PD File Format

Simon Asselbergs and Tjeerd Sietsma

October 23, 2004

Contents

Introduction . . . 3
Records . . . 4
Wiring . . . 5
Elements and objects . . . 5
Common parameters . . . 6
Reference . . . 7
A . . . 7
N . . . 8
canvas . . . 9
X . . . 10
array . . . 11
connect . . . 12
coords . . . 13
floatatom . . . 14
msg . . . 15
obj . . . 16
bng . . . 17
tgl . . . 18
nbx . . . 19
vsl . . . 20
hsl . . . 21
vradio . . . 22
hradio . . . 23
vu . . . 24
cnv . . . 25
[name] . . . 26

pd 27
restore 28
symbolatom 29
text 30

Introduction

source: https://puredata.info/docs/developer/PdFileFormat

This file describes the fileformat of patchfiles (.pd) of Miller Puckette's PureData.

It is released because:

PureData's patchfile format is officially undocumented.

It may serve as a partial tutorial to help readers understand PureData better.

This documents is currently unofficial, beware that there is no warranty whatsoever.

If you find this document incorrect or incomplete please send us an e-mail using the links at the top of this document.

There are a few things still unknown, we don't exclude the possibility those things are used for compatibility with Max.

An example

```
#N canvas 530 323 450 300 12;
#X obj 166 80 osc~ 440;
#X floatatom 166 41 5 0 500 0 freq - - 0;
#X obj 166 148 *~ 0.1;
#X obj 166 226 dac~;
#X connect 0 0 2 0;
#X connect 1 0 0 0;
#X connect 2 0 3 0;
#X connect 2 0 3 1;
```

Records

The PD fileformat is a genuine custom textfile format, not to be confused with XML. It consists of one or more records.

From a high-level each record may cover multiply lines but they all have the same syntax:

```
#[data]\r\n;
```

where `[data]` holds the record data, `\r` represents an ASCII code 13 carriage return character, and `\n` represents an ASCII code 10 line-feed character. Going forward in this document, such characters will be omitted from the examples.

Each record consists of a chunk type, element type and optional parameters like this:

```
#[chunk_type] [element_type] [p1] [p2] [p3] [...];
```

`[chunk_type]` is a single character with only three possible values:

1. `X` for an object
2. `N` for a new window, and finally
3. `A` for array data.

```python
def pd_record(chunk_type, element_type, *params):
    assert chunk_type in ['X', 'N', 'A']
    args = " ".join(str(p) for p in params)
    return f'#{chunk_type} {element_type} {args};'
```

`[element_type]` is a predefined PD element. This declaration is also used for wires.

Object elements are numbered in order of appearance in the file, all other elements are excluded from numbering. These numbers are pure virtual and can not be seen directly in the file.

`[p1] [p2] [p3] [...]` are the required parameters for each element, this differences per element.

Wiring

Almost all objects can be interconnected with wires in PureData. Each wire is stored in the file using the following syntax:

```
#X connect [source] [outlet_number] [target] [inlet_number];
```

```
def connect(source, outlet_n, target, inlet_n):
    return pd_record('X', 'connect', source.id, outlet_n, target.id, inlet_n)
```

`[source]` Is the number of the object the data is coming from.

`[outlet-number]` Represents the number of the outlet of the source object where the wire starts.

Sequentially, `[target]` (also called *sink*) is the number of the target object , and finally `[inlet_number]` specifies the inlet of the target object to which the wired is connected.

Logically, the objects (again, connects excluded) are numbered from 0 to the total number of objects in the file using the integer format. The inlets and outlets are numbered likewise. Please keep this in mind, to prevent making often made off-by-one errors.

Elements and objects

There's a difference between elements and objects. Elements are the parts that together make up the entire layout of a patch, including windowsizes and position. Objects are the building blocks of PureData that contain functionality, gui-related or not.

