## Review

- Random numbers
- mouseX, mouseY
- setup() & draw()
- frameRate(), loop(), noLoop()
- Mouse and Keyboard interaction
- Arcs, curves, bézier curves, custom shapes
- Hue-Saturation-Brightness vs. Red-Green-Blue color
- Example Sketches
- OpenProcessing website

## Odds and Ends

- Dropbox installation is a two-step process
  - Sign up for an account with dropbox
  - Install the dropbox application on your computer
- After you have installed dropbox
  - Invitation to join a shared folder named with your email user name
  - This is where all the submissions go!

- Processing programs carry the extension **`.pde`**

- Processing programs must be in a folder with the same name
  - **`myProgram.pde`** must be inside a folder called **`myProgram`**

## Syntax

- Function call
  - **`line( 10, 10, 50, 80 );`**
  - Name
  - The commas
  - The parens ()
  - The semicolon

- Code block
  - The curly braces {}

- Comments
  - //
  - /* and */

## Images

**`loadImage(`*`filename`*`);`**

- Loads an image from a file in the *data* folder in sketch folder.
- Must be assigned to a variable of type PImage.

**`image(`*`img, X, Y, [X2, Y2]`*`);`**

- Draws the image *img* on the canvas at X, Y
- Optionally fits image into box X,Y and X2,Y2

**`imageMode(CORNER);`**

- X2 and Y2 define width and height.

**`imageMode(CORNERS);`**

- X2 and Y2 define opposite corner.

## Image Example

```
imageExample
   ∟ imageExample.pde
   ∟ data
         ∟ natura-morta.jpg
```

```
PImage img;

void setup()
{
  size(500, 400);
  img = loadImage("natura-morta.jpg");
  image(img, 50, 40);
}
```

## Variables

- A <u>name</u> to which data can be assigned
- A variable name is <u>declared</u> as a specific <u>data type</u>
- Names must begin with a letter, "_" or "$" and can container letters, digits, "_" and "$"

```
boolean bReady = true;
int i;
int j = 12;
float fSize = 10.0;
color _red = color(255,0,0);
String name123 = "Fred";
PImage img;
```

## Variable Uses

- Use a value throughout your program,
  - but allow it to be changed
- As temporary storage for a intermediate computed result
- To parameterize – instead of hardcoding coordinates
- Special variables (preset variables)
  - **`width, height`**
  - **`screen.width, screen.height`**
  - **`mouseX, mouseY`**
  - **`pmouseX, pmouseY`**

## Primitive Data Types

| Type | Range | Default | Bytes |
|---|---|---|---|
| boolean | { true, false } | false | ? |
| byte | { 0..255 } | 0 | 1 |
| int | { -2,147,483,648 .. 2,147,483,647 } | 0 | 4 |
| long | { -9,223,372,036,854,775,808 .. 9,223,372,036,854,775,807 } | 0 | 8 |
| float | { -3.40282347E+38 .. 3.40282347E+38 } | 0.0 | 4 |
| double | *much larger/smaller* | 0.0 | 8 |
| color | { #00000000 .. #FFFFFFFF } | *black* | 4 |
| char | *a single character* 'a', 'b', … | '\u0000' | 2 |

## Other "things" …

| Type | Range | Default | Bytes |
|---|---|---|---|
| String | a series of chars in quotes “abc” | null | ? |
| PImage | an image | null | ? |
| PFont | a font for rendering text | null | ? |
| … | | | |

```
String message = "Hello World!";
```

## Data Type Conversion

- Variables of some types can be converted to other types.
- Type conversion function names are the types to which data will be converted

```
// binary(…), boolean(…), byte(…),
// char(…), float(…), str(…)

float f = 10.0;
int i;

//i = f;                // Throws a runtime error
i = int(f);

println( char(65) );    // Prints the character 'A'
```

## Mixing types and Integer Division

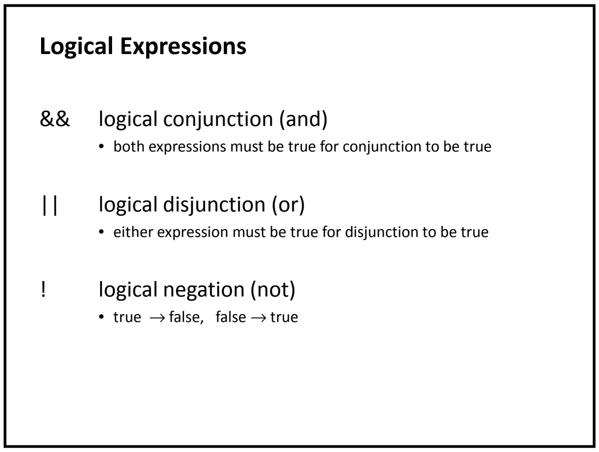
- 3*1.5
  - value?
  - type?
- 3/2
- 2/3
- x/y

