NAME: PETER PARKER.
MOOD: Baby.
HOBBIES: Sleeping & Eating.
AGE: 1.

NAME: PETER PARKER.
MOOD: Scared!
AGE: 14.
HOBBIES: - Skateboard
- Play with dogs.

NAME: SPIDERMAN.
MOOD: Confused
AGE: 15
HOBBIES: - Skateboard
- Play with dogs

NAME: SPIDERMAN.
MOOD: Adventurous
AGE: 16
HOBBIES: - Save the World
- Skateboard
- Play with dogs
Variables

Constant Variable
User-defined Variable
Declaration, Initialization, Assignment
Memory

Chapter 4, Examples 4-1, 4-2, 4-3, 4-4, Robot 2 (p. 54-56).
Chapter 8, Examples 8-1, 8-2, 8-3, 8-4, 8-5, 8-7, 8-8, 8-9 8-10, 8-11.
Review: Which image is drawn by this code …
function setup() {
    createCanvas(100, 100);
    background(255);
}
function draw() {
    rect(25, 25, 50, 50);
}
function mousePressed() {
    fill(0); // black
    ellipse(50, 50, 25, 25);
}
Review: Built-in Variables

mouseX, mouseY
frameCount
width, height

pmouseX, pmouseY
mouseIsPressed
mouseButton

keyIsPressed
key, keycode

https://p5js.org/reference/
Review: Variable

- A symbolic **name** used to refer to a **value**.
  - The value may change, but the symbolic name doesn’t.

[link to website]
Variable Types

1. Built-in (or “System”)
   - e.g. mouseX, width

2. Constant

3. User-defined
Variable Types

1. Built-in (or “System”)
   - e.g. mouseX, width

2. Constant

3. User-defined
Constant Variable

- All letters are CAPITALIZED
- Their values do NOT change
Why Constant Variable?

- Represent values that do **NOT** change
- Examples?
Using Constant Variable

e.g.,

PI, HALF_PI, ...

strokeJoin(join)
// where join = BEVEL, MITER, or ROUND

strokeCap(cap)
// cap = SQUARE, PROJECT, or ROUND

https://editor.p5js.org/sanghosuh/sketches/vEAliMLca
Variable Types

1. Built-in (or “System”)
   - e.g. mouseX, width

2. Constant
   - e.g. PI, BEVEL

3. User-defined
Variable Types

1. Built-in (or “System”)
   - e.g. mouseX, width

2. Constant
   - e.g. PI, BEVEL

3. User-defined
Variable Types

1. **Built-in (or “System”)**
   - e.g. mouseX, width
   - Available by default

2. **Constant**
   - e.g. PI, BEVEL
   - Some available by default

3. **User-defined**
   - Unavailable unless created
User-defined vs Built-in Variable

- You **need to** ...
  - create
  - name

- You **may or may not** (optional)
  - assign value
User-defined vs Built-in Variable

- You need to ...
  - create
  - name

- You may or may not (optional)
  - assign value

<table>
<thead>
<tr>
<th>Time</th>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day 0</td>
<td>piggyBank</td>
<td></td>
</tr>
<tr>
<td>Day 1</td>
<td></td>
<td>$1</td>
</tr>
<tr>
<td>Day 2</td>
<td></td>
<td>$2</td>
</tr>
<tr>
<td>Day 3</td>
<td></td>
<td>$3</td>
</tr>
</tbody>
</table>

JANE STARTED PUTTING IN $1 EVERY DAY TO HER PIGGY BANK
Creating ("Declaring") User-defined Variables

```
let piggyBank;
```

- **create**
- **name**
- variable declaration
- statement
Assigning Value to User-defined Variables

(let)

name: piggyBank
assign: = 1;

variable assignment statement
Declaring & Assigning ("Initializing") User-defined Variables

create → name → assign

let piggyBank = 1;
Declare, Assign later vs Declare & Assign ("Initializing")

// Day 0

let piggyBank;

// Day n (n = 1, 2, ...)
piggyBank = 1;

// Day 0

let piggyBank = 1;
Declaring & Assigning ("Initializing") User-defined Variables

let piggyBank = 1;
Declaring & Assigning ("Initializings") User-defined Variables

```
let piggyBank = 1;
```

- **Declaration**: `let` specifies the variable is local.
- **Type**: Not specified in this example.
- **Name**: `piggyBank`
- **Value Assignment**: `1`

**Variable Initialization Statement**
Declaring & Assigning ("Initializing") User-defined Variables

```
let piggyBank = 1;  
```

- declaration
- type
- variable
- name
- value assignment (optional)

variable initialization statement
let, const, and var

- JavaScript has three keywords to declare variables:
  - **let** are for “user-defined” variables
    - variables declared with let can be re-assigned
    - In this course, we will almost always use let
  - **const** are for “constants”
    - These kinds of variables can only be assigned once
    - Useful to hold a value that will never change in your program
      (so you make sure you don’t re-assign it by accident)

**var**

- Do not use **var**
  - Most developers agree var should be avoided
  - In this course, you’ll lose marks if you use var
Declaring & Assigning ("Initializing") User-defined Variables

```
let piggyBank = 1;
```

- declaration
- type
- variable name
- value assignment (optional)
- variable initialization statement
Declaring & Assigning ("Initializing") User-defined Variables

- declaration type
- variable name
- value assignment (optional)

```
let piggyBank = 1;
```

variable initialization statement
We're not “solving for” variables ...

\[
\frac{x + 3}{4} + \frac{(2x - 12) - 1}{3} = 1
\]

\[
12 \left[ \frac{x + 3}{4} + \frac{(2x - 12) - 1}{3} \right] = 12 [1]
\]

\[
3(x + 3) + 4(2x - 13) = 12
\]

\[
5x + 9 + 8x - 52 = 12
\]

\[
11x = 55
\]

\[x = 5\]

