wSlider

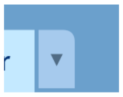
Simple in Both Practice and Execution

Overview

wSlider is a simple class, only ~10kb, which allows for the implementation of sliders into your processing program!

Implementation

You have two options:

1. In a processing document,  click this little arrow, (or use ctrl+shift+n), and enter the name “wSlider” into the box. Then, copy and paste the entire code into this new tab. That’s it!
2. You can also just add the wSlider.pde file into the folder containing your main pde file!

Use

While wSlider is an incredibly flexible option, full of a variety of customization and adjustment, so you can get exactly what you want, it is completely usable straight out of the box.

wSlider has 3 constructors:

1. wSlider(int sliderX, int sliderY, int sliderMin, int sliderMax)
2. **wSlider (int sliderX, int sliderY, int sliderMin, int sliderMax, int sliderW)**
3. wSlider (int sliderX, int sliderY, int sliderMin, int sliderMax, int sliderW, int sliderH)

Generally, #2 will be the go-to.

sliderX = the X position of the line that will be drawn

sliderY = the Y position of the line that will be drawn

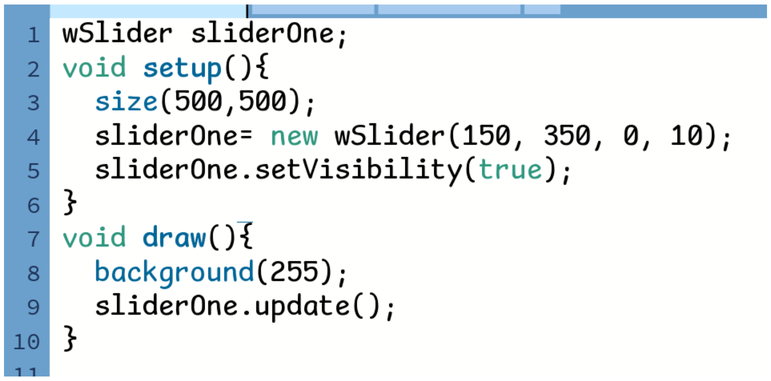
sliderMin = minimum value on the slider

sliderMax = maximum value on the slider (note, should work with negatives and having a higher min than max, but not thoroughly tested)

sliderW = the width of the slider line. Default = 200

sliderH = the height of the slider line. Default = 6

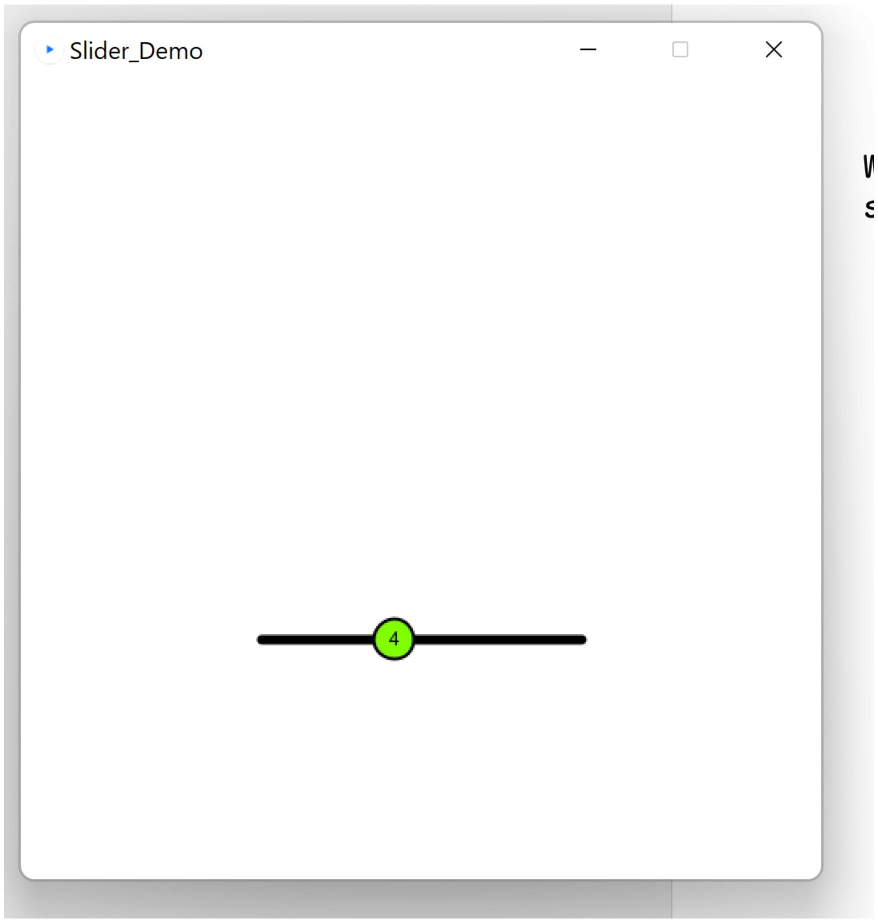
Example:



```
wSlider sliderOne;
void setup(){
  size(500,500);
  sliderOne= new wSlider(150, 350, 0, 10);
  sliderOne.setVisibility(true);
}
void draw(){
  background(255);
  sliderOne.update();
}
```

