Tips’n’tricks

Perkins + Will

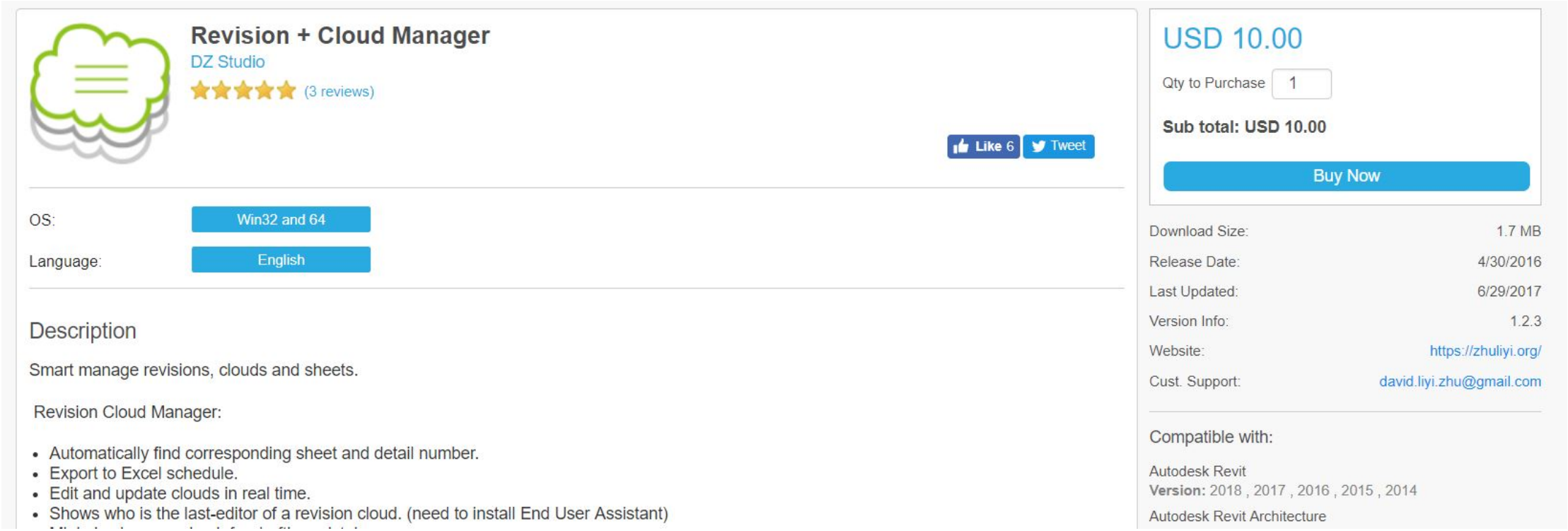
26/07/2017

Manage Revision Clouds With Dynamo

Mauro Sabiu - Perkins+Will

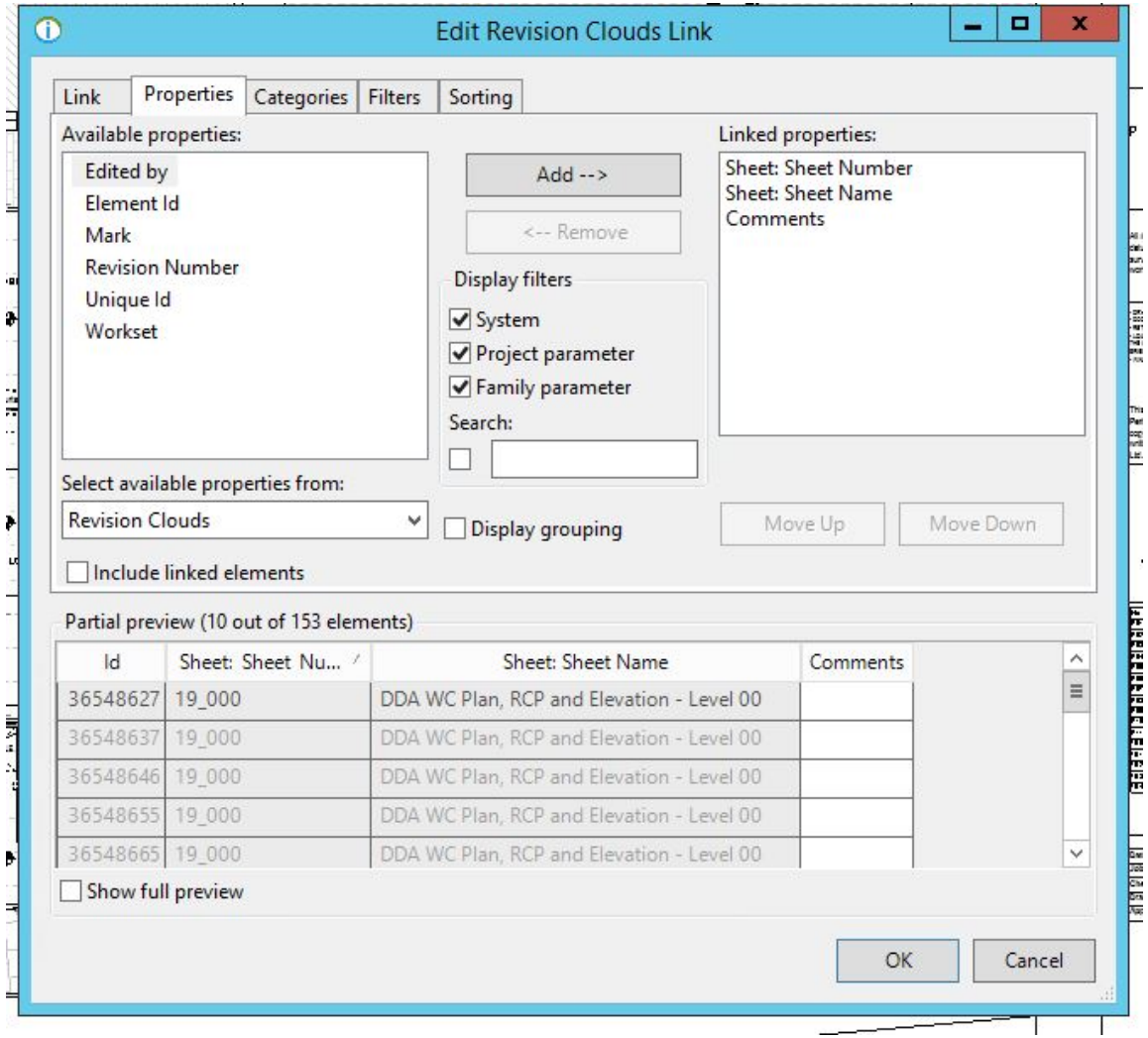
Current Tools

Revision + Cloud Manager



Current Tools

Revision + Cloud Manager

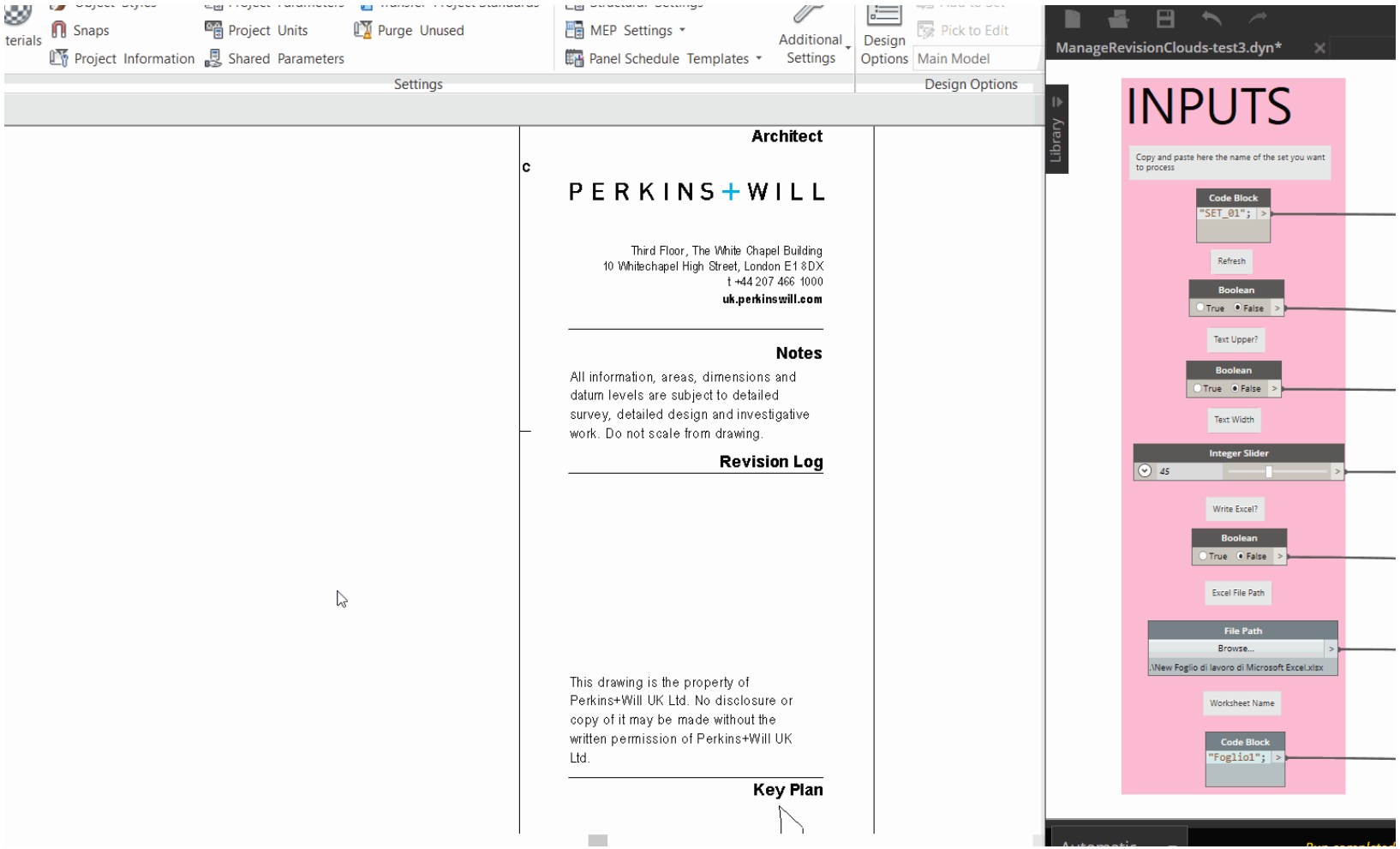


Dynamo Workflow

Why?

- Place Clouds Comments on Sheets
- Print Set as input
- Customization

Dynamo Workflow

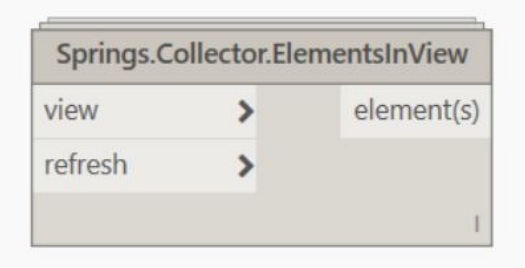


How

Collector.ElementsInView

SpringNodes Package

By Dimitar Venkov



Edit Python Script...

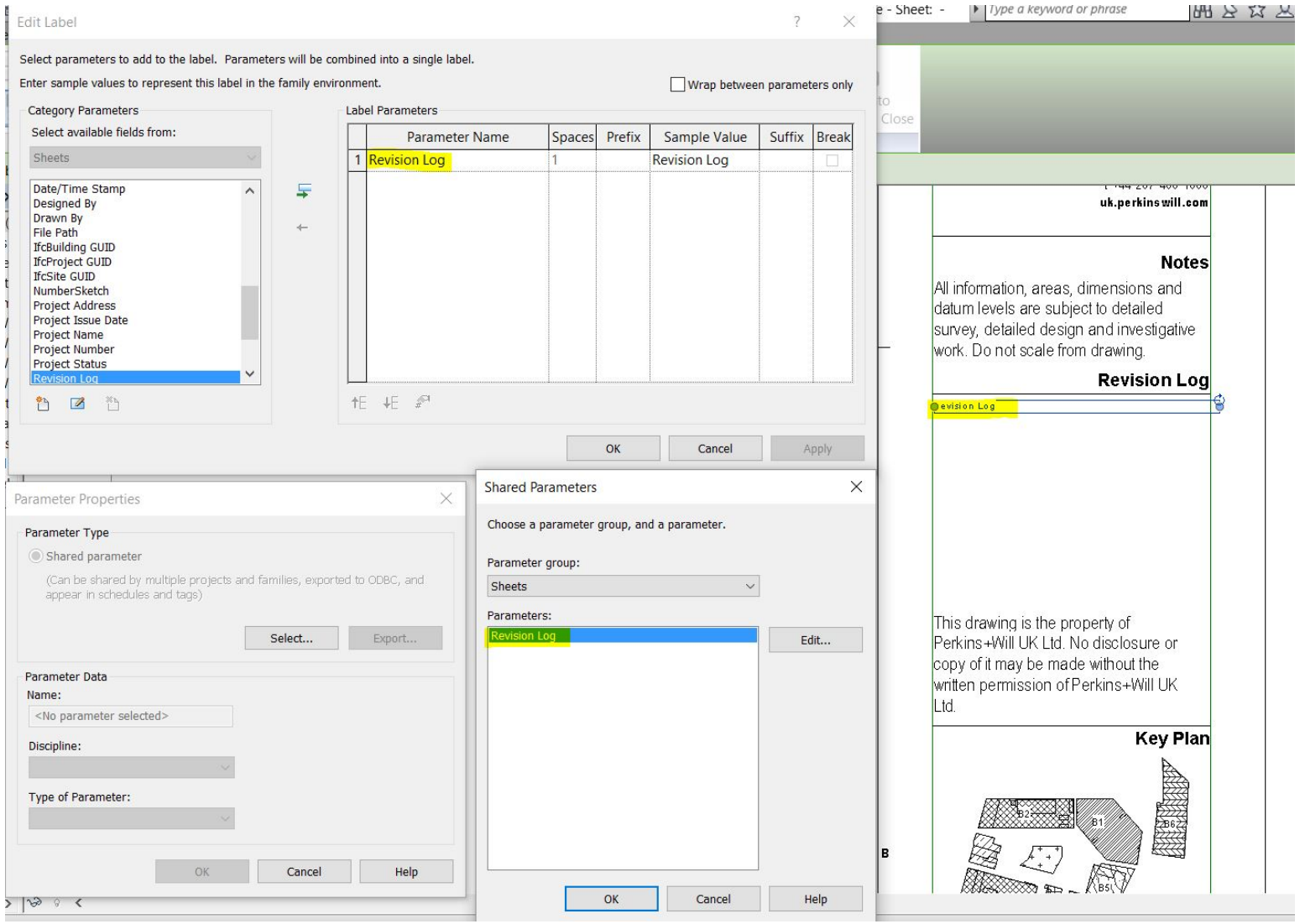
```python
#Copyright(c) 2016, Dimitar Venkov
# @5devene, dimitar.ven@gmail.com

import clr

clr.AddReference('RevitServices')
import RevitServices
from RevitServices.Persistence import DocumentManager
doc = DocumentManager.Instance.CurrentDBDocument

clr.AddReference('RevitAPI')
from Autodesk.Revit.DB import *

clr.AddReference('RevitNodes')
import Revit
clr.ImportExtensions(Revit.Elements)

def tolist(obj1):
    if hasattr(obj1,'__iter__'): return obj1
    else: return [obj1]
def output1(l1):
    if len(l1) == 1: return l1[0]
    else: return l1

views = UnwrapElement(tolist(IN[0]))
elements = []

for i in xrange(len(views)):
    fec = FilteredElementCollector(doc, views[i].Id).OfClass(RevisionCloud).GetElementIterator()
    fec.Reset()
    view_el = []
    while fec.MoveNext():
        view_el.append(fec.Current.ToDSType(True))
    elements.append(view_el)
OUT = output1(elements)
```

