mass></attribute:lineic_mass>
 9.    <attribute:CATALOG:Material></attribute:CATALOG:Material>
10.   </import-csv>
11.  </pipe-class>
12.  <pipe-class name="CustomClass/Class2">
13.   <import-csv path="CustomPipeClass\CopperClass2.csv" separator=";">
```

```
14.     <DN></DN>
15.     <OD></OD>
16.     <thickness></thickness>
17.     <attribute:lineic_mass></attribute:lineic_mass>
18.     <attribute:CATALOG:Material></attribute:CATALOG:Material>
19.    </import-csv>
20.   </pipe-class>
21. </category>
```

Here is an example of the file "CopperClass2.csv", with thicker pipes and the DN50 available:

```
32;35;2.5;1.41 kg/m;Copper
40;42;3;1.61 kg/m;Copper
50;53;3;1.8 kg/m;Copper
```

You can download this tutorial's example here.

How to create a pipe class catalog attribute

For this tutorial, we assume you followed the previous one: "Creating a pipe class"

As for the pipe class file's paths, the attributes dedicated to pipe classes are all declared in the file pipe-classes.xml

This file is located in Rhinopiping's folder "catalogs\pipe". With windows 7 the default full path is: "C:\ProgramData\Navinn\catalogs\pipe\pipe-classes.xml"

In the previous example, two attributes were already defined:

An attribute lineic-mass is typed to define a lineic mass. An attribute CATALOG:Material. The use of ":" is a way to define an attribute's tree position. In RhinoPiping we will see an attribute Material under a group CATALOG.

```
1. <?xml version="1.0"?>
2. <pipe-classes>
3.   <attribute name="lineic_mass" type="lineic-mass"
   default="N/A"></attribute>
4.   <attribute name="CATALOG:Material" default="N/A"></attribute>
5.   <include path="stainless-steel-ANSI-ASME-B36.19M-1985.xml"></include>
6.   <include path="steel-ANSI-ASME-B36.10M-1995.xml"></include>
7.   <include path="Copper.xml"></include>
8. </pipe-classes>
```

To add an attribute Provider in the group CATALOG, we must add a line in the file:

```
1. <?xml version="1.0"?>
```

```
2. <pipe-classes>
3.   <attribute name="lineic_mass" type="lineic-mass"
   default="N/A"></attribute>
4.   <attribute name="CATALOG:Material" default="N/A"></attribute>
5.   <attribute name="CATALOG:Provider" default="N/A"></attribute>
6.   <include path="stainless-steel-ANSI-ASME-B36.19M-1985.xml"></include>
7.   <include path="steel-ANSI-ASME-B36.10M-1995.xml"></include>
8.   <include path="Copper.xml"></include>
9. </pipe-classes>
```

Now if we restart Rhinoceros, in the attributes tab we will see our attributes as defined:

Nevertheless, our new attribute Provider always has its default value (here "N/A"), since we did not define its values for the different pipe classes. To customize the providers for the two copper pipe classes we created in the previous tutorial, we must update the file "copper.xml" in \Navinn\catalogs\pipe , then add a column in the corresponding CSV files:

```
1. <?xml version="1.0"?>
2. <category name="CopperClass" OD-unit="mm" thickness-unit="mm">
3.     <pipe-class name="CustomClass/Class1">
4.         <import-csv path="CustomPipeClass\CopperClass1.csv"
   separator=";">
5.             <DN></DN>
6.             <OD></OD>
7.             <thickness></thickness>
8.             <attribute:lineic_mass></attribute:lineic_mass>
9.             <attribute:CATALOG:Material></attribute:CATALOG:Material>
10.            <attribute:CATALOG:Provider></attribute:CATALOG:Provider>
11.         </import-csv>
12.    </pipe-class>
```

```
13.    <pipe-class name="CustomClass/Class2">
14.         <import-csv path="CustomPipeClass\CopperClass2.csv"
    separator=";">
15.              <DN></DN>
16.              <OD></OD>
17.              <thickness></thickness>
18.              <attribute:lineic_mass></attribute:lineic_mass>
19.              <attribute:CATALOG:Material></attribute:CATALOG:Material>
20.              <attribute:CATALOG:Provider></attribute:CATALOG:Provider>
21.         </import-csv>
22.    </pipe-class>
23. </category>
```

Our two classes CSV files will have to be modified accordingly:

