VVVV **BASIC**

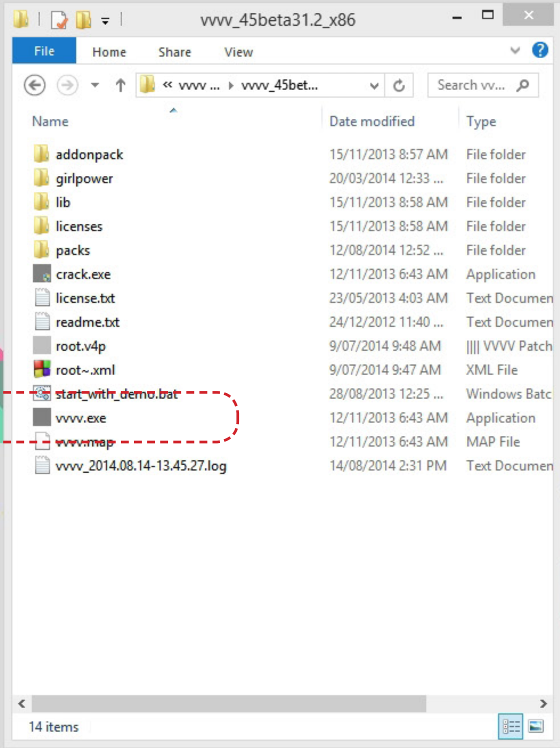
LMB / **L**eft **M**ouse **B**utton

RMB / **R**ight **M**ouse **B**utton

MMB / **M**iddle **M**ouse **B**utton

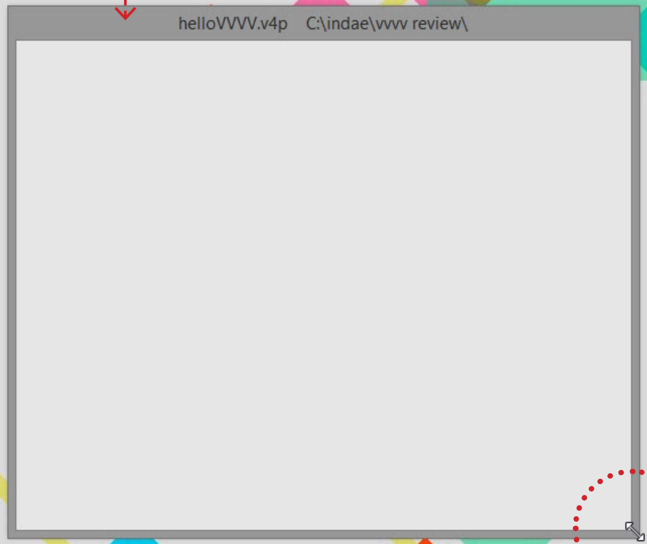
01. Run “vvvv” and patch windows

Click this to run VVVV



Name	Date modified	Type
addonpack	15/11/2013 8:57 AM	File folder
girlpower	20/03/2014 12:33 ...	File folder
lib	15/11/2013 8:58 AM	File folder
licenses	15/11/2013 8:58 AM	File folder
packs	12/08/2014 12:52 ...	File folder
crack.exe	12/11/2013 6:43 AM	Application
license.txt	23/05/2013 4:03 AM	Text Documen
readme.txt	24/12/2012 11:40 ...	Text Documen
root.v4p	9/07/2014 9:48 AM	|||| VVVV Patch
root~.xml	9/07/2014 9:47 AM	XML File
start_with_demo.bat	28/08/2013 12:25 ...	Windows Batc
vvvv.exe	12/11/2013 6:43 AM	Application
vvvv.map	12/11/2013 6:43 AM	MAP File
vvvv_2014.08.14-13.45.27.log	14/08/2014 2:31 PM	Text Documen

This is a patch window you can work on. The patch window is like an empty canvas to draw on. By adding elements (nodes and ioboxes) in this canvas you can make something “wow” or “useful”.



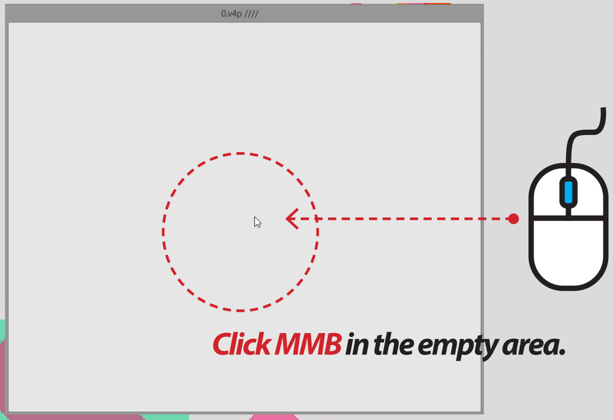
Resize the patch window by dragging the corner of the patch window.

ctr + **p**

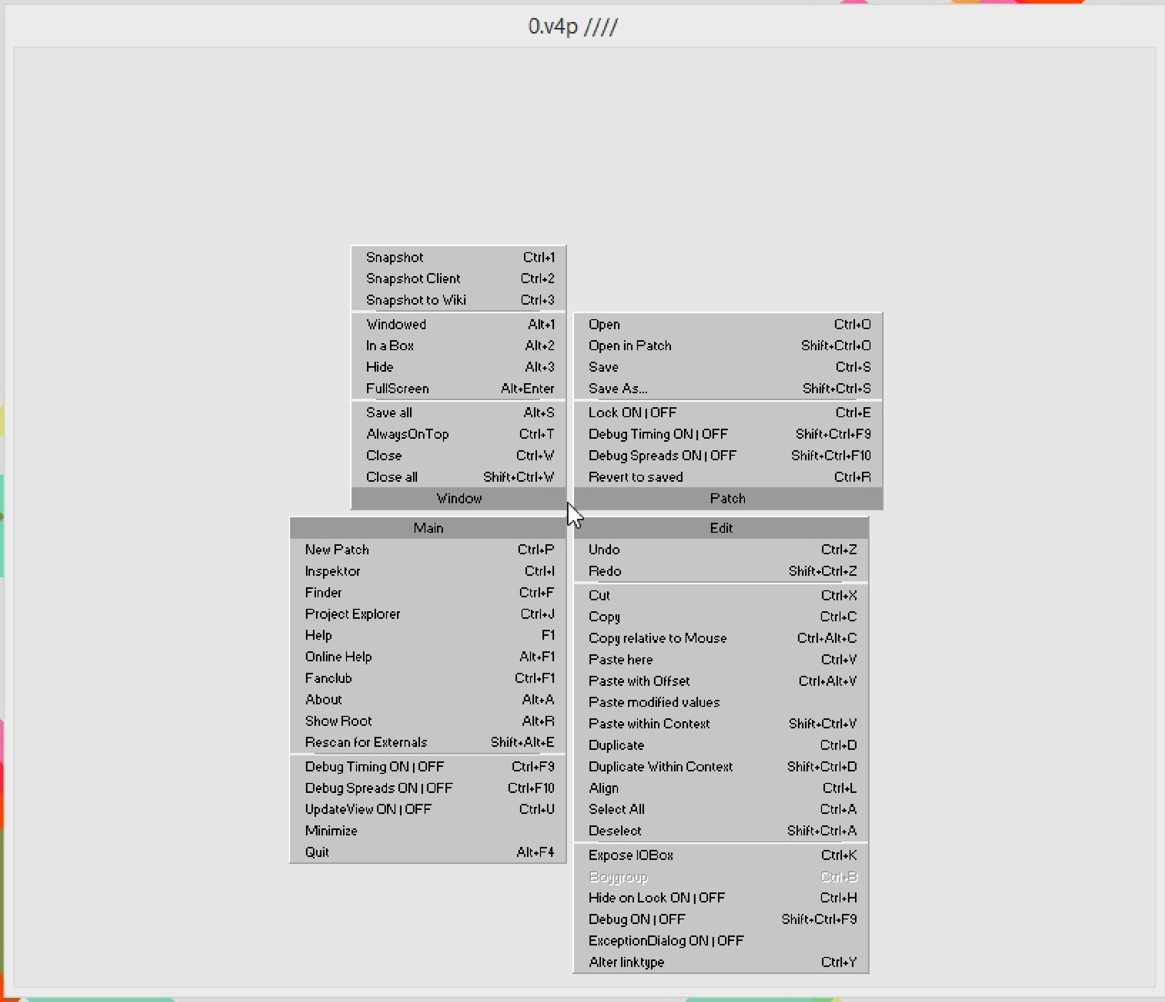
Create a new patch window

Quit vvvv : Alt + F4

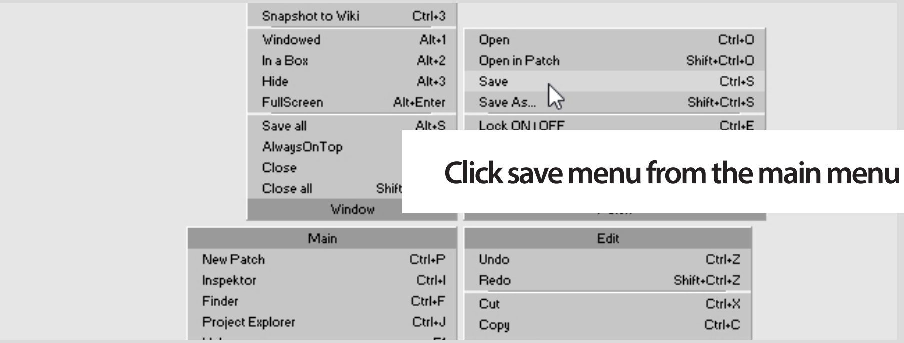
02. Show the main menu



***Click MMB** in the empty area.*



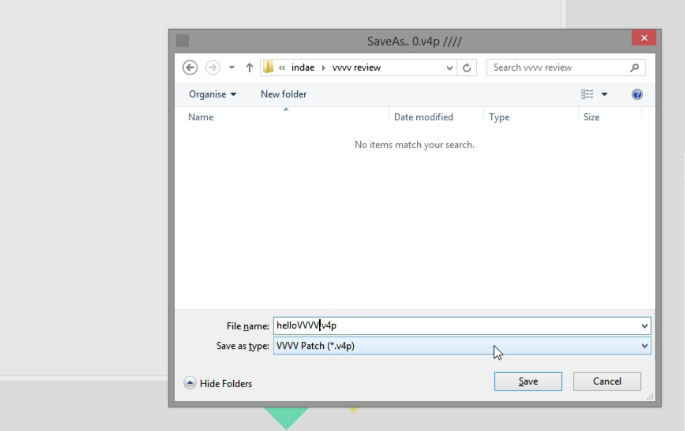
03. Save a 4v file



Click save menu from the main menu

or **ctr** + **s**

*** Try save as.**
Click the save as menu or "ctr + Shift + s".

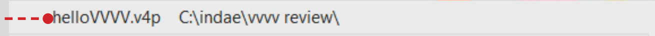


No items match your search.

File name: helloVVVV.v4p

Save as type: VVVV Patch (*.v4p)

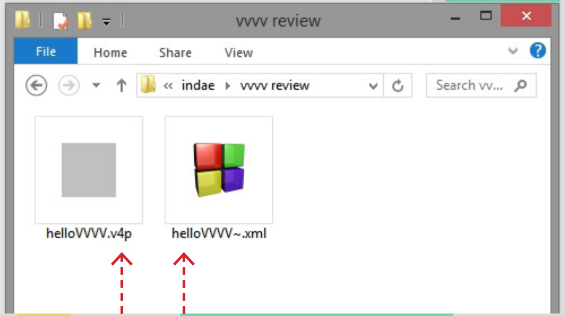
1.
Select a location (directory/folder) you want to save your patch and give it a name.



helloVVVV.v4p C:\indae\vvvv review\

Now the title of the patch window is changed.

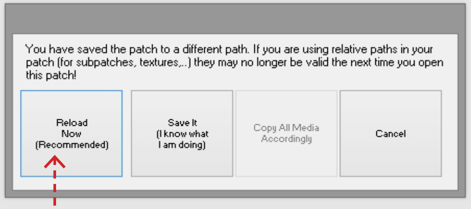
0.v4p ////



"v4p" is the extension of a vvvv patch file.

This is the backup file for your patch(v4p).

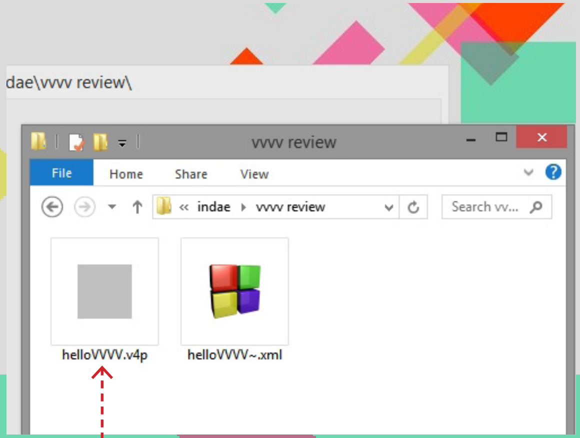
To use
Change XML to v4p and delete "~".



You have saved the patch to a different path. If you are using relative paths in your patch (for subpatches, textures,...) they may no longer be valid the next time you open this patch!

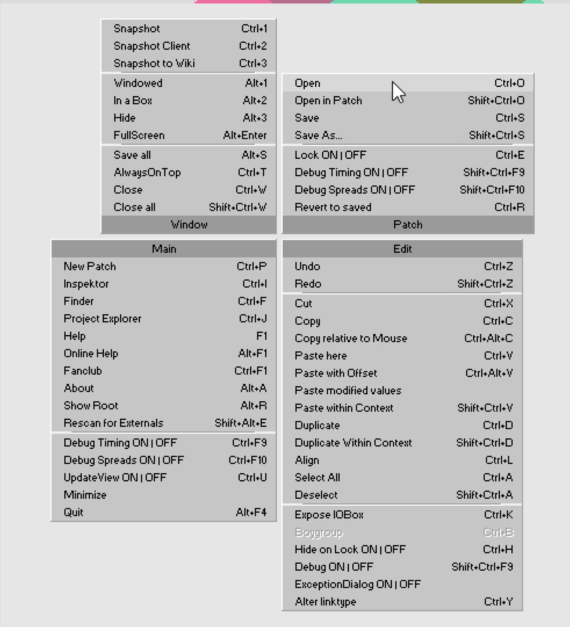
2.
Click this button: this pop-up window will be shown at the first time you save the 4v file.

04. Open a 4v file



Double click a v4p file with LMB.

Or



Or

** Open*

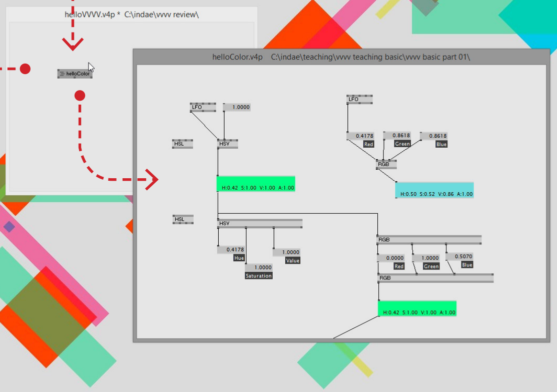
ctr + **o**

** Open in Patch*

shift + **ctr** + **o**

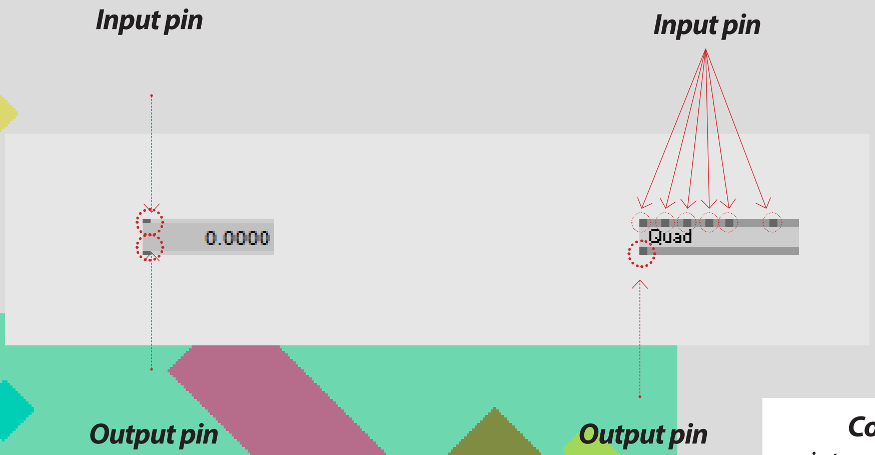
** Try "open in patch"*

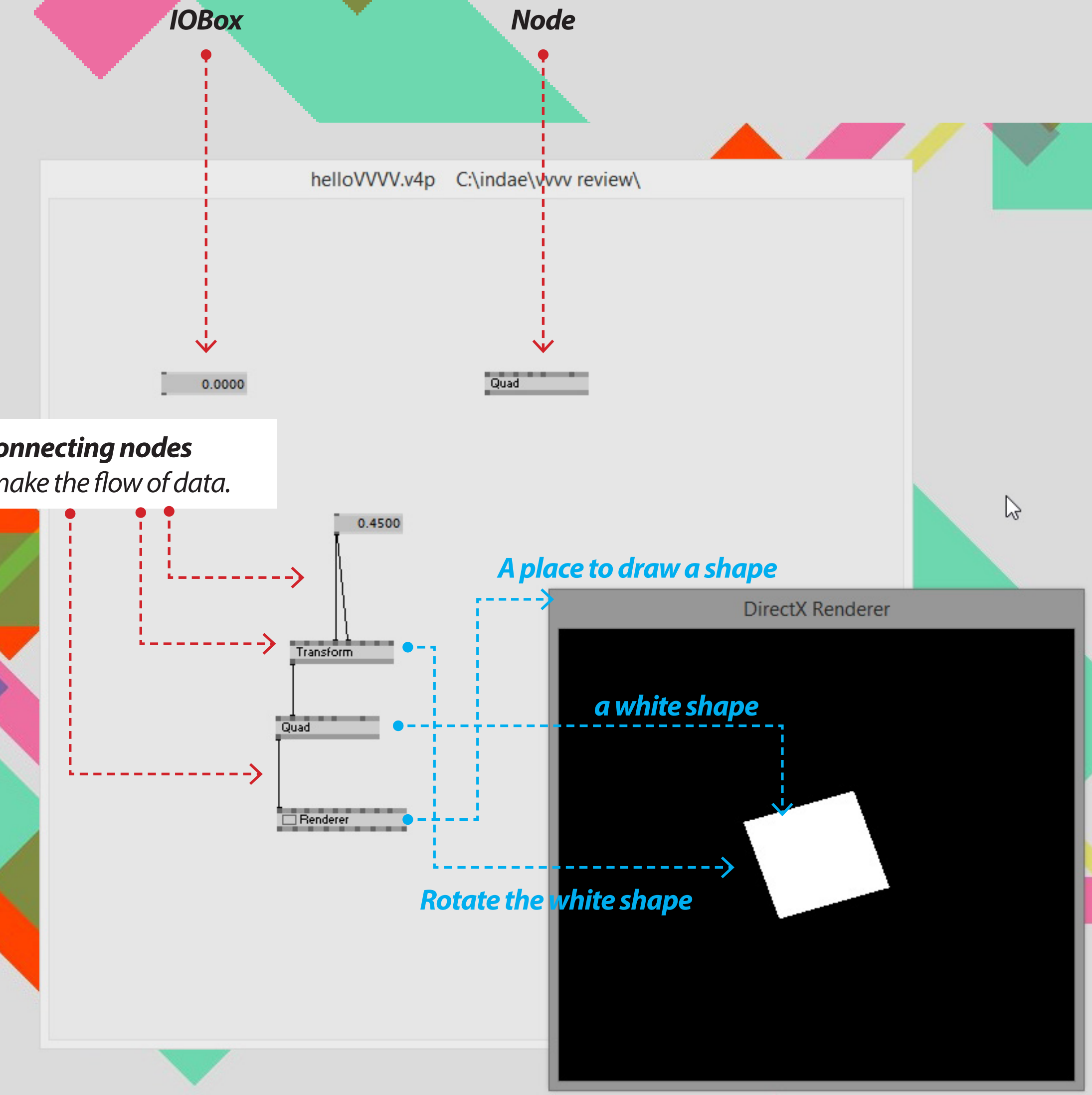
This is the patch file you select to "open in patch".


helloVVVV.v4p * C:\indae\vvvv review\
helloColor.v4p C:\indae\teaching\vvvv teaching basic\vvvv basic part 01\

Close a patch : ctr + w

05. *Make Something in 4V with IOBoxes and Nodes*

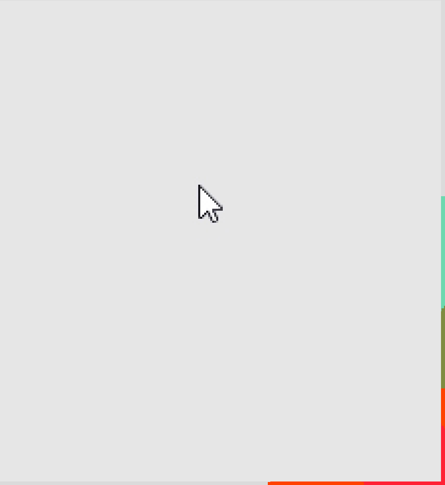


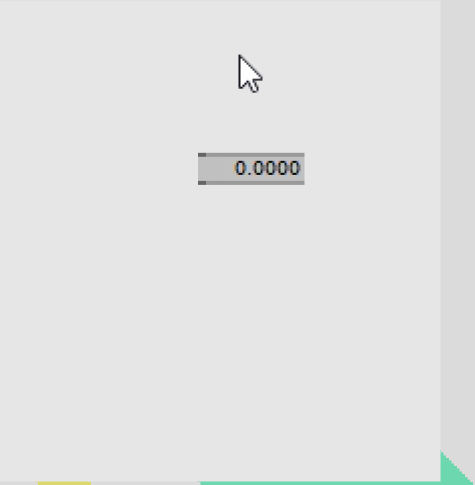


Connecting nodes
is to make the flow of data.

