VARIABLES

Storing and Manipulating Data

BOOTCAMP DAY 2

agenda

- everybody send me your HW so we can look at it together now!
- review
- variables
- basic data types
- naming conventions
- basic operators
- debugging
- basic interaction
- saving images
- ProcessingJS / openProcessing

Review:

What's an IDE?

What's a good analogy for code?

How does the coordinate system work?

What's void setup() {...} ?

What's void draw() {...} ?

Variables


GUIDING PRINCIPLES
DESIGN ETHICS + AESTHETICS
NEW DESIGN PRINCIPLES
HERITAGE
BRAIN
BUSINESS ETHICS
SMALL CAUSES
BIG EFFECTS
GUIDING PRINCIPLES

Variables

Think of variables as **containers** for holding pieces of information.

You need different containers for different types of information.

Variables

More technically:

“A variable is a named pointer to a location in the computer’s memory where data is stored. Since computers only process information one instruction at a time, a variable allows a programmer to save information from one point in the program and refer back to it at a later time.”

Variables

Two really useful features:

1. Data can be stored in a computer's memory so a program can access it and use it over and over.

2. Even better, a program can **change** this information while running.

Variables

What sort of things can you do?

Keep track of information related to shapes: color, size, location.

Data types

- int
  - whole numbers
  - 1, 2, -3, 40, 2013
- float
  - floating point number (decimal points)
  - 0.5, 1.34, 11.5, 7.0/2.0

Data types

- String
  - ASCII characters, declared in quotation marks
  - "lemon", "meringue", "pie"
- boolean
  - true, false
- color
  - colors; by default uses RGB
  - (255, 177, 80)

Variables are **declared** and **initialized(assigned)**.

Structure

Declarations

```
int x; //declare a variable for x
float y; //declare a variable for y
```

Structure

Declarations

```
int x; //declare a variable for x
float y; //declare a variable for y
```

Setup

```
void setup() {
    size(500,500);
    x = 0; //initialize x to 0
    y = 5.3; //initialize y to 5.3
}
```

Structure

Declarations

```
int x; //declare a variable for x
float y; //declare a variable for y
```

Setup

```
void setup() {
    size(500,500);
    x = 0; //initialize x to 0
    y = 5.3; //initialize y to 5.3
}
```

Draw

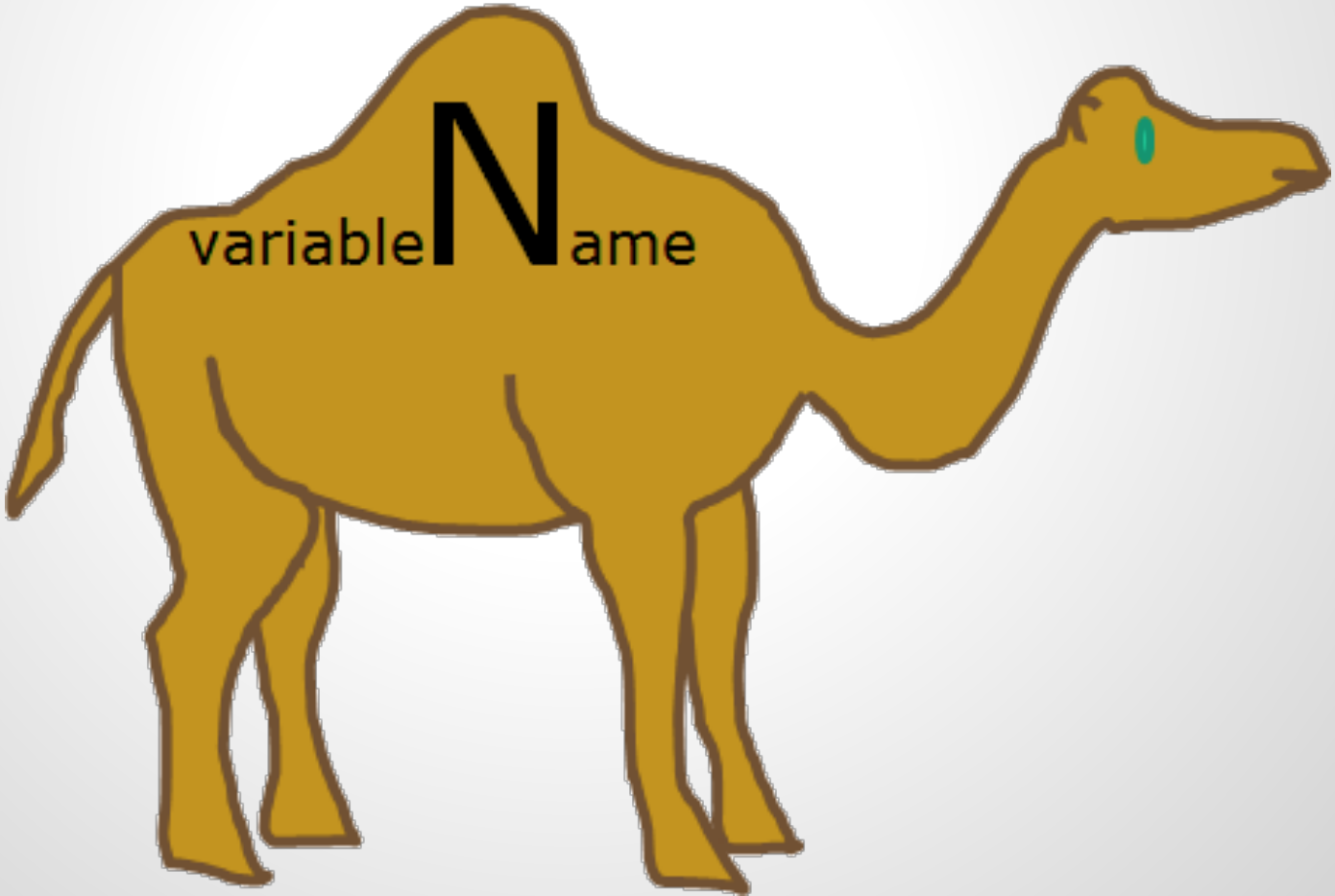
```
void draw() {
    ellipse(50, 200, x, x); //draw a circle
    rect(100, 100, 30, 50); //draw a rectangle
}
```

