Pixel Art

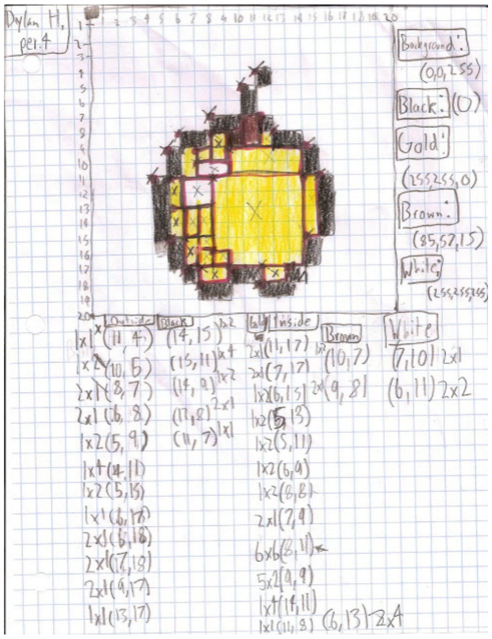
Have you ever wondered how to make pixel art? Looking for a retro look? Go no further, you can draw your own in Processing and you will become acquainted with some simple drawing tools while you are doing it!

PROCESSING
SKILL REQUIREMENTS
DIFFICULTY 1-5

ELECTRICAL PROTOTYPING	○○○○○
ROBOTICS	○○○○○
SOLDERING	○○○○○
PROGRAMMING	●○○○○
DIY	○○○○○

MATERIALS LIST

- Computer
- Processing
- Graph Paper
- Colored Pencils
- Super fine tip sharpie



STEP 1: Draw Something!

First off we need to setup your analog, or paper drawing canvas. You will be using graph paper, ⅛” grid works perfectly! Draw a rectangle that is 20 squares by 20 squares. This is the space that you will be creating your drawing in. The next thing is find something that you want to pixelate. If you want something easy you can google search for “pixel art” and take your pick. If you are looking for more of a challenge you can draw your own.

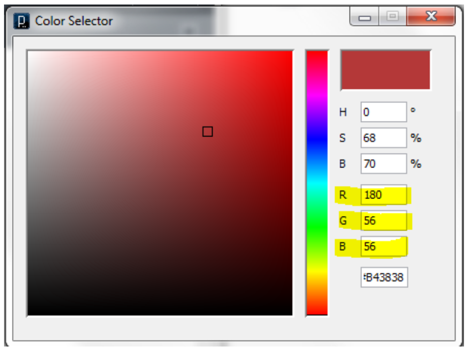
The rules are that you have to use full squares of color (no triangles, or partially filled boxes) See example image for a finished product

STEP 2: Color your Drawing and Block it out!

Once you have outlined your boxes add color with colored pencils. If there are 2 similar color next to one another make sure that you can tell the difference between the two. Once you have added color use a super fine tip sharpie to outline as big of rectangles as you possibly can...this will help you in the long run, I promise.

STEP 3: Plotting your Rectangles

Once you have outlined all of your rectangles you will need to put a point at the upper left hand corner of each one. You are going to need to plot these points and list them out. Create a table that has 5 columns. The columns are going to be the X coordinate, Y coordinate, width, height and color of each box. To find the X and Y coordinate start in the upper left hand corner and count how many boxes over for the X coordinate, and how many boxes down for the Y coordinate. The width and height are in boxes as well.



H 0 °
S 68 %
B 70 %
R 180
G 56
B 56
#B43838

STEP 4: Setup and RGB Colors

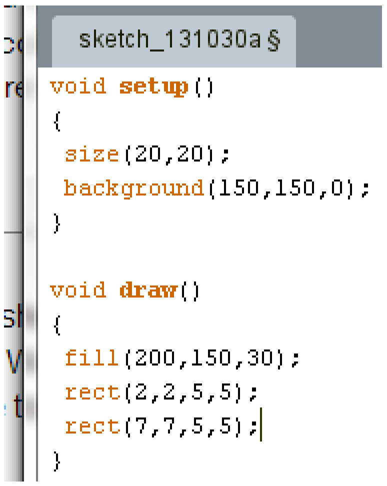
Once you have completed listing all of your boxes you can move into Processing. Open Processing and your setup `size(20,20);` should be the width and height of your palette (20x20). **Note:** If you want it bigger you can make it 200x200 but you will have to multiple all of your dimensions by 10! You can also set your background color using the `background(R,G,B);` function. R,G and B are how much red, green, or blue there is in your color. The numbers range from 0-255. You can use the color selector tool under your tools menu to find the RGB value of specific colors. You will use the same method to color your drawing as well, so remember how to get to the color selector tool.

STEP 5: Drawing Rectangles

Now that you have a window size() and a background(), we can start drawing rectangles! Remember that we need to have a void draw() and inside of that we are going to start creating rectangles. Your code should look similar to the image before you go any further.

To draw your first rectangle you are going to use the rect(X,Y,W,H); function. X and Y are the coordinates for the upper right hand corner, while W and H are the width and height. Use your table and start to enter rectangles into your draw. Draw a few and then hit play to check on your progress. What color are the rectangles?

To change the color or a give rectangle in the line above it type fill(R,G,B); with R,G,B being the RGB values of the color you want. This fill command will fill all of the shapes below it. A good habit to get into is grouping all of the rectangle of one color so you only have to fill once for each color. So, I would fill(255,0,0) and then all of the rectangles I would want red, then another fill( ) color for another set of rectangles. Follow these directions for all of your rectangles until you have finished your table. Make sure that you run your code every so often to double check for correct placement and color of each rectangle.



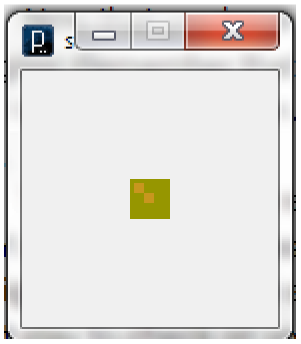
```
void setup()
{
 size(20,20);
 background(150,150,0);
}

void draw()
{
 fill(200,150,30);
 rect(2,2,5,5);
 rect(7,7,5,5);
}
```

STEP 6: noStroke()

Now that you are finished with all of your rectangles you have probably noticed by now that they are outlined in black. This is called the shapes "stroke" or outline. We have left it alone up to this point so that you can see each individual rectangle as you create them. Now that you are finished go to the top of your draw code. In the first line type noStroke(); and then hit play. You should have no outlines! here is an image of code above with a noStroke(); added produces this tiny image!

Remember that this is only a 20 by 20 pixel image. If you want to make it larger you can multiple all of your dimensions by 10 to get a 200 by 200 image.



STEP 8: Share your work!

If you would like to share your work...which you do! You can sign up at openprocessing.org. Open Processing allows you to share your code, view the drawing online, as well as allows others to fork or borrow your code to manipulate and change it on their own while still keeping your sketch intact for others to check out.

OpenProcessing

TAKING IT FURTHER

- Use both the tint() and filter() function together to make some pretty cool images!
- Use multiple images with different filters!
- play around with text("text",x,y);