05. Creating *IOBox - How to create it*

IO in IOBox stands for: Input/Output. Denoting that those nodes are useful for both purposes: As a means for the user to input data into the running program. On the other hand they can be used to output/display data from the running program.





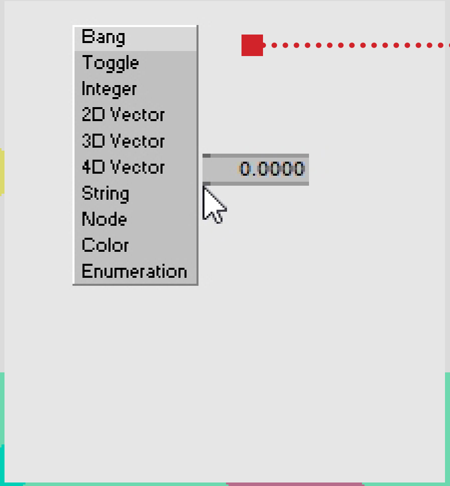
x 2

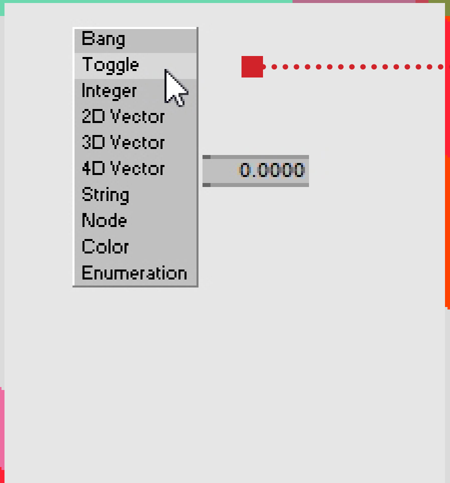
To create an IOBox
Double click **RMB** at empty area in an
patch window

Move your mouse ***left*** on the IOBox
list and ***up/down***, and select one.

Move your mouse ***right*** toward.
Now you can have float IOBox

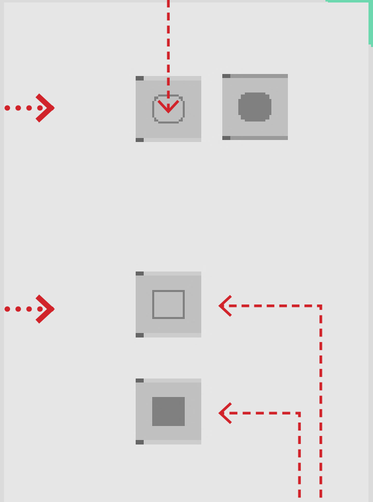
05. Creating *IOBox - Bang/Toggle*





Bang

Toggle



x 1

Click on the round box with RMB

Off → ← ***On***

When you click the IOBox, it outputs "on (1) " then back to "off (0) " ***right after that***.

Off → ***On*** → ***Off***

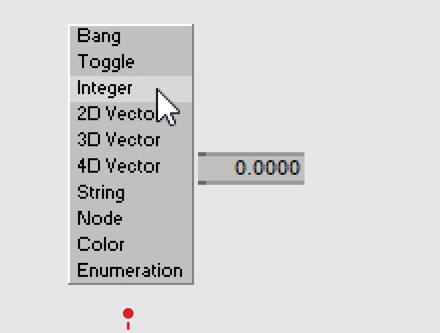
When you click the IOBox, it outputs "on (1) " then back to "off (0) " ***next time you click it***.

Click on the square box with RMB

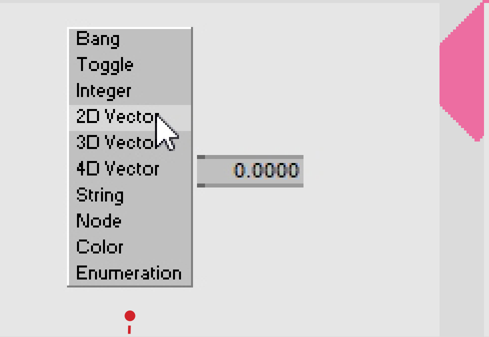
Click on the square box with RMB

x 1

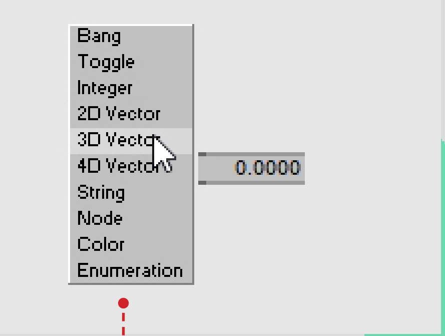
05. Creating *IOBox - Integer and 2D/3D/4D Vector*



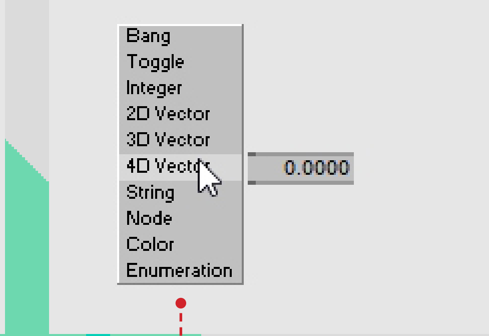
Integer is a number that can be written without a fractional component.



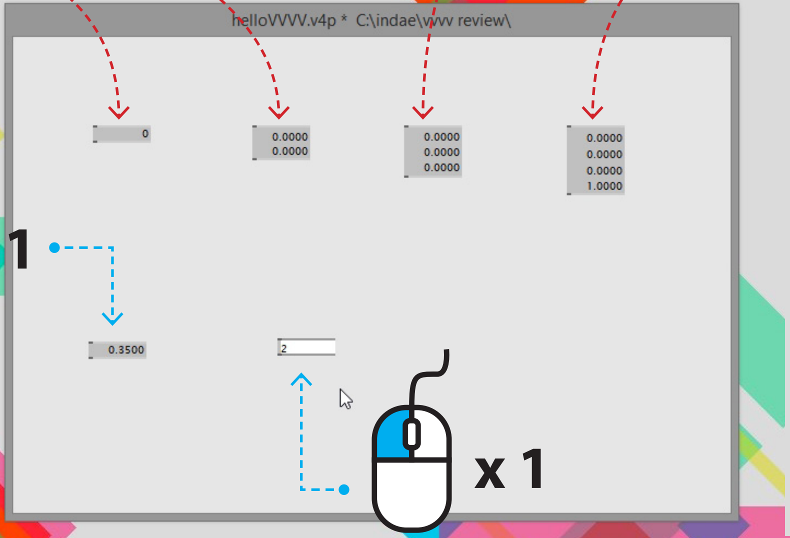
2D vector is a set of two real numbers, such as 1.222.

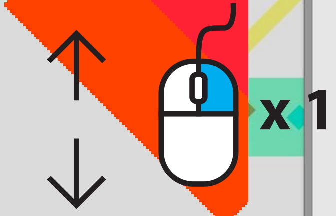


3D vector is a set of three real numbers ,such as 1.222.



4D vector is a set of four real numbers, such as 1.222.



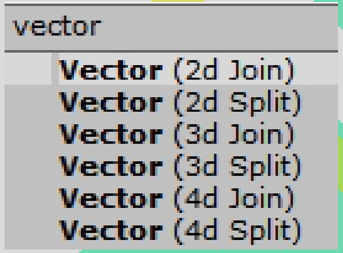


Changing a value of IOBox:
*Hover mouse over the IOBox and **click RMB and hold-drag up/down** mouse to change the value.*

Changing a value of IOBox
*Hover mouse over the IOBox and **click it with LMB and type** a value with keyboard.*

** 2D/3D/4D vector have other cases of use, such as "Vector mathematics". For now let's consider them as data containers.*

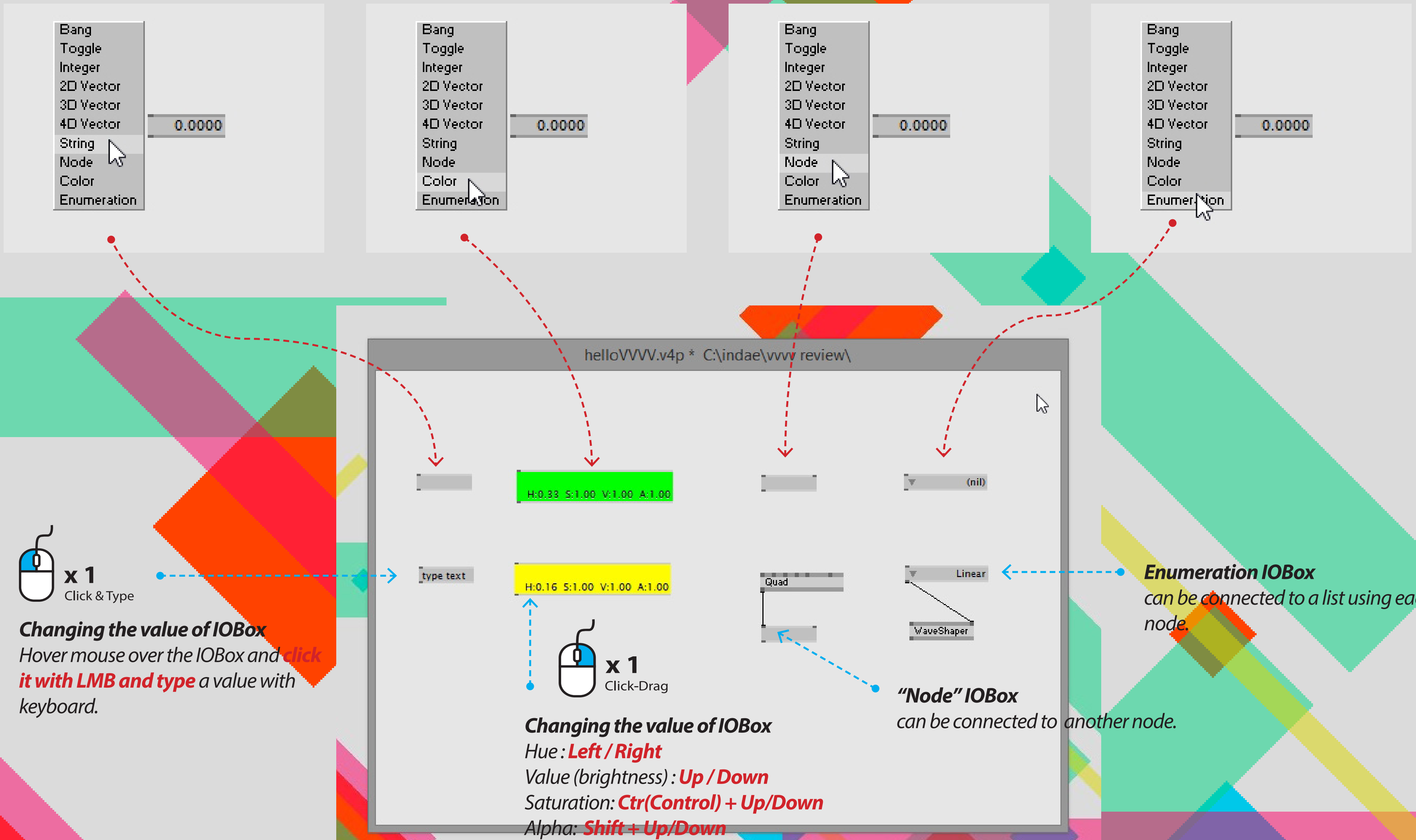
Related Nodes



vector

Vector (2d Join)
Vector (2d Split)
Vector (3d Join)
Vector (3d Split)
Vector (4d Join)
Vector (4d Split)

05. Creating *IOBox - String, color, and Enumeration*



Changing the value of IOBox

*Hover mouse over the IOBox and **click it with LMB and type** a value with keyboard.*

Changing the value of IOBox

*Hue : **Left / Right***

*Value (brightness) : **Up / Down***

*Saturation: **Ctr(Control) + Up/Down***

*Alpha: **Shift + Up/Down***

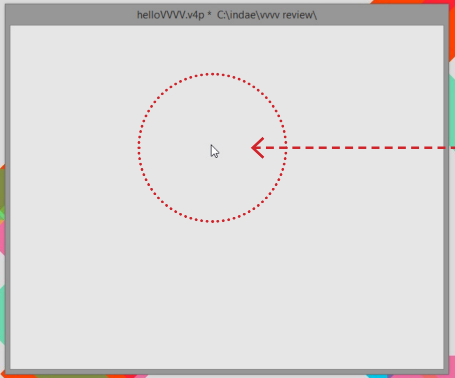
"Node" IOBox

can be connected to another node.

Enumeration IOBox

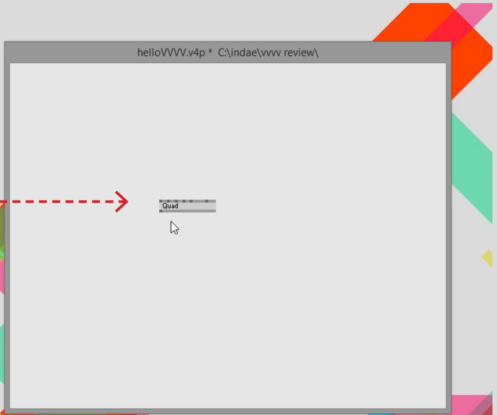
can be connected to a list using each node.

06. Creating a *Node*



x 2

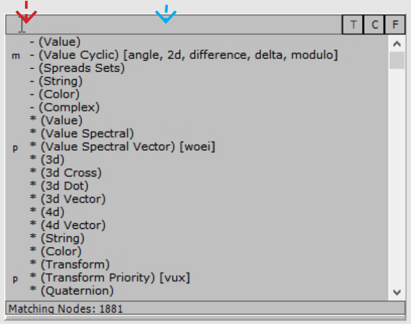
To create a Node
Double click at an empty area in patch window.



After double click LMB, the drop down menu appears.

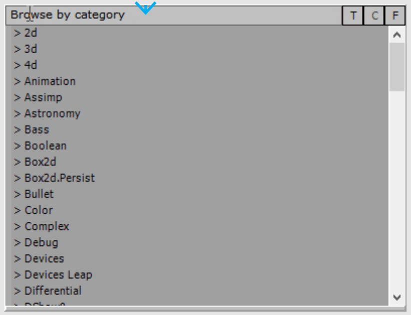
Typing the name of a node in the text field and press enter key to create a node.

x 1 ***Right-click on the text field to Switch between different drop down menu modes.***



```
    - (Value)
m   - (Value Cyclic) [angle, 2d, difference, delta, modulo]
    - (Spreads Sets)
    - (String)
    - (Color)
    - (Complex)
    * (Value)
    * (Value Spectral)
p   * (Value Spectral Vector) [woei]
    * (3d)
    * (3d Cross)
    * (3d Dot)
    * (3d Vector)
    * (4d)
    * (4d Vector)
    * (String)
    * (Color)
    * (Transform)
p   * (Transform Priority) [vux]
    * (Quaternion)
Matching Nodes: 1881
```

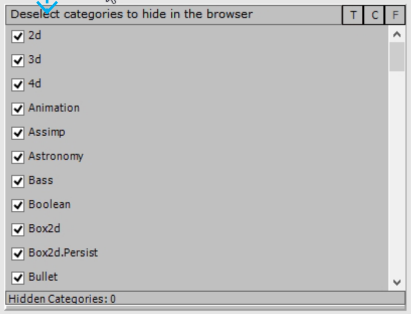
Order by alphabet.



```
Browse by category
> 2d
> 3d
> 4d
> Animation
> Assimp
> Astronomy
> Bass
> Boolean
> Box2d
> Box2d.Persist
> Bullet
> Color
> Complex
> Debug
> Devices
> Devices Leap
> Differential
```

Order by category.

Click a name of category with LMB to expand it. Select a node by clicking LMB.



```
Deselect categories to hide in the browser
☑ 2d
☑ 3d
☑ 4d
☑ Animation
☑ Assimp
☑ Astronomy
☑ Bass
☑ Boolean
☑ Box2d
☑ Box2d.Persist
☑ Bullet
Hidden Categories: 0
```

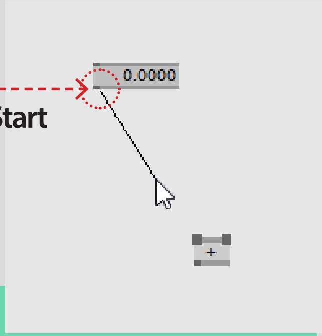
Manage categories.

06. Connection between *Nodes*

Connect nodes

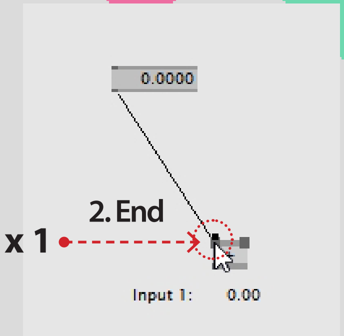
x 1

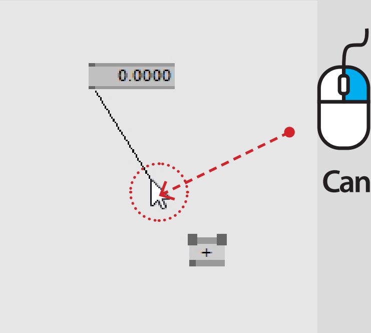
1. Start



x 1

2. End



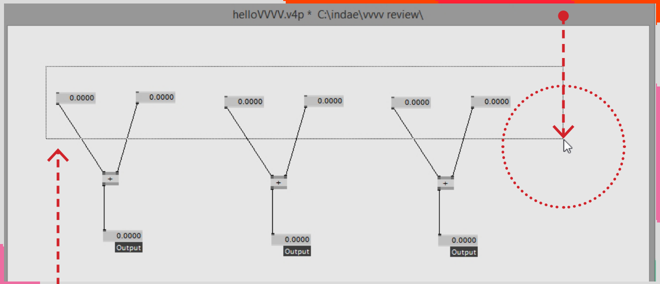


x 1

Cancel connecting.