Variables

```
int x = 0;
float var1 = 1.3;
boolean fred = true;
```

Naming Conventions

just use camelCase, actually. (but there are others)



Commenting Review

```
int variable2;          //We can leave comments for ourselves
float readingValue;     //Anything after the double slash is ignored
                        //by the compiler
```

```
/*
This type of comment also works. Everything between these two
symbol pairs is ignored
*/
```

ALL VARIABLES MUST HAVE...

A TYPE

why? - so the computer knows exactly how much memory should be allocated to store that variable's data.

A NAME

why? - so the computer (and you) can make sense of everything.

VARIABLE NAMING TIPS:

- Avoid using words that appear elsewhere in Processing. (e.g. - don't name a variable *mouseX*)

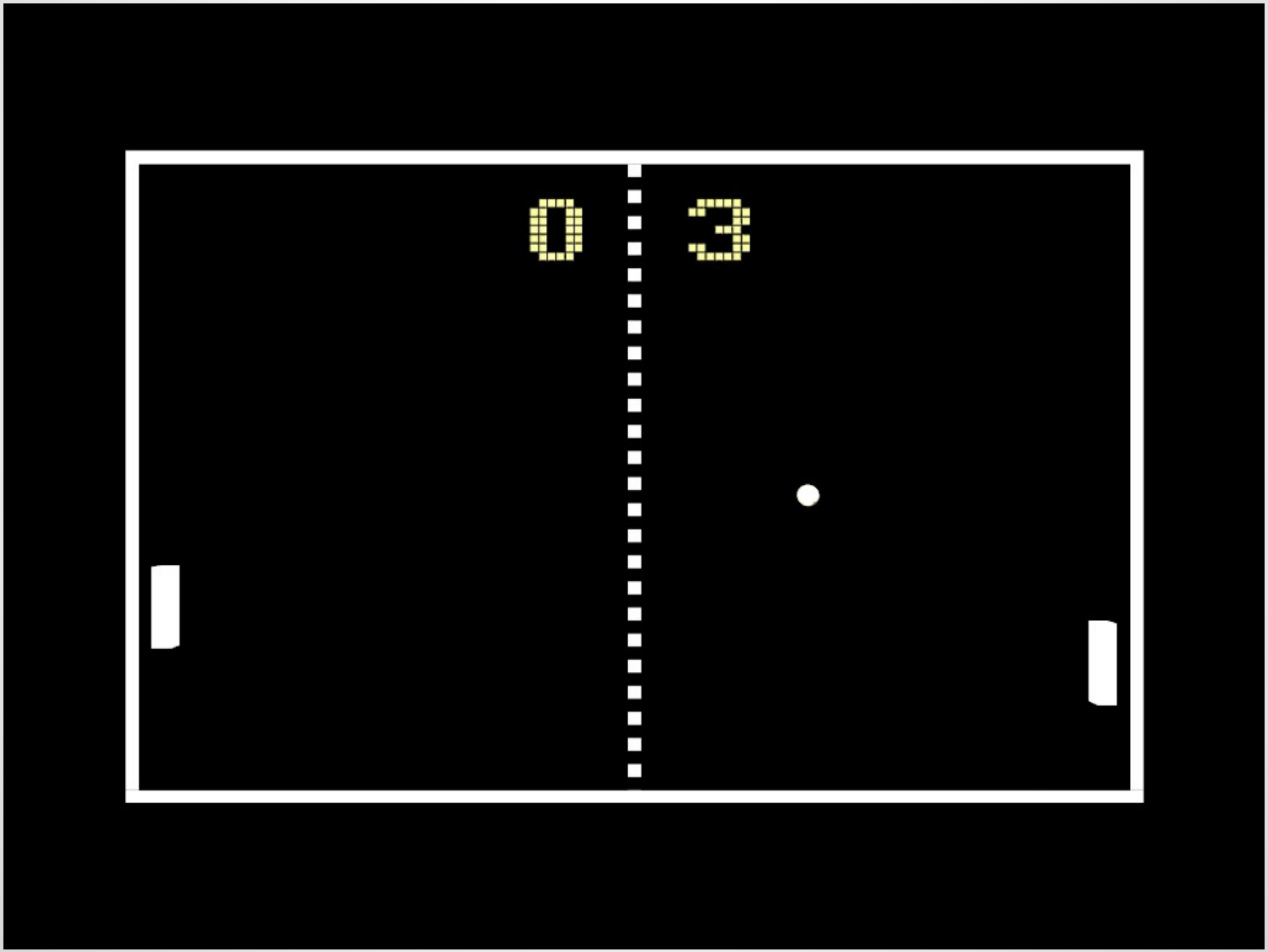
- Use names that mean something connected to what the variable is to be used for. (duh)

- Don't start with capital letters! Use camelCase. (there's a good reason for this but we'll get there later)

Examples

```
int count = 0;
// Declare an int named count, assigned the value 0
char letter = 'a';
// Declare a char named letter, assigned the value 'a'
double i = 132.32;
// Declare a double named i, assigned the value 132.32
boolean happy = false;
// Declare a boolean named happy, assigned the value false
float x = 4.0;
// Declare a float named x, assigned the value 4.0
float y;
// Declare a float named y (no assignment)
y = x + 5.2;
// Assign the value of x plus 5.2 to the previously declared y
float z = x*y + 15.0;
// Declare a variable named z, assign it the value which is x times y plus 15.0.
```

What sorts of variables could we use to have a functioning Pong game?