All other elements are used to build actual PureData objects:

- array - visual two dimensional array
- canvas - specifications for the windowsizes and position
- coords - used for graphs
- floatatom - number
- msg - message
- obj - object, empty, subpatch or gui:
  - bang
  - toggle
  - nbx
  - vslider
  - hslider
  - vu
  - canvas - gui element
  - pd [name]
- [name]
- restore - for exiting subwindows and graphs
- symbolatom - symbol
- text - comment

Common parameters

Positions and size: as PureData uses a graphical interface every gui-related element (`object`, `message`, `number`, `symbol`, `comment`, `bang`, `toggle`, `number2`, `vslider`, `hslider`, `vradio`, `hradio`, `vu`, `canvas`, `graph`, `array`) have a horizontal and vertical position in the window that holds the (sub)patch.

Records of gui elements that have adjustable sizes also contain the horizontal and/or vertical size.

Positions and sizes are stored in pixels.

Note that in graphs (element: coords) the pixel sizes of its canvas are relative of the coordinates of the graph, when the option "graph on parent" is selected.

Color: Some graphical elements have color attributes. Per color only one signed integer value is stored that contains the three 8-bit color components (RGB).

Formula to calculate color attribute values:

```
def color(red, green, blue):
    return ((red * -65536) + (green * -256) + (blue * -1))
```

Where `[red]`, `[green]`, `[blue]` obviously represent the three color components, their values range from 0 to 255.

They apply to the attributes `[background color]`, `[front color]`, `[label color]` of various elements.

Fonts: elements that have a unsigned integer `[font]` attribute (e.g. the element text) have a choice out of three different fonts:

```
0 = Courier
1 = Helvetica
2 = Times
```

`Courier` is the only available fixed width font.

The `[fontsize]` attribute contains the font size in pixels.

Labels: the many GUI-elements that have a `[label]` attribute can be named. In its label string no spaces ASCII code 32 is allowed. It will generally be recognized as a seperator character for the next parameter/attribute.

Other parameters: Because of the difference between element parameters there is no default syntax for all elements. They may or may not look much like other elements, depending on their visual and functional similarities.

Reference

A

Announces array data

Syntax:

`#A [p1] [p2] [p3] [...];`

Parameters:

`[p1] [p2] [p3] [...]` floating point variables representing array elements

Example:

See Array

Notes:

- Used only in combination of an array definition

N

Announces a frameset

Syntax:

`#N [new_frame];`

Parameters:

`[new_frame]` new frameset, currently only a new patchwindow can be defined

Example:

see canvas

Notes:

- Currently used only for new canvas definitions

canvas

Defines window properties

Syntax:

```
#N canvas [x_pos] [y_pos] [x_size] [y_size] [name] [open_on_load];
```

Parameters:

```
[x_pos] - horizontal position offset of frameset (window)
[y_pos] - vertical position offset of frameset (window)
[x_size] - horizontal size of frameset (window)
[y_size] - vertical size of frameset (window)
[name] - name / handle of frameset
[open_on_load] - when flag is set the canvas is opened when the patch is loaded
```

Example:

```
#N canvas 0 0 452 302 12;
```

Example:

```
#N canvas 0 0 452 302 thiscanvas 0;
#X restore 41 136 pd thiscanvas;
```

Notes:

In this example patch there are two canvas definitions:

1. The first example has a special syntax, the attributes `[name]` and `[open_on_load]` have been replaced by a `[font_size]` attribute. This is only the case when the canvas definition is the first record in the patchfile, to define the position and size of the main patcher window, and set the default font size. When a first canvas definition is absent the default values of PureData are used. Note that in this case you can alter the default font size with a command line parameter.
2. The second example shows a regular internal subpatch definition. Normally a canvas defintion is always postceeded with a restore element to define the position within the canvas, and the name of the subpatch.

Presumably, the canvas name in the canvas defintion and restore definition should be thesame.

X

Announces regular elements

Syntax:

```
#X [element];
```

Parameters:

```
[element] - element definition
```

Example:

```
#X obj 50 36;
```

Notes:

- Used with every element definition except canvas definitions

array

Array

Syntax:

```
#X array [array_name] [array_size] float [save_flag];
```

Parameters:

```
[array_name] - name / handle of the array
[array_size] - total number of elements of an array
[save_flag] - with this flag set the array data is stored in the patch
```

Example:

```
#N canvas 0 0 450 300 graph4 0;
#X array array3 10 float 1;
#A 0 0 0 0 0 0 0 0 0 0;
#X coords 0 1 99 -1 200 140 1;
#X restore 270 193 graph;
```