Accept Changes | Cancel

How

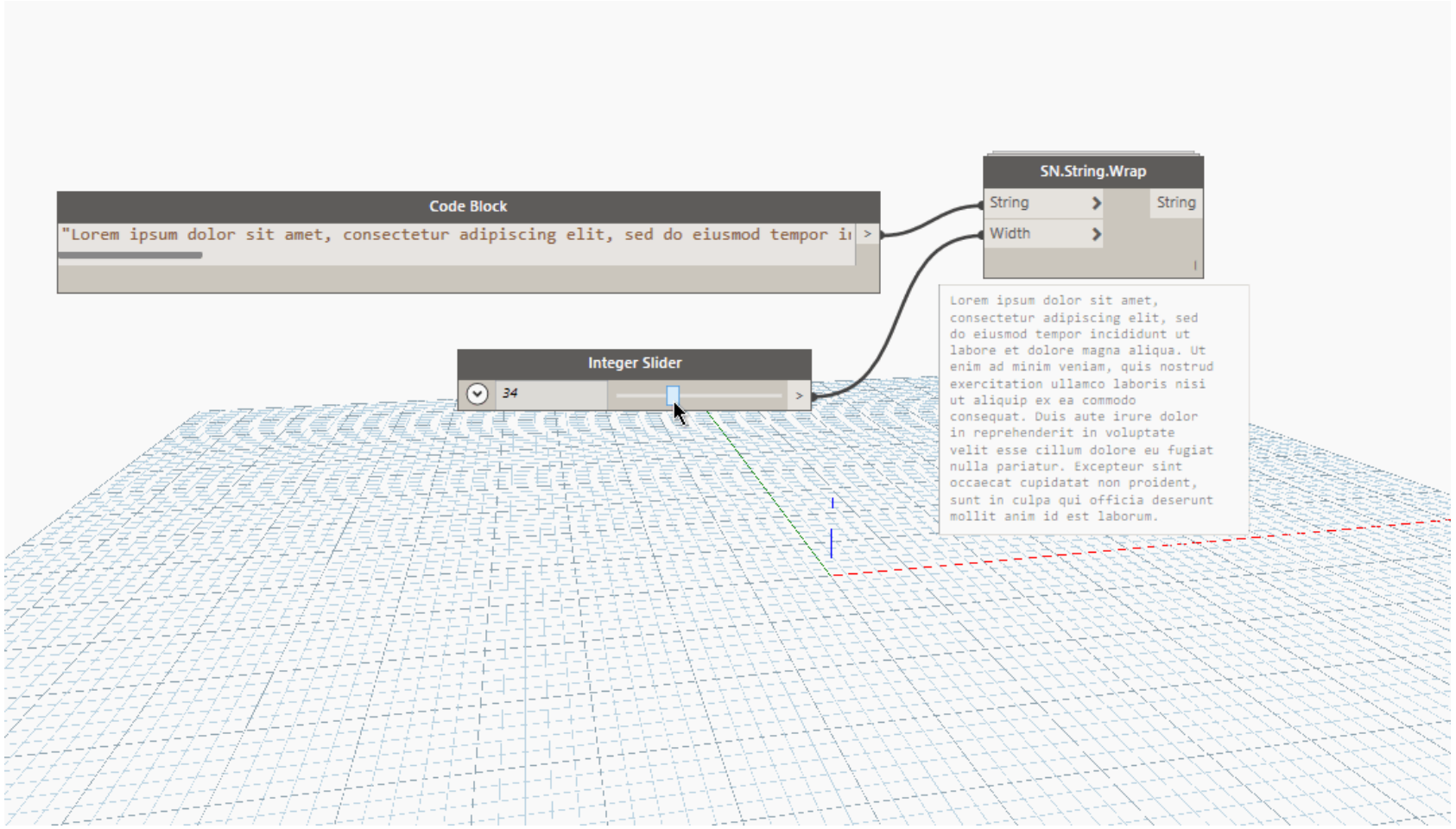
Set sheets text shared parameter.

And place the label in the titleblocks.