Click LMB and hold drag to select multiful nodes

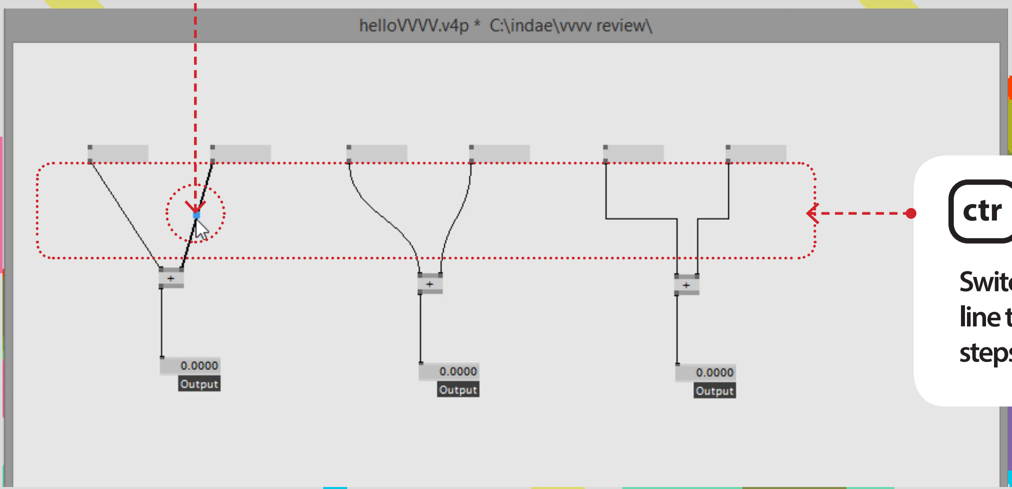
x 1



ctr + l

Arrange selected nodes.
(Ctrl(Control) + l)

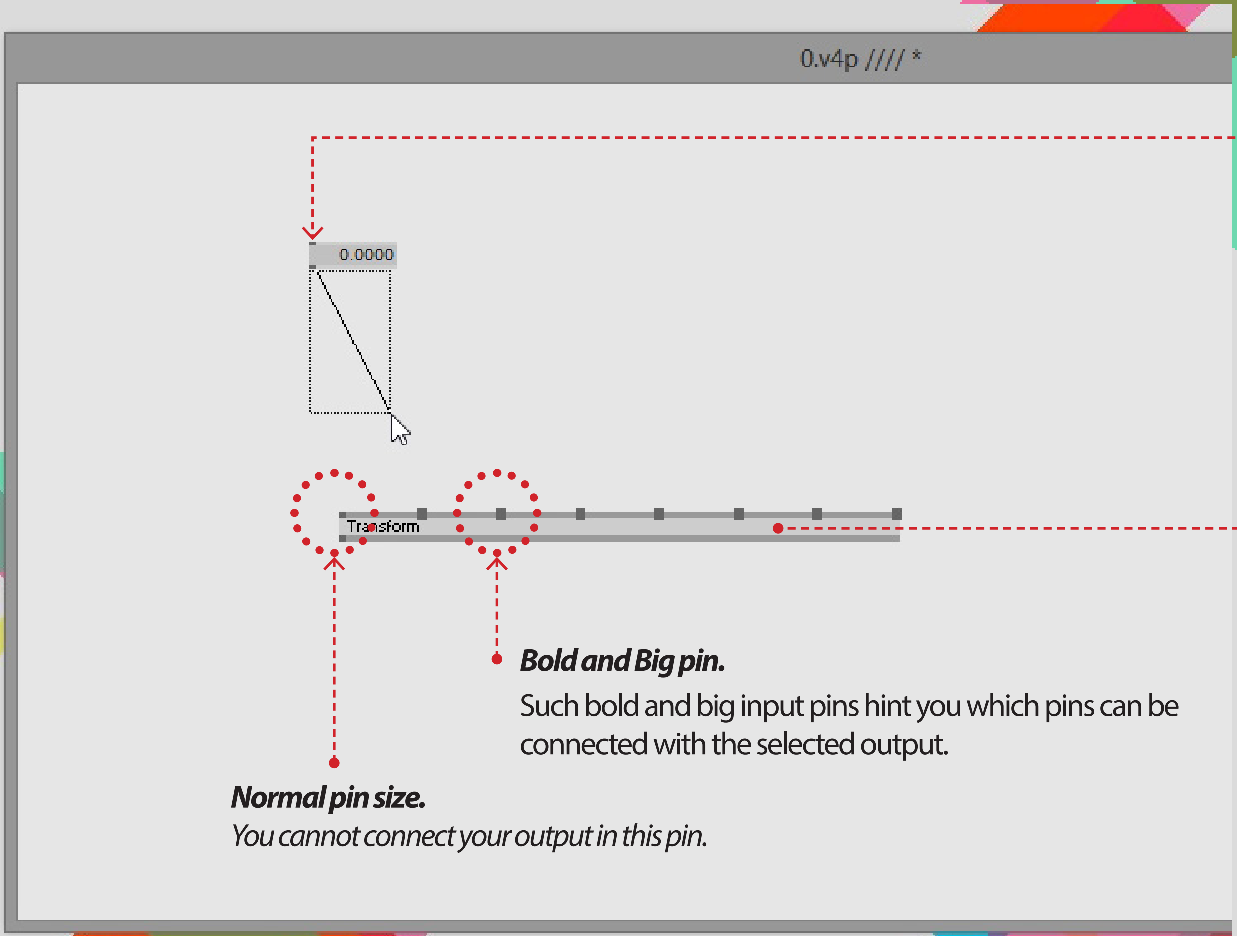
Select each line by clicking LMB. The selected line is dark and bold with a blue dot. You can modify the line by dragging the blue dot. You can use delete key or click the selected line with RMB.



ctr + y

Switching between different line types (straight, curve, steps)

06. Connecting: *Where to where and reset nodes*



Bold and Big pin.

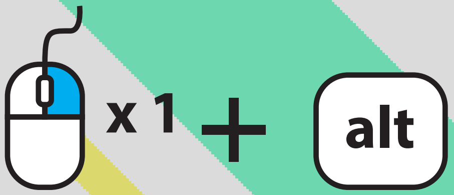
Such bold and big input pins hint you which pins can be connected with the selected output.

Normal pin size.

You cannot connect your output in this pin.

To reset values of any pins or any nodes, hover mouse on the pin you want to reset and ***click RMB + alt key.***

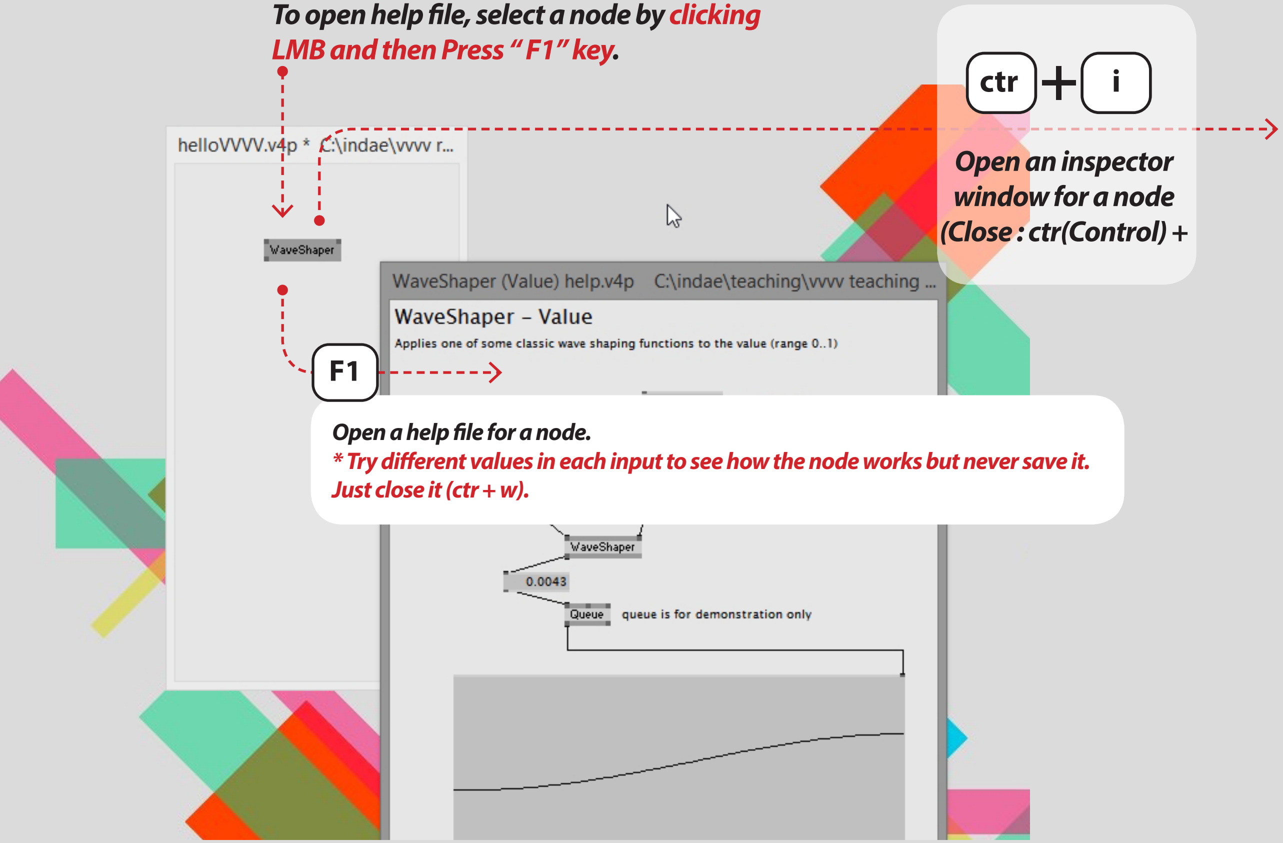
If you want to reset a node, hover mouse ***inside the node and click RMB + alt key***.



07. Help file and Inspector window

The "help file" is useful to understand how a node works and what the node does.

To open help file, select a node by clicking LMB and then Press " F1" key.



helloVVVV.v4p * C:\indae\vvvv r...

WaveShaper (Value) help.v4p C:\indae\teaching\vvvv teaching ...

WaveShaper – Value

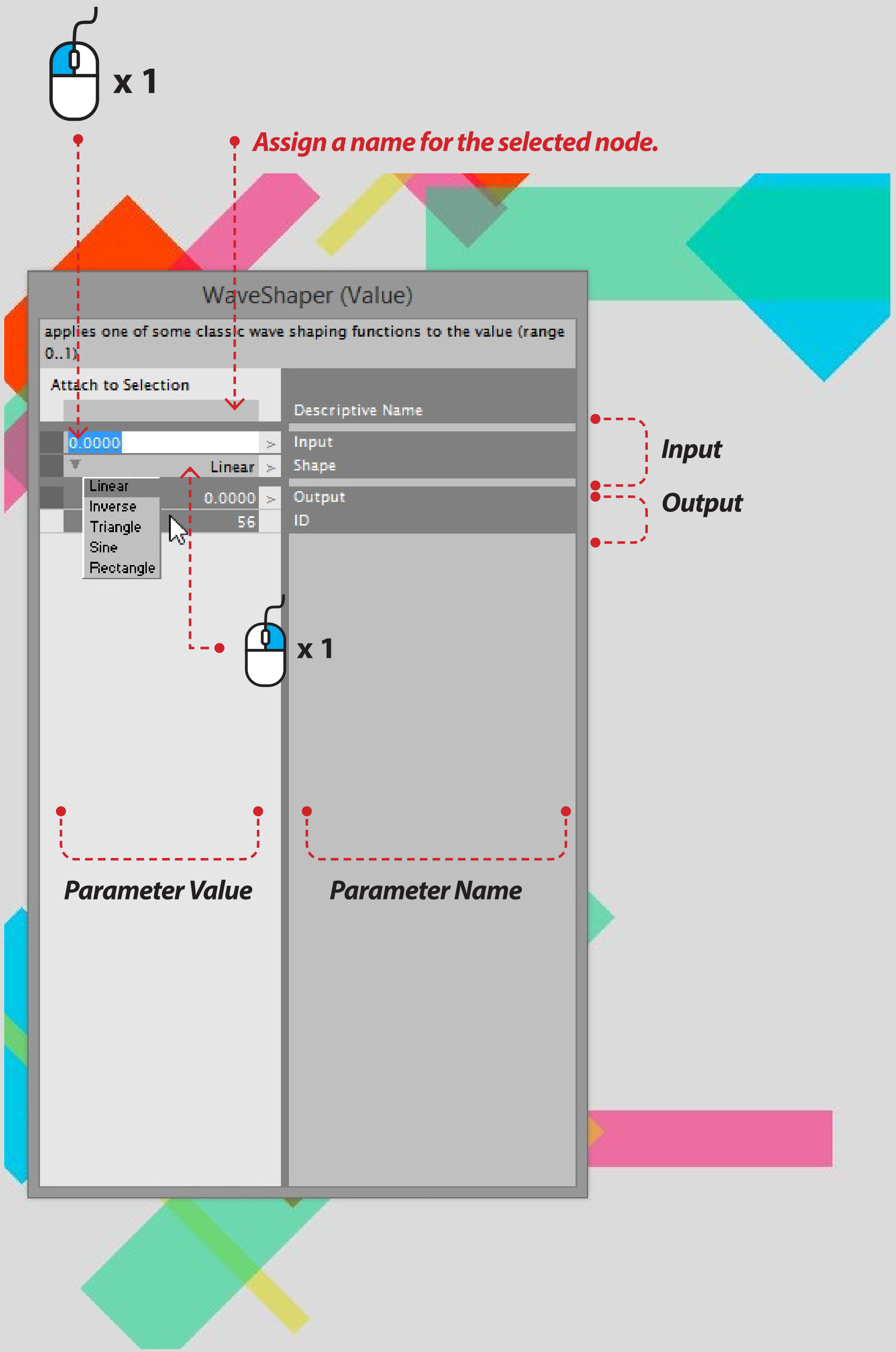
Applies one of some classic wave shaping functions to the value (range 0..1)

queue is for demonstration only

Open an inspector window for a node (Close : ctr(Control) +

Open a help file for a node.

** Try different values in each input to see how the node works but never save it. Just close it (ctr + w).*



WaveShaper (Value)

applies one of some classic wave shaping functions to the value (range 0..1)

Attach to Selection

	Descriptive Name
0.0000	Input
Linear	Shape
0.0000	Output
56	ID

Assign a name for the selected node.

Input

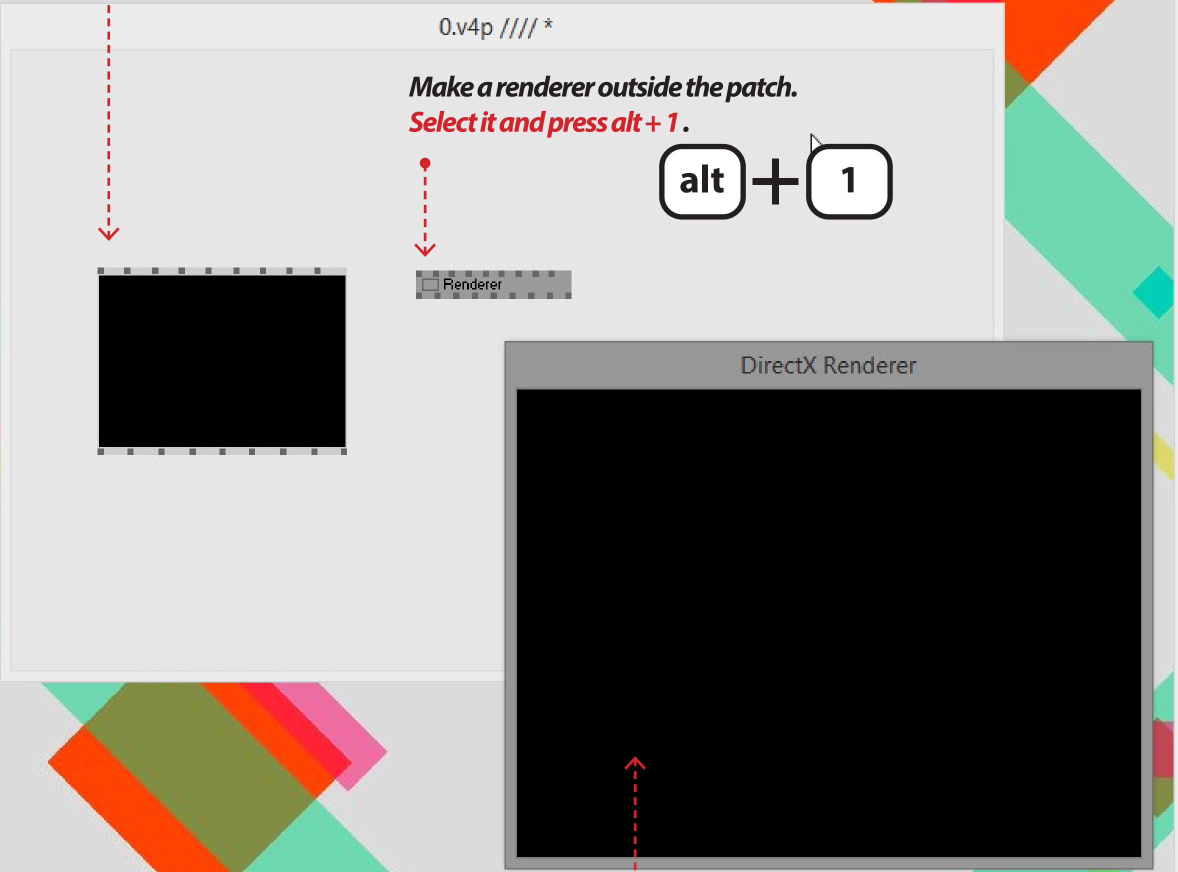
Output

Parameter Value

Parameter Name

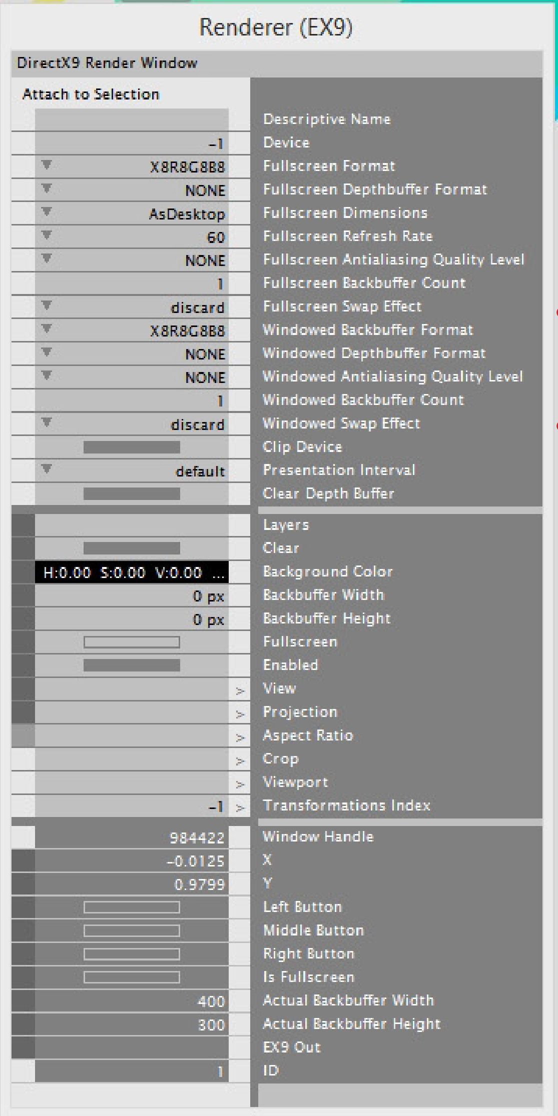
08. Hello *renderer*

alt + 2 ***Make a renderer inside the patch.*** ***Select it and press alt + 2.***



Make a renderer outside the patch. ***Select it and press alt + 1.*** alt + 1

alt + enter ***Make full screen.*** ***Select it and alt + enter / Back to normal (alt + enter).***



Renderer (EX9)