Notes:

- An array is a gui-form of a table, always visualised using a graph. Optionally the array data can be saved in the patch - which can be very space-consuming and CPU intensive when you want to store a large number of floating point values typed out in a textfile.

connect

Wires GUI-elements

Syntax:

```
#X connect [source] [outlet] [target] [inlet];
```

Parameters:

```
[source]
[outlet]
[target]
[inlet]
```

Example:

```
#X obj 30 27 midiin;
#X obj 26 59 midiout;
#X connect 0 0 1 0;
#X connect 0 1 1 1;
```

Notes:

- Objects are virtually numbered in order of appearance in the file, starting from zero. Inlets and outlets of the objects are numbered likewise.

coords

Visual ranges of a frameset (window)

Syntax:

```
#X coords [x_from] [y_to] [x_to] [y_from] [width] [heigth] [graph_on_parent];
```

Parameters:

```
[x_from] - first index to display
[y_to] - dynamic range of graph upper border
[x_to] - last index to display
[y_from] - dynamic range of graph lower border
[width] - relative horizontal size of graph
[heigth] - relative vertical size of graph
[graph_on_parent] - when set displayes child path, when unset graph is displayed like an object
```

Example:

```
#N canvas 0 0 452 302 graph1 0;
#X coords 0 1 100 -1 200 140 1;
#X restore 58 26 graph;
```

Notes:

- A coords statement must always be preceded with a canvas statement which also holds the graph name.
- A coords statement must always be terminated with a restore statement that has the reserved handle graph
- Off limit values will be displayed outside the graph in the PureData GUI.

floatatom

Defines a number box

Syntax:

```
#X floatatom [x_pos] [y_pos] [width] [lower_limit] [upper_limit] [label_pos] [label]
[receive] [send];
```

Parameters:

```
[x_pos] - horizontal position within window
[y_pos] - vertical position within window
[width] - number of digits
[lower_limit] - minimal value
[upper_limit] - maximum value
[label_pos] - label position relative to floatatom position. 0 = left, 1 = right, 2 = top, 3 = bottom
[label] - floatatom label/name
[receive] - receive symbol name
[send] - send symbol name
```

Example:

```
#X floatatom 32 26 5 0 0 0 - - -;
```

Notes:

- When the value of [upper_limit] minus the value of `[lower_limit]` is less than one, or the `[width]` attribute is set to one, PureData resets these values both to zero.
- Floatatom and symbolatom are the only elements that uses "-" characters to indicate that no value has been assigned to its attributes `[label]`, `[receive]` and `[send]`.

msg

Defines a message

Syntax:

```
#X msg [x_pos] [y_pos] [p1] [p2] [p3] [...];
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[p1] [p2] [p3] [...] the content of the message
```

Example:

```
#X msg 61 48 read audio.wav;
```

obj

Defines an object

Syntax:

```
#X obj [x_pos] [y_pos] [object_name] [p1] [p2] [p3] [...];
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[object_name] - name of the object (optional)
[p1] [p2] [p3] [...] the parameters of the object (optional)
```

Example:

```
#X obj 55 50;
#X obj 132 72 trigger bang float;
```

Notes:

- The first line is an example of an empty object. The second line describes a trigger object with its parameters.

bng

Defines a bang

Syntax:

```
#X obj [x_pos] [y_pos] bng [size] [hold] [interrupt] [init] [send] [receive] [label]
[x_off] [y_off] [font] [fontsize] [bg_color] [fg_color] [label_color] ;
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[size] - square size of the gui element
[hold] - hold time in milliseconds, ranges from 50 to 1000000000
[interrupt] - interrupt time in milliseconds, ranges from 10 to 250
[init] - bang on load
[send] - send symbol name
[receive] - receive symbol name
[label] - label
[x_off] - horizontal position of the label text relative to the upperleft corner of the object
[y_off] - vertical position of the label text relative to the upperleft corner of the object
[font] - font type
[fontsize] - font size
[bg_color] - background color
[fg_color] - foreground color
[label_color] - label color
```

Example:

```
#X obj 27 32 bng 15 10000 100 1 empty empty empty 0 -6 0 8 -262144 -1 -1;
```