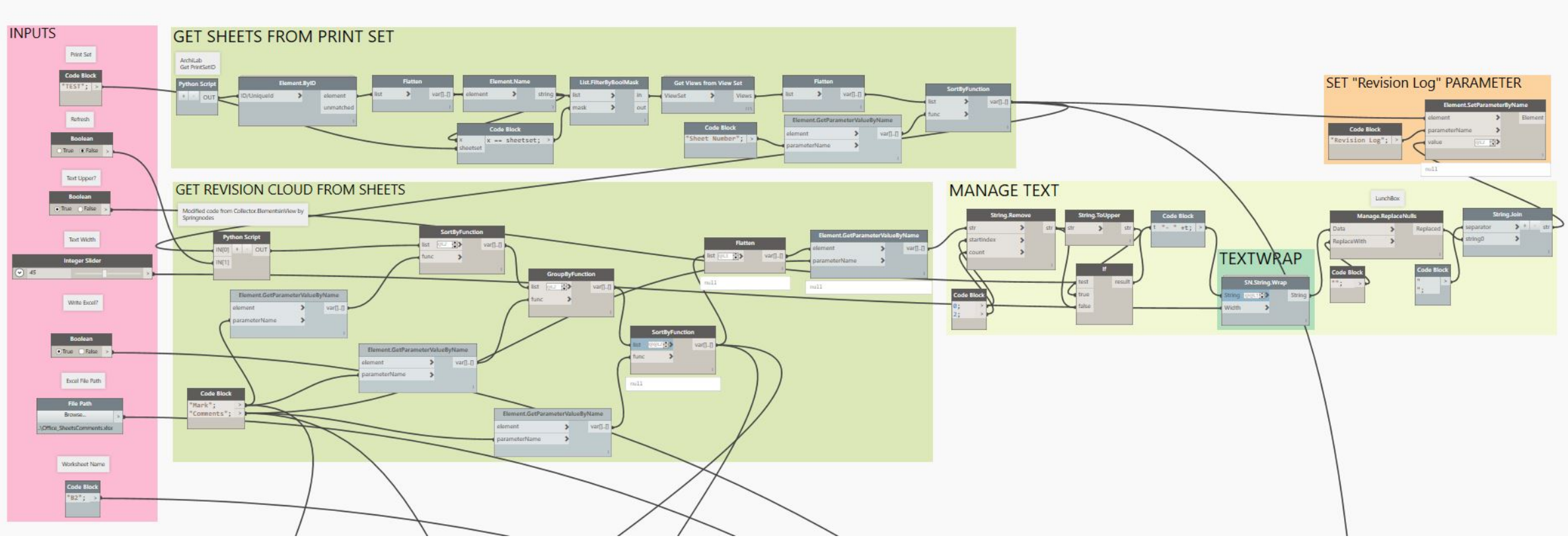


TextWrap

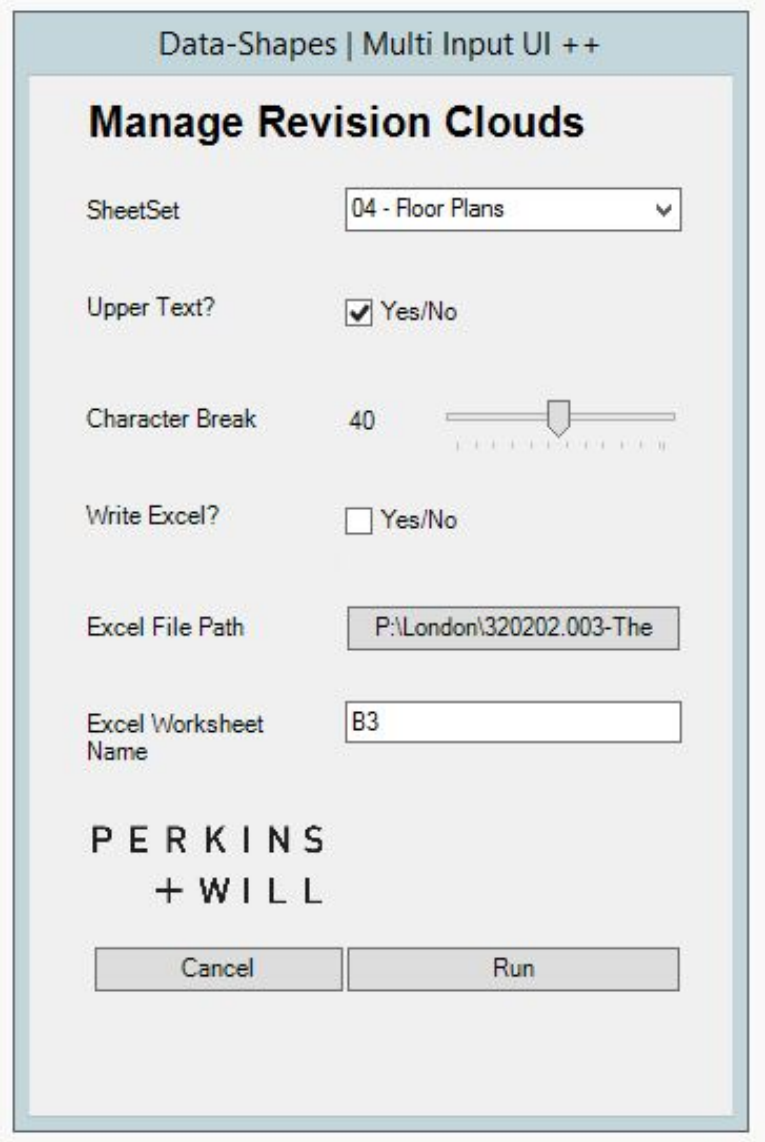
```
import clr
clr.AddReference('ProtoGeometry')
from Autodesk.DesignScript.Geometry import *
import sys
sys.path.append(r'C:\Program Files (x86)\IronPython 2.7\Lib')
import textwrap

#The inputs to this node will be stored as a list in the IN variables.
dataEnteringNode = IN
text = IN[0]
len = IN[1]

def wraptext(tx, l):

    b = textwrap.dedent(tx).strip()
    if l<=1:
        l = 1
    for width in [l]:
        c = textwrap.fill(b, width=width)
        return c

#Assign your output to the OUT variable.

OUT = wraptext(text, len)
```



Dyn Definition



Inputs



Thank you

Design Script - Functions

The easy way to control parameters

Defining a Function within a Code Block

Extract from Dynamo Primer –

"Functions can be created in a code block and recalled elsewhere in a Dynamo definition. This creates another layer of control in a parametric file, and can be viewed as a text-based version of a custom node. In this case, the "parent" code block is readily accessible and can be located **anywhere** on the graph. No wires needed!"

http://dynamoprimer.com/en/07_Code-Block/7-4_functions.html

Code Block

```
def SearchMark()
{
    return = "RAD";
};

def TypeParams()
{
    return = {
    "Manufacturer", //Index 0
    "Model", //Index 1
    "Thermal Output", //Index 2
    "Water Flow Rate", //Index 3
    "Pipe Connections Size", //Index 4
    "Type Comments", //Index 5
    "Type Mark"};
 //Index 6;
};

def InstanceParams()
{
    return = {
    "Mark", //Index 0
    "Level", //Index 1
    "Schedule Level", //Index 2
    "Type Id", //Index 3
    "Height", //Index 4
    "Length"};
 //Index 5;
};

def ExcelHeaders()
{
    return = {"TypeID","UniqueID","Ref No.","Location", "Manufacturer",
    "Model", "Output (W)", "Flow Rate (kg/s)","Size (WxH)", "Connections",
    "Notes"};
};
```

Defining a Function within a Code Block

Within the Function you can define lists of Strings, Numbers or Boolean values. Anything you can have in a list, you can have in a list within a Function.

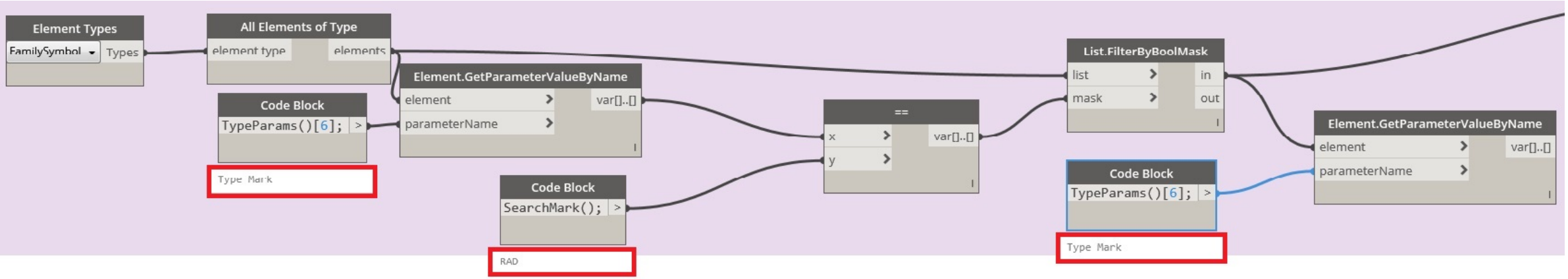
These can then be recalled anywhere on the graph without the need for lengthy wires, or node duplication.

If you need to change the parameter value simply change it in the Function no need to search the graph.

Code Block

```
def SearchMark()
{
    return = "RAD";
};

def TypeParams()
{
    return = {
    "Manufacturer", //Index 0
    "Model", //Index 1
    "Thermal Output", //Index 2
    "Water Flow Rate", //Index 3
    "Pipe Connections Size", //Index 4
    "Type Comments", //Index 5
    "Type Mark"};
 //Index 6;
};

def InstanceParams()
{
    return = {
    "Mark", //Index 0
    "Level", //Index 1
    "Schedule Level", //Index 2
    "Type Id", //Index 3
    "Height", //Index 4
    "Length"};
 //Index 5;
};

def ExcelHeaders()
{
    return = {"TypeID","UniqueID","Ref No.","Location", "Manufacturer",
    "Model", "Output (W)", "Flow Rate (kg/s)","Size (WxH)", "Connections",
    "Notes"};
};
```

Using the defined Function in a graph



As with normal lists you recall the whole list –

FunctionName() ;

Or an index of that list -

FunctionName() [0];

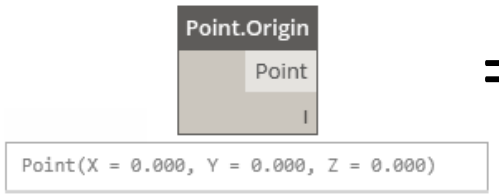
Dynamo Coordinate System

The easy way to control parameters

Dynamo Coordinate System

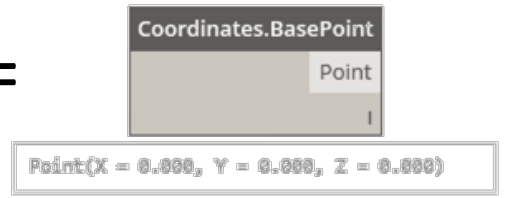
Assuming that:

Dynamo

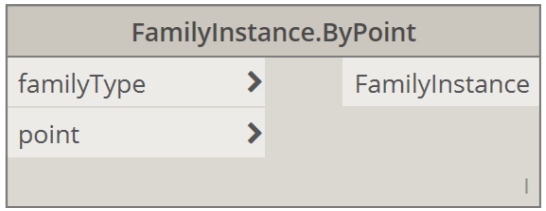


=

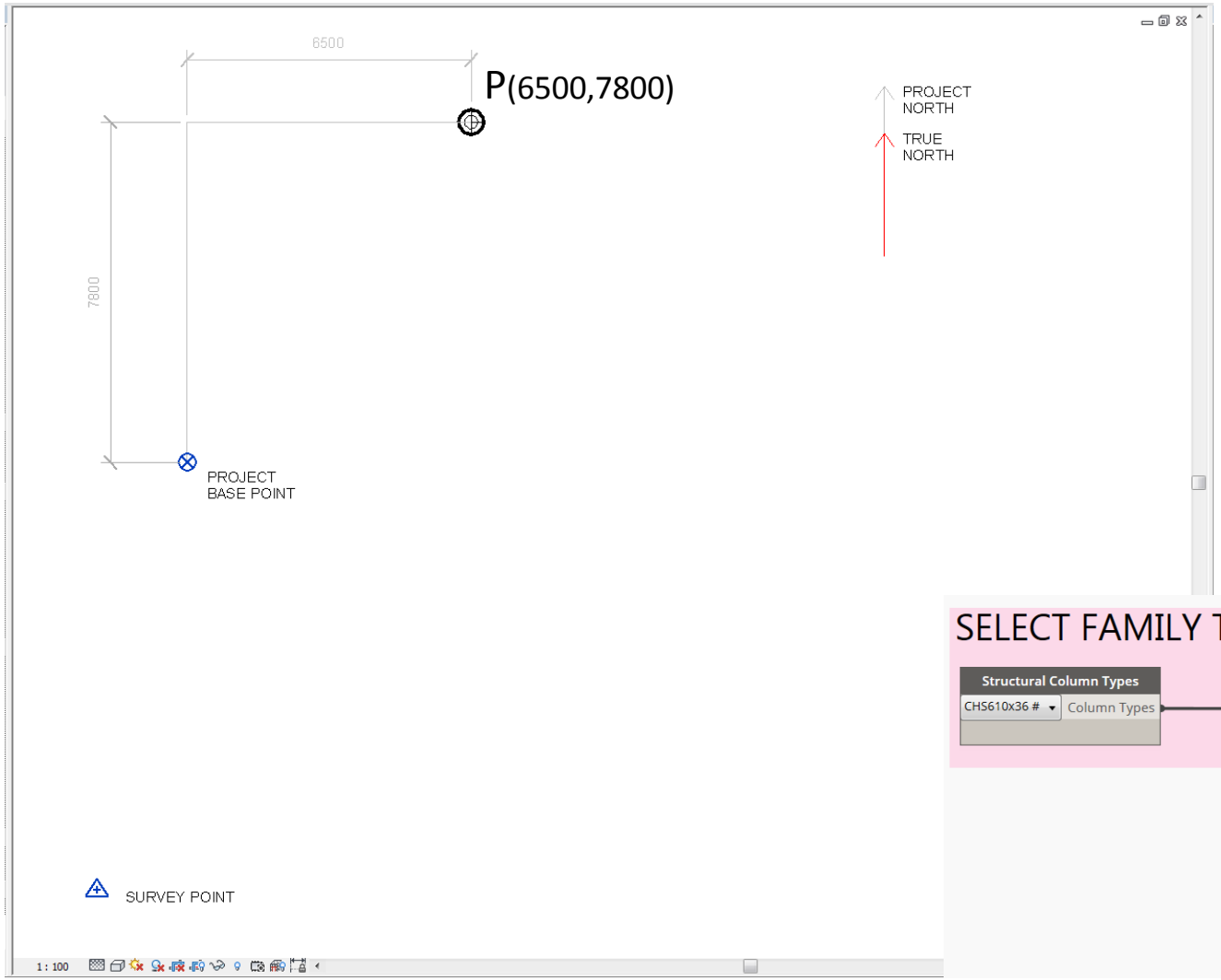
Revit

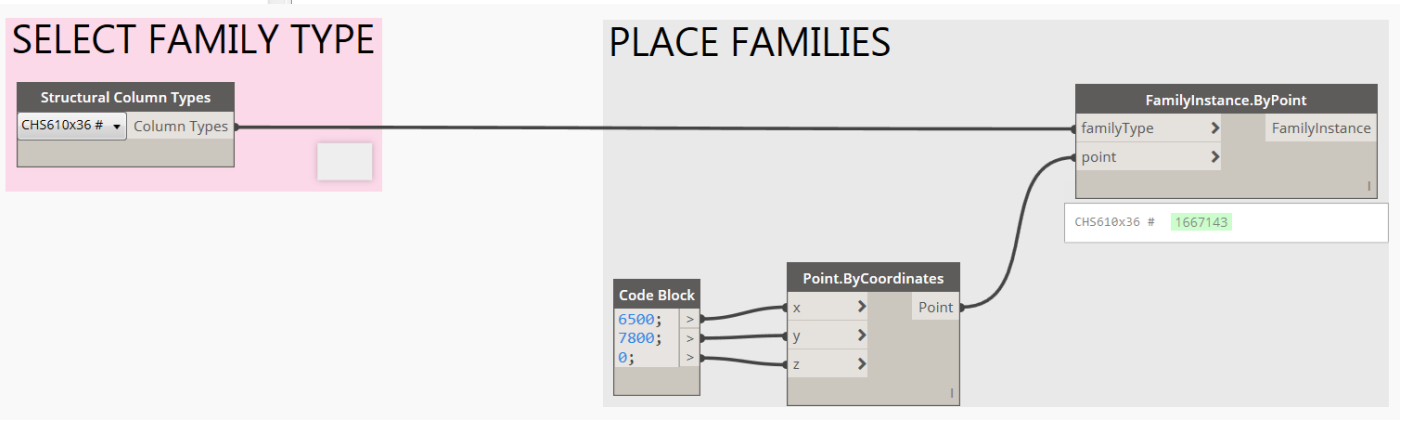


How do I place a family at a datum point P in Revit?

