Variables & Datatypes

CSE 120 Spring 2017

Instructor: Justin Hsia

Teaching Assistants: Anupam Gupta, Braydon Hall, Eugene Oh, Savanna Yee

Please stop charging your phone in public ports

"Just by plugging your phone into a [compromised] power strip or charger, your device is now infected, and that compromises all your data," Drew Paik of security firm Authentic8 explained.

Public charging stations and wi-fi access points are found in places like airports, planes, conference centers and parks, so people can always have access to their phones and data. But connecting your phone to an unknown port has its risks.

The cord you use to charge your phone is also used to send data from your phone to other devices. If a port is compromised, there's no limit to what information a hacker could take, Paik explained.

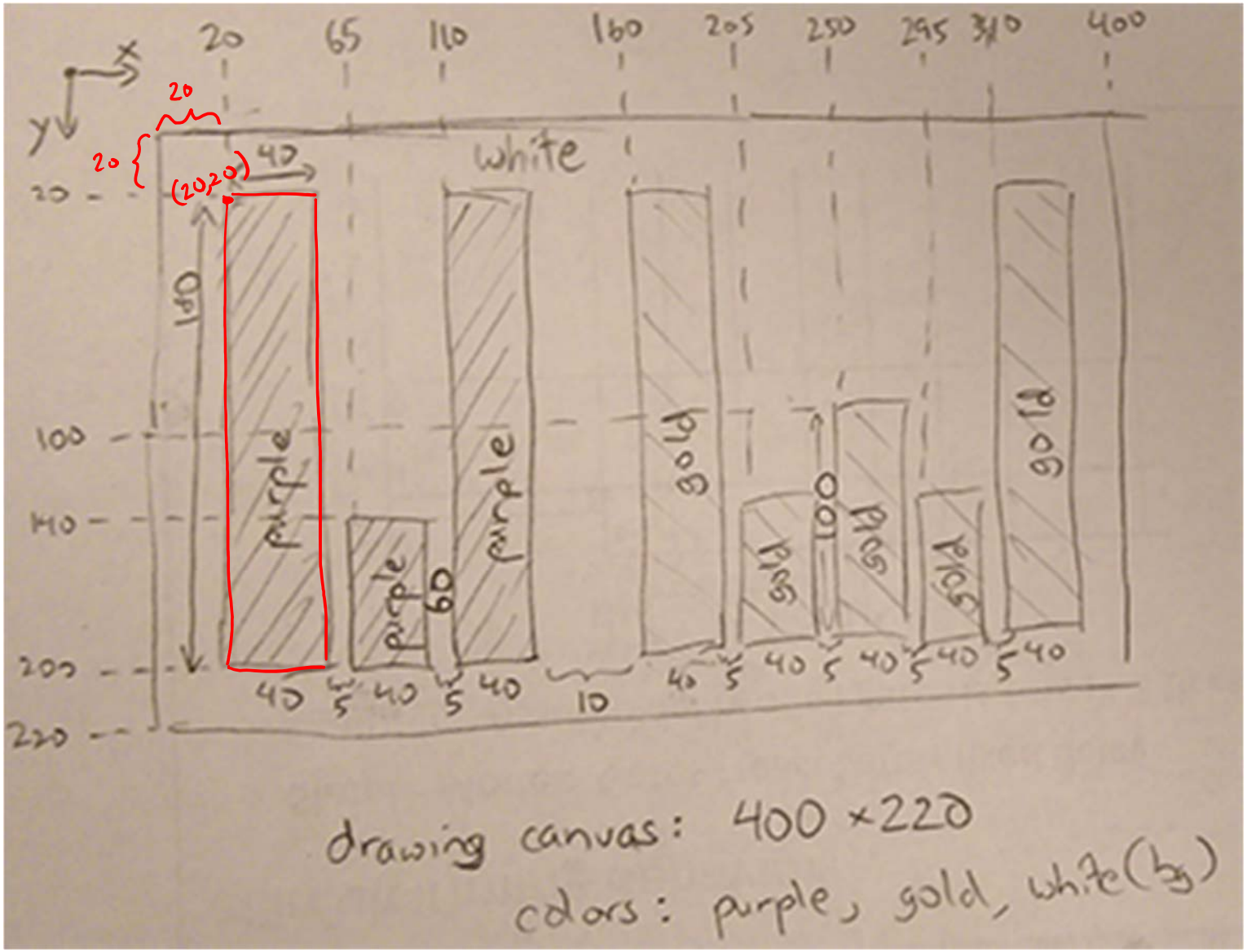
- http://money.cnn.com/2017/02/15/technology/public-ports-charging-bad-stop/