```
32;35;1.5;1.41 kg/m;Copper;unimat
40;42;2;1.61 kg/m;Copper;unimat
```

```
32;35;2.5;1.41 kg/m;Copper;unimat
40;42;3;1.61 kg/m;Copper;unimat
50;53;3;1.8 kg/m;Copper;altitube
```

After restarting Rhinoceros, the attribute Provider now displays the appropriate values. You can download this tutorial example here.

How to create an elbow class

RhinoPiping can use two different kinds of elbows:

1. The components owning the stereotype "elbow".
   They are created like any other components, so each of them has to be drawn and created manually, for each ND (Nominal Diameter) needed. This can be a bother, but gives us two interesting points:
   - The component-type elbows are compatible with every pipe class from a software point of view. Once you created them, you can use them with any pipe classes. You will also be able to insert them with the "RP_AddComponent" command, as for any other end of line components.
   - Since this kind of elbow is designed manually, it can have any form-factor and size. We can make it longer just on one side, or take into account the overlap needed if it is screwed on the pipe for example.
2. The elbow classes. Less flexible, they have the huge advantage to be very fast to set up. In this tutorial, we will create two different elbow classes for the two pipe classes created in the previous tutorials.

Every elbow class is defined in only one file named elbow-classes.xml.

This file is located in RhinoPiping's folder "catalogs\elbow". With windows 7 the default full path is: "C:\ProgramData\Navinn\catalogs\elbow\elbow-classes.xml"

The elbow classes can be defined in two different ways: from a curvature factor, or from curvature values. An elbow class is only compatible with one pipe class but a pipe class can have several elbow classes.

Example of a curvature factor pipe class:

For each factor needed, we add a line with these properties:

- "key": the elbow class name
- "factor": The curvature factor (It will be multiplicated by the diameter)
- "pipe-class": The associated pipe class name and tree path

```
1. <CurvatureFactors>
2. <item key = "D1" factor = "1" pipe-
   class="CopperClass/CustomClass/Class1"></item>
3. <item key = "D1.5" factor = "1.5" pipe-
   class="CopperClass/CustomClass/Class1"></item>
4. <item key = "D2" factor = "2" pipe-
   class="CopperClass/CustomClass/Class1"></item>
5. <item key = "D3" factor = "3" pipe-
   class="CopperClass/CustomClass/Class1"></item>
6. </CurvatureFactors>
```

Example of a tabulated curvature pipe class:

For each DN (Nominal diameter) used in each class, we define the curvature radius.