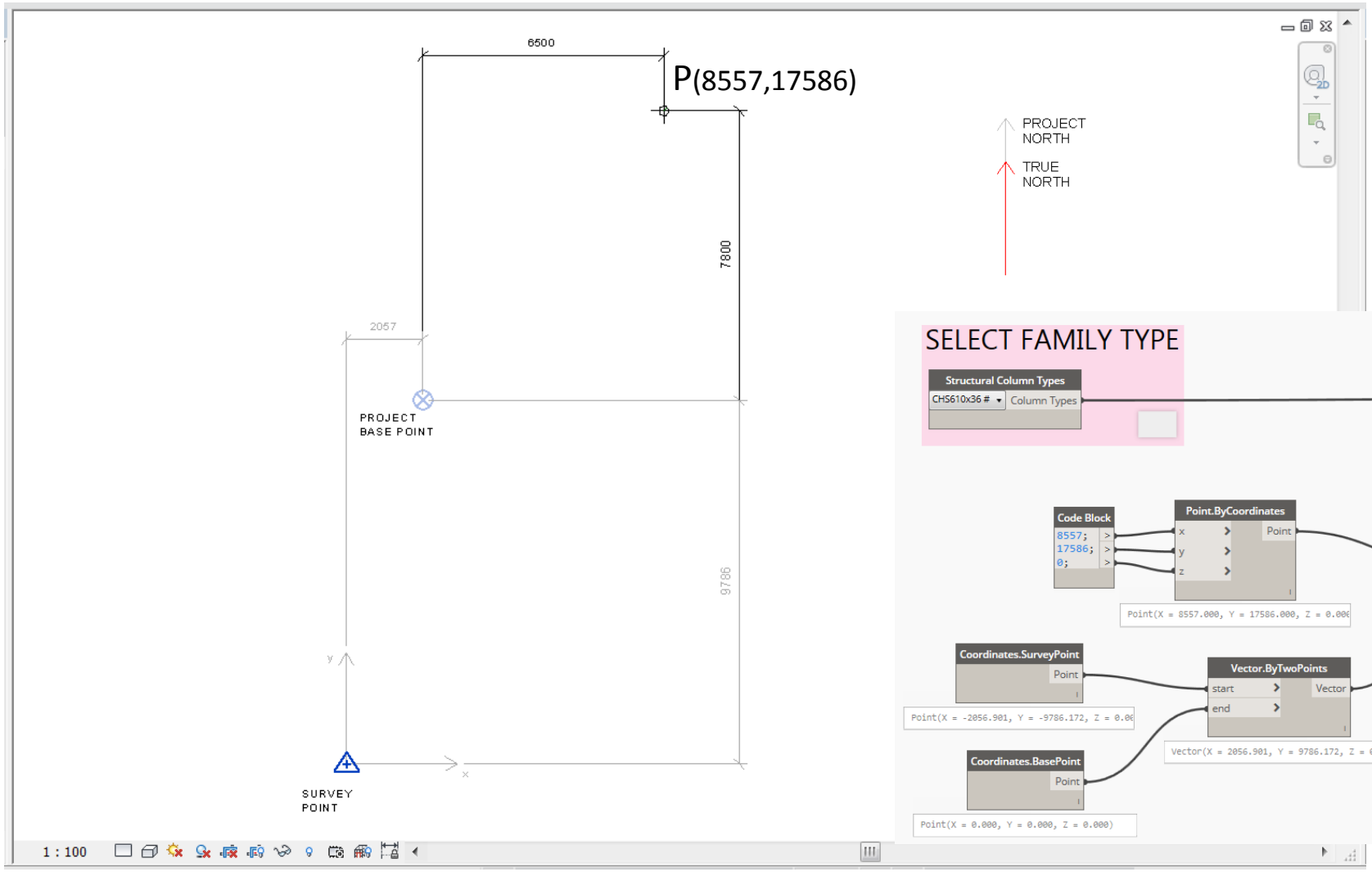


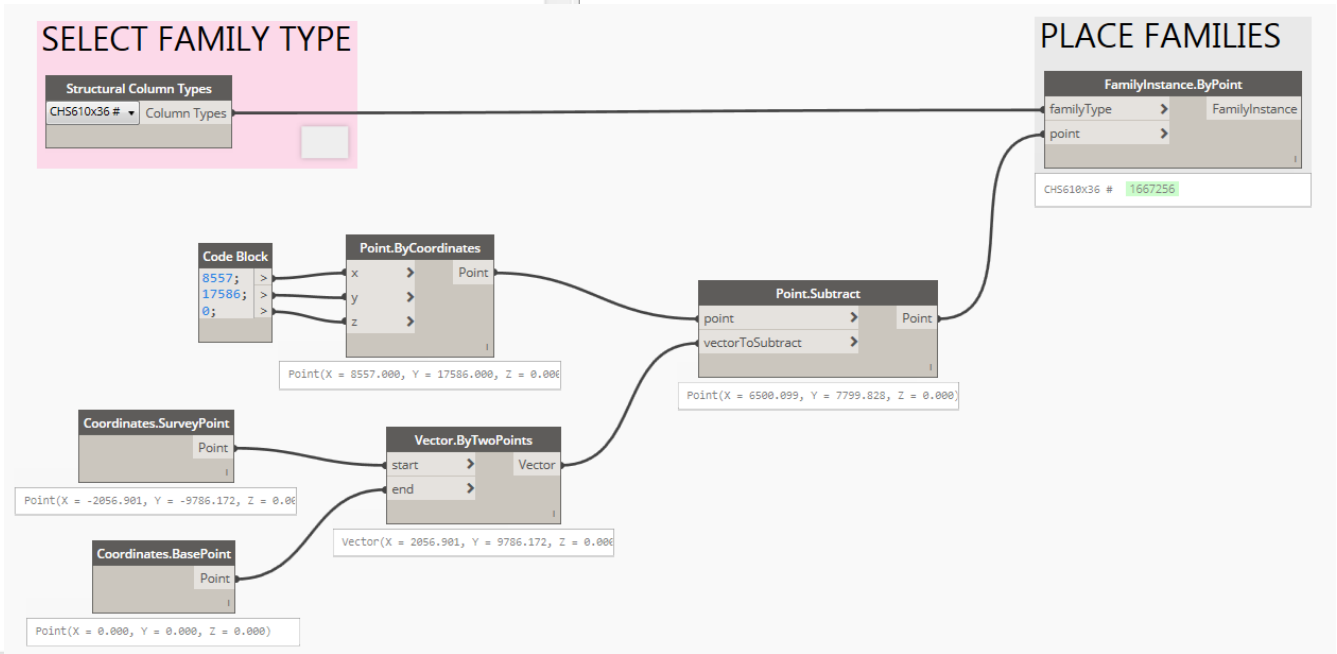
Scenario #1: P_{coord} from PBP, Project North = True North