Administrivia

- Assignments:
  - Taijitu due today (4/5)
  - Reading Check 2 due tomorrow (4/6)
  - Custom Logo due Friday (4/7)
- Assignment rubrics on Canvas
- No "big ideas" lecture this week
  - More time on programming

Lab: Custom Logo

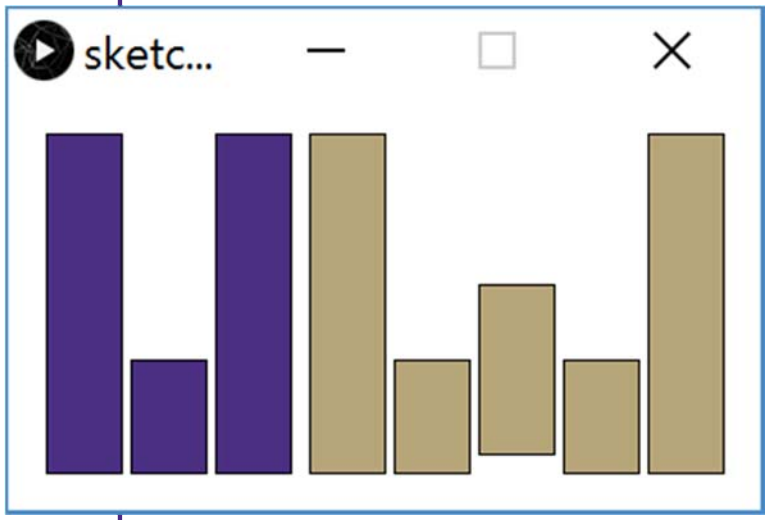
Coordinate system reminder



Lab: Custom Logo

uw_logo

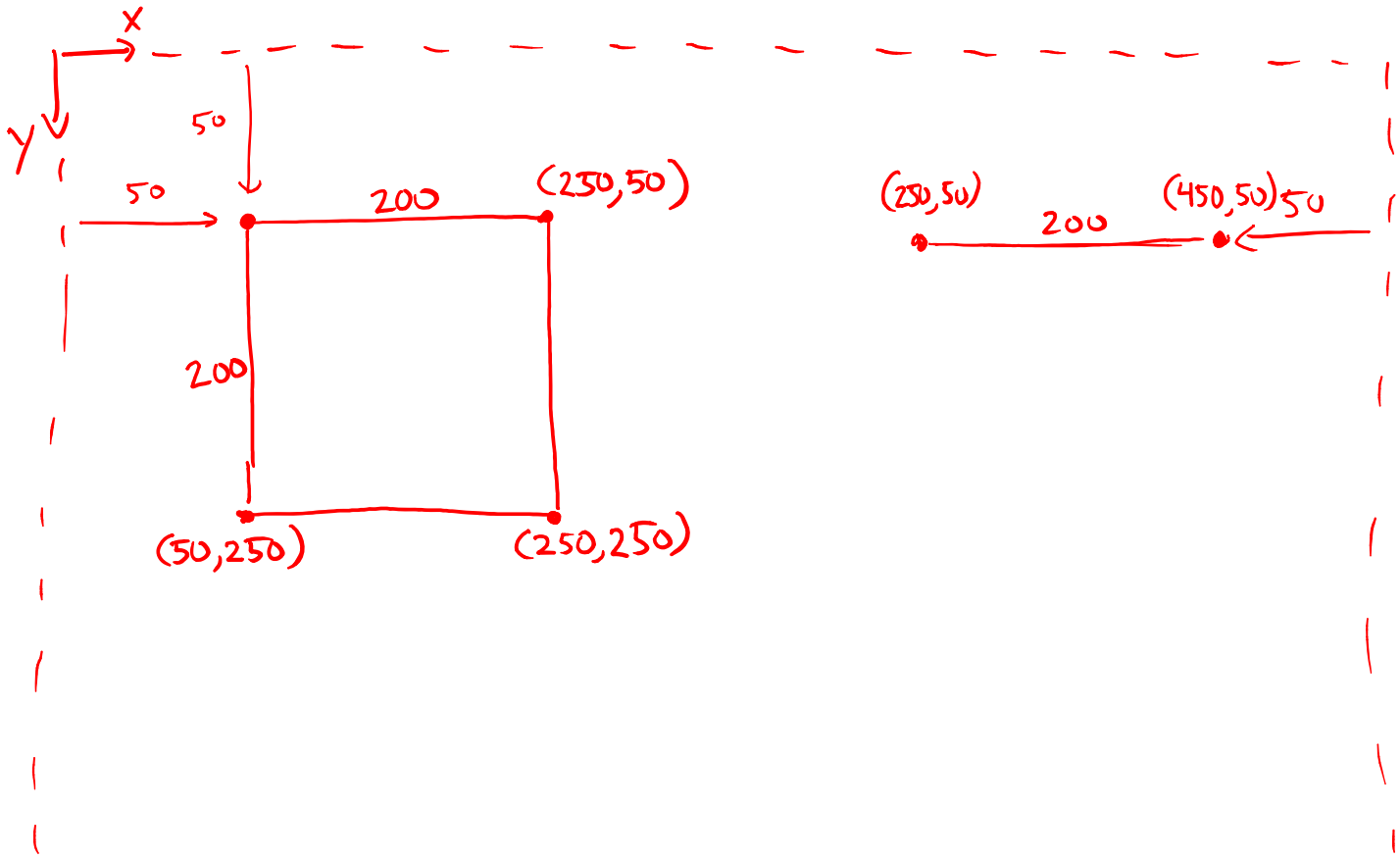
```
/* uw_logo.pde
   Created by Justin Hsia

   UW logo made out of rectangles in school colors.
*/

size(400,220);     // drawing canvas of 400x220
background(255);   // white background

// The letter 'U' in purple
fill( 75,  47, 131);       // purple fill
rect( 20,  20, 40, 180);   // left side of U
rect( 65, 140, 40,  60);   // middle base of U
rect(110,  20, 40, 180);   // right side of U

// The letter 'W' in gold
fill(183, 165, 122);       // gold fill
rect(160,  20, 40, 180);   // left segment of W
rect(205, 140, 40,  60);   // left base of W
rect(250, 100, 40,  90);   // middle segment of W
rect(295, 140, 40,  60);   // right base of W
rect(340,  20, 40, 180);   // right segment of W
```



Drawing a Square

- [See Demo on Panopto]

to move square from left side of drawing canvas to the right side, need to update x-position of all 4 end points!



Variables

- Piece of your program that holds the value of something
  - Every variable must be given a *name* and a *datatype*
- The values of these variables can change (*i.e.* vary) during the execution of your program
  - Warning: Not like a variable in Algebra (*i.e.* an unknown)

- Assignment: give a variable a specific value
  - *e.g.* `x = 12;`
- Read: use the current value of a variable
  - *e.g.* `y = x + 1;`

Datatypes

- `int`: integers
- `float`: decimal/real numbers
- `color`: a triple of numbers representing RGB
- `boolean`: true or false

- Many more exist and can be found in the Processing Reference:

Primitive

boolean
byte
char
color
double
float
int
long

Declarations

- We declare a variable by telling Processing the variable's datatype, followed by the variable's name:

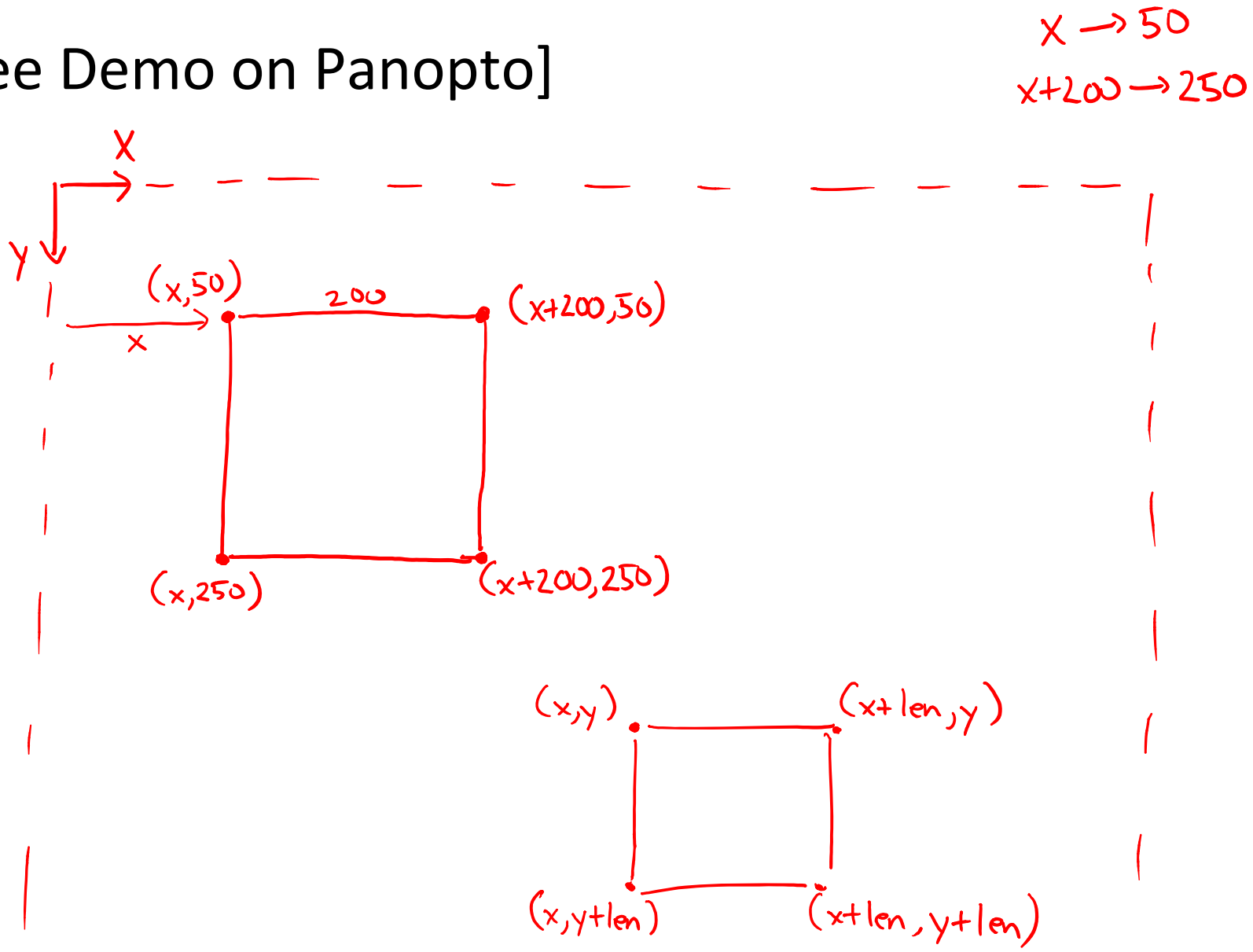
```
1 int x;
2 float half;
3 color yellow;
```

- You can also give a variable a starting value (initialization) in the same line as the declaration:

```
1 int x = 4;
2 float half = 0.5;
3 color yellow = color(255, 255, 0);
```

Drawing a Square with Variables

- [See Demo on Panopto]



Variable Rules & Guidelines

- Variables are case-sensitive
  - e.g. `leftside` is not the same as `leftSide`

- Variable names are meaningless to computers, but meaningful to humans
  - Choosing informative names improves readability and reduces confusion

- In this class, most of our variables will be declared and initialized at the very top of our programs

Variable Manipulation

- Executed sequentially, just like other statements
- For variable assignments, compute right-hand side *first*, then store result in variable
- Example:

```
int x = 4;
x = x + 1;
```

right-hand side

1) Read the current value of x (4) for the right-hand side
2) Add 1 to the current value of x
3) Store the result (5) back into x