DirectX9 Render Window

Attach to Selection

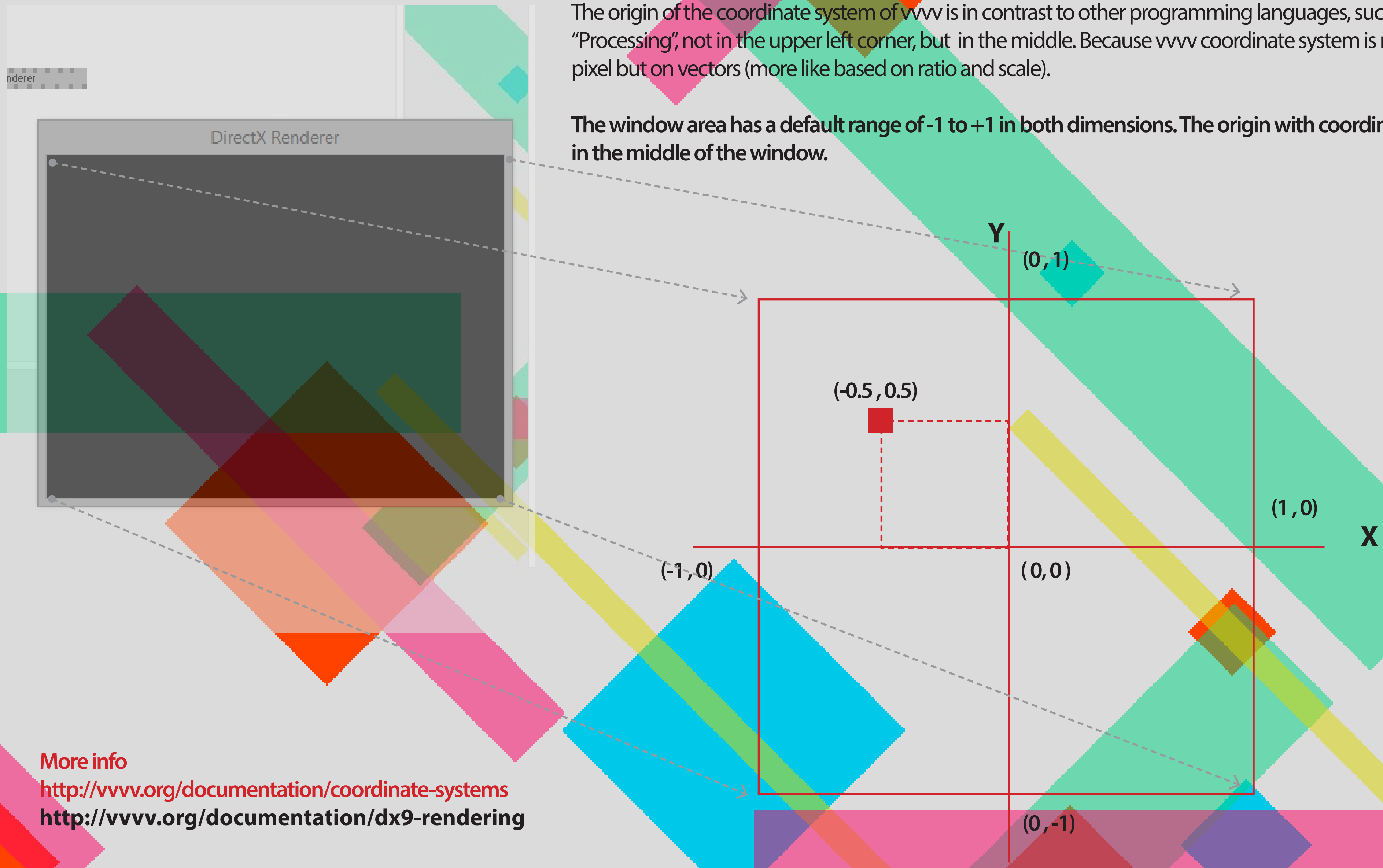
	Descriptive Name
-1	Device
X8R8G8B8	Fullscreen Format
NONE	Fullscreen Depthbuffer Format
AsDesktop	Fullscreen Dimensions
60	Fullscreen Refresh Rate
NONE	Fullscreen Antialiasing Quality Level
1	Fullscreen Backbuffer Count
discard	Fullscreen Swap Effect
X8R8G8B8	Windowed Backbuffer Format
NONE	Windowed Depthbuffer Format
NONE	Windowed Antialiasing Quality Level
1	Windowed Backbuffer Count
discard	Windowed Swap Effect
■	Clip Device
default	Presentation Interval
■	Clear Depth Buffer
	Layers
■	Clear
H:0.00 S:0.00 V:0.00 ...	Background Color
0 px	Backbuffer Width
0 px	Backbuffer Height
□	Fullscreen
■	Enabled
>	View
>	Projection
>	Aspect Ratio
>	Crop
>	Viewport
-1 >	Transformations Index
984422	Window Handle
-0.0125	X
0.9799	Y
□	Left Button
□	Middle Button
□	Right Button
□	Is Fullscreen
400	Actual Backbuffer Width
300	Actual Backbuffer Height
	EX9 Out
1	ID

Anti-aliasing.
Basic setting (4)

08. Hello *renderer: Coordinate System of renderer*

The origin of the coordinate system of vvvv is in contrast to other programming languages, such as "Processing", not in the upper left corner, but in the middle. Because vvvv coordinate system is not on pixel but on vectors (more like based on ratio and scale).

The window area has a default range of -1 to +1 in both dimensions. The origin with coordinate 0 is in the middle of the window.

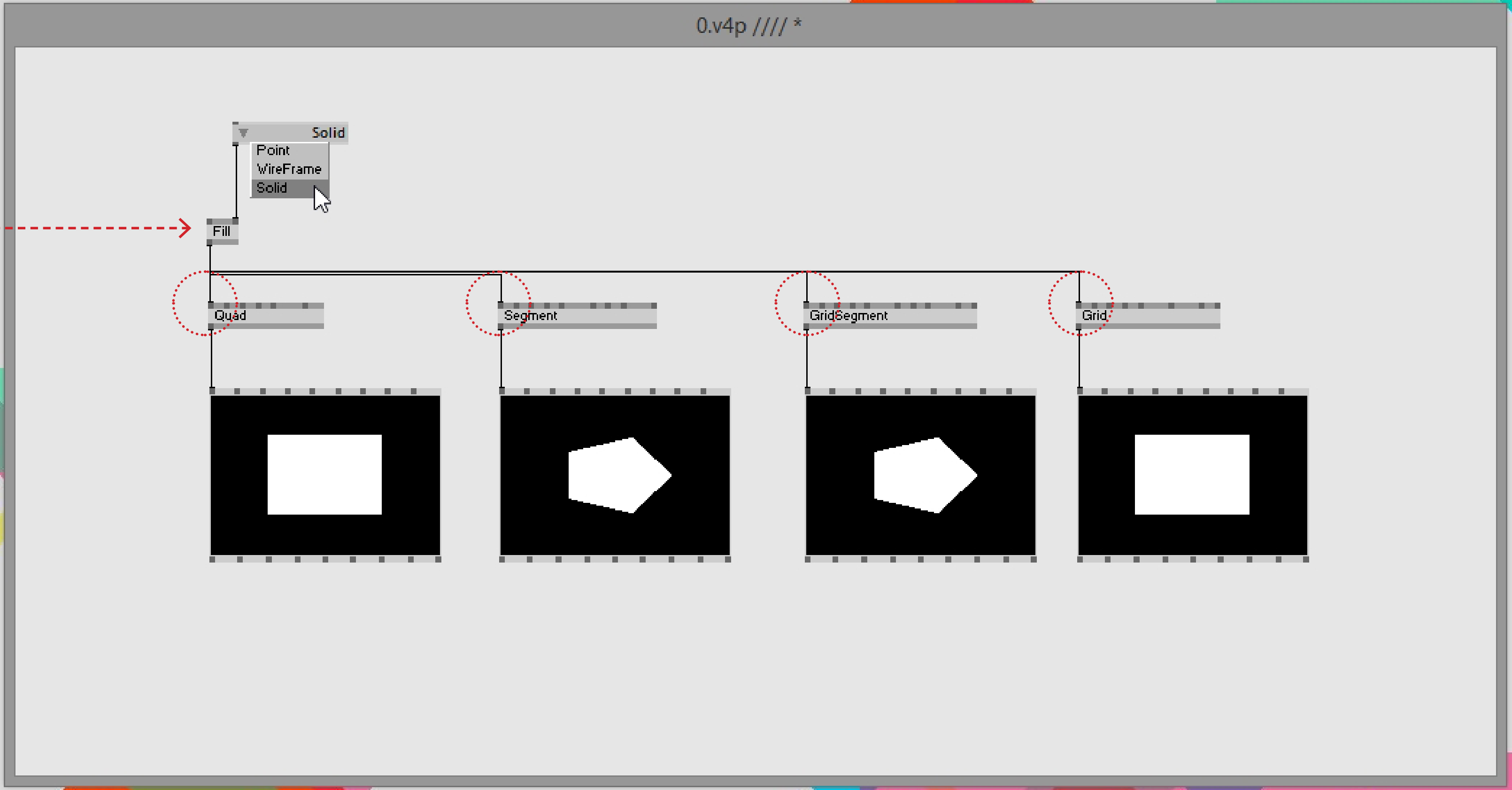


More info
http://vvvv.org/documentation/coordinate-systems
http://vvvv.org/documentation/dx9-rendering

09. Hello *Shapes and Render state*

Fill(EX9 Render state)

Set fill mode to point/ wire frame/ sold



Primitive shapes in 4v

Quad, segment, grid segment, grid, rope, line, pillow

**** Try each node. Create nodes, select them and Press F1 key to see the help file of each node.***

09. Hello *Shapes : Basic use of a shape*

Transform.
Define translate(position), scale(size), rotate, and centerXY(offset).

Render state.
Fill mode: point or wire frame or solid.

Quad.
It is like a rectangle.

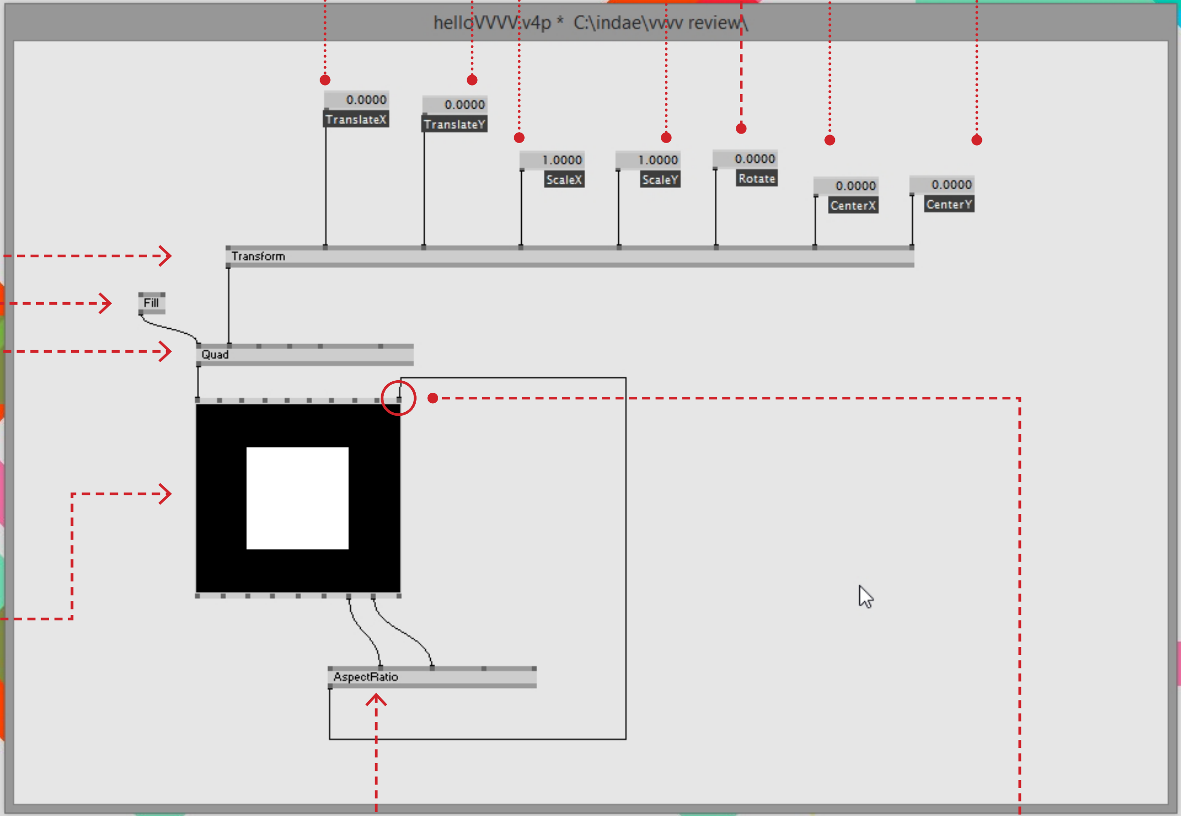
Renderer.
The final destination of the shape.

Translate :Position.

Scale :Size

Rotation

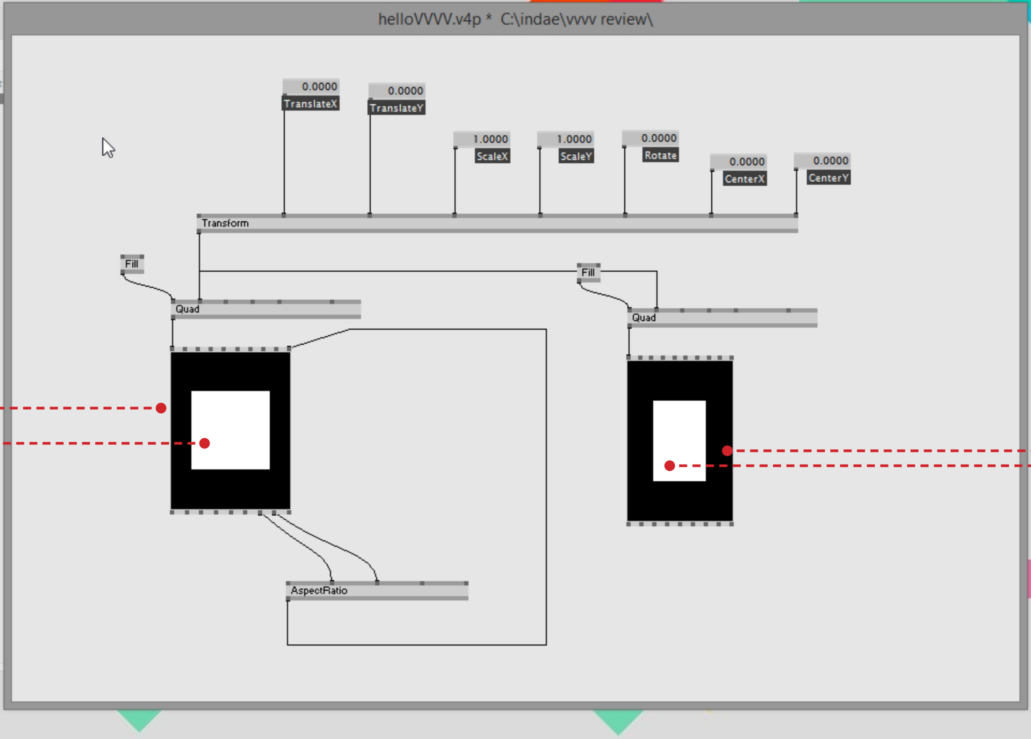
Center XY :Offset

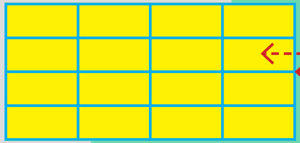


AspectioRatio.
Squeezes the incoming transformation to the given aspect ratio.

** The output of this node needs to be connected the last input pin of the renderer (it is invisible but you can connect it).*

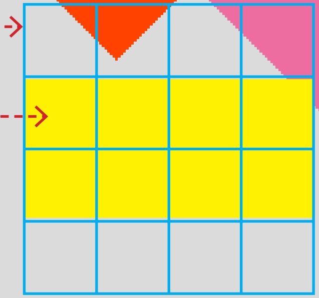
09. Hello *Shapes : AspectRatio*





Without AspectioRatio.

The shape will be stretched according to AspectRatio of renderer.



With AspectioRatio.

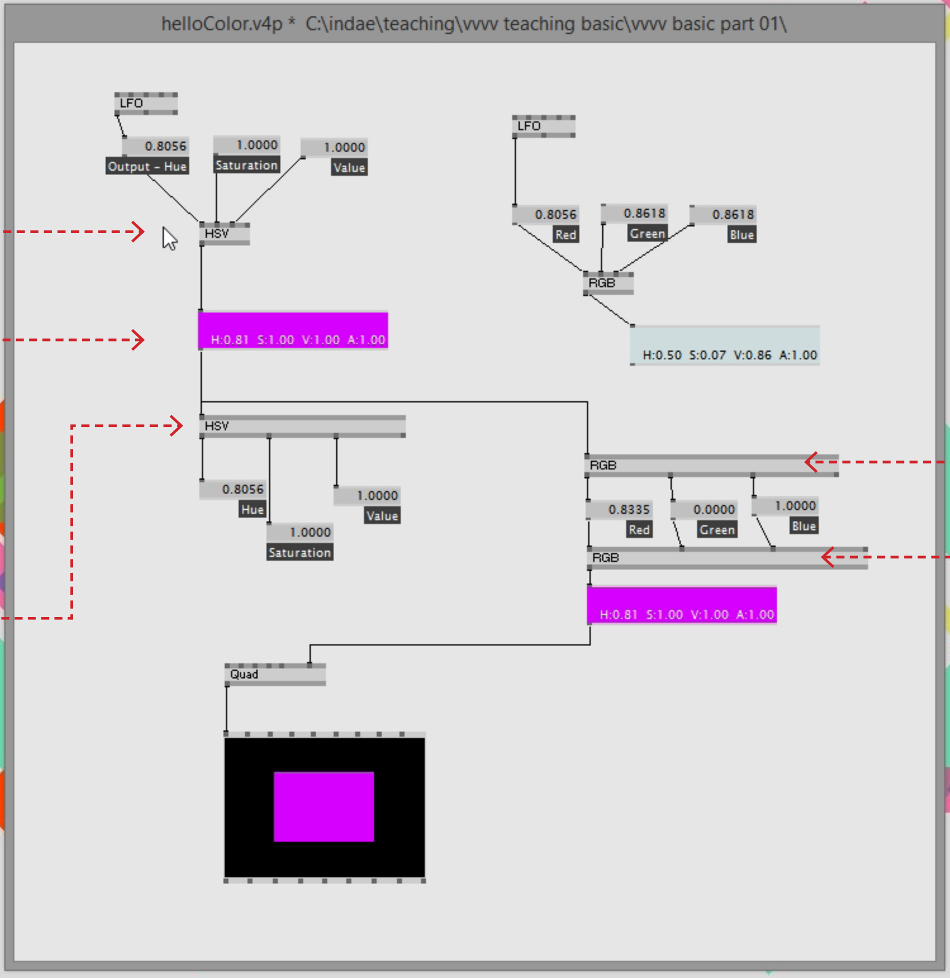
The shape will be stretched but keeping its own AspectioRatio.

10. Hello *Colour*

HSV (joint).
creates color by providing 'Hue', 'Saturation', and Value(Brightness) value.

Color IOBox.
This IOBox here is only to preview the output color. You can directly connect the output of HSV(joint) to other input pins.

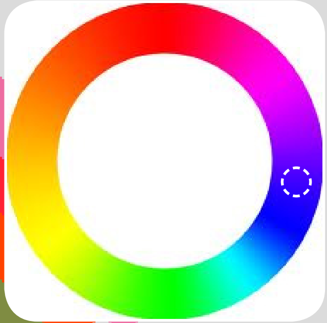
HSV (split).
outputs each value of the input color. Hue, Saturation, value, and Alpha.



helloColor.v4p * C:\indae\teaching\vvvv teaching basic\vvvv basic part 01\

RGB (joint).
creates a color by providing 'Red, 'Green', and Blue value.

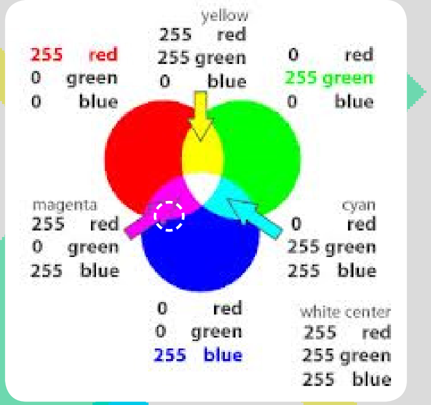
RGB (split).
retrieves the value of 'Red, 'Green', and Blue from the input color.



Hue value determines which colour it will be.

HSV vs. *RGB*

A color comes from the harmony of Red, Green, and Blue. Three values are needed to make the color.



10. Hello *Image*

fileTexture.
Imports an Image as texture.
Click the first pin of this node with RMB to open up a file browser.