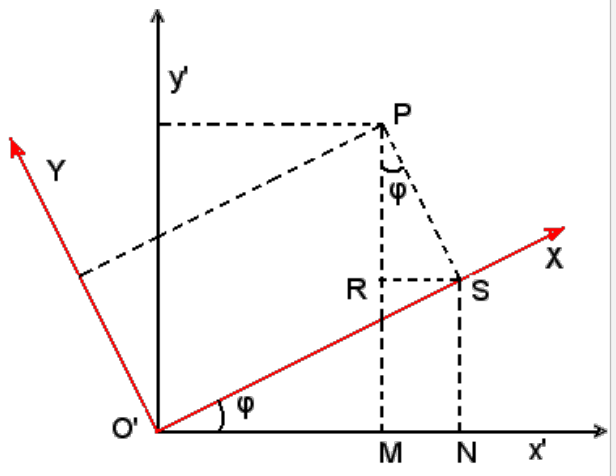


Scenario #2: P_{coord} from SP, Project North = True North





Scenario #3: P_{coord} from SP, Project North ≠ True North

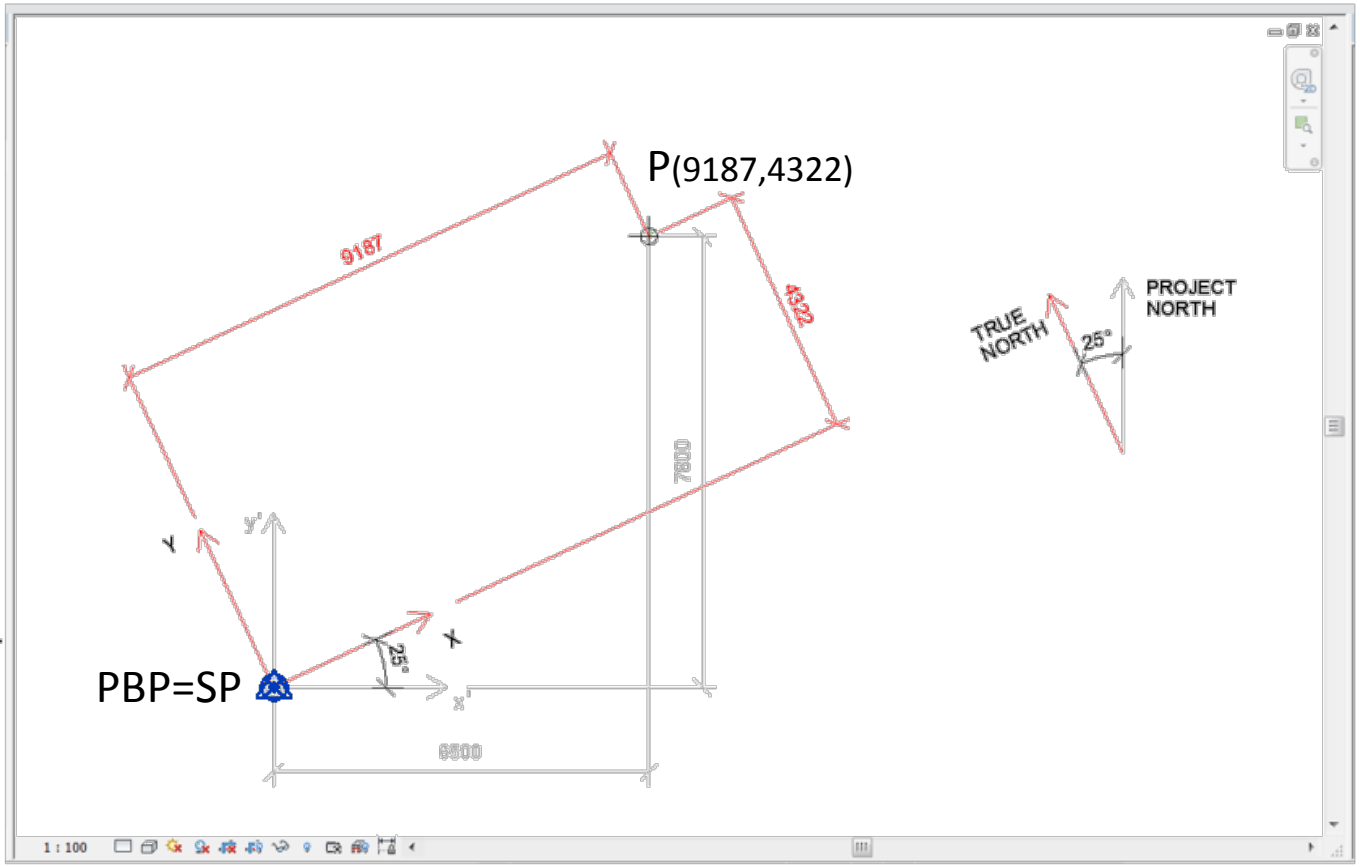


$X = x' \cos\varphi + y' \operatorname{sen}\varphi$

$Y = -x' \operatorname{sen}\varphi + y' \cos\varphi$

$x' = X \cos\varphi - Y \operatorname{sen}\varphi$

$y' = X \operatorname{sen}\varphi + Y \cos\varphi$

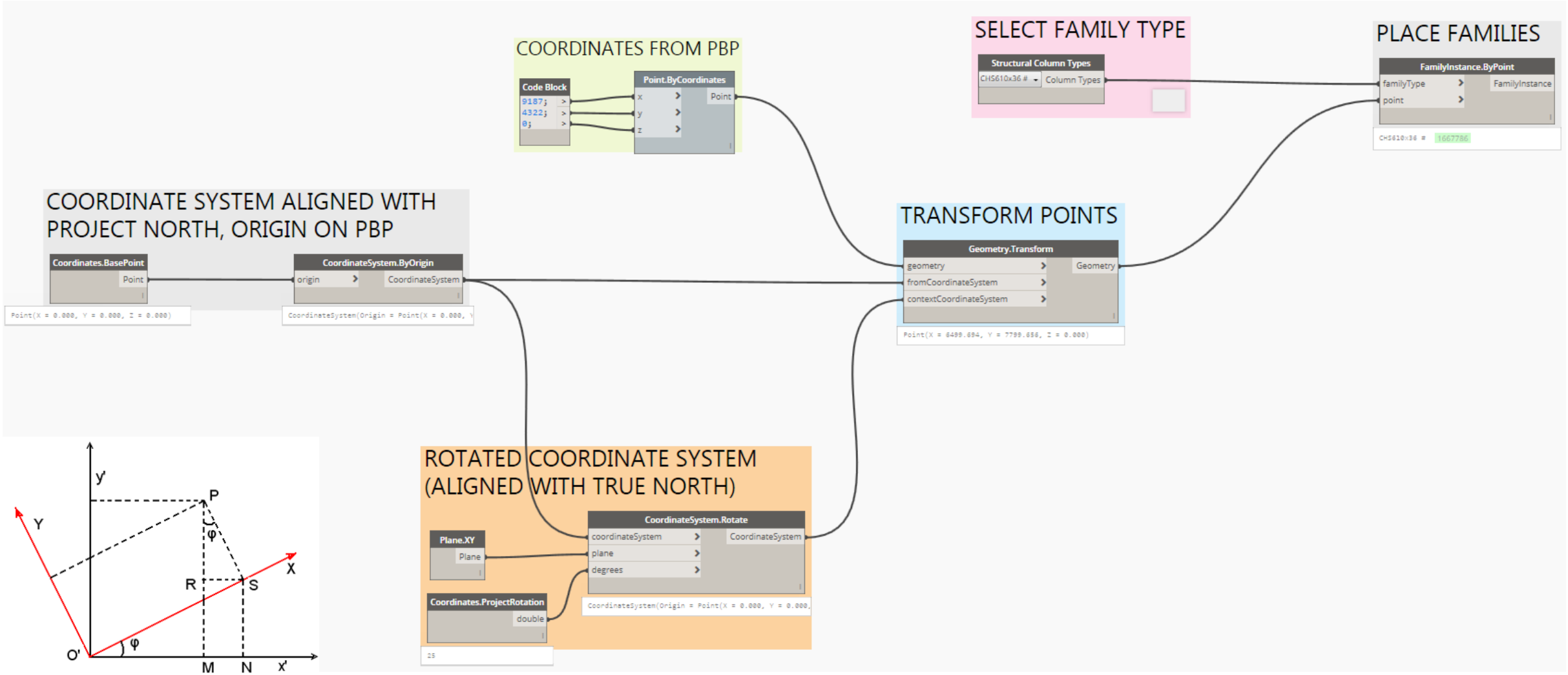


	degree	radians
ANGLE	25	0.436
sin	0.423	
cos	0.906	

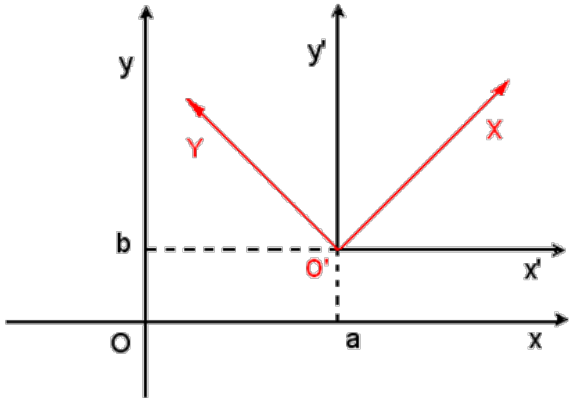
POINT COORDINATES	
X	Y
9187	4322

TRANSFORMED COORDINATES	
x'	y'
6500	7800

