PV259
Generative Design Programing

Week 1

Introduction to p5.js

Marko Řeháček & Megi Kejstová
rehacek@mail.muni.cz

join

https://discord.gg/CcxnBVPtcR

Where to look for p5

p5.js website, the tool, docs
https://p5js.org
https://p5js.org/reference/
https://p5js.org/examples/

There are useful tutorials also on Processing website:
https://processing.org/tutorials

Daniel Shiffman, best p5 and Processing tutorials
Intro to p5.js - playlist, **Coding train,** the channel

Sketches, ideas
https://openprocessing.org

Books with examples
generative-gestaltung.de
Data-driven Graphic Design
Form+Code

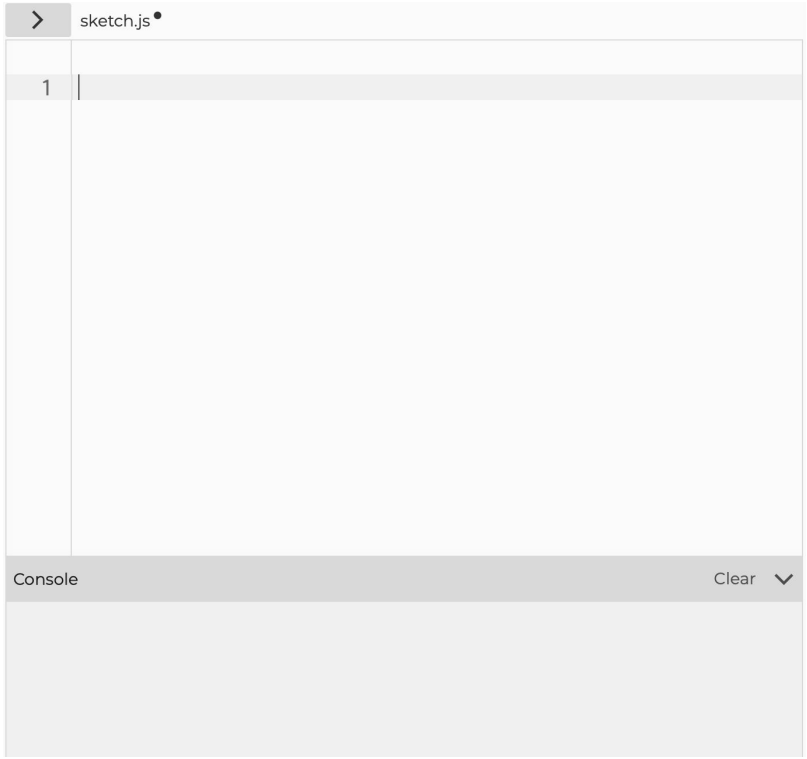
The coding train playlist

Loads of good video tutorials...

The online IDE

https://editor.p5js.org/
that's what we use (for now) during classes
video tutorial

Alternatively, you can setup VS Code at home
(most popular IDE for JavaScript).
You'll need a plugin. I also recommend using some
AI code completion (Github copilot, etc.).
VS Code video tutorial



The inevitable functions

```
function setup ( )  {
   // runs only once,
       at the beginning of the program
}

function draw ( ) {
   // runs in loop 30 times per second
}
```

You can use loop( ) and noLoop( ) functions,
or set the speed using frameRate().

Controlling the canvas

createCanvas (width, height)

createCanvas (windowWidth, windowHeight)
stretch it

fullscreen (boolean)

background (int)

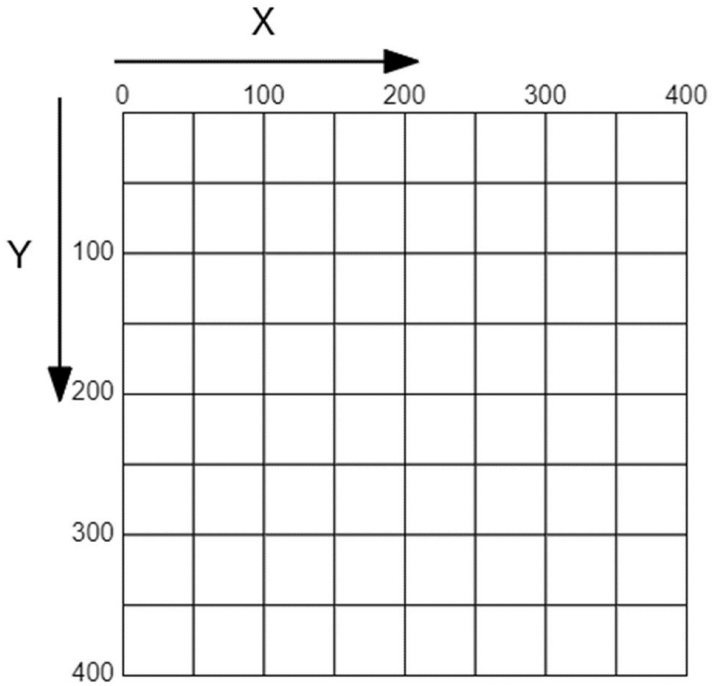
print (string)