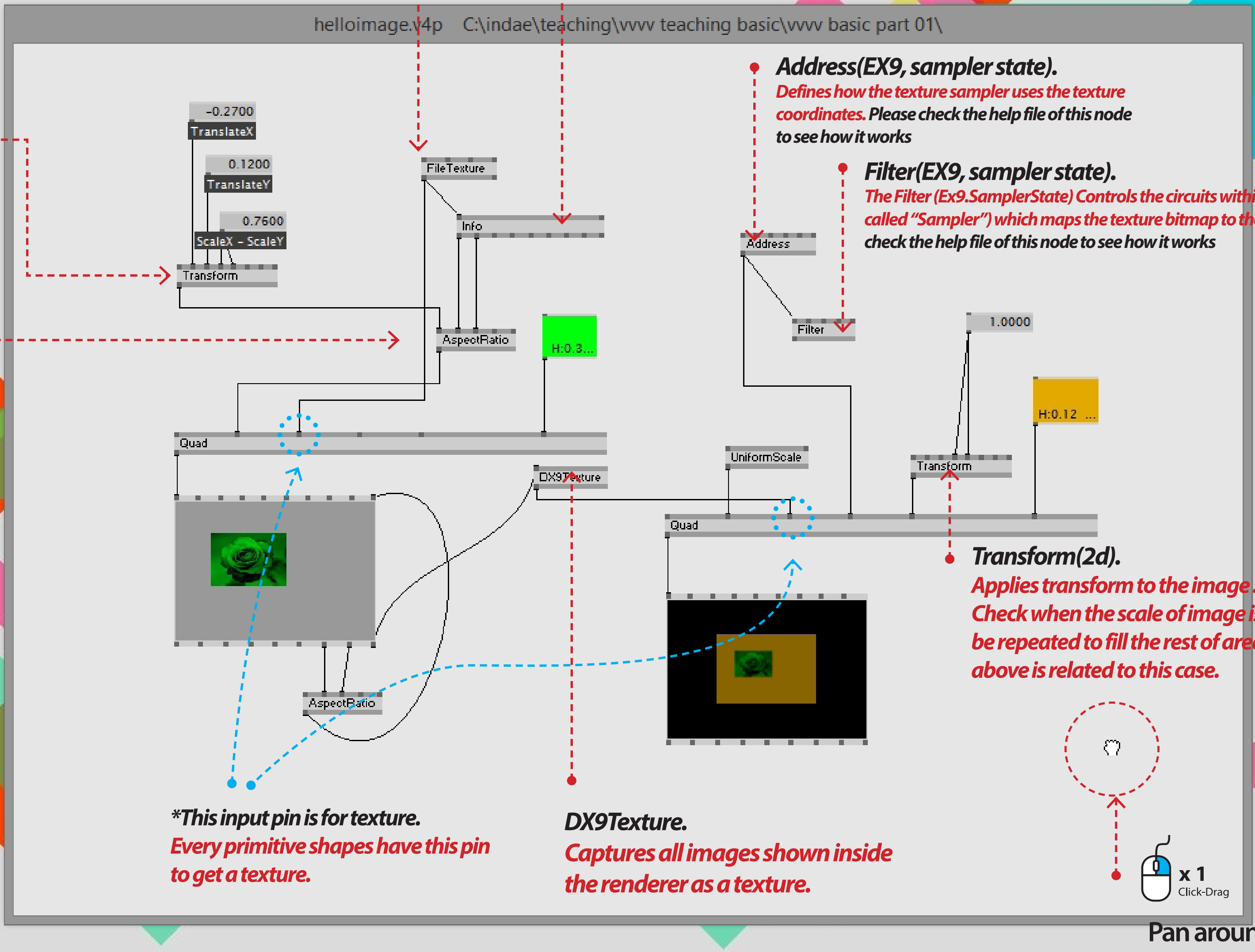
info(texture).
This node gives you detail information of the image such as width and height.

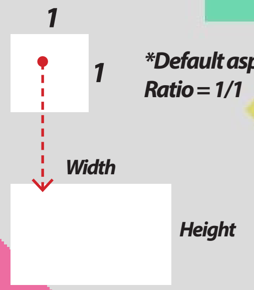
Transform(2d).
It is about where the quad position and how big the quad is.

****AspecRatio(transform).***
****This node is important to keep the ratio of your image .***

Address(EX9, sampler state).
Defines how the texture sampler uses the texture coordinates. Please check the help file of this node to see how it works

Filter(EX9, sampler state).
The Filter (Ex9.SamplerState) Controls the circuits within the graphics card (the so called "Sampler") which maps the texture bitmap to the geometry mesh. Please check the help file of this node to see how it works





****Default aspect ratio of a quad.***
Ratio = 1/1

****Selected image.***
Aspect ratio = height/width
This value needs to be applied to rescale the quad. AspectRatio node does this job for you.

****This input pin is for texture.***
Every primitive shapes have this pin to get a texture.

DX9Texture.
Captures all images shown inside the renderer as a texture.

Transform(2d).
Applies transform to the image.
Check when the scale of image is smaller than 1. The image will be repeated to fill the rest of area in the quad. The node address above is related to this case.

x 1
Click-Drag

Pan around the patch window.

10. Hello *Text*

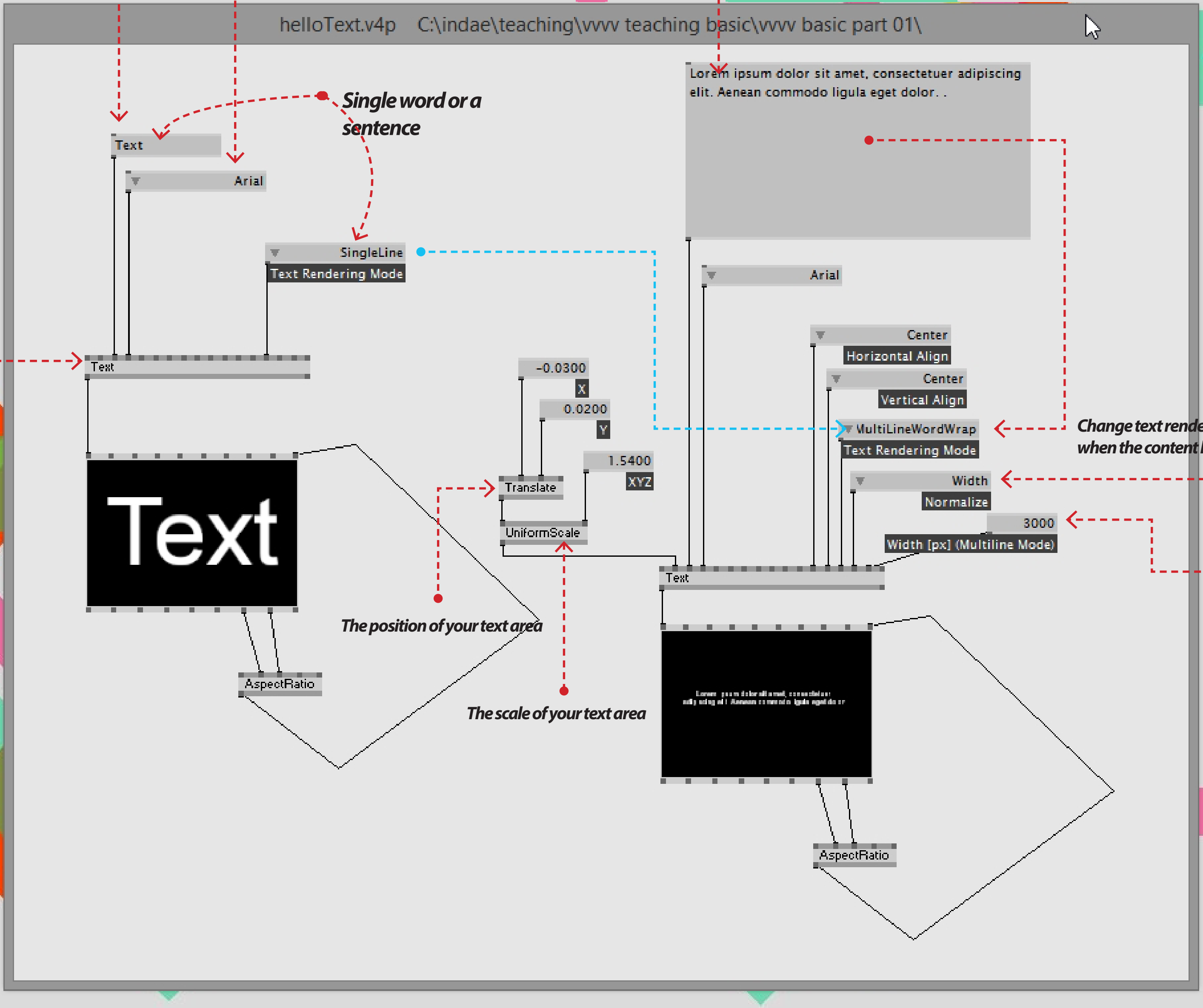
Contents.
Text IOBox

Choose a font

Multi line contents
in TextIOBox

Text (EX9).
Draws text in the renderer.

Single word or a sentence



helloText.v4p   C:\indae\teaching\vvvv teaching basic\vvvv basic part 01\

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. .

SingleLine
Text Rendering Mode

Center
Horizontal Align

Center
Vertical Align

MultiLineWordWrap
Text Rendering Mode

Width
Normalize

3000
Width [px] (Multiline Mode)

-0.0300
X

0.0200
Y

1.5400
XYZ

The position of your text area

The scale of your text area

Change text rendering Mode when the content has multi lines.

Change normalise to width with multi line contents

It will determine the word limit for each line.

10. Hello *input: mouse*

Mouse X.

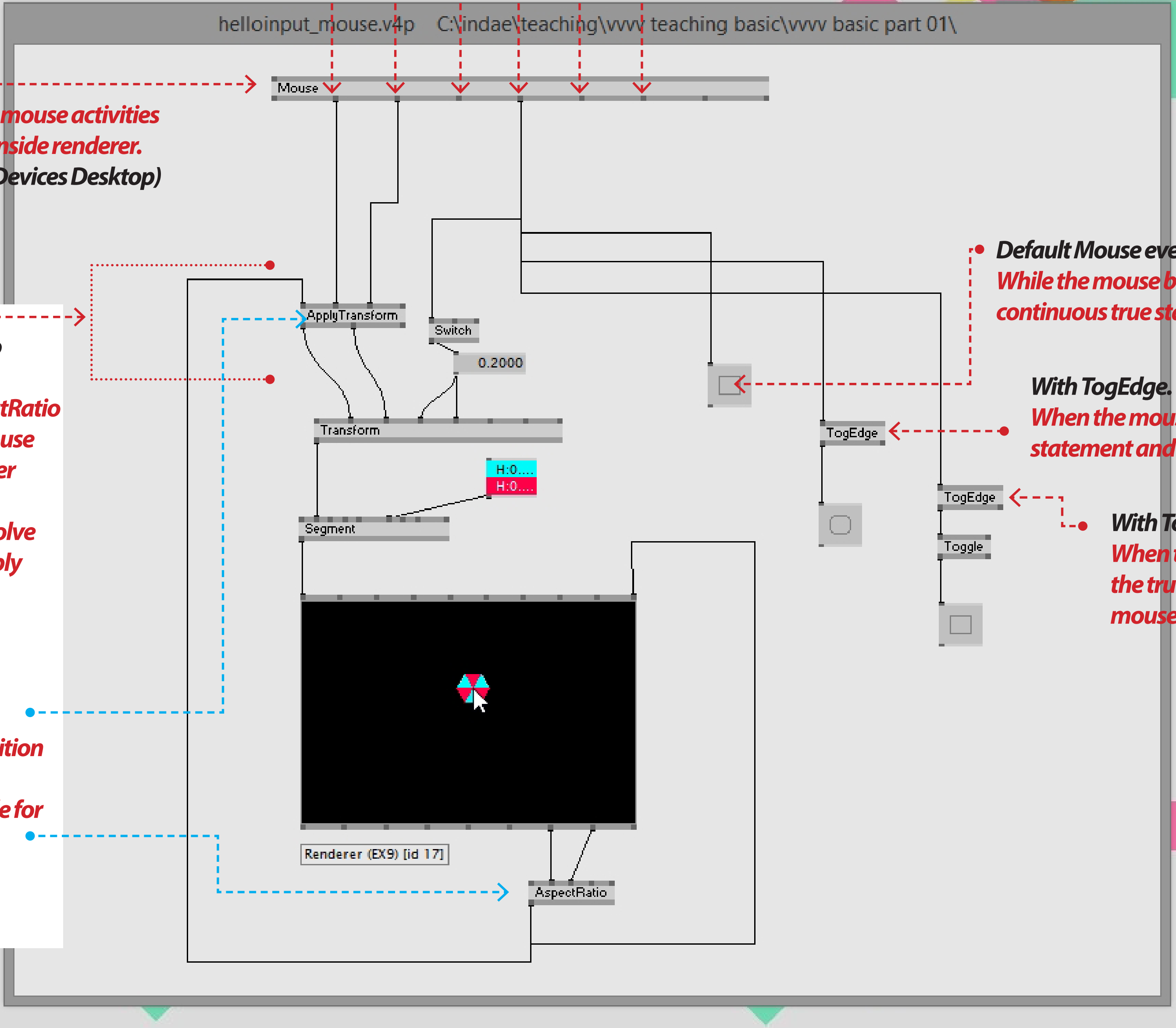
Mouse Y.

Wheel value

Left mouse button

Middle mouse button

Right mouse button



helloinput_mouse.v4p   C:\indae\teaching\vvvv teaching basic\vvvv basic part 01\

Mouse(window).
This node detects the mouse activities when mouse moves inside renderer.
-> Compare: Mouse(Devices Desktop)

Re-map your Mouse position in relation to AspectRatio.
When you connect the AspectRatio node to the renderer, the mouse coordination inside the render differs from your expected mouse position. In order to solve this problem you need to apply AspectRatio to your mouse position as well.

The node ApplyTransform calculates the translated position by AspectRatio.
** More info: check the help file for ApplyTransform.*

Default Mouse event.
While the mouse button is being pressed, it creates the continuous true statement : "On" state.

With TogEdge.
When the mouse button is pressed, it creates the true statement and back to false: Like "Bang".

With TogEdge + Toggle.
When the mouse button is pressed, it creates the true statement and stay that state until the mouse button is pressed again: Like "Toggle".

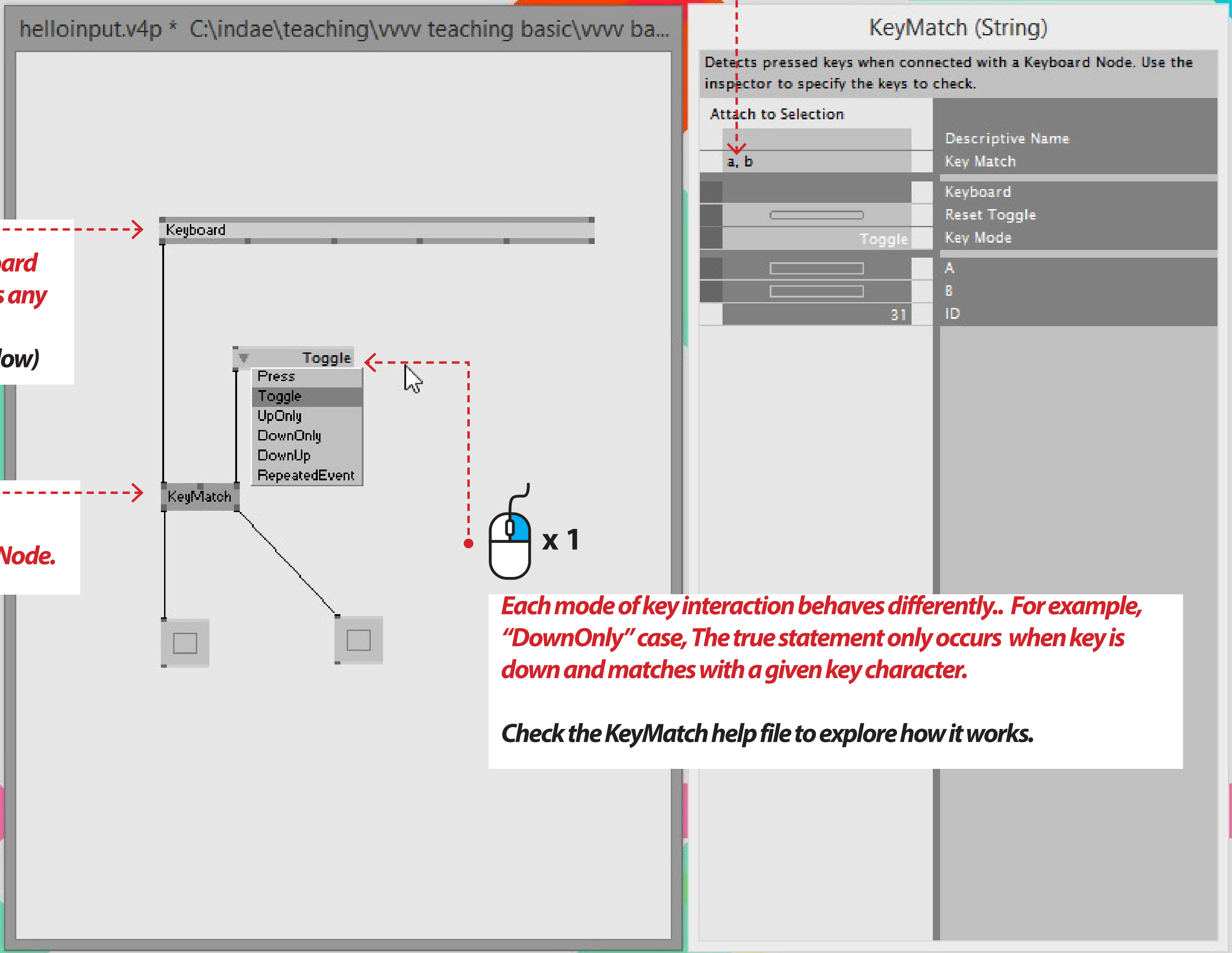
10. Hello *input: Keyboard*

x 1
Click & Type

Use the inspector to specify the keys to be checked.

Keyboard(Global).
This node detects your keyboard activities whenever you press any keyboard key.
-> Compare : Keyboard(window)

KeyMatch.
Detects pressed keys when connected with a Keyboard Node.


helloinput.v4p * C:\indae\teaching\vvvv teaching basic\vvvv ba...

KeyMatch (String)

Detects pressed keys when connected with a Keyboard Node. Use the inspector to specify the keys to check.

Attach to Selection

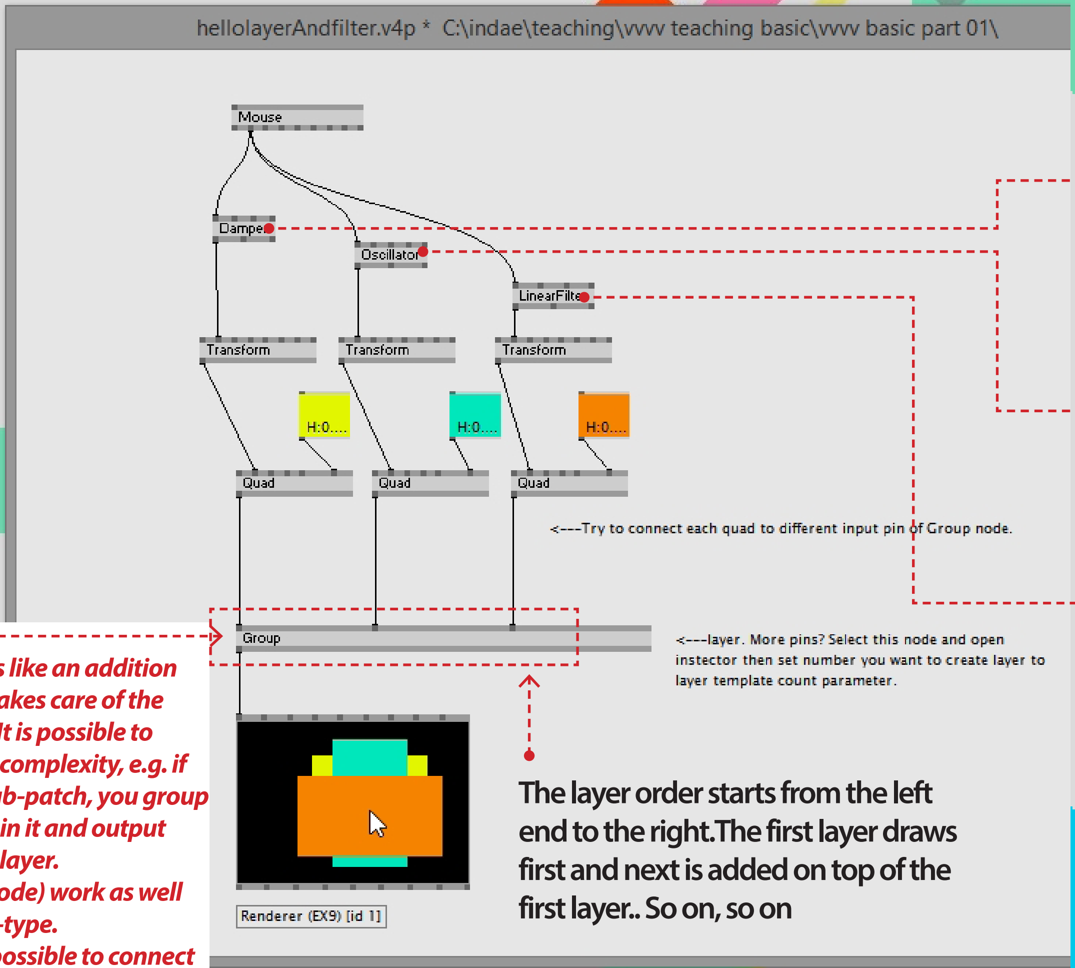
	Descriptive Name
a, b	Key Match
	Keyboard
	Reset Toggle
Toggle	Key Mode
	A
	B
31	ID

x 1

Each mode of key interaction behaves differently.. For example, "DownOnly" case, The true statement only occurs when key is down and matches with a given key character.