Variable Practice

1)
```
int x = 1;
int y = 2;
int z = 3;

x = x + 1;
y = y - 1;
z = z + 2;
```

x	y	z
2	1	5

2)
```
int x = 7;
int y = 2;
int z = 0;

x = x + 3;
y = y - 2;
z = x + y;
```

x	y	z
10	0	10

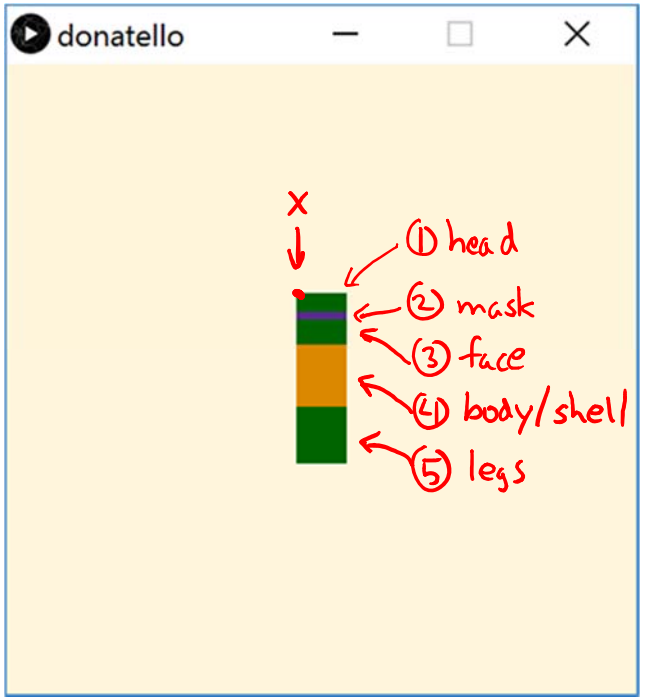
3)
```
int x = -1;
int y = 0;
int z = 5;

x = x + z;
y = y - x;
z = x + z;
```

x	y	z
4	-4	9

TMNT: Donatello

donatello

```
size(500,500);
noStroke();
background(255,245,220);

// Donatello
fill(0,100,0);          // dark green
rect(230,182,40,15);    // top of head

fill(88,44,141);        // purple
rect(230,197,40,6);     // bandana mask

fill(0,100,0);          // dark green
rect(230,203,40,20);    // bottom of head

fill(219,136,0);        // dark yellow
rect(230,223,40,50);    // shell

fill(0,100,0);          // dark green
rect(230,273,40,45);    // lower body
```