Notes:

- Hold time is for how long you see a flash when you click on the bang, interrupt time is for how long you don't see it flash when you click on this object while it's flashing.
- `[send]`, `[receive]` and `[label]` cannot be named "empty", this is a reserved name for when no value is assigned.

tgl

Defines a toggle

Syntax:

```
#X obj [x_pos] [y_pos] tgl [size] [init] [send] [receive] [label] [x_off] [y_off] [font]
[fontsize] [bg_color] [fg_color] [label_color] [init_value] [default_value] ;
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[size] - square size of the gui element
[init] - set on load
[send] - send symbol name
[receive] - receive symbol name
[label] - label
[x_off] - horizontal position of the label text relative to the upperleft corner of the object
[y_off] - vertical position of the label text relative to the upperleft corner of the object
[font] - font type
[fontsize] - font size
[bg_color] - background color
[fg_color] - foreground color
[label_color] - label color
[init_value] - value sent when the [init] attribute is set
[default_value] - default value when the [init] attribute is not set
```

Example:

```
#X obj 29 44 tgl 15 1 empty empty empty 0 -6 192 8 -262144 -1 -1 234 234;
```

Notes:

- `[send]`, `[receive]` and `[label]` cannot be named "empty", this is a reserved name for when no value is assigned.
- The `[default_value]` attribute can be changed in a patch and saved, this is why there are different values for `[init_value]` and `[default_value]`

nbx

Defines a Number2 number box

Syntax:

```
#X obj [x_pos] [y_pos] nbx [size] [height] [min] [max] [log] [init] [send] [receive]
[label] [x_off] [y_off] [font] [fontsize] [bg_color] [fg_color] [label_color] [log_height];
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[size] - number of digits the element displays
[height] - vertical size of element in pixels
[min] - minimum value, typically -1e+037
[max] - maximum value, typically 1e+037
[log] - linear when unset, logarithmic when set
[init] - when set outputs
[send] - send symbol name
[receive] - receive symbol name
[label] - label
[x_off] - horizontal position of the label text relative to the upperleft corner of the object
[y_off] - vertical position of the label text relative to the upperleft corner of the object
[font] - font type
[fontsize] - font size in pixels
[bg_color] - background color
[fg_color] - foreground color
[label_color] - label color
[log_height] - logarithmic steps, accepts values from 10 to 2000, default is 256
```

Example:

```
#X obj 39 48 nbx 5 14 -1e+037 1e+037 0 0 empty empty empty 0 -6 0 10 -262144 -1 -1 0 256;
```

Notes:

- This element resembles the floatatom element, but is extended with gui and logarithmic parameters and (probably) more accurate.
- The attributes `[font]`, `[font_size]` and `[fg_color]` also apply to the digits.

vsl

Defines a vertical slider

Syntax:

```
#X obj [x_pos] [y_pos] vsl [width] [height] [bottom] [top] [log] [init] [send] [receive]
[label] [x_off] [y_off] [font] [fontsize] [bg_color] [fg_color] [label_color] [default_value]
[steady_on_click];
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[width] - horizontal size of gui element
[height] - vertical size of gui element
[bottom] - minimum value
[top] - maximum value
[log] - when set the slider range is outputted logarithmically, otherwise it's output is linair
[init] - sends default value on patch load
[send] - send symbol name
[receive] - receive symbol name
[label] - label
[x_off] - horizontal position of the label text relative to the upperleft corner of the object
[y_off] - vertical position of the label text relative to the upperleft corner of the object
[font] - font type
[fontsize] - font size
[bg_color] - background color
[fg_color] - foreground color
[label_color] - label color
[default_value] - default value times hundred
[steady_on_click] - when set, fader is steady on click, otherwise it jumps on click
```

Example:

```
#X obj 50 38 vsl 15 128 0 127 0 0 empty empty empty 0 -8 0 8 -262144 -1 -1 0 1;
```

Notes:

- The vertical slider object and the horizontal slider are the only objects which have a default value multiplied by hundred. This purpose is unknown.