```
1. <Standard name="PieceOfCake">
2.       <Class name="3D" pipe-class="CopperClass/CustomClass/Class2">
3.       <item DN="32" curvature="47.5"></item>
4.       <item DN="40" curvature="57"></item>
5.       <item DN="50" curvature="76"></item>
6.       </Class>
7. </Standard>
```

When updated, our elbow-classes.xml file should look like this:

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <angle-elements>
3.    <CurvatureFactors>
4.      <item key = "D1" factor = "1" pipe-class="stainless-steel-ANSI-
   ASME-B36.19M-1985/Schedule-5S"></item>
5.      <item key = "D1" factor = "1" pipe-class="stainless-steel-ANSI-
   ASME-B36.19M-1985/Schedule-10S"></item>
6.      <item key = "D1" factor = "1" pipe-class="stainless-steel-ANSI-
```

```
   ASME-B36.19M-1985/Schedule-40S"></item>
7.    <item key = "D1" factor = "1" pipe-class="stainless-steel-ANSI-
   ASME-B36.19M-1985/Schedule-80S"></item>
8.    <item key = "D1" factor = "1" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Standard"></item>
9.    <item key = "D1" factor = "1" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Extra-Strong"></item>
10.    <item key = "D1" factor = "1" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Double-Extra-Strong"></item>
11.    <item key = "D1" factor = "1" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-10"></item>
12.    <item key = "D1" factor = "1" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-20"></item>
13.    <item key = "D1" factor = "1" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-30"></item>
14.    <item key = "D1" factor = "1" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-40"></item>
15.    <item key = "D1" factor = "1" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-60"></item>
16.    <item key = "D1" factor = "1" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-80"></item>
17.    <item key = "D1" factor = "1" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-100"></item>
18.    <item key = "D1" factor = "1" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-120"></item>
19.    <item key = "D1" factor = "1" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-140"></item>
20.    <item key = "D1" factor = "1" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-160"></item>
21.    <item key = "D1.5" factor = "1.5" pipe-class="stainless-steel-ANSI-
   ASME-B36.19M-1985/Schedule-5S"></item>
22.    <item key = "D1.5" factor = "1.5" pipe-class="stainless-steel-ANSI-
   ASME-B36.19M-1985/Schedule-10S"></item>
23.    <item key = "D1.5" factor = "1.5" pipe-class="stainless-steel-ANSI-
   ASME-B36.19M-1985/Schedule-40S"></item>
24.    <item key = "D1.5" factor = "1.5" pipe-class="stainless-steel-ANSI-
   ASME-B36.19M-1985/Schedule-80S"></item>
25.    <item key = "D1.5" factor = "1.5" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Standard"></item>
26.    <item key = "D1.5" factor = "1.5" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Extra-Strong"></item>
27.    <item key = "D1.5" factor = "1.5" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Double-Extra-Strong"></item>
28.    <item key = "D1.5" factor = "1.5" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-10"></item>
29.    <item key = "D1.5" factor = "1.5" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-20"></item>
30.    <item key = "D1.5" factor = "1.5" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-30"></item>
```

```
31.    <item key = "D1.5" factor = "1.5" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-40"></item>
32.    <item key = "D1.5" factor = "1.5" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-60"></item>
33.    <item key = "D1.5" factor = "1.5" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-80"></item>
34.    <item key = "D1.5" factor = "1.5" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-100"></item>
35.    <item key = "D1.5" factor = "1.5" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-120"></item>
36.    <item key = "D1.5" factor = "1.5" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-140"></item>
37.    <item key = "D1.5" factor = "1.5" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-160"></item>
38.    <item key = "D2" factor = "2" pipe-class="stainless-steel-ANSI-
   ASME-B36.19M-1985/Schedule-5S"></item>
39.    <item key = "D2" factor = "2" pipe-class="stainless-steel-ANSI-
   ASME-B36.19M-1985/Schedule-10S"></item>
40.    <item key = "D2" factor = "2" pipe-class="stainless-steel-ANSI-
   ASME-B36.19M-1985/Schedule-40S"></item>
41.    <item key = "D2" factor = "2" pipe-class="stainless-steel-ANSI-
   ASME-B36.19M-1985/Schedule-80S"></item>
42.    <item key = "D2" factor = "2" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Standard"></item>
43.    <item key = "D2" factor = "2" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Extra-Strong"></item>
44.    <item key = "D2" factor = "2" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Double-Extra-Strong"></item>
45.    <item key = "D2" factor = "2" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-10"></item>
46.    <item key = "D2" factor = "2" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-20"></item>
47.    <item key = "D2" factor = "2" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-30"></item>
48.    <item key = "D2" factor = "2" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-40"></item>
49.    <item key = "D2" factor = "2" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-60"></item>
50.    <item key = "D2" factor = "2" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-80"></item>
51.    <item key = "D2" factor = "2" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-100"></item>
52.    <item key = "D2" factor = "2" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-120"></item>
53.    <item key = "D2" factor = "2" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-140"></item>
54.    <item key = "D2" factor = "2" pipe-class="steel-ANSI-ASME-
   B36.10M-1995/Schedule-160"></item>
55.    <item key = "D3" factor = "3" pipe-class="stainless-steel-ANSI-
   ASME-B36.19M-1985/Schedule-5S"></item>
56.    <item key = "D3" factor = "3" pipe-class="stainless-steel-ANSI-
```

```
            ASME-B36.19M-1985/Schedule-10S"></item>
57.         <item key = "D3" factor = "3" pipe-class="stainless-steel-ANSI-
            ASME-B36.19M-1985/Schedule-40S"></item>
58.         <item key = "D3" factor = "3" pipe-class="stainless-steel-ANSI-
            ASME-B36.19M-1985/Schedule-80S"></item>
59.         <item key = "D3" factor = "3" pipe-class="steel-ANSI-ASME-
            B36.10M-1995/Standard"></item>
60.         <item key = "D3" factor = "3" pipe-class="steel-ANSI-ASME-
            B36.10M-1995/Extra-Strong"></item>
61.         <item key = "D3" factor = "3" pipe-class="steel-ANSI-ASME-
            B36.10M-1995/Double-Extra-Strong"></item>
62.         <item key = "D3" factor = "3" pipe-class="steel-ANSI-ASME-
            B36.10M-1995/Schedule-10"></item>
63.         <item key = "D3" factor = "3" pipe-class="steel-ANSI-ASME-
            B36.10M-1995/Schedule-20"></item>
64.         <item key = "D3" factor = "3" pipe-class="steel-ANSI-ASME-
            B36.10M-1995/Schedule-30"></item>
65.         <item key = "D3" factor = "3" pipe-class="steel-ANSI-ASME-
            B36.10M-1995/Schedule-40"></item>
66.         <item key = "D3" factor = "3" pipe-class="steel-ANSI-ASME-
            B36.10M-1995/Schedule-60"></item>
67.         <item key = "D3" factor = "3" pipe-class="steel-ANSI-ASME-
            B36.10M-1995/Schedule-80"></item>
68.         <item key = "D3" factor = "3" pipe-class="steel-ANSI-ASME-
            B36.10M-1995/Schedule-100"></item>
69.         <item key = "D3" factor = "3" pipe-class="steel-ANSI-ASME-
            B36.10M-1995/Schedule-120"></item>
70.         <item key = "D3" factor = "3" pipe-class="steel-ANSI-ASME-
            B36.10M-1995/Schedule-140"></item>
71.         <item key = "D3" factor = "3" pipe-class="steel-ANSI-ASME-
            B36.10M-1995/Schedule-160"></item>
72.         <item key = "D1" factor = "1" pipe-
            class="CopperClass/CustomClass/Class1"></item>
73.         <item key = "D1.5" factor = "1.5" pipe-
            class="CopperClass/CustomClass/Class1"></item>
74.         <item key = "D2" factor = "2" pipe-
            class="CopperClass/CustomClass/Class1"></item>
75.         <item key = "D3" factor = "3" pipe-
            class="CopperClass/CustomClass/Class1"></item>
76.     </CurvatureFactors>
77.
78.     <TabulatedCurvatures>
79.       <Standard name="VALLOUREC">
80.         <Class name="3D" pipe-class="stainless-steel-ANSI-ASME-
            B36.19M-1985/Schedule-10S">
81.           <item DN="20" curvature="28.5"></item>
82.           <item DN="25" curvature="38"></item>
83.           <item DN="32" curvature="47.5"></item>
84.           <item DN="40" curvature="57"></item>
```

```
 85.            <item DN="50" curvature="76"></item>
 86.            <item DN="65" curvature="95"></item>
 87.            <item DN="80" curvature="114.5"></item>
 88.            <item DN="100" curvature="152.5"></item>
 89.            <item DN="125" curvature="190.5"></item>
 90.            <item DN="150" curvature="228.5"></item>
 91.            <item DN="200" curvature="305"></item>
 92.            <item DN="250" curvature="381"></item>
 93.            <item DN="300" curvature="457"></item>
 94.            <item DN="350" curvature="533.5"></item>
 95.            <item DN="400" curvature="609.5"></item>
 96.        </Class>
 97.      </Standard>
 98.      <Standard name="PieceOfCake">
 99.      <Class name="3D" pipe-class="CopperClass/CustomClass/Class2">
100.            <item DN="32" curvature="47.5"></item>
101.            <item DN="40" curvature="57"></item>
102.            <item DN="50" curvature="76"></item>
103.      </Class>
104.      </Standard>
105.    </TabulatedCurvatures>
106. </angle-elements>
```