We define sliderOne as a new wSlider, starting at (150, 300) on the screen. It has a minimum value of 0, and a maximum value of 10.

Then we just put (sliderName).update(); into our draw function. Now, when we do (sliderName).setVisibility(true), our slider will be drawn!
If we want to hide the slider, just do (sliderName).setVisibility(false).



You can click on the circle, and drag the dot along the line, which produces a different value. This value can be returned with the line **(sliderName). getVal()**. This returns an int.

Customization

The following attributes can be adjusted:
All are called by sliderName.function();

Attribute	Set value	Get current value
The value of the slider	**N/A**	**getVal()** returns int
visibility of the slider default -> false; false -> not shown true -> shown	**setVisibility(boolean)**	**visibility()** returns boolean
the X location of the slider line:	**setSlX(int)**	**getSlX()** returns int
the Y location of the slider line:	**setSlY(int)**	**getSlY()** returns int
the width of the slider line:	**setSlW(int)**	**getSlW()** returns int
the height of the slider line:	**setSlH(int)**	**getSlH()** returns int
the minimum value on the slider:	**setSlMin(int)**	**getSlMin()** returns int
the maximum value on the slider:	**setSlMax(int)**	**getSlMax()** returns int
the x position of the dot	**N/A** should never need to be manually set this way. Try **setStartingValue()**	**getDotX()** returns int

The y position of the dot	**N/A** should never need to be manually set	**getDotY()** returns int
set the starting value of the dot	**setStartingValue(int)**	the value of the dot = **getVal()**
the width of the dot:	**setDotW(int)**	**getDotW()** returns int
the height of the dot:	**setDotH(int)**	**getDotH()** returns int
the stroke width of the dot:	**setDotStroke(int)**	**getDotStroke()** returns int
the colour of the line:	**setLineColor(color)**	**getLineColor()** returns color
the colour of the dot:	**setDotColor(color)**	**getDotColor()** returns color
the colour of the dot when hovered:	**setDotHover(color)**	**getDotHover()** returns color
the colour of the dot when clicked on:	**setDotClicked(color)**	**getDotClicked()** returns color
colour of the text:	**setTextColor(color)**	**getTextColor()** returns color
the font of the text on the dot:	**setValueFont(PFont)**	**getValueFont()** returns PFont
the size of the text on the dot:	**setDotFontSize(int)**	**getDotFontSize()** returns int

whether the text should be displayed at all: true -> show text false -> hide text	**setTextState(boolean)**	**getTextState()** returns boolean
if the slider should be locked:	**setLockedState(boolean)**	**lockedState()** returns boolean
text vertical offset (default = 4)	**setTextVertOffset(int)**	**getTextVertOffset()** returns int