hsl

Defines a horizontal slider

Syntax:

```
#X obj [x_pos] [y_pos] hsl [width] [height] [bottom] [top] [log] [init] [send] [receive]
[label] [x_off] [y_off] [font] [fontsize] [bg_color] [fg_color] [label_color] [default_value]
[steady_on_click];
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[width] - horizontal size of gui element
[height] - vertical size of gui element
[bottom] - minimum value
[top] - maximum value
[log] - when set the slider range is outputted logarithmically, otherwise it's output is linair
[init] - sends default value on patch load
[send] - send symbol name
[receive] - receive symbol name
[label] - label
[x_off] - horizontal position of the label text relative to the upperleft corner of the object
[y_off] - vertical position of the label text relative to the upperleft corner of the object
[font] - font type
[fontsize] - font size
[bg_color] - background color
[fg_color] - foreground color
[label_color] - label color
[default_value] - default value times hundred
[steady_on_click] - when set, fader is steady on click, otherwise it jumps on click
```

Example:

```
#X obj 53 44 hsl 128 15 0 127 0 0 empty empty empty -2 -6 0 8 -262144 -1 -1 0 1;
```

Notes:

- The horizontal slider object and the vertical slider are the only objects which have a default value multiplied by hundred. This purpose is unknown.

vradio

Defines a vertical radio button selector

Syntax:

```
#X obj [x_pos] [y_pos] vradio [size] [new_old] [init] [number] [send] [receive] [label]
[x_off] [y_off] [font] [fontsize] [bg_color] [fg_color] [label_color] [default_value];
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[size] - horizontal size and vertical size, depending on the number of radio buttons
[new_old] - send new and old value, or only the new value
[init] - send default value on init
[number] - amount of radio buttons
[send] - send symbol name
[receive] - receive symbol name
[label] - label
[x_off] - horizontal position of the label text relative to the upperleft corner of the object
[y_off] - vertical position of the label text relative to the upperleft corner of the object
[font] - font type
[fontsize] - font size
[bg_color] - background color
[fg_color] - foreground color
[label_color] - label color
[default_value] - default value to be sent on patch load when the [init] attribute has been set.
```

Example:

```
#X obj 48 42 vradio 15 1 0 8 empty empty empty 0 -6 0 8 -262144 -1 -1 0;
```

Notes:

- The purpose of the [new_old] switch is unknown.

hradio

Defines a horizontal radio button selector

Syntax:

```
#X obj [x_pos] [y_pos] hradio [size] [new_old] [init] [number] [send] [receive] [label]
[x_off] [y_off] [font] [fontsize] [bg_color] [fg_color] [label_color] [default_value];
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[size] - vertical size and horizontal size, depending on the number of radio buttons
[new_old] - send new and old value, or only the new value
[init] - send default value on init
[number] - amount of radio buttons
[send] - send symbol name
[receive] - receive symbol name
[label] - label
[x_off] - horizontal position of the label text relative to the upperleft corner of the object
[y_off] - vertical position of the label text relative to the upperleft corner of the object
[font] - font type
[fontsize] - font size
[bg_color] - background color
[fg_color] - foreground color
[label_color] - label color
[default_value] - default value to be sent on patch load when the [init] attribute has been set.
```

Example:

```
#X obj -50 54 hradio 15 1 0 8 empty empty empty 0 -6 0 8 -262144 -1 -1 0;
```

Notes:

- The purpose of the [new_old] switch is unknown.

vu

Defines a VU-meter

Syntax:

```
#X obj [x_pos] [y_pos] vu [width] [height] [receive] [label] [x_off] [y_off] [font]
[fontsize] [bg_color] [label_color] [scale] [];
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[width] - horizontal size of element
[height] - vertical size of element
[receive] - receive symbol name
[label] - label
[x_off] - horizontal position of the label text relative to the upperleft corner of the object
[y_off] - vertical position of the label text relative to the upperleft corner of the object
[font] - font type
[fontsize] - font size
[bg_color] - background color
[label_color] - label color
[scale] - when set the logarithmic scale is displayed
[] - unknown value, default is zero
```

Example:

```
#X obj 40 44 vu 15 120 empty empty -1 -8 0 8 -66577 -1 1 0;
```

Notes:

- We looked inside the source code but still couldn't see the purpose of the final value.