Coordinate system

```
circle ( 0, 0, 150 ); // at origin, of diameter 150 px
```

Where does it appear?

Coordinate system



sketch.js

```
function setup() {
  createCanvas(400,400);
}

function draw() {
  background(230);
  circle(0,0,150);
}
```

Preview

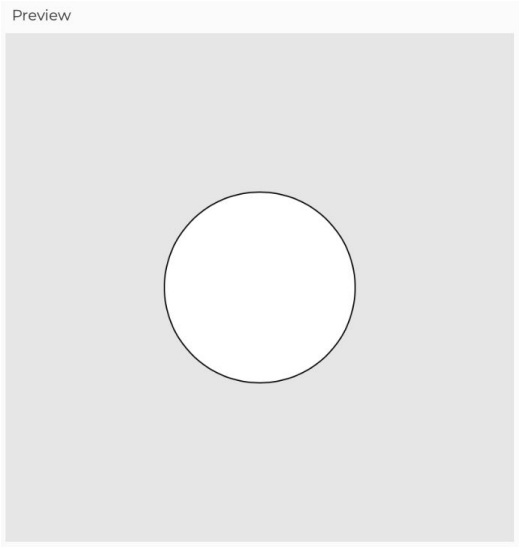
Console

[Coordinates, shapes: tutorial]