Check the KeyMatch help file to explore how it works.

11. Hello *Layers and animation fileters.*



<---Try to connect each quad to different input pin of Group node.

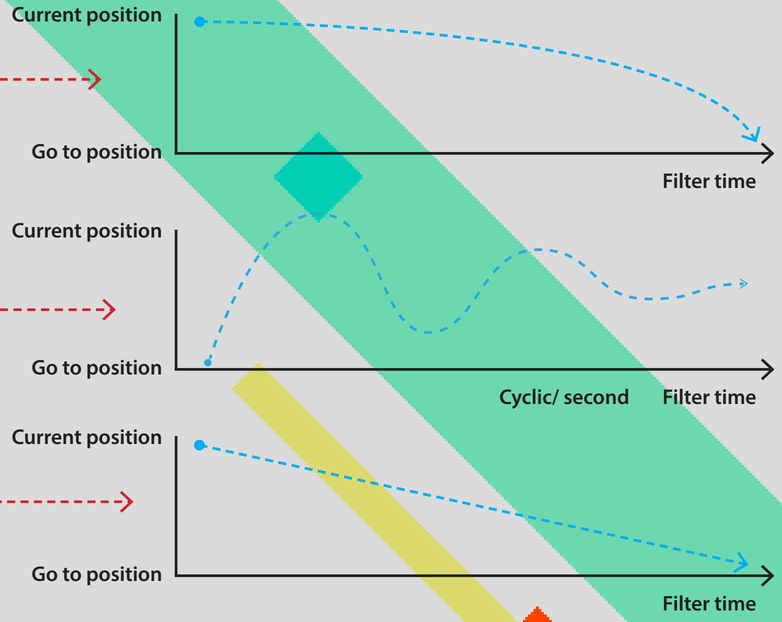
<---layer. More pins? Select this node and open instector then set number you want to create layer to layer template count parameter.

The layer order starts from the left end to the right.The first layer draws first and next is added on top of the first layer.. So on, so on

Group.
The Group node is like an addition for Layers and it takes care of the drawing priority. It is possible to build trees of any complexity, e.g. if you are using a sub-patch, you group all render objects in it and output only the grouped layer.
S (Node) and R (Node) work as well for the layer data-type.
Furthermore it's possible to connect one layer to many grouped nodes .

Add more layers
Open the inspector window and change the value of layer template count.

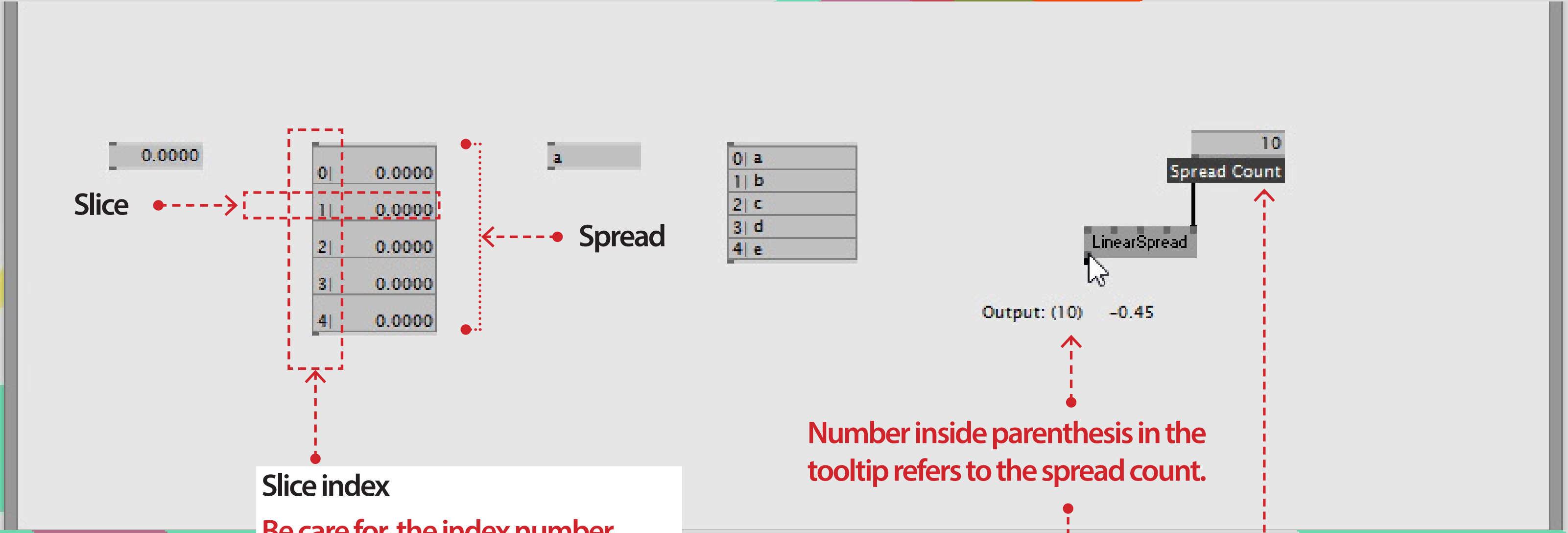
Diagram of Animation filter.



Animation filters, such as "Damper", "Oscillator", and "LinearFilter", apply a force to the input value which prevents the output to reach the new value for a given time.

*** Check input pin of each node and try to set new value to input pin. -> See help file (F1) for Damper, Oscillator, and LinearFilter.**

12. Hello *Spread-Basic.*



Number inside parenthesis in the tooltip refers to the spread count.

Slice index

Be care for, the index number always starts from “0”

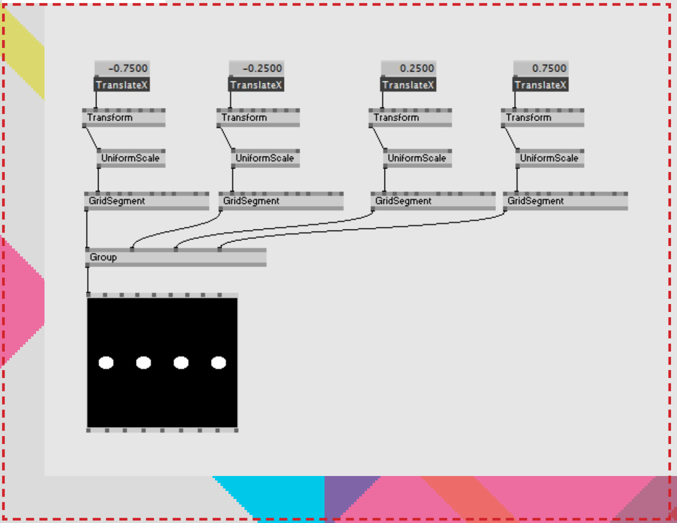
**Slice count = 5*

Why Spread?

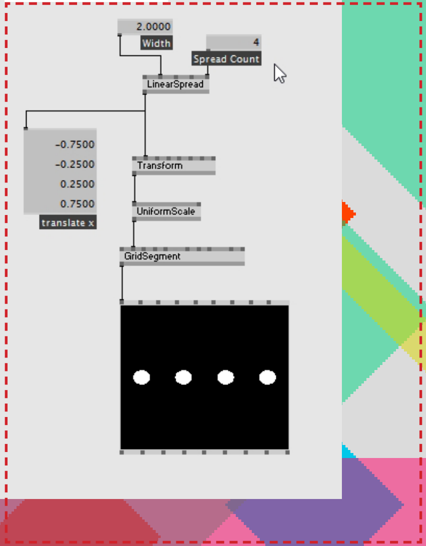
Case A and B produce the same graphics. However the ways to produce the result are different. Do you think which one is more effective?

Case B uses a spread. It is much easy to handle and mange input data.

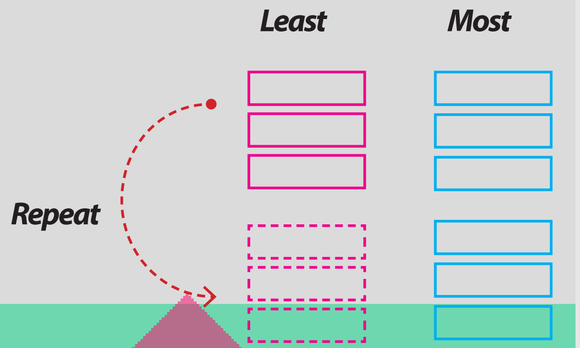
Case A



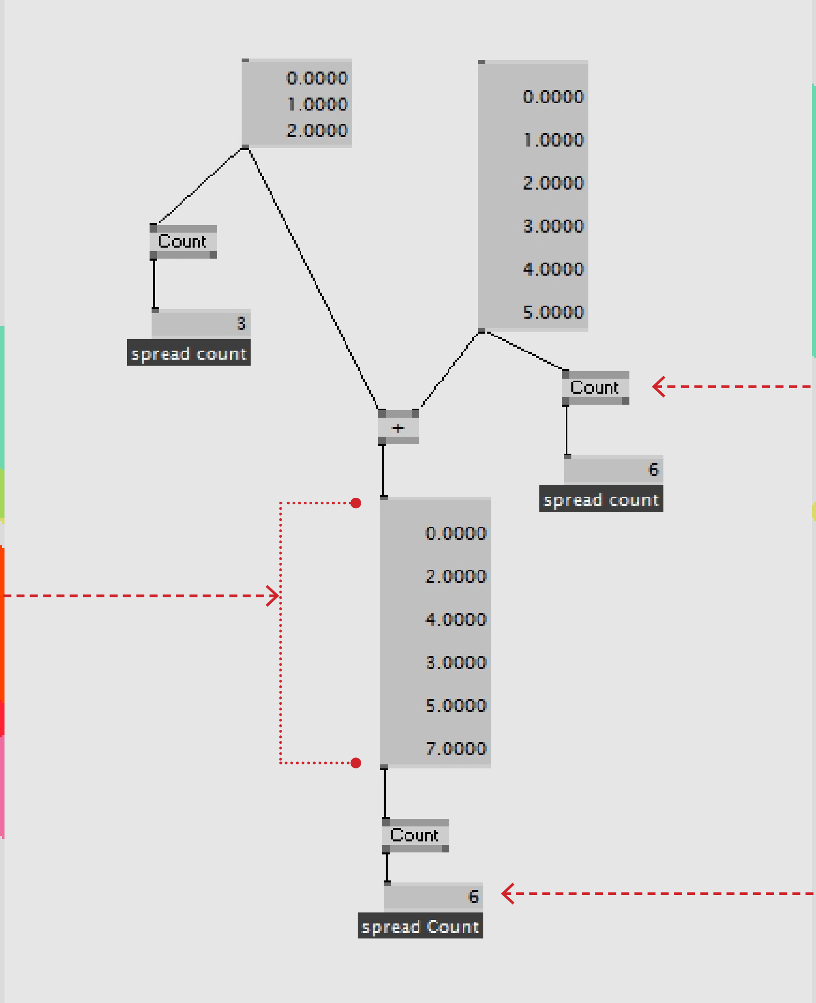
Case B



12. Hello *Spread-Basic.*



***Look at the result.
How two spreads have added up each others. The least count spread is repeated again to match the most count one.***

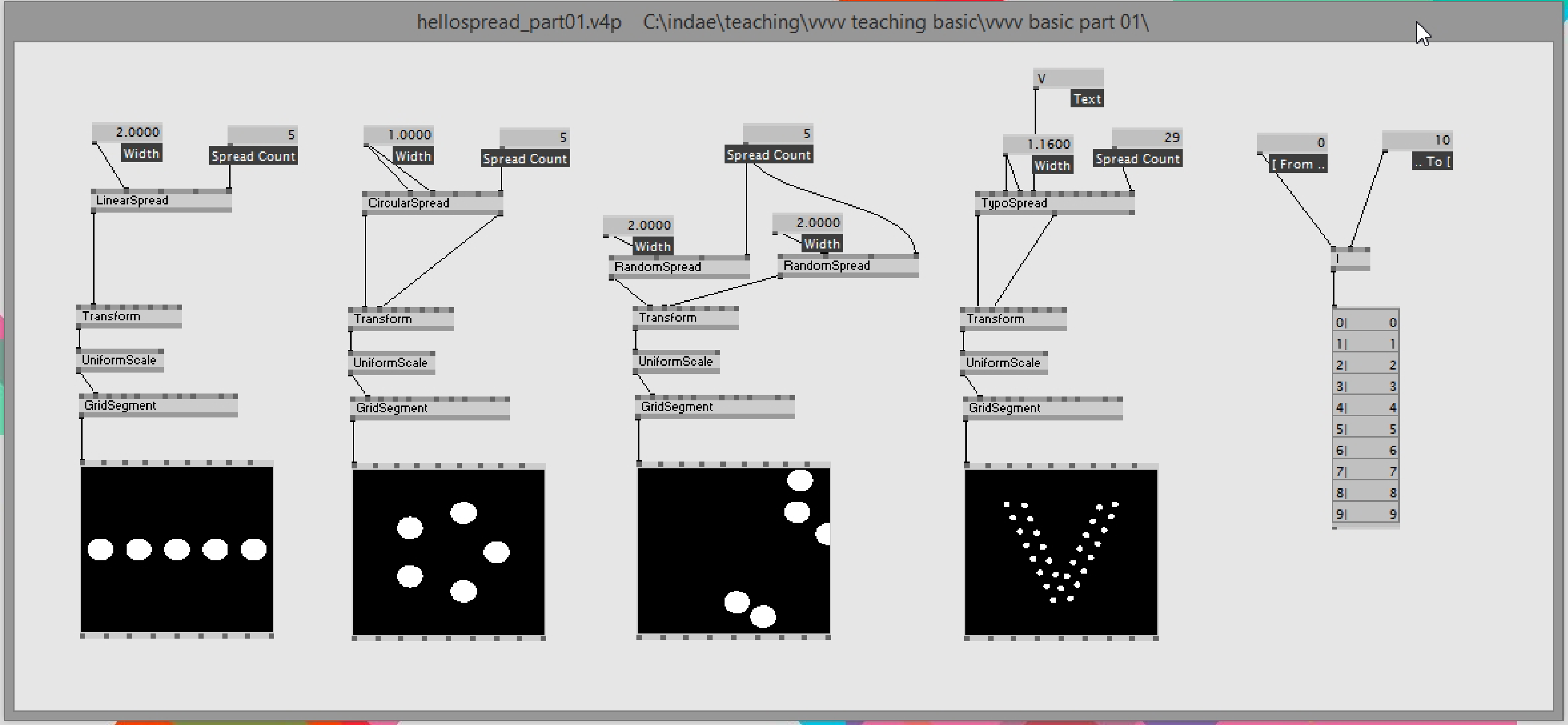


Count(value).
The node counts the number of slice count when input spreads.

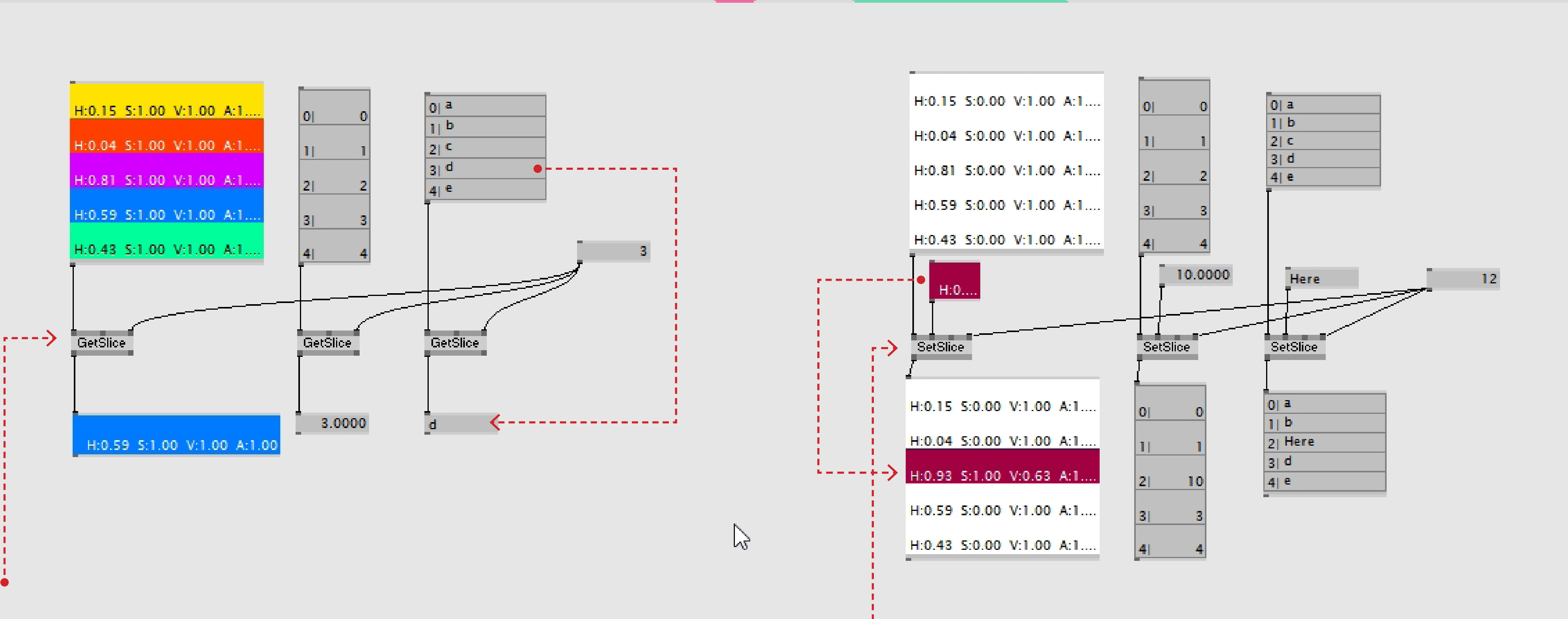
The spread count of the end result is always same as the spread count of the most one.

12. Hello *Spread-Basic nodes for create spread.*

*** Please Check help files of Linearspread, circularspread, randomspread, typospread, and i spread.**



12. Hello *Spread-Basic nodes for create spread.*


H:0.15 S:1.00 V:1.00 A:1....
H:0.04 S:1.00 V:1.00 A:1....
H:0.81 S:1.00 V:1.00 A:1....
H:0.59 S:1.00 V:1.00 A:1....
H:0.43 S:1.00 V:1.00 A:1....
0| 0
1| 1
2| 2
3| 3
4| 4
0| a
1| b
2| c
3| d
4| e
3
GetSlice
GetSlice
GetSlice
H:0.59 S:1.00 V:1.00 A:1.00
3.0000
d
H:0.15 S:0.00 V:1.00 A:1....
H:0.04 S:0.00 V:1.00 A:1....
H:0.81 S:0.00 V:1.00 A:1....
H:0.59 S:0.00 V:1.00 A:1....
H:0.43 S:0.00 V:1.00 A:1....
0| 0
1| 1
2| 2
3| 3
4| 4
0| a
1| b
2| c
3| d
4| e
H:0....
10.0000
Here
12
SetSlice
SetSlice
SetSlice
H:0.15 S:0.00 V:1.00 A:1....
H:0.04 S:0.00 V:1.00 A:1....
H:0.93 S:1.00 V:0.63 A:1....
H:0.59 S:0.00 V:1.00 A:1....
H:0.43 S:0.00 V:1.00 A:1....
0| 0
1| 1
2| 10
3| 3
4| 4
0| a
1| b
2| Here
3| d
4| e

GetSlice.