Caution! To create an elbow component, it can be faster to create first an automatic elbow, then explode it in Rhino3D and make it a component.

You can download this tutorial's example here.

How to create global attributes settings

See the demo here (youtube)

How to change an object color to match an attribute value

It is possible to change quite easily some Rhino/RhinoPiping objects color to match a specific attribute. In our example, we with to have an attribute "Status", with a choice between the value Built and the value Todo.

Creating the attribute

In default-attribute-spec.xml we must add:

```
 1. <enum name = "STATUS_ENUM">
 2.            <literal value = "Built"></literal>
 3.            <literal value = "ToDo"></literal>
 4. </enum>
```

```
5. <attribute name = "Status" type = "STATUS_ENUM" read-only = "false">
6.    <description>Status of the pipeline</description>
7. </attribute>
```

Rhinoscript code

We will write the script as shown below, then run the command "ChangeColors" every time we want to update the colors.

```
 1. Option Explicit
 2. 'Script written by Matthieu Arnold
 3. 'Script copyrighted by NAVINN
 4. 'Script version Thursday, November 27, 2014 5:28:33 PM
 5.
 6. Rhino.Print "-------------------------------------------"
 7. Rhino.AddAlias "ChangeColors", "_-RunScript (ChangeColors)"
 8.
 9. Call ChangeColors()
10. Dim MyAttribute
11.
12. Sub ChangeColors
13.     Dim arrObjects, strObject
14.     arrObjects = Rhino.AllObjects
15.     If IsArray(arrObjects) Then
16.         For Each strObject In arrObjects
17.             Rhino.Print strObject
18.             'Rhino.SelectObject strObject
19.             If VarType(strObject) = vbString Then
20.                 MyAttribute = Rhino.GetUserText(strObject, "Status")
21.                 If IsNull(MyAttribute) = False Then
22.                     Rhino.Print MyAttribute
23.                     If MyAttribute = "Built" Then
24.                         Rhino.ObjectColor strObject, RGB(0, 255, 0)
25.                     Else
26.                         Rhino.ObjectColor strObject, RGB(255, 0, 0)
27.                     End If
28.                 End If
29.             End If
30.         Next
31.     End If
32. End Sub
```

How to create a template file for the command AA_Report

The template file is formated like a kind of .csv with a .tpl extension. It can be opened with a text

editor like notepad.

When you create your template file, you must follow those rules:

- The separator must be ;
- If you are using a spreadsheet program like Excel for edition, check it won't add double-quotes.
- Make sure all the attributes you are referring to have a default value. If they don't, the command AA_Report will stop when meeting an attribute with no value. [1]
- A "cell" can contain text, attribute values, or both.