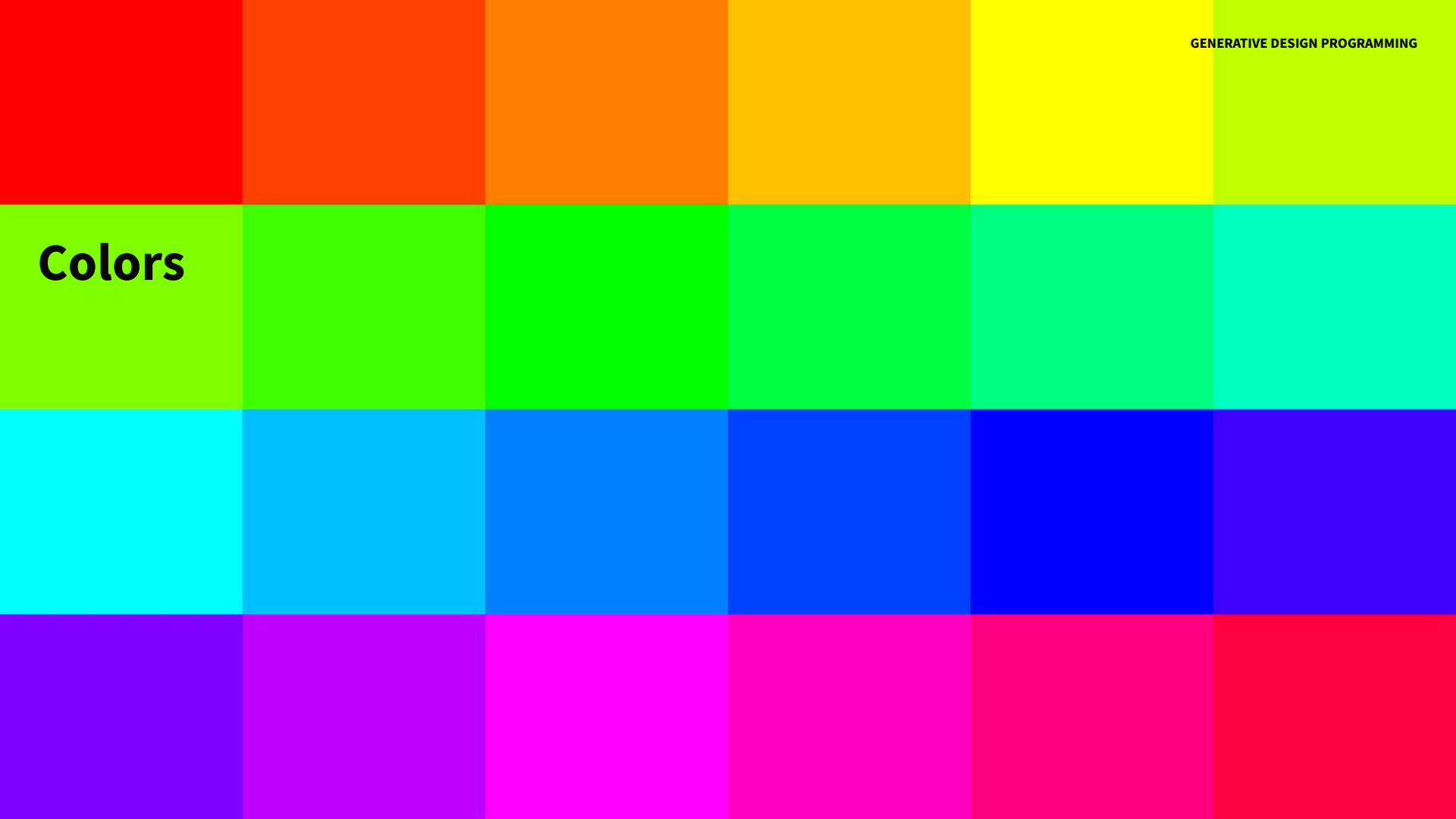
Center it

To help, we have some built-in variables...

width, height // of the artboard in pixels



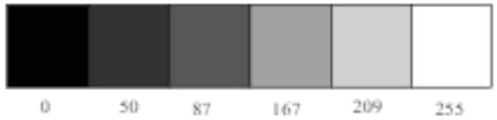
Colors



Digital colors

Grayscale (8bit)

0-255

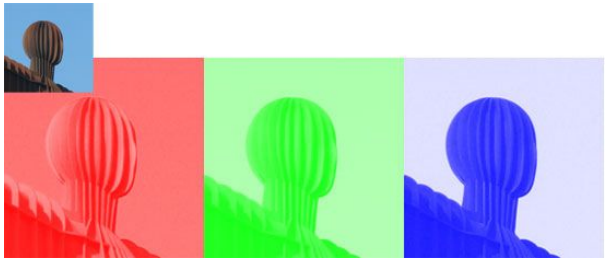


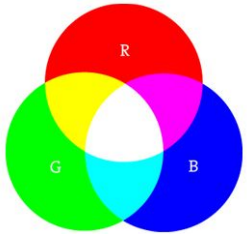
eg.

background (230 **)**

RGB, Red-Green-Blue (8bit * 3 *channels*)

0-255, 0-255, 0-255





eg.

background (0, 210, 0 **)**

About digital color (article)

Color functions

Before drawing shapes, we use these to set the colors:

Fill	**fill** (color) **noFill** ( )
Stroke	**stroke** (color) **noStroke** ( ) **strokeWeight** (number)

And for the background:

Background	**background** (color) **clear** ( )

The ***color*** can be either 1, 3, or 4 arguments:

number 0-255 for grayscale
fill (255)
string hexcode, rgb(a) css-style
fill (“#CD5C5C”)
fill (“rgb(205, 92, 92)”)

3 numbers 0-255 for RGB
fill (0, 244, 12)
+ 1 number 0-1.0 for opacity/alpha
fill (0, 244, 12, 0.23)

object - variable with stored color

Saving colors

You can also store colors in variables:
let c_magenta = **color (** red, green, blue **);**

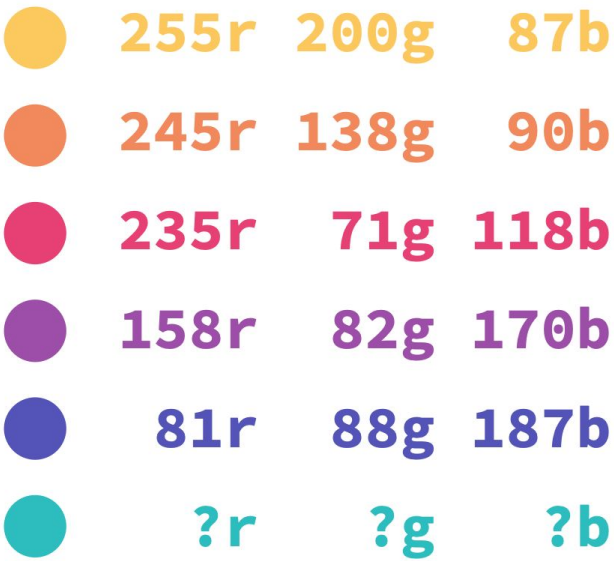
fill (c_magenta **);**

Inside the functions (in **draw()**), consider using **const** keyword instead to declare a constant.