player 1 x position

player 1 y position

player 2 x position

player 2 y position

player 1 score

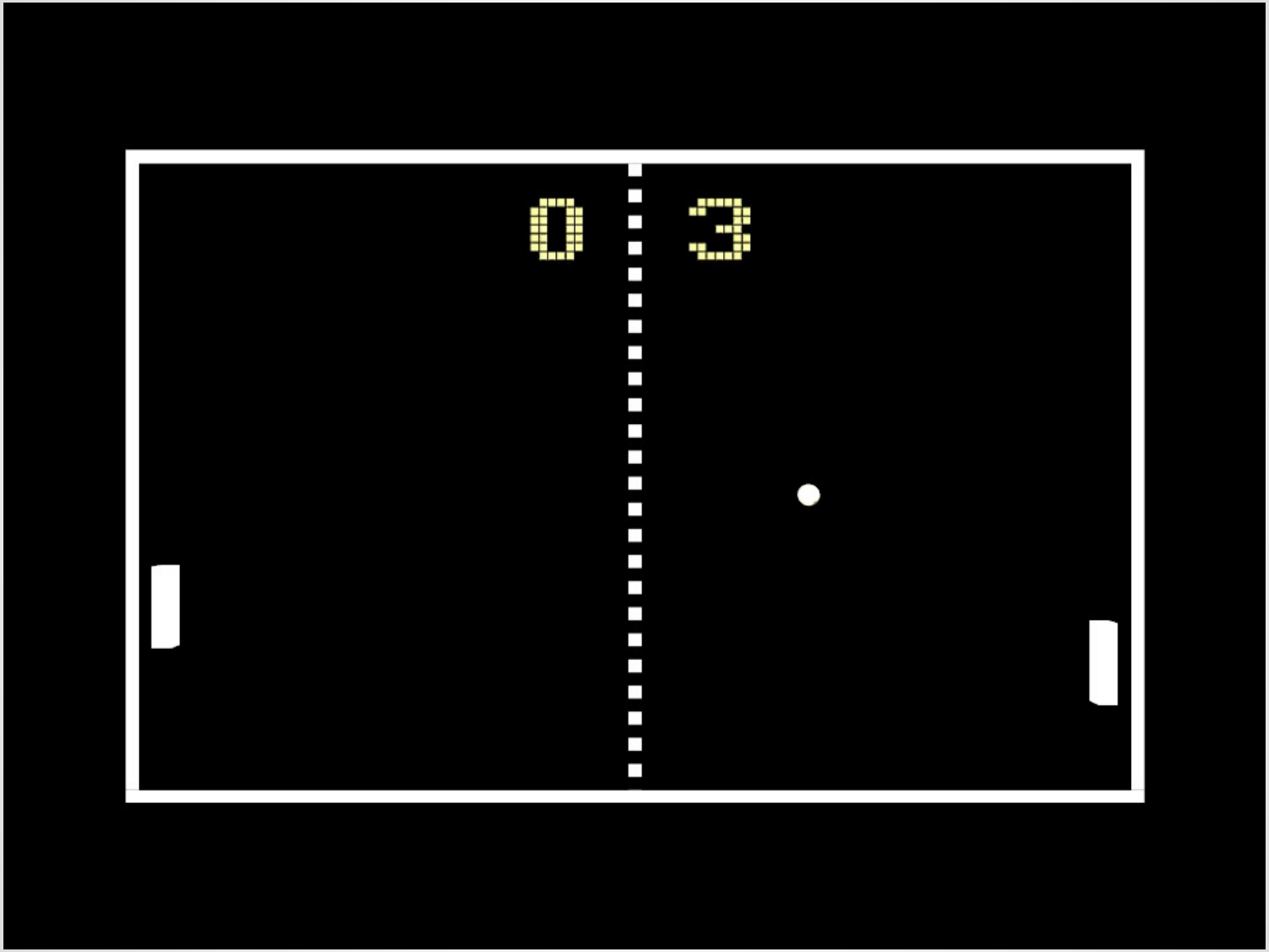
player 2 score

ball x position

ball y position

ball direction

etc.


0
3

Looking back at our material from yesterday:

Let’s use variables in place of our previously hard-coded values.

TRY NOW:

- using ints or floats to set positions
- using
- use the “color” datatype to save colors
- use an int to set individual RGB components that color

organize and name things in a way that makes sense to you.

There are also in-built System Variables for us to make use of

- width
- height
- displayWidth
- displayHeight
- mouseX
- mouseY
- frameCount

Try using system variables to set your object's parameters

Yesterday we made some basic shapes. Now we’re going to use operations to do some basic interaction explorations.

Operators for simple math

An operator is a symbol that represents an *operation.*

The basics:

+ **-** ***** **/** **=**

This may be new to you:

%

```
int a = 5 % 4;              // Sets 'a' to 1
int b = 125 % 100;          // Sets 'b' to 25
float c = 285.5 % 140.0;  // Sets 'c' to 5.5
float d = 30.0 % 33.0;    // Sets 'd' to 30.0
```

What happens if you try to add a float and an int together?

Give it a try now, then we'll figure out what's happening.

Debugging

println();
print();

Use to track your variables and give yourself other messages during runtime.

Use it to see if things are going to plan, or help find out where things are going awry!

Try doing some basic operations and printing the result.

Now take your variables and use them in basic operations instead

Try making:

- A counter that counts up every frame, like framecount

- An “xPos” variable that moves your object across the screen

- A color changer variable that changes the color of an object over time

Map Function - extremely useful

Re-maps a number from one range to another.