... we’re “assigning values to” variables (and then using variables in place of numbers).
Declaring & Assigning ("Initializing") User-defined Variables

- Declaration type
- Variable name
- Value assignment (optional)

```let piggyBank = 1;```
Declaring & Assigning ("Initializing") User-defined Variables

<table>
<thead>
<tr>
<th>declaration</th>
<th>type</th>
<th>variable name</th>
<th>value assignment (optional)</th>
</tr>
</thead>
<tbody>
<tr>
<td>let piggyBank = 1;</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Variable initialization statement
JavaScript Data Types

Number
- whole numbers like -5, 0, 12, ...
- decimal numbers like 0.123, -10.1, 32.0, 0.0

Boolean
- true or false

String
- any literal text, e.g. "hello", "123", etc.

Object
- an Array
- a collection of information you define
- a p5 image
- a p5 colour
... (many more)
What does this represent?

A. Declaring variable
B. Assigning value to variable
C. Creating constant
D. None of the above
Recap: Declaring versus Assigning Variables

- Declaring:
  ```javascript
  let piggyBank;
  ```

- Assigning:
  ```javascript
  piggyBank = 1;
  ```

- Declaring and Assigning (aka "Initializing"):
  ```javascript
  let piggyBank = 1;
  ```
Why create our own variables?

- They ...
... make code easier to change and easier to read

Starter: https://editor.p5js.org/cs105/sketches/-DpiDsI5s

https://editor.p5js.org/cs105/sketches/Y48n3yKis
spiderman

... make creating & expressing ideas easier

https://editor.p5js.org/sanghosuh.sketches/XWUC3nedD
Spreadsheet Analogy

Untitled spreadsheet

File  Edit  View  Insert

100%

$4.95

B4

A1*C3

37

=A1*C3
Spreadsheet Analogy

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$4.95</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>37</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>=cost*items</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Computer Memory and Variables

- **items**: 37
- **cost**: 4.95
- **mouseX**: 67
- **PI**: 3.141...
- **studentid**: "smith567"
Recap: Why create our own variables?

- They can make...
  - changing & reading code easier
  - creating & expressing your ideas easier
  - writing code easier
## Recap: Variable Types

1. **Built-in (or “System”)**
   - e.g. `mouseX`, `width`
   - Available **by default**

2. **Constant**
   - e.g. `PI`, `BEVEL`
   - Some available **by default**

3. **User-defined**
   - e.g. `donutSize`, `bgShade`
   - Unavailable **unless created**

**NOTE:** They serve different purposes
Using Variables

- We don't always know the value (or even care what the exact value is), we just refer to it with a name.

mouseX

PI

donutSize
let vs const
where to declare
assigning multiple times
++ -- += -= *= /= shortcuts
implicit "conversion" of types

starter: https://editor.p5js.org/sanghosuh/sketches/J6PHkL HvH
https://editor.p5js.org/sanghosuh/sketches/J6PHkL HvH
Math

“Math can be an important aspect of programming, but it’s not necessary to be good at math to understand or enjoy programming.

There are as many styles of programming as there are people who program, and it’s the decision of the individual to utilize or ignore math as they prefer. People who enjoy math often write programs to visualize equations or take delight in exploring phenomena such as fractals. People who struggled with math in school sometimes find they enjoy and understand it better when it is applied to form and motion.”

(from Reas and Fry, Processing Handbook)
(math expression demos)

+  add

-  subtract
   ...  negative numbers

*  multiply

( )  brackets

/  divide
   ...  divide by 0 makes Infinity

%  modulo
   (remainder after division)

Order of operations: B E DM AS

https://editor.p5js.org/sanghosuh/sketches/ipgSM5c01
(squares1)

using math expressions with built-in variables

https://editor.p5js.org/cs105.sketches/IJTBU4rj
What does this code draw?

function setup() {
  createCanvas(100, 50);
}

function draw() {
  rect(width - 5, height - 5, 10, 10);
}
(squares2)

using user-defined variables

https://editor.p5js.org/cs105.sketches/wUnLfEl9Q
Functions that Return a Value

- Some function calls return a value:
  - random, min, max, (and many more we’ll see later)

- Assign the "result" of these functions to a variable:
  // assign a random number from 0 to 99.99999 to x
  x = random(100);

  // assign whichever is smallest, 50 or mouseX
  x = min(50, mouseX);

- Use the “result” of these functions as a function argument:
  // draw a point at a random x and y location
  point(random(width), random(height));

  // mouseY changes weight, but only up to 25 pixels
  strokeWeight(min(mouseY, 25));
min(a, b)

Determines the smallest value in a sequence of numbers, and then returns that value. min() accepts any number of Number parameters.

Syntax

min(a, b)

Parameters

a  Number:  Number to compare
b  Number:  Number to compare

Returns

Number: minimum Number

From p5 Reference for min: https://p5js.org/reference/#/p5/min
function draw() {
    ...
    // set w to be the smallest of
    // mouseY and 25
    let w = min(mouseY, 25);
    strokeWeight(w);
    ...
}

function draw() {
    ...
    // set to be the smallest of
    // mouseY and 25
    strokeWeight(min(mouseY, 25));
    ...
}

Starter: https://editor.p5js.org/cs105/sketches/NOgT7IORD

https://editor.p5js.org/cs105/sketches/0QZM7wHns
let name = "PETER PARKER";
let mood = "Baby";
let age = 1;

function setup() {
age = 16;
print(age);
}

function draw() {
age = 20;
print(age);
}
What does this code draw?

```javascript
let a = 50;
function setup() {
  createCanvas(100, 50);
  strokeWeight(20);
  a = 100;
}
function draw() {
  point(a, height / 2);
}
```