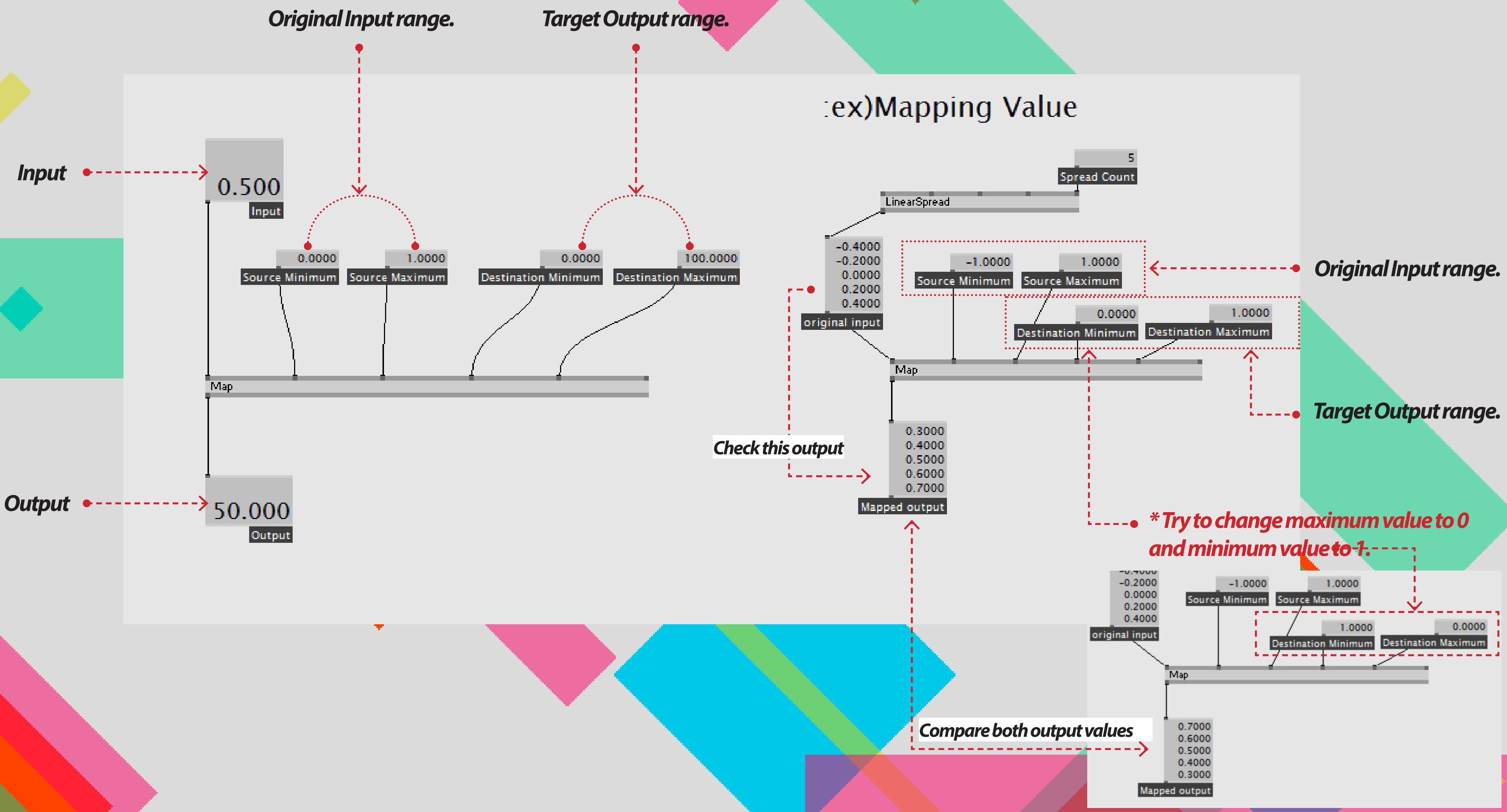
Gets all slices specified in the index input from the input spread

Setslice

Gets all slices specified in the index input from the input spread

13. Hello *Map.*

Maps the value in the given range to a proportional value in the given output range.



Original Input range.

Target Output range.

Input

Output

Check this output

Original Input range.

Target Output range.

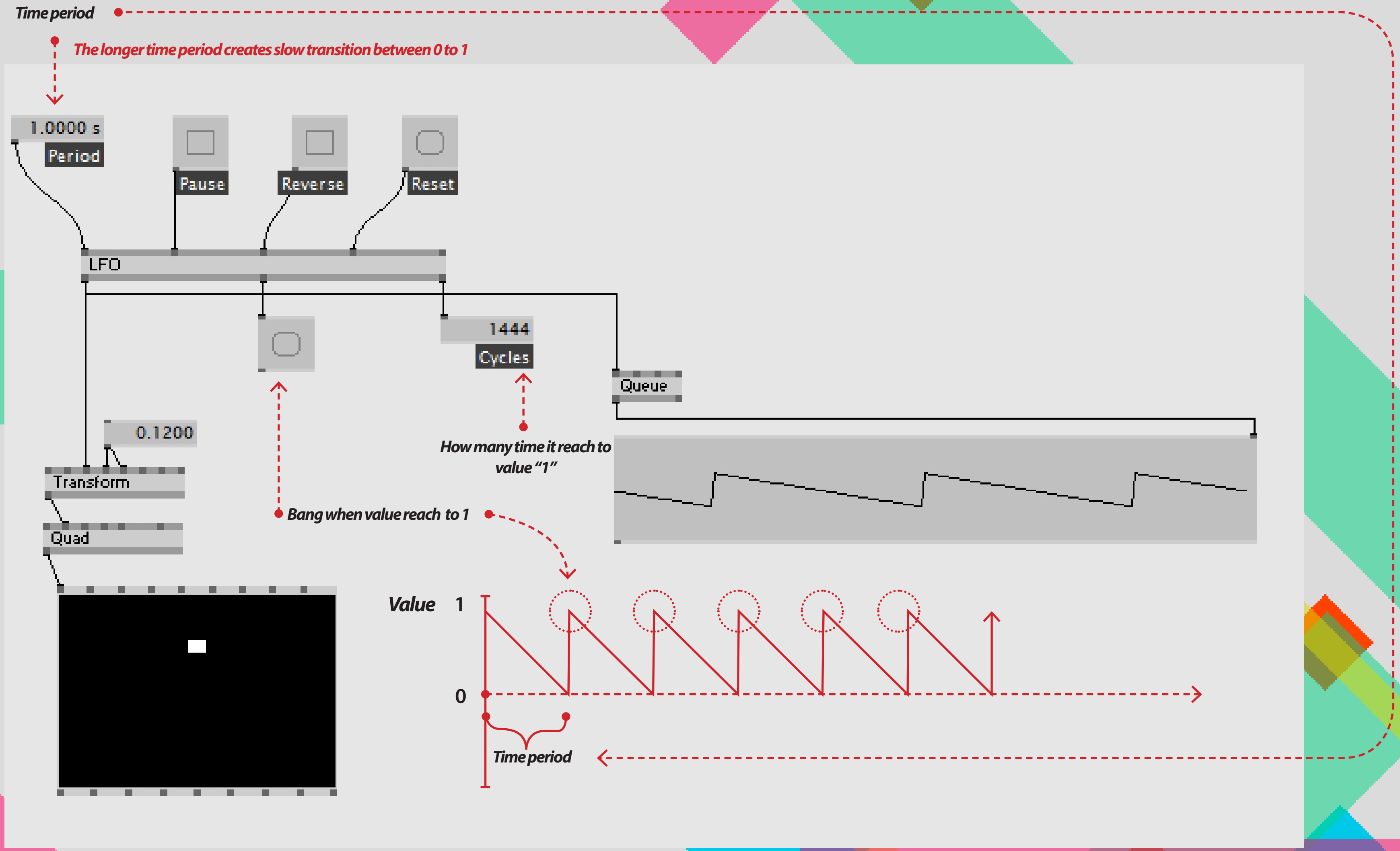
****Try to change maximum value to 0 and minimum value to 1.***

Compare both output values

ex)Mapping Value

14. Hello *LFO.*

Creates a changing value, going linearly from 0 to 1 and jumping back to 0. To change the shape, you can use a Waveshaper (Value).



Time period

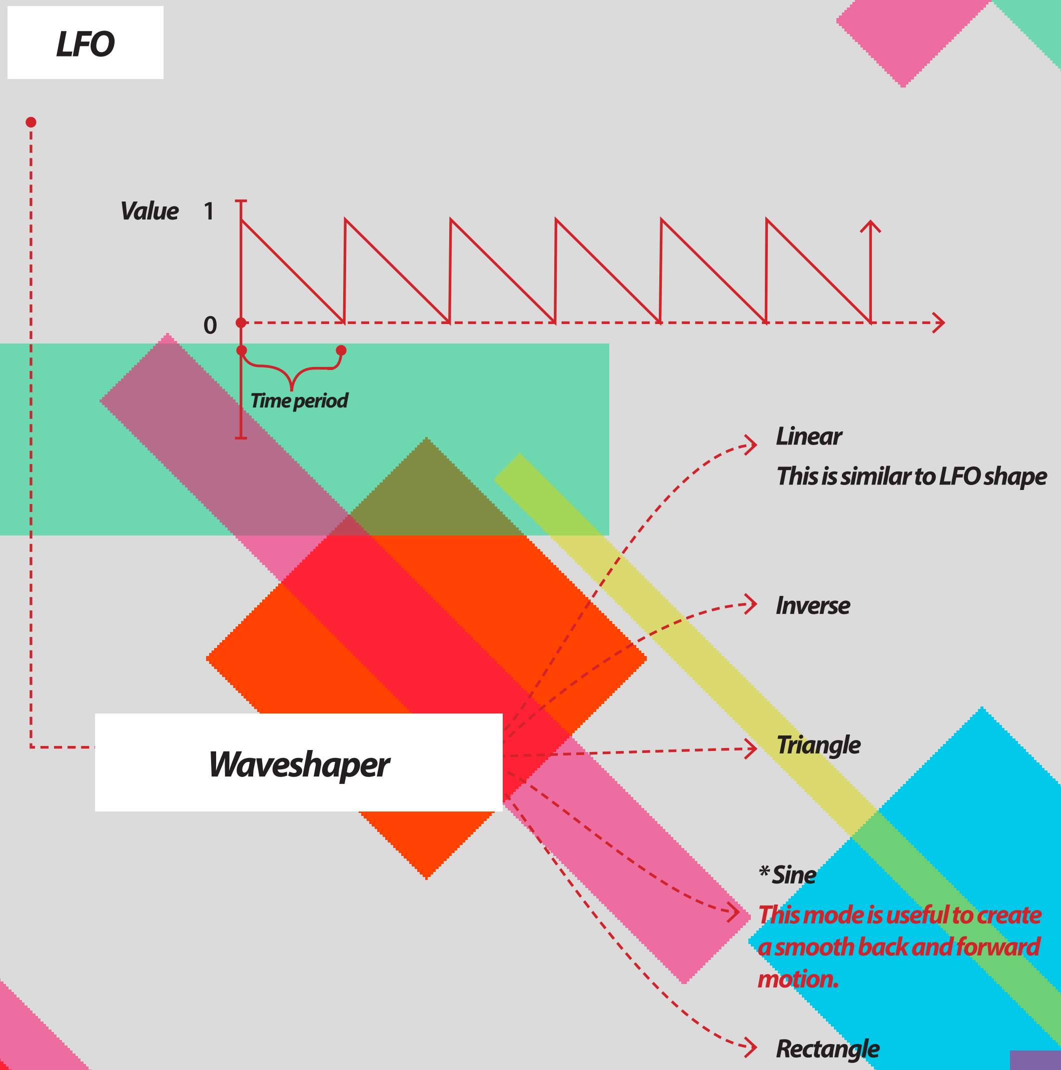
The longer time period creates slow transition between 0 to 1

How many time it reach to value "1"

Bang when value reach to 1

15. Hello *Waveshaper.*

Applies one of some classic wave shaping functions to the value (range 0..1)



LFO

Waveshaper

Linear

This is similar to LFO shape

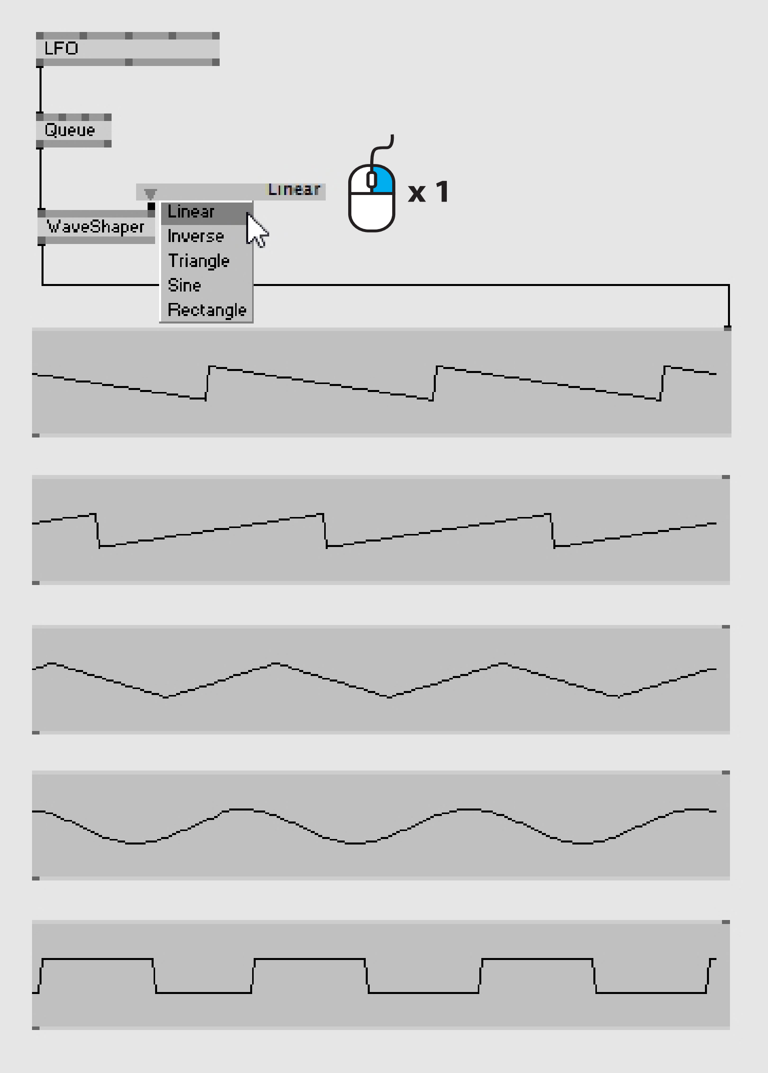
Inverse

Triangle

**** Sine***

This mode is useful to create a smooth back and forward motion.

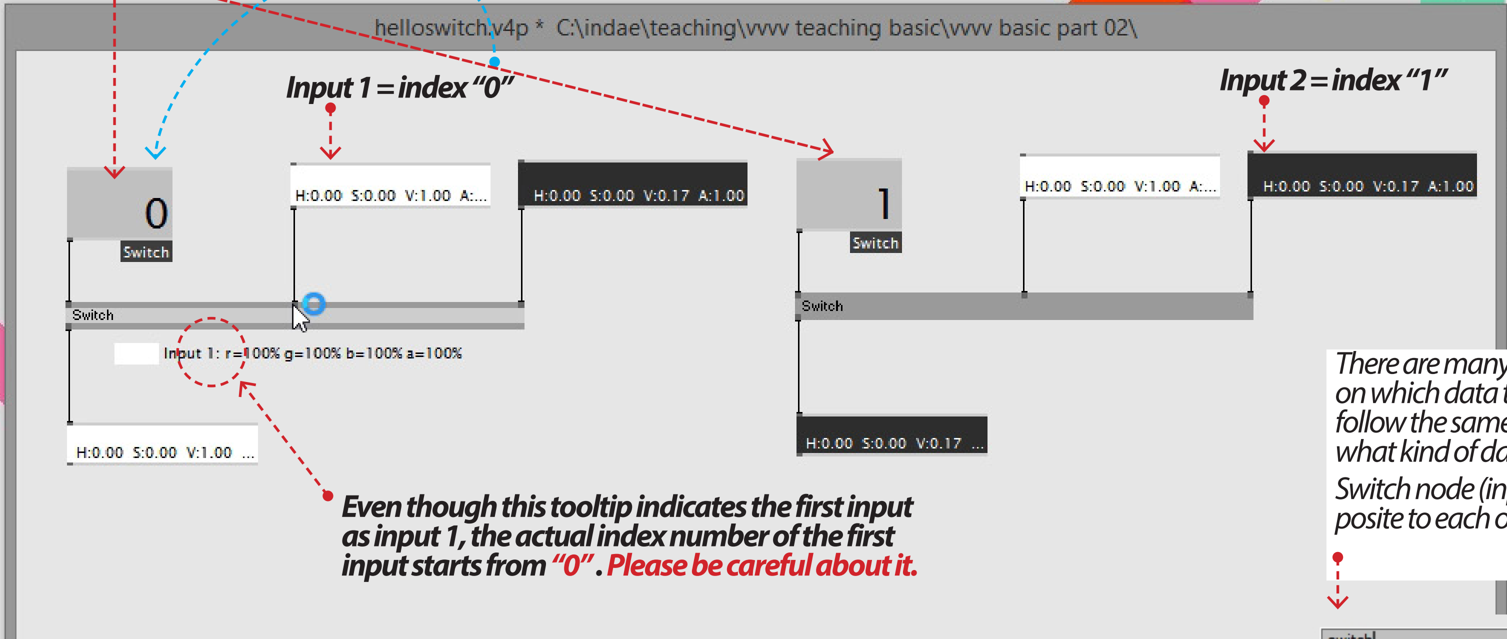
Rectangle



15. Hello *Switch.*

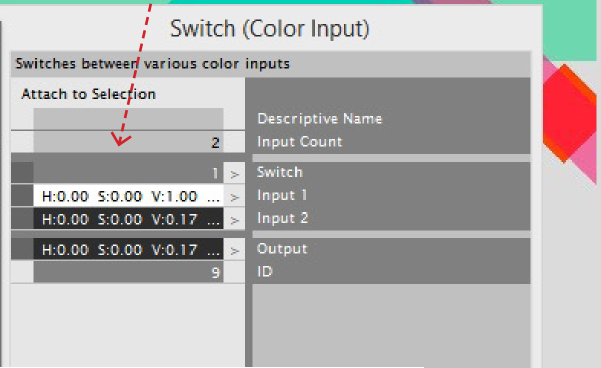
Switches between various inputs

This number decides which input goes to output


helloswitch.v4p * C:\indae\teaching\vvvv teaching basic\vvvv basic part 02\
Input 1 = index "0"
Input 2 = index "1"
0
Switch
H:0.00 S:0.00 V:1.00 A:...
H:0.00 S:0.00 V:0.17 A:1.00
Switch
Input 1: r=100% g=100% b=100% a=100%
H:0.00 S:0.00 V:1.00 ...
1
Switch
H:0.00 S:0.00 V:1.00 A:...
H:0.00 S:0.00 V:0.17 A:1.00
Switch
H:0.00 S:0.00 V:0.17 ...

Even though this tooltip indicates the first input as input 1, the actual index number of the first input starts from "0". Please be careful about it.

Add more inputs x 1 Click & Type



Switch (Color Input)

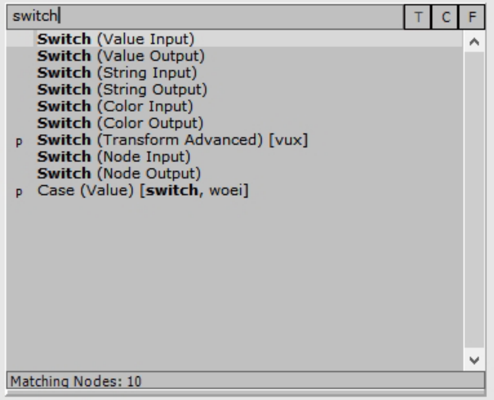
Switches between various color inputs

Attach to Selection

	Descriptive Name
2	Input Count
1	Switch
H:0.00 S:0.00 V:1.00 ...	Input 1
H:0.00 S:0.00 V:0.17 ...	Input 2
H:0.00 S:0.00 V:0.17 ...	Output
9	ID

There are many switch nodes you can select, based on which data type you are working on. All the node follow the same basic logic. The only difference is what kind of data type a switch node accepts.

