

## 15-104 Introduction to Computing for Creative Practice *Fall 2022*

### 03 Variables and a little bit of Logic

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

1



## Conditionals (the `if/else` statement)

- An `if-else` statement allows to test a logical condition to determine whether to run some code or some other code.
- General forms for `if` and `if-else`:

```
if ( condition ) {           if ( condition ) {  
    instruction(s) if true          instruction(s) if true  
}                           } else {  
                           instruction(s) if false  
}
```

2



## Example (McCarthy, Reas, Fry)

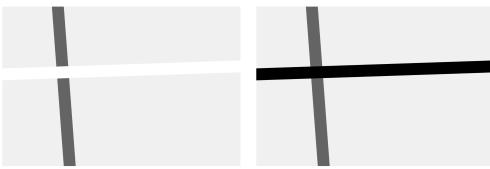
`mouseIsPressed` is a p5.js system variable that evaluates to true if the mouse is pressed down and false otherwise.

```

function setup() {
  createCanvas(600, 400);
  strokeWeight(30);
}

function draw() {
  background(240);
  stroke(102);
  line(140, 0, 170, height);
  if (mouseIsPressed) {
    stroke(0);
  } else {
    stroke(255);
  }
  line(0, 170, width, 150);
}

```



3



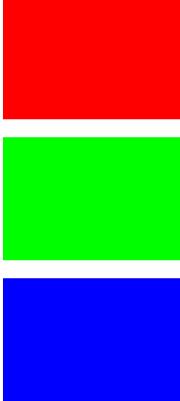
## Example (layered ifs)

```

function setup() {
  createCanvas(600, 400);
}

function draw() {
  if (mouseX < (width * 0.33)) {
    background(255, 0, 0);
  } else if (mouseX > (width * 0.66)) {
    background(0, 255, 0);
  } else {
    background(0, 0, 255);
  }
}

```

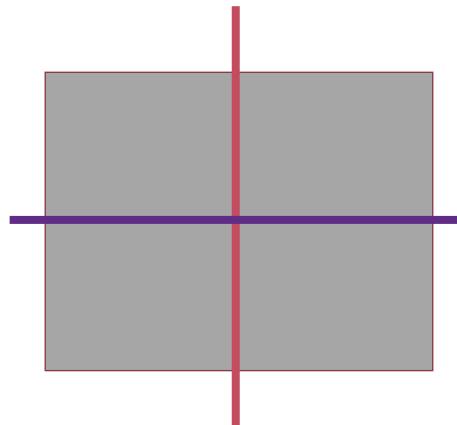


4



## Four quadrants

```
function draw() {
    if (mouseX < width/2) {
        if (mouseY < height/2) {
            background(255,0,0); // red
        } else {
            background(0,255,0); // green
        }
    } else {
        if (mouseY < height/2) {
            background(0,0,255); // blue
        } else {
            background(0,0,0); // black
        }
    }
}
```



5



## Logical AND &&

```
function draw() {
    if (mouseX < width/2 && mouseY < height/2) {
        background(255,0,0); // red
    } else if (mouseX < width/2 && mouseY >= height/2) {
        background(0,255,0); // green
    } else if (mouseX > width/2 && mouseY < height/2) {
        background(0,0,255); // blue
    } else {
        background(0,0,0); // black
    }
}
```

Logical AND (&&): Both conditions must be true for the whole condition to be true.

6



## Variables

- A variable is a container that holds some data value.
- We use variables in program statements to use that data value to perform a computation.
- p5.js has some variables that are predefined in the language to mean something:  
`mouseX`, `mouseY`, `width`, `height`, `mouseIsPressed`
- Variables in p5.js have implicit data types.  
(e.g. number, Boolean)

7



## We can define our own variables

- Global variables – defined before the setup function (i.e. not inside any specific function)
  - Pro: Accessible throughout the entire program
  - Con: Any part of the program could modify the variable, making it harder to debug
- Local variables – defined within a function
  - Pro: Only one function can access and modify the variable, making it easier to debug
  - Con: Not accessible by other functions, but we can pass the value in a variable into another function using a parameter

8



## Example

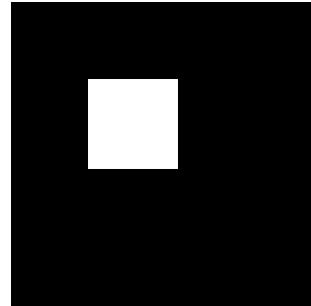
We declare a variable using the keyword `var`.

```
var boxWidth = 60;

function setup() {
    createCanvas(200, 200);
}

function draw() {
    background(0);
    rect(50, 50, boxWidth, boxWidth);
}
```

Here, the `=` operator does not mean equality. Instead, it means **assignment** (i.e. Assign the variable `boxWidth` the value of 60. Symbolically: `boxWidth ← 60`)



9



## Modifying variables

- Within our code, we can modify variables.
- Example: As soon as the mouse crosses to the right half, double the box width.

```
function draw() {
    background(0);

    if (mouseX > width / 2) {
        boxWidth = 120;
    }

    rect(50, 50, boxWidth, boxWidth);
}
```

10



## Motion

```
var boxWidth = 60;
var boxX = 50;
function setup() {
    createCanvas(200, 200);
}
function draw() {
    background(0);
    boxX = boxX + 1; ← What does this line do?
    rect(boxX, 50, boxWidth, boxWidth);
}
```

11



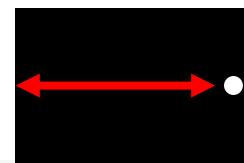
## Modify draw so box reappears

```
function draw() {
    background(0);
    if (boxX > width) {
        boxX = -boxWidth; ← Same as:
                           boxX = -1 * boxWidth;
    }
    boxX = boxX + 1;
    rect(boxX, 50, boxWidth, boxWidth);
}
```

12

## A tennis match

```
var x = 300;
var dir = 1;
var speed = 5;
function setup() {
  createCanvas(600, 400);
}
// cont'd in next column
```



```
Same as:  
x = x + dir * speed;  
  
function draw() {  
  background(0);  
  ellipse(x, height/2, 50, 50);  
  x += dir * speed;  
  if (x > width - 25) {  
    dir = -dir;  
  }  
  if (x < 25) {  
    dir = -dir;  
  }  
}
```

Why 25?

13

## Logical OR

```
var x = 300;
var dir = 1;
var speed = 5;
function setup() {
  createCanvas(600, 400);
}
// cont'd in next column
```

```
function draw() {  
  background(0);  
  ellipse(x, height/2, 50, 50);  
  x += dir * speed;  
  if (x > width - 25 || x < 25) {  
    dir = -dir;  
  }  
}
```

Logical OR (||): Only one condition needs to be true for the whole condition to be true.

14



## Expanding and Contracting

```

var dir = 1;
var speed = 5;
var diam = 50;

function setup() {
  createCanvas(600, 400);
}

function draw() {
  background(0);
  ellipse(width/2, height/2, diam, diam);
  diam += dir * speed;
  if (diam > 400) {
    dir = -dir;
    diam = 400;
  } else if (diam < 0) {
    dir = -dir;
    diam = 0;
  }
}

```

15



## Try these:

- Make the circle have separate horizontal and vertical speeds and have it bounce off of each of the four sides of the canvas.
- Have the white circle expand until it completely fills the canvas. Then have a black circle expand until it completely fills the canvas. Repeat forever.

16