Example

Let's build a typical example of a template file to generate a Bill of Materials. Create a text file and rename it for example "BillOfMaterials.tpl". The template file will be read from top to bottom when `AA_Report` is running. So you may wish to begin with a simple line of texts for columns titles:

```
Label_ISO ;DN;Part number;Mass;Lineic mass;Material
```

The attributes values will be called like that: (example with label_ISO)

```
$(attribute:label_ISO)
```

We can look for different attributes for each kind of population. This is nearly always necessary since all piping elements don't have the same attributes (for example only the elbows have a curvature attribute). To separate the population, we use filters. The most common way would be to separate pipes, elbows, and others:

```
$(filter:population:Pipe)$(attribute:label_ISO);;;;;
$(end-filter)
$(filter:population:Elbow)$(attribute:label_ISO);;;;;
$(end-filter)
$(filter:population:ComponentNode:)$(attribute:label_ISO);;;;;
$(end-filter)
```

As you can see, the line begins with the filter, and there is no separator before the first column definition. (We take the following line to end the filter.) The population:Elbow stands for the automatic elbows (not the components). If you leave nothing between two separators, it will leave a blank cell in the Bill Of Material. In the example above, `AA_Report` will create a CSV file with only the labels in the first column. Now we can complete it and add the line for column titles:

```
Label_ISO ;DN;Mass;Lineic mass;Material
$(filter:population:Pipe)$(attribute:label_ISO);$(attribute:rhinopiping:sort
ed_DN);$(attribute:CATALOG:Weight);$(attribute:lineic_mass);$(attribute:CATA
LOG:Material);
$(end-filter)
$(filter:population:Elbow)$(attribute:label_ISO);$(attribute:rhinopiping:sor
ted_DN);;;$(attribute:CATALOG:Material);
$(end-filter)
$(filter:population:ComponentNode:)$(attribute:label_ISO);$(attribute:rhinop
iping:sorted_DN);$(attribute:CATALOG:Weight);;$(attribute:CATALOG:Material);
```

```
$(end-filter)
```

How to remove a component with the free LT edition

As you may have noticed, the command *RP_RemoveComponent* is not available for RhinoPiping LT free licenses. You can still remove a component with one more step:

1. Disconnect the component with the command: *RP_Disconnect*
2. Delete the component with the command: *_Delete*
3. Re-draw the missing pipe portion with the command: *RP_CreatePipeline*

How to replace an elbow with the free LT edition

As you may have noticed, the command *RP_ChangeElbows* is not available for RhinoPiping LT free licenses. You can still remove a component with one more step:

1. Disconnect the elbow with the command: *RP_Disconnect*
2. Delete the component with the command: *_Delete*
3. Re-draw the missing pipe portion with the command *RP_CreatePipeline.* While using the command select the new elbow class.

Commands (LT and Pro editions)

RP_CreatePipeline

This one is RhinoPiping's most important command since it is the one with which you are going to draw your pipelines with.

Usage

1. Run the command *RP_CreatePipeline*
2. Pick the vertices of the polyline which will be pipeline's axis,
3. Hit Enter (or right-click) to end the pipeline.

Snaps

If the Rhinoceros® snap "Point" is enabled, the command can:

1. Extend an existing pipeline.
2. Connect to an existing pipeline (these snaps are only defined after the second vertex picking)
3. Start from tapping into an existing pipeline (these snaps disappear after the first vertex is

picked)
4. End with tapping into an existing pipeline.

It may occur that RhinoPiping has difficulties connecting to an existing pipeline if too many snaps are enabled. Depending on the case, you may need to temporarily disable every snap except "Point".

Pre-selection

If pipelines are pre-selected when the command starts, the snaps for connection or tapping are built only for the pre-selected elements.

Consistency

RhinoPiping always makes sure every piping element is compatible with the others in the same pipeline. When you change an option, it might change others. For example, if you switch from DN40 to DN50: if the current pipe class doesn't have DN50, the pipe class should change automatically.

A pipeline can only contain one pipe class with compatible elbow classes. (You won't be able to change the pipe class if you try to connect to an existing pipeline). The components are not associated with a specific pipe class. (You may use them with every pipe class).

Options

- Offset

When this option is enabled, the pipe will be drawn with an offset. Through the *OffsetParam* menu, you can choose its *axis* (X, Y, Z) and *gap* (Distance between the picked points and the pipe's side).

- Angle

This option lets you define if the pipeline's angles should be locked to a list of specific angles or free.
 AngleValues Let you register a list of allowed angles in degrees. This option is active only when *Angle*=Locked.

> ⚠️ When using ortho mode, if you removed 90 from the parameter AngleValues, it may happen that no angle proposed by the component family satisfies the constraints induced by the ortho mode. In that case, Rhino Piping won't propose any elbow.

- Draw

This option configures the dynamic display of future pipeline during the vertices picking. You can choose between:

1. Display axes (value *Axes*),
2. Display pipes and elbows geometric forms (value *PipesAndBents*).

- Elbows

This option lets you change the elbows class of the pipeline. It can be changed after several vertices are picked.