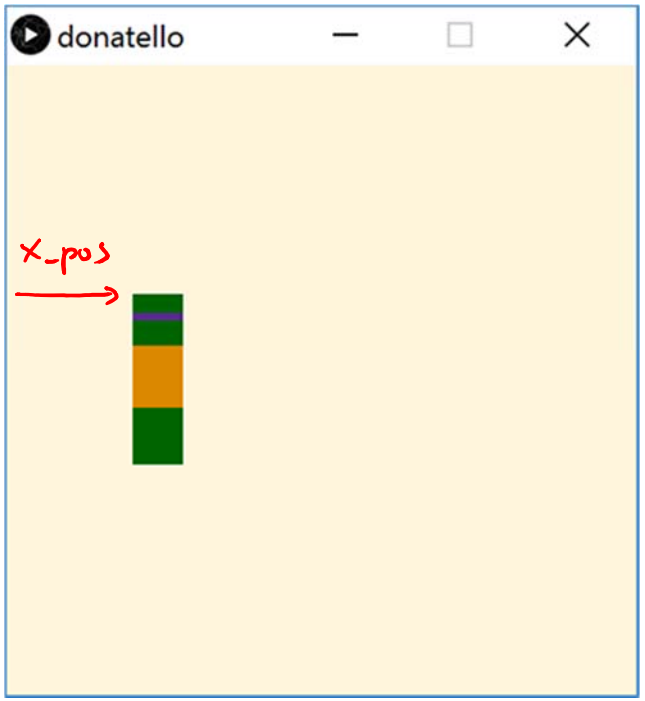
Donatello with a Variable

new variable

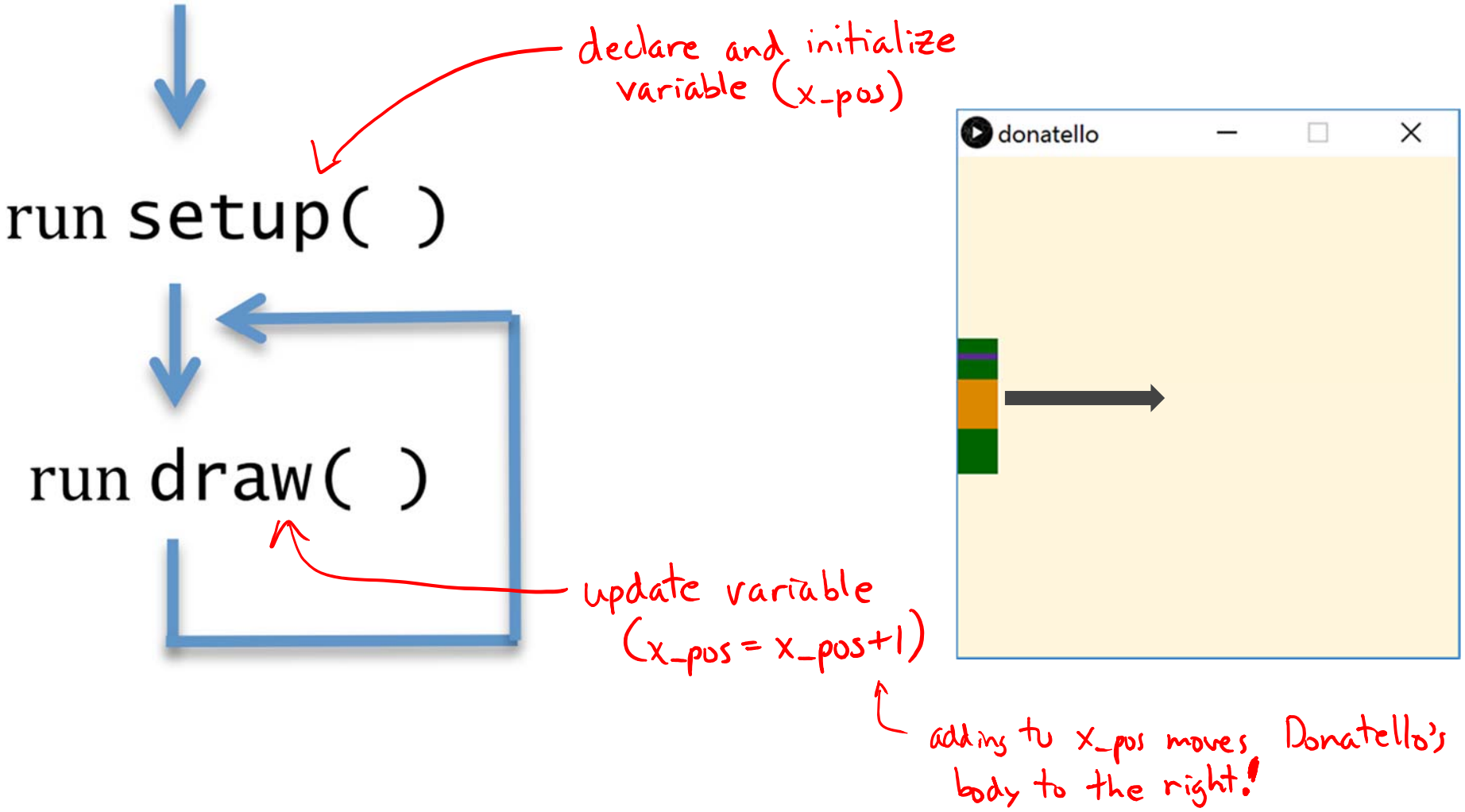
donatello

```
int x_pos = 100;          // x-position

size(500,500);
noStroke();
background(255,245,220);

// Donatello
fill(0,100,0);            // dark green
rect(x_pos,182,40,15);    // top of head

fill(88,44,141);          // purple
rect(x_pos,197,40,6);     // bandana mask

fill(0,100,0);            // dark green
rect(x_pos,203,40,20);    // bottom of head

fill(219,136,0);          // dark yellow
rect(x_pos,223,40,50);    // shell

fill(0,100,0);            // dark green
rect(x_pos,273,40,45);    // lower body
```

x_pos moves entire drawing!