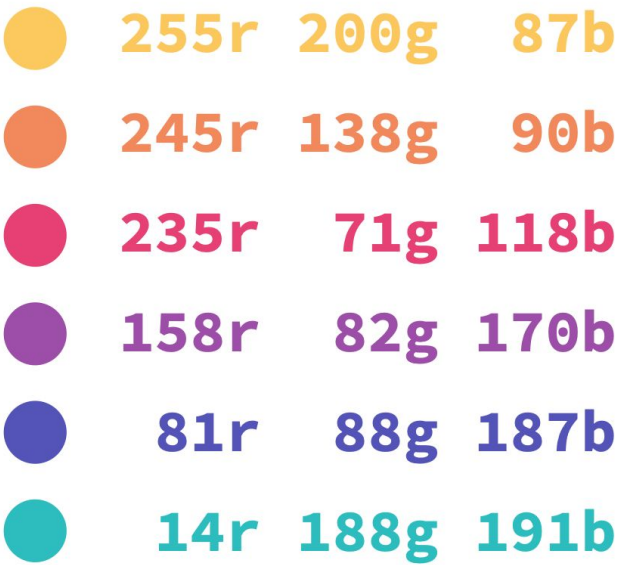
You may also see declaration with **var** from older version of JS, which has sort of weird behavior. Stick to **let** or **const**.

Look at Javascript basics (MDN)

RGB (red - green - blue)

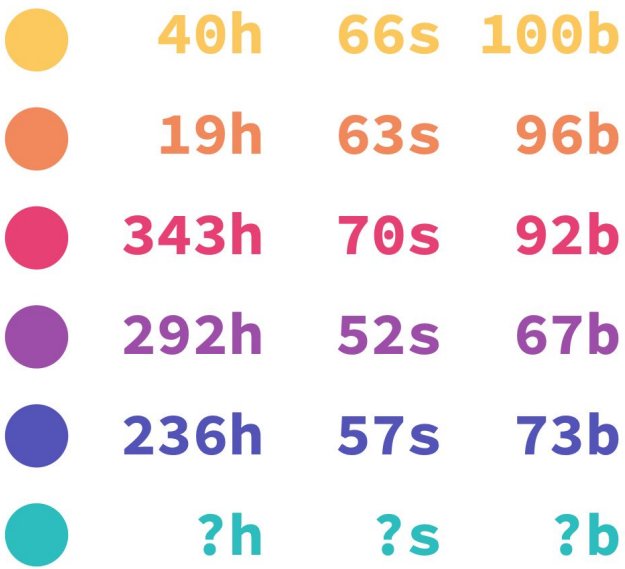


RGB (red - green - blue)

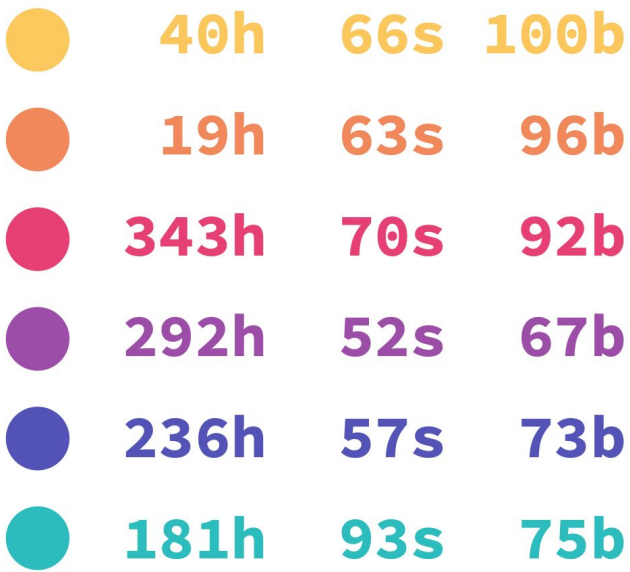


HSB = Hue
Saturation
Brightness

HSB (hue-saturation-brightness)