> ⚠ When using ortho mode, if the elbows class is based on a component family, it may happen that no angle proposed by the component family satisfies the constraints induced by the ortho mode. In that case, Rhino Piping won't propose any elbow.

- Guide

If the value is Keep, the axis polyline of the pipeline is kept after the command. The default value is Destroy.

- Nominal Diameter

This option lets you choose the nominal diameter of the pipeline. This option is disabled when the pipeline starts with a continuation or a tapping.

- Pipes

This option lets you change the pipes class of the pipeline. The pipes class can still be changed after several vertices are picked. This option is disabled when the pipeline starts with a continuation or a tapping.

- Snap

This option lets you configure specific piping snaps. You can allow or forbid:

1. Free points,
2. Continuations[2],
3. Tappings[3],
4. Bypasses[4]

- Tappings

This option lets you change the tapping class of the pipeline.

RP_AddComponent

Presentation

This command adds a component to the model. The component can be inserted:

1. At any position
2. At a pipeline's end
3. Into a pipe (if the component owns two coaxial connection circles).

The component is chosen among the catalog. The choice is limited to the components compatible with the context (selected pipeline…).

Usage

1. Run the command *RP_AddComponent*
2. Choose the pipeline into which you want to insert the component (right-click if you don't want to choose a pipeline)
3. Choose the family of the component you want to insert
4. If no pipeline has been chosen, choose which version you want to insert
5. Click to insert the component.

Pre-selection

If a pipeline is selected before running the command, it will be used as command context (the step of the pipeline's choice is skipped).

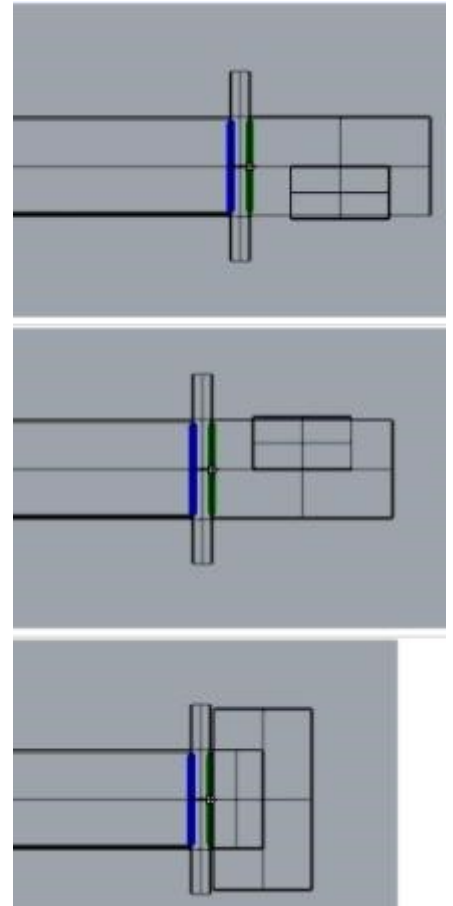## Options

- ChangeComponent

This option allows changing the component to add. It is proposed when several components of the same family can be added. (For example when inserting tees)

> Please note that the current component's diameter(s) is displayed in the state bar.

- ShiftCircle

Permutes the connection circles to choose which circle on the new
component is connected to the end of the pipeline.

> ℹ️ This option is not considered when inserting the component
> into a pipe.

- Swap

Permutes the connection circles to choose which circle is placed at the mouse cursor. This option
applies when inserting a component into a pipe, not when connecting to the end of the line.

- TurnOver

This option turns the component over.

RP_Slide

This command slides a component along its pipeline.

Usage

1. Run the command *RP_Slide*

2. Choose the component you want to slide
3. Pick the new position.

Pre-selection

If a component is pre-selected, this component will be transformed.

Options

- Swap

Permutes the connection circles to choose which circle is placed at the mouse cursor. This option applies when inserting a component into a pipe, not when connecting to the end of the line.

RP_Connect

This command connects two pipelines or two piping elements

Usage

1. Run the command *RP_Connect*
2. Choose the first end to connect,
3. Choose the second end to connect.
4. If the two connection circles were identical, they are just connected and become highlighted in blue. If not, then first selected will be moved with all its piping line, and then connected.

Pre-selection

Pre-selection is ignored.