cnv

Defines a canvas, a gui component

Syntax:

```
#X obj [x_pos] [y_pos] cnv [size] [width] [height] [send] [receive] [label] [x_off]
[y_off] [font] [font_size] [bg_color] [label_color] [];
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[size] - size of selectable square
[width] - horizontal size of the GUI-element
[height] - vertical size of the GUI-element
[send] - send symbol name
[receive] - receive symbol name
[label] - label
[x_off] - horizontal position of the label text relative to the upperleft corner of the object
[y_off] - vertical position of the label text relative to the upperleft corner of the object
[font] - font type
[fontsize] - font size
[bg_color] - background color
[label_color] - foreground color
[] - unknown value, default is zero
```

Example:

```
#X obj 27 40 cnv 15 100 60 empty empty empty 20 12 0 14 -233017 -66577 0;
```

Notes:

- We couldn't find the purpose of the final value.

[**name**]

Defines an external subpatch or library patch

Syntax:

```
#X obj [x_pos] [y_pos] [name] [p1] [p2] [p3] [...] ;
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[name] - name of the subpatch / library patch
[p1] [p2] [p3] [...] - optional parameters
```

Example:

```
#X obj 121 102 subpatch;
```

Notes:

- If the external object isn't in the loaded libraries PureData searches in the folder of the main frame (window) to locate the object.

pd

Defines an internal subpatch

Syntax:

```
#X restore [x_pos] [y_pos] pd [name];
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[size] - size of selectable square
[width] - horizontal size of the GUI-element
[name] - name of the subpatch
```

Example:

```
#N canvas 0 0 454 304 inc 0;
#X obj 34 40 inlet;
#X obj 34 95 outlet;
#X obj 34 67 + 1;
#X connect 0 0 2 0;
#X connect 2 0 1 0;
#X restore 90 124 pd inc;
#X floatatom 90 99 5 0 0 0 - - -;
#X floatatom 90 151 5 0 0 0 - - -;
#X connect 0 0 2 0;
#X connect 1 0 0 0;
```

Notes:

- Naturally the restore element which invokes a subpatch is preceded with a canvas element and the subpatch elements.
- Objects within a subpatch are counted seperately from a parent frame (window).

restore

Ends a canvas definition

Syntax:

```
#X restore [x_pos] [y_pos] [type] [name];
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[type] - type of canvas, values are either "graph" or "pd"
[name] - name of the subpatch, only used when the [type] attribute is set to "pd"
```

Example:

```
#N canvas 0 0 450 300 graph2 0;
#X coords 0 1 100 -1 200 140 1;
#X restore 27 30 graph;
```

Example:

```
#N canvas 0 0 452 302 subpatch 0;
#X restore 64 69 pd subpatch;
```

Notes:

- The restore element is used only in combination with canvas elements. There are two uses:
- In the first example it defines a graph, in this case a coords element is required.
- In the second example it defines a subpatch. In this case only the canvas attribute [name] and the restore attribute [name] should correlate.

symbolatom

Defines an symbol box

Syntax:

```
#X symbolatom [x_pos] [y_pos] [width] [lower_limit] [upper_limit] [label_pos] [label]
[receive] [send];
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[width] - amount of digits/characters
[lower_limit] - minimum value
[upper_limit] - maximum value
[label_pos] - label position relative to floatatom position. 0 = left, 1 = right, 2 = top, 3 = bottom
[label] - label
[receive] - receive symbol value
[send] - send symbol value
```

Example:

```
#X symbolatom 36 37 10 0 0 0 - - -;
```

Notes:

- When the value of [upper_limit] minus the value of [lower_limit] is less than one, PureData resets these values both to zero.
- Symbolatom and floatatom are the only elements that uses "-" characters to indicate that no value has been assigned to its attributes [label], [receive] and [send].

text

Defines a comment

Syntax:

```
#X text [x_pos] [y_pos] [comment];
```

Parameters:

```
[x_pos] - horizontal position within the window
[y_pos] - vertical position within the window
[comment] - custom string, spaces allowed
```

Example:

```
#X text 28 25 comment;
```

Notes:

- ASCII return codes 13 and 10 are not stored, a semicolon character is preceded with the escape character backslash.