HSB (hue-saturation-brightness)

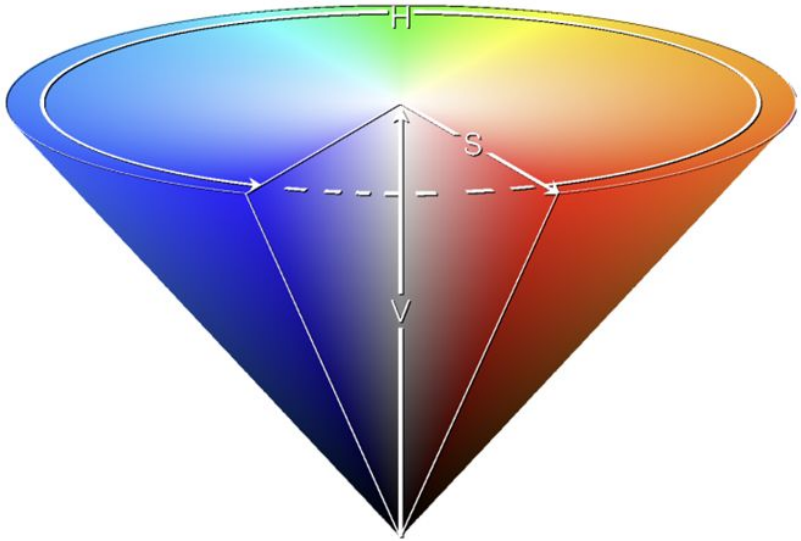


Color modes

colorMode (*string* mode, *number* max, ... **)**

```
//colorMode(RGB, r, g, b, alpha)
colorMode ( RGB, 255, 255, 255, 255 ) // rgba defaults

//colorMode(HSB, h, s, b, alpha)
colorMode ( HSB, 360, 100, 100, 1.0 ) // hsba defaults
```



Color opacity

People also call it **alpha**, **transparency.**

RGB

```
fill ( 255, 0, 0, 255)
    // range from 0 to 255,
    0 being completely transparent and 255 is 100% opaque
```

HSB

```
fill ( 320, 100, 100, 0.23 )

    // default range is 0.0-1.0
```

Sketch #1: Hello circles

Make two circles of your favorite color.
Make the color transparent.
Position the circles such that they overlap.

Code ->

Basic shapes

point(x, y **)**

line(x1, y1, x2, y2 **)**

square(x, y, length **)**

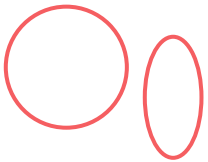
rect(x, y, width, height **)**

rectMode(CENTER | CORNERS)

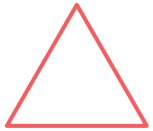
// is x,y center or top left?

circle(x, y, radius **)**

ellipse(x, y, width, height **)**



triangle(x1, y1, x2, y2, x3, y3 **)**



Sketch #2: Mickey

Make an abstract figure using all the shapes you know.

https://en.wikipedia.org/wiki/Mickey_(Damien_Hirst)

Back to circle

Mouse interaction

mouseX, mouseY

// x and y mouse position

sketch.js

```
function setup() {
  createCanvas(400,400);
}

function draw() {
  background(230);
  circle(mouseX,mouseY,150);
}
```

Console Clear

Mouse interaction

Function **mousePressed ( )** { // here goes your art }

mouseClicked ( ) { ... }
mouseDragged ( ) { ... }
mousePressed ( ) { ... }
mouseReleased ( ) { ... }

keyPressed ( ) { ... }
keyTyped( ) { ... }
key
keyCode

Let me take a selfie

```
keyPressed ( ) {
    if (key === “s”) {
        save();
    }
    …
}
```

Check **keyCode** for special keys (LEFT_ARROW, BACKSPACE, ESC, … see more):

```
if (keyCode === RETURN) …
```

Use **keyTyped()** if you want to distinguish between lower and uppercase letters…

Code ->

Fading effect

```
function draw ( ) {
    background ( 255, 10 );
    fill ( “#f55e61ff” );
    ellipse ( random( width ), random( height ), 100, 100 );
}
```

Code ->

Every sketch from class available at
https://editor.p5js.org/mrehacek/collections/Y7yY_s7PN