Options

- ConnectAllMatchingCircles

When selected, Rhino Piping will check every circle on the active document, and connect them automatically if they are matching.
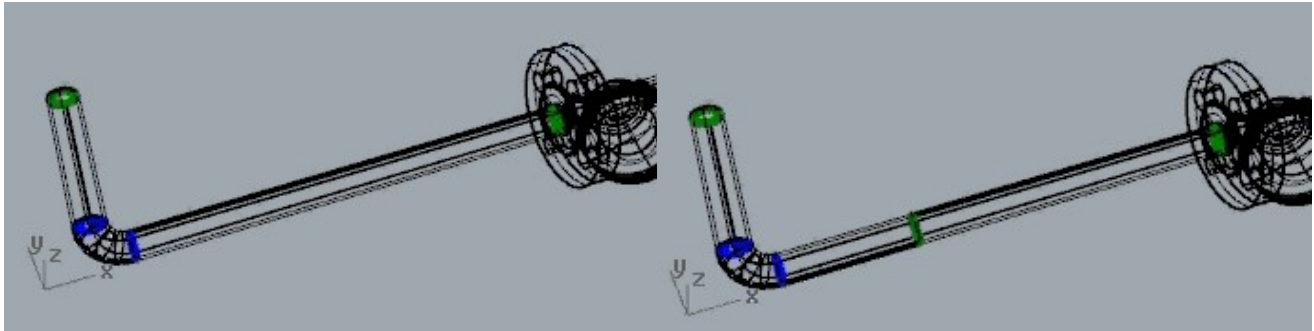
RP_Disconnect

This command disconnects the selected piping element from its pipeline. The pre-selection is canceled if several piping elements were selected.

RP_Cut

This command lets you cut a pipe. It becomes then two different pipes, with a green (disconnected) connection circle between them.

RP_Replace

This command replaces a component with any other one compatible in shape and size. If components are pre-selected, the command will replace, among these components, the ones of the chosen family. (RhinoPiping will then propose to select which family should be replaced)

RP_Rotate

This command rotates an element around the axis of one of its connection circles. It can also be used to rotate a larger part of a pipeline.

Usage

1. Run the command *RP_Rotate*
2. Choose the element to rotate (The highlighted circle defines the rotation's axis.)
3. Pick a reference point or type the rotation's angle (The angle value must be expressed in degrees. If you type the angle value, the following step is skipped.)
4. Pick reference point's new position.
5. The rotation is applied to the element. If the new position of a previously connected circle is no longer compatible with the connection, the connection circle is disconnected.

Pre-selection

By default, you choose among all elements of the model. If elements are pre-selected, you choose among the selection.

Options

- RotateNeighbours

If the value is "True", the transformation is also applied to the neighbors of the transformed element. This option is disabled if the neighbors cannot follow the transformation. (It's the case when the transformed element is on a loop.)

RP_TurnAround

This command turns a component around an axis while keeping the connections. You just have to run the command, choose the component, and pick the symmetry's axis.

Commands (Pro edition only)

RP_ChangeElbows

This command replaces an elbow (or several) with any other compatible elbow.

RP_ReplaceDiameters

This command changes the diameter of a pipeline. Every piping element of the selected pipeline, including components, will be replaced. You will only be able to select compatible diameters. (For example, if a valve included in the selected pipeline only exists for DN40 and DN50, you will only be able to choose between these two sizes.)

RP_Stretch

This command lets you stretch/shorten one pipe or several parallel pipes.

Usage

1. Run the command *RP_Stretch*
2. Choose the pipe to stretch
3. (optionally) Choose other parallel pipes (In some cases (loop), this step is mandatory.)
4. Pick a reference point
5. Pick the new position of reference point, or type translation value

Options

- Roll

Switch the unchanged and transformed ends.

RP_Mirror

This command creates the symmetry of a pipeline portion. The created components are the same as the original ones, only their position is symmetrical. The pre-selection is taken into account.
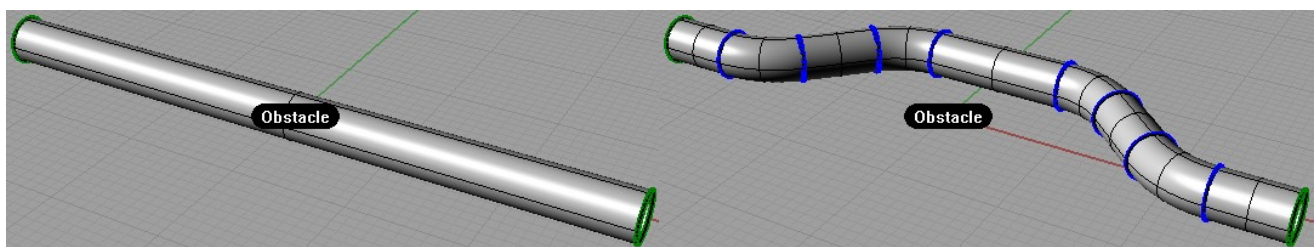
Usage

1. Run the command *RP_Mirror*
2. Select a pipeline or part of it (The *completion* option might help)
3. Choose the mirror plane.
4. Validate

Options

- Copy: This option is available only if you selected the whole pipeline. If you select yes, the original line is kept, else it is removed.
- Completion: This option helps with the selection:
  - *NoCompletion* (Default): you will select components and pipes one by one ▯ *CoaxialSection*: you will select coaxial sections of your pipeline one by one
  - *SameDNsection*: Your selection will be extended across the pipeline until there is a change of nominal diameter
  - *WholePipeline*: The whole pipeline will be selected
- Mirror plane:
  - Default: Select two points within the current CPlane
  - 3points: Define a custom plane
  - XAxis: Symmetry based on X axe
  - YAxis: Symmetry based on Y axe

RP_Avoidance

This command builds an avoidance on a pipe, in order to avoid an obstacle for example.