Switch node (input) and switch node (output) are opposite to each other. Check the switch node (output).



switch| T C F

```
  Switch (Value Input)
  Switch (Value Output)
  Switch (String Input)
  Switch (String Output)
  Switch (Color Input)
  Switch (Color Output)
p Switch (Transform Advanced) [vux]
  Switch (Node Input)
  Switch (Node Output)
p Case (Value) [switch, woei]
```

Matching Nodes: 10

Diagram for switch node (input)

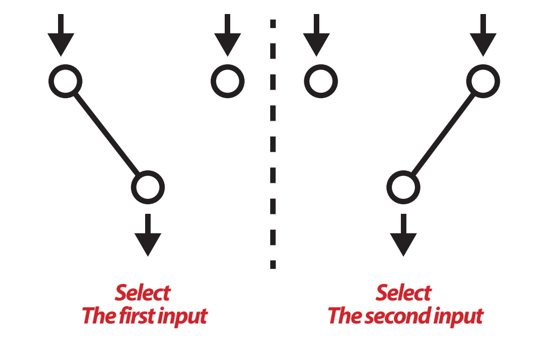
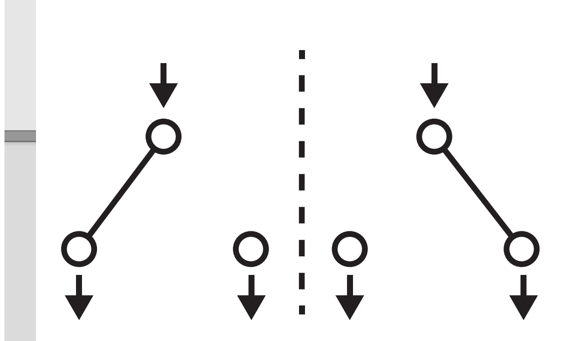
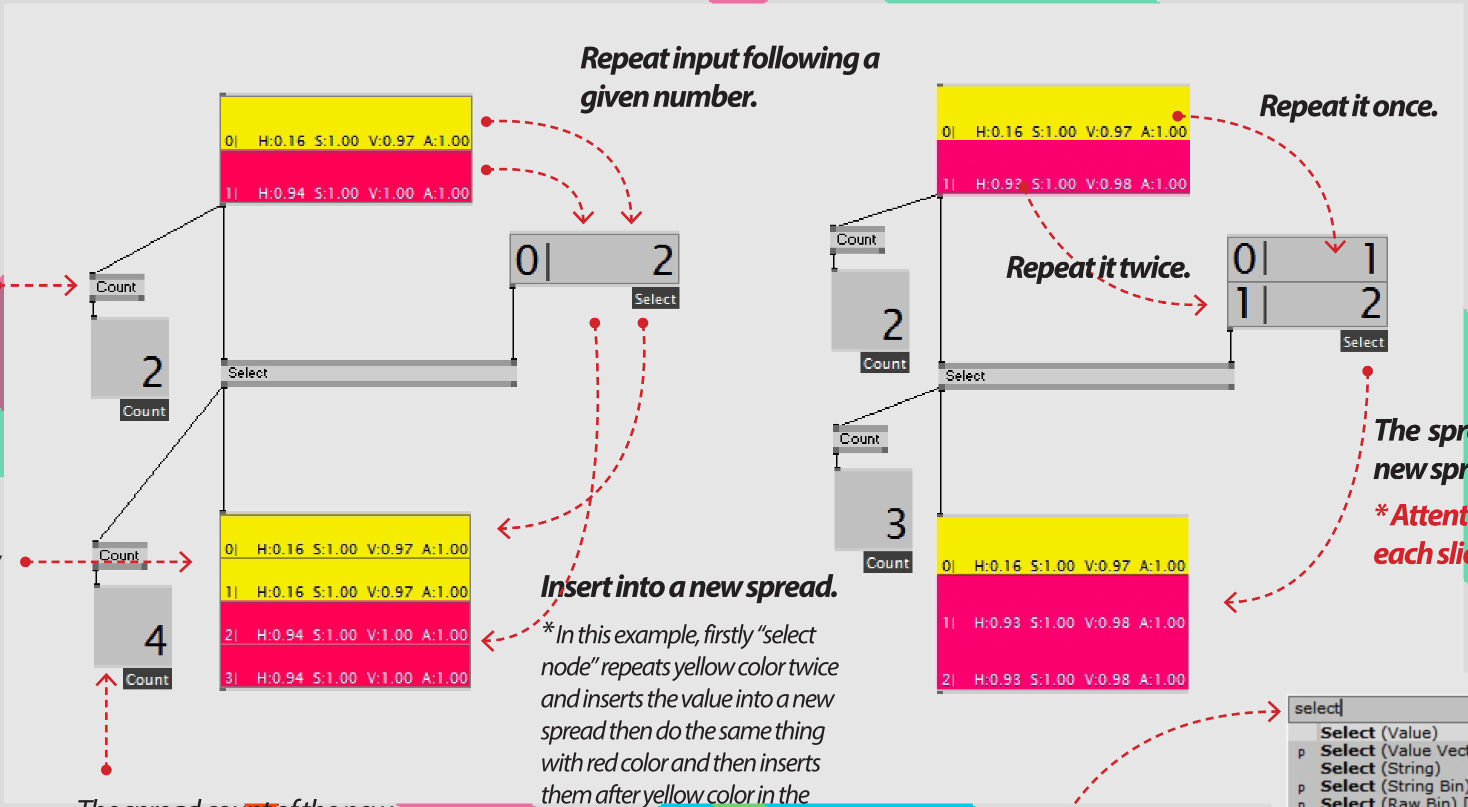


Diagram for switch node (Output)



16. Hello *Select.*

Select selects, how often a slice from a given spread is inserted into a new spread.



```
0|   H:0.16  S:1.00  V:0.97  A:1.00
1|   H:0.94  S:1.00  V:1.00  A:1.00
```

```
0|   H:0.16  S:1.00  V:0.97  A:1.00
1|   H:0.16  S:1.00  V:0.97  A:1.00
2|   H:0.94  S:1.00  V:1.00  A:1.00
3|   H:0.94  S:1.00  V:1.00  A:1.00
```

```
0|   H:0.16  S:1.00  V:0.97  A:1.00
1|   H:0.93  S:1.00  V:0.98  A:1.00
```

```
0|   H:0.16  S:1.00  V:0.97  A:1.00
1|   H:0.93  S:1.00  V:0.98  A:1.00
2|   H:0.93  S:1.00  V:0.98  A:1.00
```

Original Input

New Output as a new spread

Repeat input following a given number.

Insert into a new spread.

** In this example, firstly "select node" repeats yellow color twice and inserts the value into a new spread then do the same thing with red color and then inserts them after yellow color in the spread.*

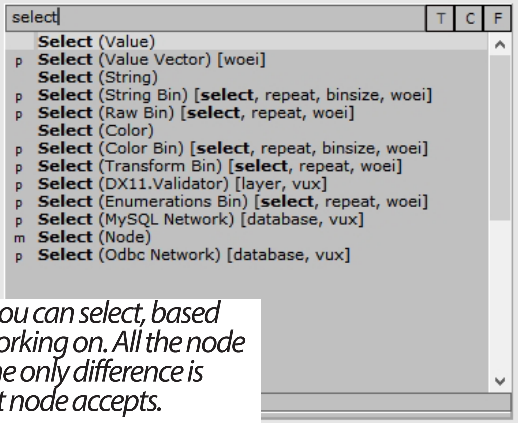
The spread count of the new spread is "4" as each input value is repeated twice.

Repeat it once.

Repeat it twice.

The spread account of the new spread is "3"

**** Attention to the order of each slice in the new spread.***

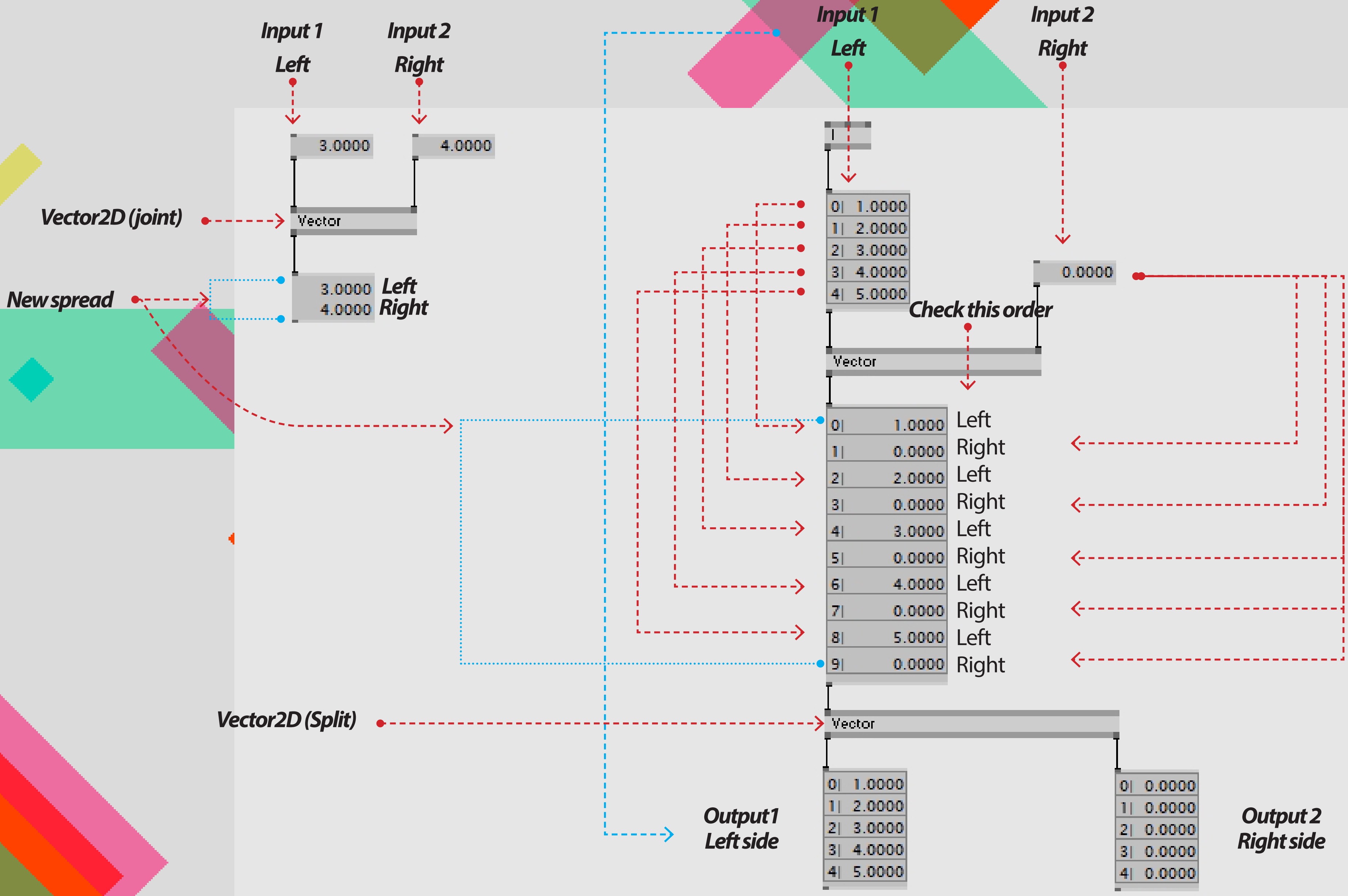


```
select
  Select (Value)
p Select (Value Vector) [woei]
  Select (String)
p Select (String Bin) [select, repeat, binsize, woei]
p Select (Raw Bin) [select, repeat, woei]
  Select (Color)
p Select (Color Bin) [select, repeat, binsize, woei]
p Select (Transform Bin) [select, repeat, woei]
p Select (DX11.Validator) [layer, vux]
p Select (Enumerations Bin) [select, repeat, woei]
p Select (MySQL Network) [database, vux]
m Select (Node)
p Select (Odbc Network) [database, vux]
```

There are many select nodes you can select, based on which data type you are working on. All the node follow the same basic logic. The only difference is what kind of data type a select node accepts.

17. Hello *Vector2D (Joint/ split).*

Joins a 2d vector (a pair of two inputs) from single values



Input 1
Left

Input 2
Right

```
3.0000    4.0000
```

Vector2D (joint)

```
Vector
```

New spread

```
3.0000  Left
4.0000  Right
```

Input 1
Left

Input 2
Right

```
I

0|  1.0000
1|  2.0000
2|  3.0000
3|  4.0000
4|  5.0000
```

```
0.0000
```

Check this order

```
Vector
```

```
0|      1.0000   Left
1|      0.0000   Right
2|      2.0000   Left
3|      0.0000   Right
4|      3.0000   Left
5|      0.0000   Right
6|      4.0000   Left
7|      0.0000   Right
8|      5.0000   Left
9|      0.0000   Right
```

Vector2D (Split)

```
Vector
```

Output1
Left side

```
0|  1.0000
1|  2.0000
2|  3.0000
3|  4.0000
4|  5.0000
```

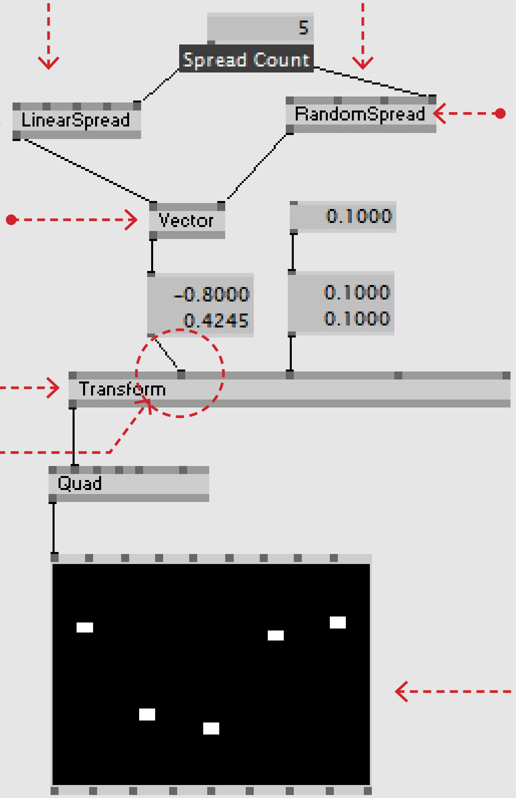
Output 2
Right side

```
0|  0.0000
1|  0.0000
2|  0.0000
3|  0.0000
4|  0.0000
```

17. Hello *Vector2D (Joint/ split) - In use.*

Input 1 Left

Input 2 Right



Use this value for X position

Use this value for Y position

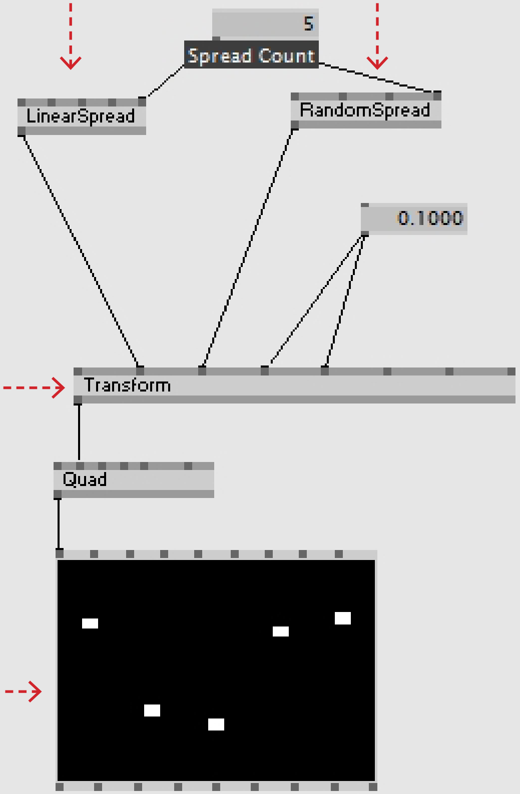
Joint Left and right side input value as a new spread

Transform (2D Vector):

** Transform (2D Vector) only arrows one spread for XY coordination.*

X input

Y input

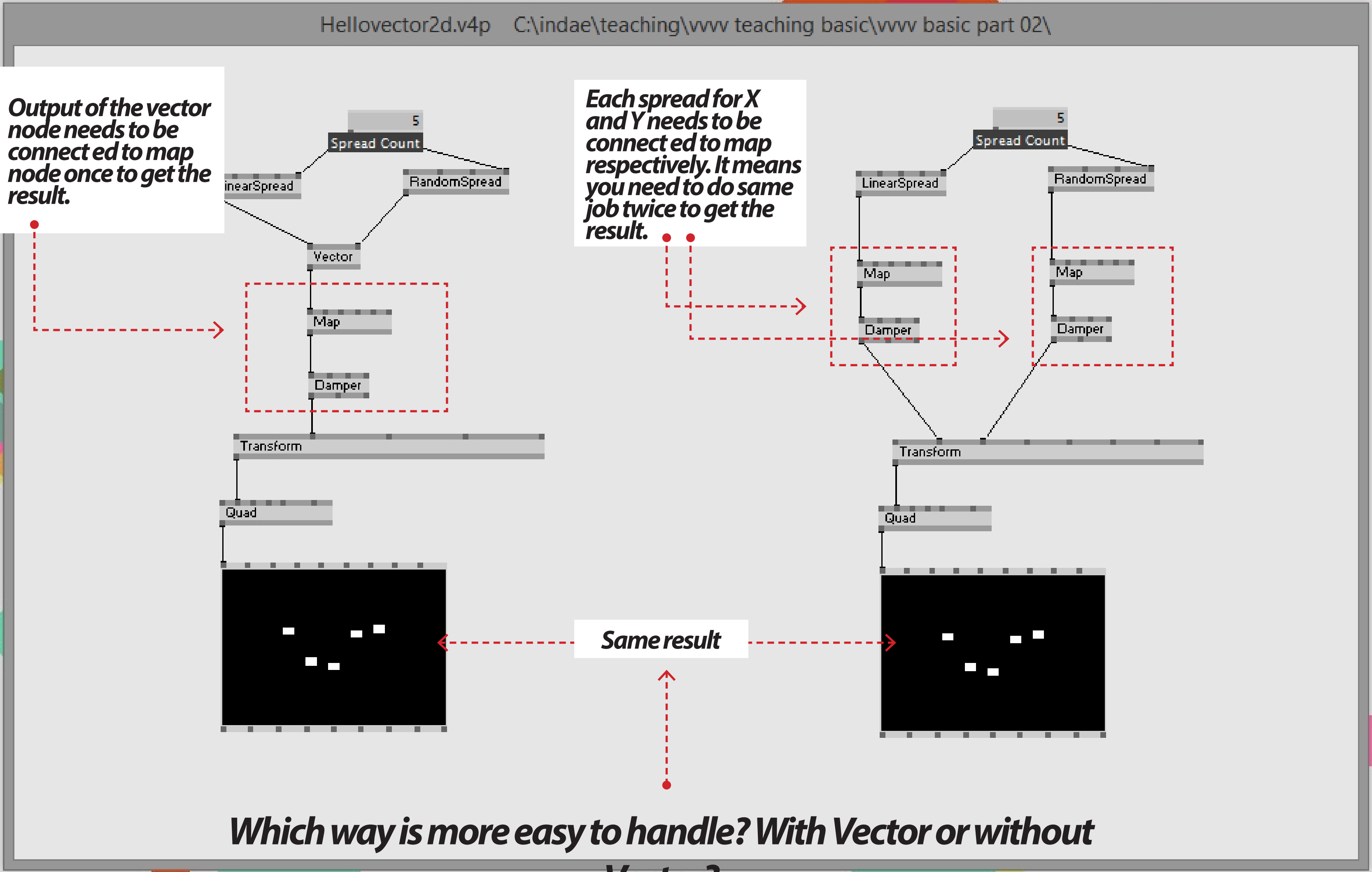


Transform

Same result

Why Transform (2D vector)? If the results are same for both methods?

17. Hello *Vector2D (Joint/ split) - In use.*



Output of the vector node needs to be connect ed to map node once to get the result.

Each spread for X and Y needs to be connect ed to map respectively. It means you need to do same job twice to get the result.

Same result

Which way is more easy to handle? With Vector or without Vector?