## Conditionals: if-statement

Programmatic branching …

```
if ( boolean_expression ) {
    statements;
}

// What does this do?
void draw() {
    if ( mouseX > 50 && mouseY > 50 ) {
        ellipse( mouseX, mouseY, 10, 10 );
    }
}
```

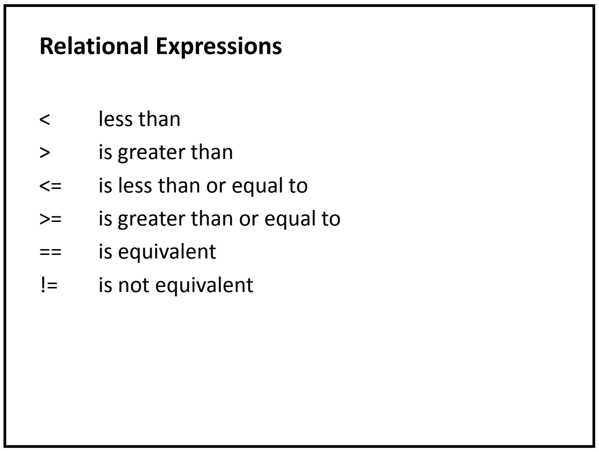


## Logical Expressions

&& logical conjunction (and)
- both expressions must be true for conjunction to be true

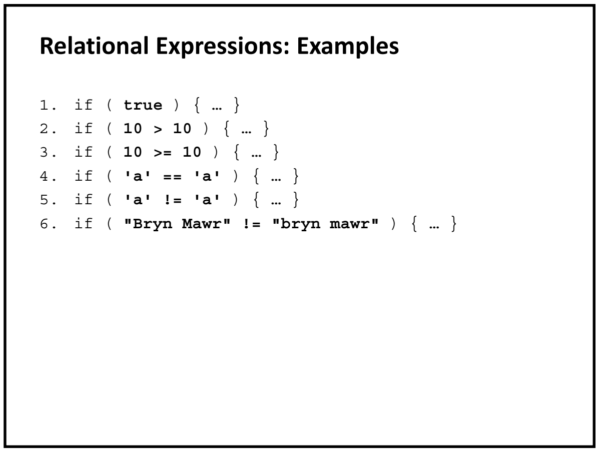
|| logical disjunction (or)
- either expression must be true for disjunction to be true

! logical negation (not)
- true → false, false → true



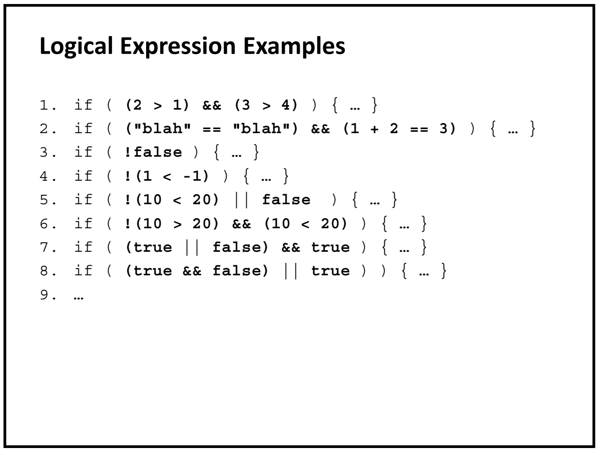
## Relational Expressions

| | |
|---|---|
| < | less than |
| > | is greater than |
| <= | is less than or equal to |
| >= | is greater than or equal to |
| == | is equivalent |
| != | is not equivalent |