For example:

```
float myColorRange = map(mouseX, 0, width, 0, 255);
```

DON'T WORRY ABOUT THIS RIGHT NOW- WE'LL PUT IT IN ANYWAY

```
//inside draw loop


if (keyPressed) {
    if (key == 's') {
        saveFrame();
    }
}
```

simple way for us to save screenshots of our fancy pretty fancy work.

pretty straightforward - this will make more sense later if it doesn't now.

ps. try mousePressed

Homework

CAPTURE A PROCEDURAL IMAGE!

Ideas:

- Parameterize your previous HW to make it interactive and/or animated
- Use variables and operators to change the sketch in some way over time
- Try using system variables too

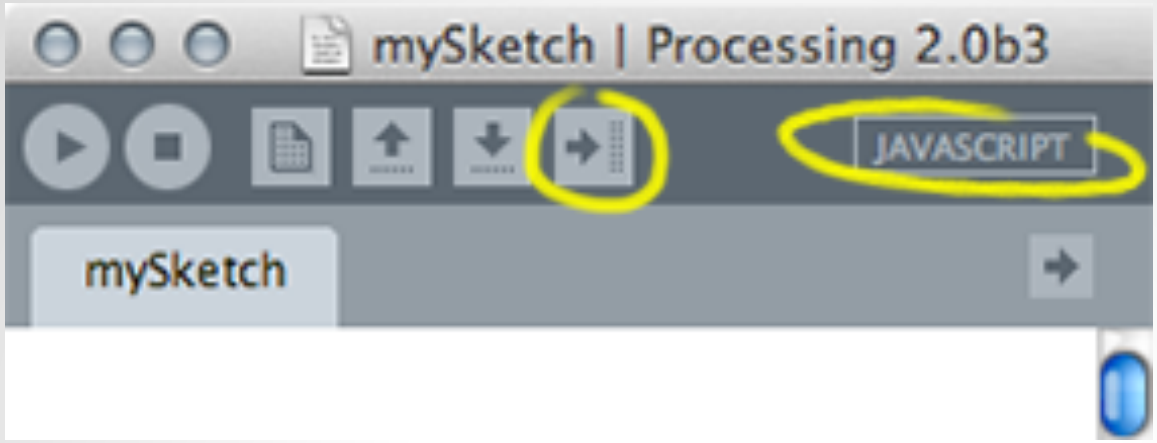
Try not to plan out the result - instead let the look happen naturally with exploration and screenshot something that you think is cool!

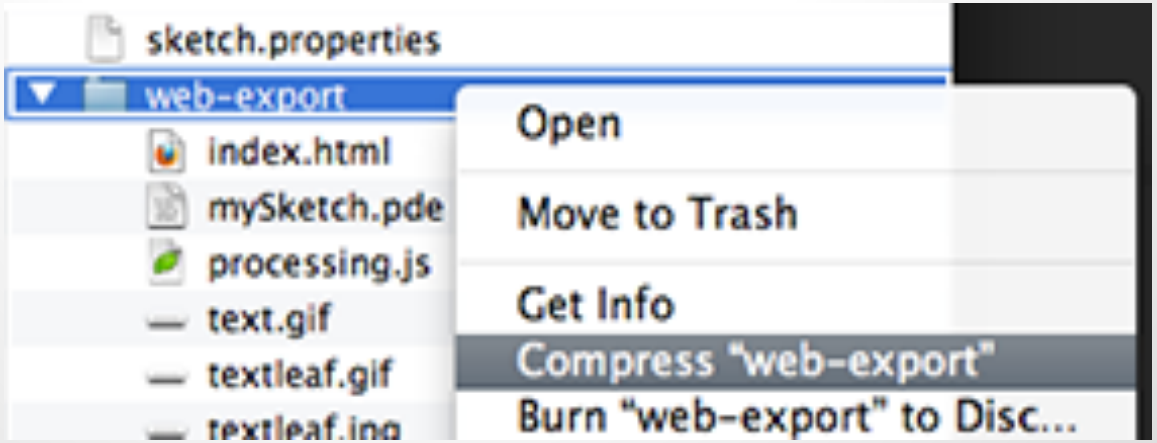
extra areas to explore, if you want -

- rotation
- iteration
- recursion

ProcessingJS

we’ll also be turning in work on openprocessing.org


mySketch | Processing 2.0b3
JAVASCRIPT
mySketch


sketch.properties
web-export
index.html
mySketch.pde
processing.js
text.gif
textleaf.gif
Open
Move to Trash
Get Info
Compress "web-export"
Burn "web-export" to Disc...