

## 15-104 Introduction to Computing for Creative Practice

*Fall 2022*

**35 More About DOM** (from [Make: Getting Started with p5.js](#), by McCarthy, Reas and Fry)

Instructor: Tom Cortina, [tcortina@cs.cmu.edu](mailto:tcortina@cs.cmu.edu), GHC 4117, 412-268-3514

1



## Document Object Model

- The DOM stands for Document Object Model.
- The idea is that the entire web page is processed to create nested objects: Every heading, paragraph, table, image, and list, originally in HTML, is accessible as an object via p5.js and JavaScript programming.
- To access “the DOM” from p5.js, you need the “all” template, which includes p5.dom.js, an extension of the core p5.js functions that includes new functions to access DOM objects.

2



## Capturing The Webcam

- The function `createCapture()` accesses the webcam on your computer and creates an HTML element that displays the audio and video feed from it.
- Once the capture element is created, it can be drawn onto the canvas and manipulated further.
- CAUTION: It also plays the audio feed which can cause major feedback!**

3



## Example: Webcam Fun

```
var capture;

function setup() {
    createCanvas(480, 480);
    capture = createCapture();
    capture.hide();
}

function draw() {
    var aspectRatio = capture.height/capture.width;
    var h = width * aspectRatio;
    image(capture, 0, 0, width, h);           Draw the webcam image and invert it.
    filter(INVERT);
}
```

A reference to the capture device.  
We hide it since we will be drawing its data in the draw function.



4



## Filter Options

**THRESHOLD** Converts the image to black and white pixels depending if they are above or below the threshold defined by the level parameter. The parameter must be between 0.0 (black) and 1.0 (white). If no level is specified, 0.5 is used.

**GRAY** Converts any colors in the image to grayscale equivalents. No parameter is used.

**OPAQUE** Sets the alpha channel to entirely opaque. No parameter is used.

**INVERT** Sets each pixel to its inverse value. No parameter is used.

**POSTERIZE** Limits each channel of the image to the number of colors specified as the parameter. The parameter can be set to values between 2 and 255, but results are most noticeable in the lower ranges.

**BLUR** Executes a Gaussian blur with the level parameter specifying the extent of the blurring. If no parameter is used, the blur is equivalent to Gaussian blur of radius 1. Larger values increase the blur.

**ERODE** Reduces the light areas. No parameter is used.

**DILATE** Increases the light areas. No parameter is used.

5

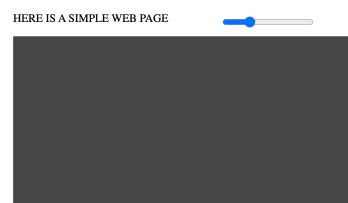


## Example: Slider

```
var slider;

function setup() {
  createCanvas(480, 240);
  slider = createSlider(0, 255, 100);           Range of slider: 0 to 255
  slider.position(300, 20);                   Initial position of slider: 100
}                                              Position of slider on webpage: (300, 20)

function draw() {
  var grayvalue = slider.value();
  background(grayvalue);                     Get the current value of the slider
}                                              and set the background color to that value.
```



6



## Example: Color Slider

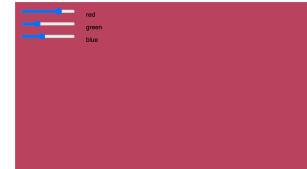
```

var rSlider;
var gSlider;
var bSlider;

function setup() {
  // create canvas
  createCanvas(710, 400);
  textSize(15);
  noStroke();
  rSlider = createSlider(0, 255, 100);
  rSlider.position(20, 20);
  gSlider = createSlider(0, 255, 0);
  gSlider.position(20, 50);
  bSlider = createSlider(0, 255, 255);
  bSlider.position(20, 80);
}

function draw() {
  var r = rSlider.value();
  var g = gSlider.value();
  var b = bSlider.value();
  background(r, g, b);
  text('red', rSlider.x * 2 +
    rSlider.width, 35);
  text('green', gSlider.x * 2 +
    gSlider.width, 65);
  text('blue', bSlider.x * 2 +
    bSlider.width, 95);
}