Donatello with Motion



Stopping Motion

- Stop Donatello from running off the right side of the screen:

returns minimum of these two numbers

```
x_pos = min(x_pos + 1, 460);
```

old x-pos	x_pos+1	min(x_pos+1,460)	new x_pos
0	1	1	1
1	2	2	2
2	3	3	3
⋮	⋮	⋮	⋮
458	459	459	459
459	460	460	460
460	461	460	460

← sets maximum x_pos of 460

- Stop Donatello from running off the left side of the screen:

returns maximum of these two numbers

```
x_pos = max(x_pos - 1, 0);
// sets minimum x-pos of 0
```

Falling Into Place

- Introduce variables for each body segment:

```
3  int   head_pos = 0;     // head position
4  float mask_pos = 15;    // mask position
5  int   face_pos = 21;    // face position
6  float body_pos = 41;    // body position
7  int   leg_pos  = 91;    // leg  position
```

initial y-positions for each body segment

- Update each variable at different speeds:

```
33   head_pos = min(head_pos + 3,   364);
34   mask_pos = min(mask_pos + 3.5, 379);
35   face_pos = min(face_pos + 4,   385);
36   body_pos = min(body_pos + 4.5, 405);
37   leg_pos  = min(leg_pos  + 5,   455);
```

← higher segments fall slower

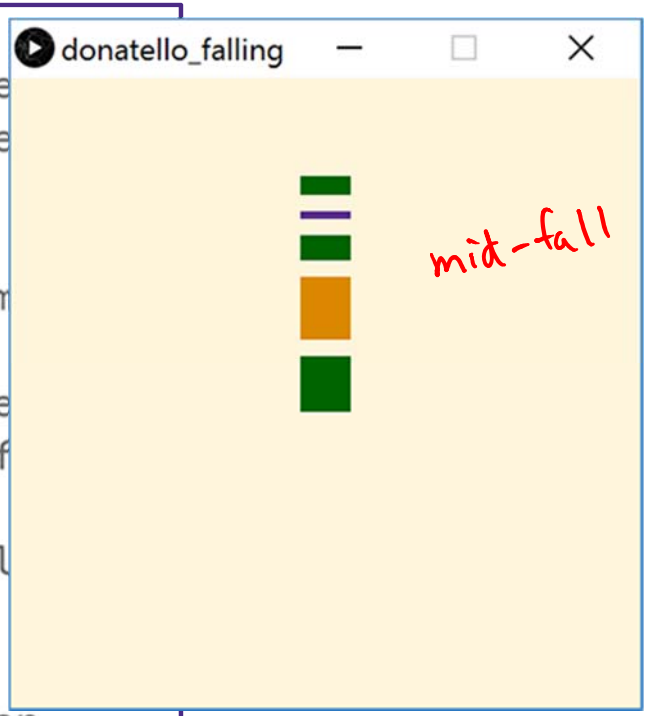
← lower segments fall faster

variables that use decimals need to be declared as float

Falling Into Place

- Update y-positions of drawing based on new variables:

```
17  // Donatello
18  fill(0,100,0);                  // dark gree
19  rect(x_pos,head_pos,40,15);     // top of he
20
21  fill(88,44,141);                // purple
22  rect(x_pos,mask_pos,40,6);      // bandana m
23
24  fill(0,100,0);                  // dark gree
25  rect(x_pos,face_pos,40,20);     // bottom of
26
27  fill(219,136,0);                // dark yell
28  rect(x_pos,body_pos,40,50);     // shell
29
30  fill(0,100,0);                  // dark green
31  rect(x_pos,leg_pos,40,45);      // lower body
```



Summary

- Variables are named quantities that can vary during the execution of a program
- Datatypes specific different forms of data
  - e.g. int, float, color, Boolean
- Variable *declarations* specify a variable datatype and name to the program
  - Generally occurs at top of program
- `min()` and `max()` functions can be used to limit or stop change in a variable value