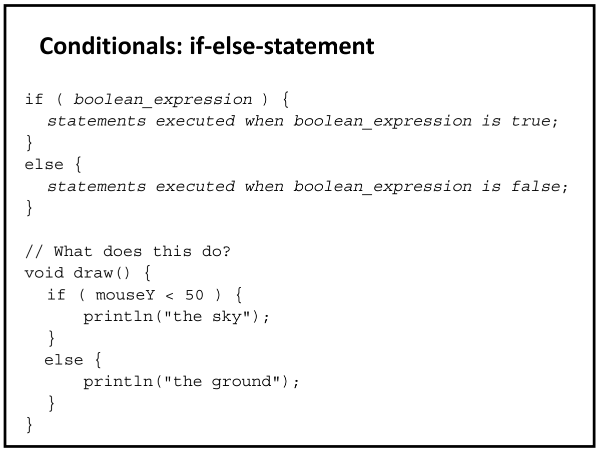
## Relational Expressions: Examples

```
1.  if ( true ) { … }
2.  if ( 10 > 10 ) { … }
3.  if ( 10 >= 10 ) { … }
4.  if ( 'a' == 'a' ) { … }
5.  if ( 'a' != 'a' ) { … }
6.  if ( "Bryn Mawr" != "bryn mawr" ) { … }
```



## Logical Expression Examples

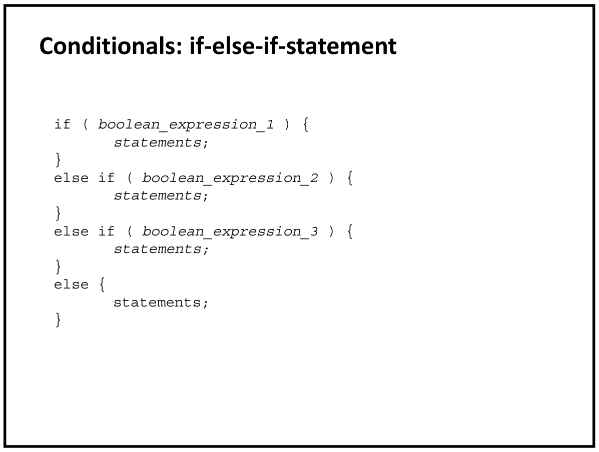
```
1.  if ( (2 > 1) && (3 > 4) ) { … }
2.  if ( ("blah" == "blah") && (1 + 2 == 3) ) { … }
3.  if ( !false ) { … }
4.  if ( !(1 < -1) ) { … }
5.  if ( !(10 < 20) || false  ) { … }
6.  if ( !(10 > 20) && (10 < 20) ) { … }
7.  if ( (true || false) && true ) { … }
8.  if ( (true && false) || true ) ) { … }
9.  …
```



## Conditionals: if-else-statement

```
if ( boolean_expression ) {
  statements executed when boolean_expression is true;
}
else {
  statements executed when boolean_expression is false;
}

// What does this do?
void draw() {
  if ( mouseY < 50 ) {
      println("the sky");
  }
  else {
      println("the ground");
  }
}
```



## Conditionals: if-else-if-statement

```
if ( boolean_expression_1 ) {
        statements;
}
else if ( boolean_expression_2 ) {
        statements;
}
else if ( boolean_expression_3 ) {
        statements;
}
else {
        statements;
}
```

What will this do?

```
void setup() {
  size(500,500);
  smooth();
  ellipseMode(CENTER);
}

void draw() {
  if (mouseX < width/2) {
    stroke(255, 0, 0);
    if (mouseY < height/2) {
      fill(0, 255, 0);
    }
    else {
      fill(0, 0, 255);
    }
  }
  else {
    stroke(0, 0, 255);
    if (mouseY < height/2) {
      fill(255, 0, 0);
    }
    else {
      fill(255);
    }
  }
  ellipse(mouseX, mouseY, 50, 30);
}
```

What does this do?

```
void setup() {
  size( 500, 500 );
}

void draw() {
  if ( mouseX > 100 ) {
    background( 255, 0, 0 );
  }
  else if ( mouseX > 200 ) {
    background( 0, 0, 255 );
  }
}
```

Does this work better?

```
void setup() {
  size( 500, 500 );
}

void draw() {

  if ( mouseX > 200 ) {
    background( 0, 0, 255 );
  }

  if ( mouseX > 100 ) {
    background( 255, 0, 0 );
  }

}
```

## Equations of Motion (Simplified)

r = displacement (position)
t = time
v = velocity
a = acceleration

- Constant acceleration (a)

  $r_{i+1} = r_i + v_i \Delta t$

  $v_{i+1} = v_i + a \Delta t$

- Assume small time intervals – i.e. $\Delta t = 1$