```



7



## Example: Input Box and Button

```

var inputbox;
var button;

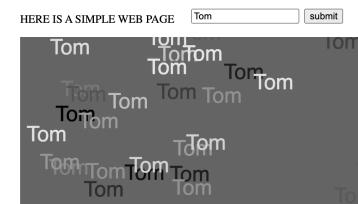
function setup() {
  createCanvas(480, 240);
  inputbox = createInput();
  inputbox.position(250, 10);
  button = createButton("submit");
  button.position(410, 10);
  button.mousePressed(drawName);
  background(100);
  noStroke();
  text("Enter your name.", 20, 20);
}

```

```

function drawName() {
  background(100);
  textSize(30);
  var name = inputbox.value();
  for (var i = 0; i < 30; i++) {
    fill(random(255));
    text(name, random(width),
      random(height));
  }
}

```



8



## Some GUI Elements

```
createSlider(min, max, [value], [step])
    min      Number: minimum value of the slider
    max      Number: maximum value of the slide
    value    Number: default value of the slider (Optional)
    step     Number: step size for each tick of the slider (if step is set to 0, the slider
                will move continuously from the minimum to the maximum value) (Optional)

createButton(label, [value])
    label    String: label displayed on the button
    value    String: value of the button (Optional)

createInput([value])
    value    String: default value of the input box (Optional)

createSelect([multiple])           (see next slide for createSelect methods)
    multiple Boolean: true if dropdown should support multiple selections (Optional)
```

9



## GUI Element: Select

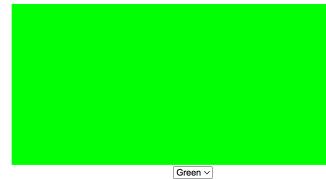
Methods that can be used with a Select object:

- .option(name, [value])** can be used to set options for the select after it is created.
- .value()** will return the currently selected option.
- .selected()** will return current dropdown element which is an instance of p5.Element
- .selected(value)** can be used to make given option selected by default when the page first loads.
- .disable()** marks whole of dropdown element as disabled.
- .disable(value)** marks given option as disabled

10

## Example: Drop Down Menu

```
var colormenu;
function setup() {
  createCanvas(480, 240);
  colormenu = createSelect();
  colormenu.option("", color(128, 128, 128)); // initial choice
  colormenu.option("Red", color(255, 0, 0));
  colormenu.option("Green", color(0, 255, 0));
  colormenu.option("Blue", color(0, 0, 255));
  colormenu.position(250, 250);
  noStroke();
}
function draw() {
  background(colormenu.value());
}
```



Green ▾

11

## Example: Color Averager

```
var inp1;
var inp2;
function setup() {
  createCanvas(100, 100);
  background('grey');
  inp1 = createColorPicker('#ff0000');
  inp1.position(35, height + 10);
  inp1.input(setShade1);
  inp2 = createColorPicker(
    color('yellow'));
  inp2.position(35, height + 35);
  inp2.input(setShade2);
  setMidShade();
}

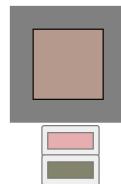
lerpColor:
Blends two colors to find a third color
somewhere between them.
```

```
function setMidShade() {
  // Finding a shade
  // between the two
  var commonShade =
    lerpColor(inp1.color(),
              inp2.color(), 0.5);
  fill(commonShade);
  rect(20, 20, 60, 60);
}

function setShade1() {
  setMidShade();
  console.log('shade 1: ', this.value());
}

function setShade2() {
  setMidShade();
  console.log('shade 2: ', this.value());
}
```

Parameter between 0 and 1:  
0.5 means halfway between  
the two colors  
0.0 means only first color  
1.0 means only second color



12