Scenario #3: P_{coord} from SP, Project North ≠ True North



Scenario #4: P_{coord} from SP, Project North ≠ True North

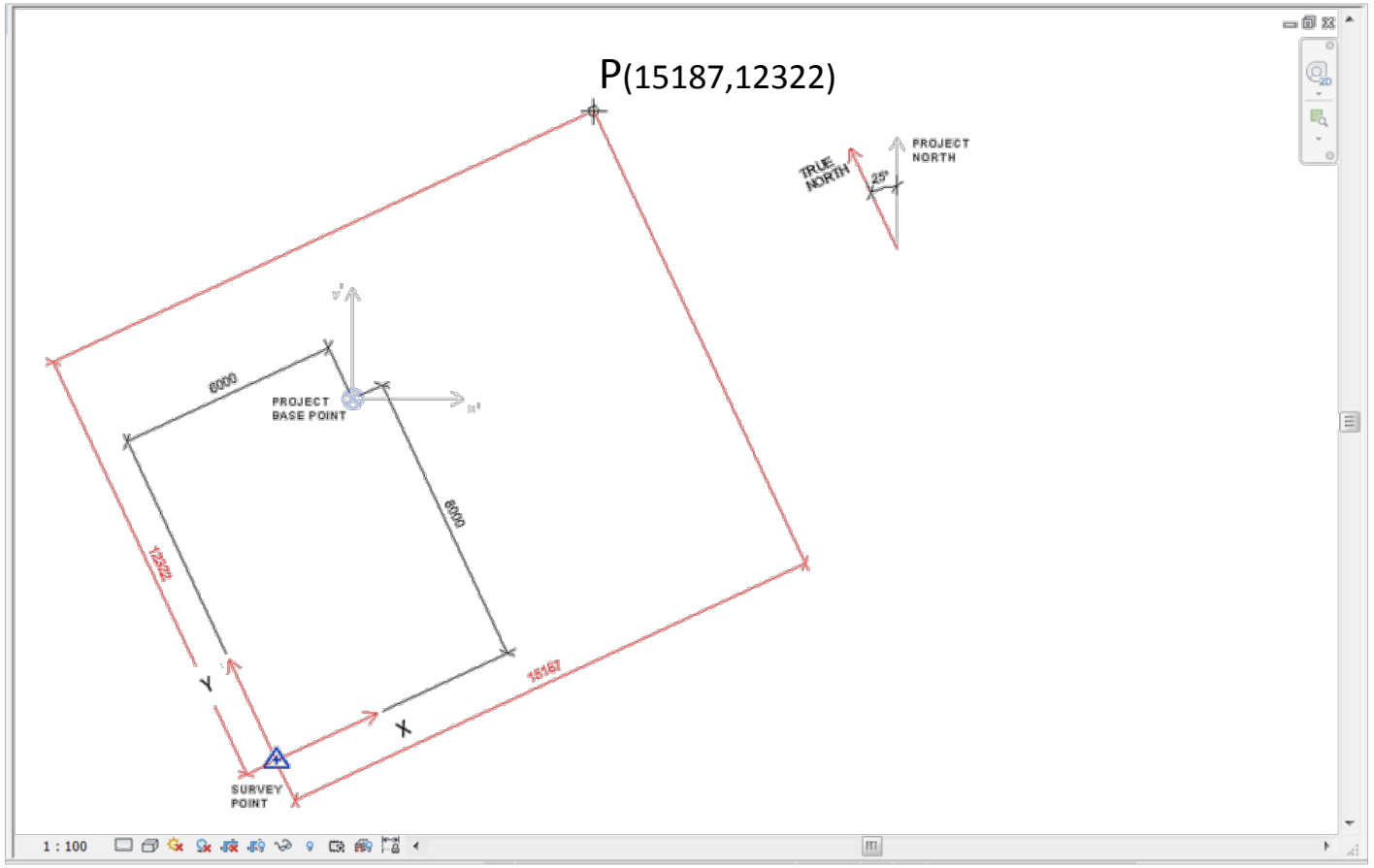


$$X = (x - a)\cos\varphi + (y - b)\sin\varphi$$

$$Y = -(x - a)\sin\varphi + (y - b)\cos\varphi$$

$$x' = (x - a)\cos\varphi - (y - b)\sin\varphi$$

$$y' = (x - a)\sin\varphi + (y - b)\cos\varphi$$



	degree	radians
ANGLE	25	0.436
sin	0.423	
cos	0.906	

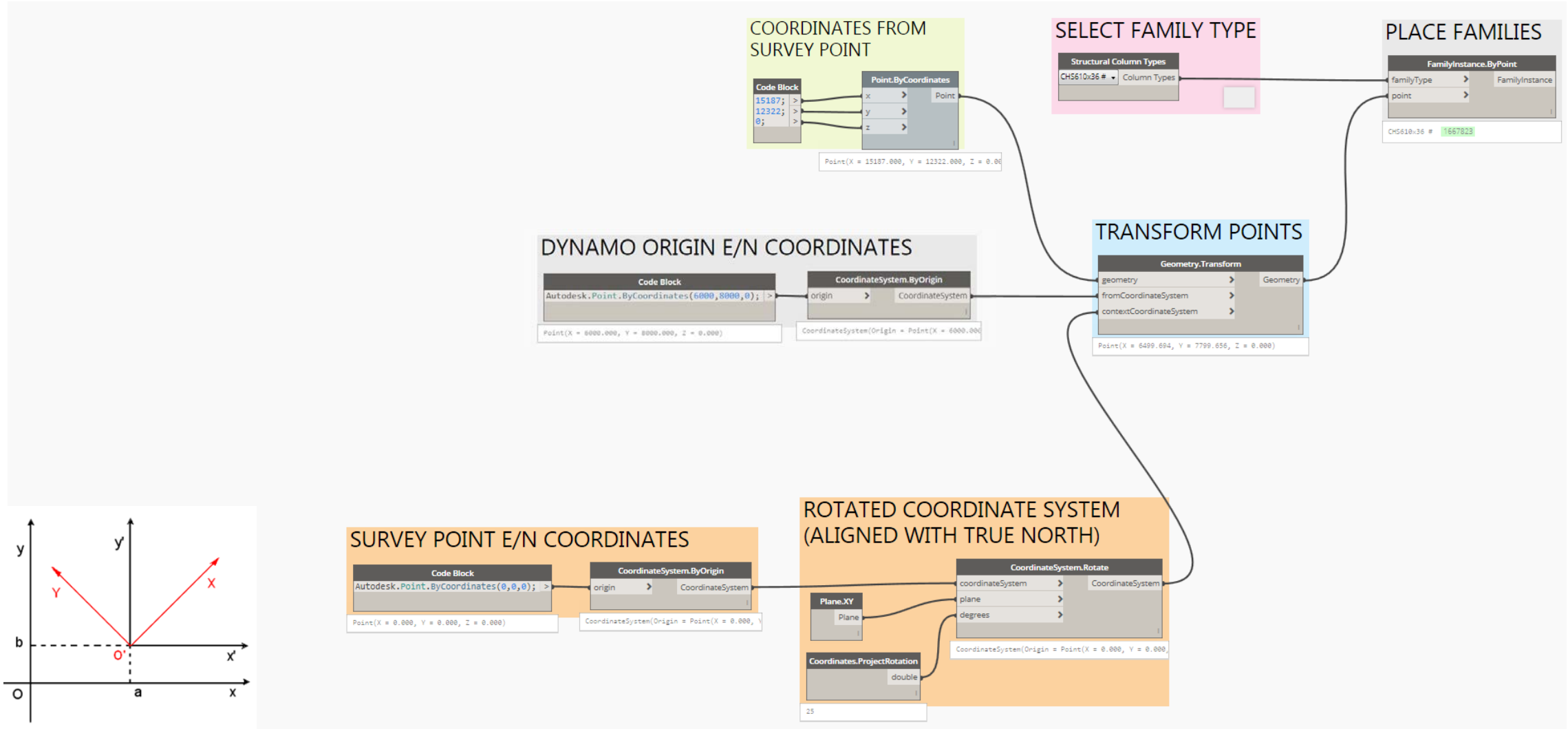
POINT COORDINATES	
X	Y
15187	12322

PBP COORDINATES	
a	b
6000	8000

X-a	Y-b
9187	4322

TRANSFORMED COORDINATES	
x'	y'
6500	7800

Scenario #4: P_{coord} from SP, Project North ≠ True North



Think in terms of Eastern and Northings