introduction to processing
colour basics

Uta Hinrichs, CPSC 583, 2010
outline

• defining colour in processing
• colour tools
• colour scales & schemes
colour in processing

- greyscale colour
- range from 0 – 255

→ background(255)
→ stroke(0)
→ fill(150)
colour in processing

• RGB colour
• based the rules of mixing light
• 3 values \((r, g, b)\) range from 0 to 255
  \(\rightarrow\) \text{fill}(r, g, b)
  \(\rightarrow\) \text{color } c = \text{color}(r,g,b)

• alternative
  – hexadecimal representation
  \(\rightarrow\) \text{color } c = \text{color}(\#FFFFFF)
  \(\rightarrow\) \text{fill}(\#FFFFFF)
transparency

• alpha value
• specified as a 4th argument in the rgb color definition
• range 0 – 255
  – 0 completely transparent (0% opaque)
  – 255 completely opaque (100% opaque)

→ fill(r,g,b,a)
void setup() {
    size(200,200); //set window size
    background(0); //set window background color
    smooth(); //for smooth lines
    noStroke();
    noLoop(); //important since we don’t see any alpha
}

void draw() {
    fill(0,0,255); // No 4th argument means 100% opacity.
    rect(0,0,100,200);

    fill(255,0,0,255); // 255 means 100% opacity.
    rect(0,0,200,40);

    fill(255,0,0,191); // 75% opacity.
    rect(0,50,200,40);

    fill(255,0,0,127); // 55% opacity.
    rect(0,100,200,40);

    fill(255,0,0,63); // 25% opacity.
    rect(0,150,200,40);
}

http://www.learningprocessing.com/examples/chapter-1/example-1-4/
noLoop

- delete “noLoop” in setup() and observe how your sketch changes
- note: draw() is called constantly
- every shape is drawn in the order it its called
noLoop

- with noLoop()
- without noLoop()
how to find RGB colour representations

- any image processing program
- processing colour selector

- adobe kuler
  - 5-step colour compositions
  - searchable themes
color schemes

- **sequential**
  - ordered data (low to high)
  - light colors = low values
  - dark colors = high values

- **diverging**
  - equal emphasis on both mid-range critical values and extremes at both ends
  - lightest color = meaningful mid-range value, e.g. mean, median, zero
  - increased darkness = difference to mid-range value

- **qualitative**
  - no magnitude differences in the data
  - best for nominal and categorical data
  - mostly differences in hue, only subtle differences in lightness
Cynthia Brewer’s color brewer

http://colorbrewer2.org/
(more) continuous color scales

- BTC (blue to cyan)
- BTY (blue to yellow)
- grey
- Heated Object
- LOCS (linearized opt. color scale)
- Magenta
- OCS (optimal color scale)
- Rainbow
(more) continuous color scales

→ use pre-calculated values
http://www.cs.uml.edu/~haim/ColorCenter/Programs/ColorScales/ColorScales.html
hands-on exercise

- arrange a number of shapes in a row
  - use Shape() class
- colour shapes according to a color scale of your choice
  - color of each shape is determined by its x position
- all color scale classes require the ColorRGB class

```
BTC btc_scale = new BTC();
float red = btc_scale.rgb[position in color scale].getR();
float green = btc_scale.rgb[position in color scale].getG();
float blue = btc_scale.rgb[position in color scale].getB();
```
Shape[] shapeArray;
BTC btc_scale;

void setup()
{
    size(800,200);
    background(230);
    shapeArray = new Shape[7];
    btc_scale = new BTC();

    smooth();
    noLoop();
}

void draw()
{
    for(int i = 0; i<shapeArray.length; i++)
    {
        int s_pos = i*((int)256/shapeArray.length);
        System.out.println(s_pos);
        float b = btc_scale.rgb[s_pos].getB() * 255; //BTC class gives values between 0 and 1, we need them from 0 to 255
        float r = btc_scale.rgb[s_pos].getR() * 255;
        float g = btc_scale.rgb[s_pos].getG() * 255;

        shapeArray[i] = new Shape(100*i + 100, 100, 50, 50);
        shapeArray[i].setShapeColor(color(r,g,b));

        fill(shapeArray[i].getShapeColor());
        shapeArray[i].drawShape();
    }
}
useful links/further readings

• color centre
  http://www.cs.uml.edu/~haim/ColorCenter/Programs/ColorScales/ColorScales.html

• color brewer
  http://colorbrewer2.org/

• color use guidelines for data representation
  http://www.personal.psu.edu/faculty/c/a/cab38/ColorSch/ASApaper.html

• how to not lie with visualization