Usage

1. Run the command *RP_Avoidance*
2. Choose the pipe on which the avoidance is to be built. This step also determines avoidance's orientation: when the "Pick" option is not set to *Middle*, the first elbow will be on the same side as the closest connection circle of chosen point.
3. Pick the first point on the pipe's axis. Depending on the "Pick" Option, it can be the Middle point or the first elbow position (vertex).
4. Pick the end point.

Options

- Elbows

This option lets you change the elbow class of the pipeline. The elbows class can be changed after several vertices are picked.

> ⚠ When using ortho mode, if the elbows class is based on a component family, it may happen that no angle proposed by the component family satisfies the constraints induced by the ortho mode. In that case, Rhino Piping doesn't propose any elbow.

- Angle

This option lets you define if the pipeline's angles should be locked to a list of specific angles or free.
⬚ AngleValues Let you register a list of allowed angles in degrees. This option is active only if Angle=Locked.

> ⚠ When using ortho mode, if you removed 90 from the parameter 'AngleValues', it may happen that no angle proposed by the component family satisfies the constraints induced by the ortho mode. In that case, Rhino Piping won't propose any elbow.

- Depth

This option lets you enter on the keyboard the avoidance's depth.

- Length

This option lets you enter on the keyboard the avoidance's length.

- Pick

This option lets you choose if the first selected point will be the centered vertex or the side vertex of the avoidance.

RP_RemoveComponent

This command deletes a component and tries to redraw a pipeline instead. Just launch the command, select the component and validate to remove it. Pre-selection is taken into account.
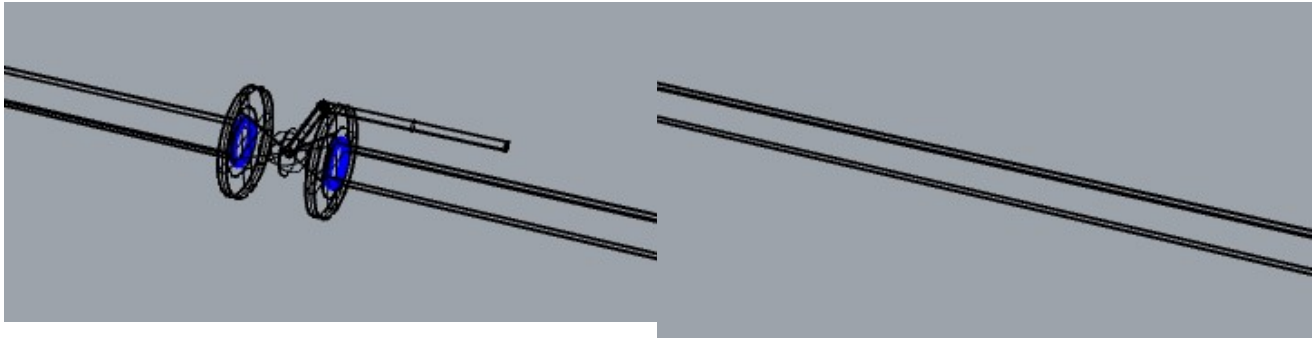
⚠️ An error will occur if the pipes connected to the removed components are not from the same pipe class. This is due to a limitation of the components: as their class is not checked, you may connect them with several non-compatible pipe classes. In this case, the *RP_RemoveComponent* command would try to connect two pipe classes that are not supposed to be compatible.

Commands for attributes

AA_Get Value

This command lets you read an attribute value for the selected rhino or RhinoPiping element (s), and display it in the command results. If the attribute path is left empty, every attribute will be read.

AA_SetValue

This command lets you set the value for a specific attribute. Several elements may be changed at once.

[1)]

Warning: Please remember that despite the attribute Label_ISO being created in the selected items when you run the command RP_Iso3D it is not included in the default attribute spec. Find the file `default-attribute-spec.xml` and add an entry point for the attribute Label_ISO with a default value to guarantee it will always have a value for the Bill of Materials. For example:

```
<attribute name = "label_ISO" default = "none"/>
```

2)

connections at pipeline start or end

3)

at pipeline start or end

4)

double-tapping linking two pipes

From:
https://wiki.tomkod.com/ - **TomKod knowledge base**

Permanent link:
https://wiki.tomkod.com/doku.php?id=start:plugins:rhinopiping:rp-usermanual&rev=1636032699

Last update: **2021/11/04 14:31**