Review

- setup() & draw()
- The event loop
- mouseX, mouseY
- Mouse and Keyboard interactions
- Arcs, curves, bézier curves, custom shapes
- Programming principals
  - Syntax is important
  - Reference manuals are your friend
  - Don’t be afraid to try different things

Text

text(theString, x, y);

- Draws theString on the sketch at (x, y)
- A string is represented by ""
- text("CS110 is fun!", width/2, height/2);

textSize(size);

- Sets the current font size

random(high);
random(low, high);

- Generate a random number in the range
  low (or 0) to high

print(something);
println(something);

- Print something to the Processing console.

Variables

- A location where data is stored
- A variable name is declared as a specific data type
- Names must begin with a letter, _, or $ and can contain letters, digits, _, and $

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

Variable Uses

- Refer to a value throughout your program
  - but allow it to be changed
  - As temporary storage for an intermediate computed result
  - To parameterize – instead of hardcoded coordinates
- Special variables (preset variables)
  - width, height
  - mouseX, mouseY, pmouseX, pmouseY
- Assigned with a single =
  - known as the assignment operator
  - left side and right side are not equal
Primitive Data Types

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

Other "things" ...

<table>
<thead>
<tr>
<th>Type</th>
<th>Range</th>
<th>Default</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>a series of chars in quotes &quot;abc&quot;</td>
<td>null</td>
<td>?</td>
</tr>
<tr>
<td>PImage</td>
<td>an image</td>
<td>null</td>
<td>?</td>
</tr>
<tr>
<td>PFont</td>
<td>a font for rendering text</td>
<td>null</td>
<td>?</td>
</tr>
</tbody>
</table>

String message = "Hello World!";

Data Type Conversion

- Types must match
- If variable types on the two sides of an assignment do not match, one must be converted
  - automatic conversion
  - explicit conversion (casting)

```java
float f = 10.0;
int i = 5;

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

Mixing types and Integer Division

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

Images

```java
save(filename);
loadImage(filename);
- Loads an image from a file in the sketch folder.
- Or in the data subfolder.
- 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 (resize)

imageMode(CORNER);
- X and Y define the upper left corner
- X2 and Y2 define width and height.
```

Image Example

```java
PImage img;
void setup()
{
  size(500, 400);
  img = loadImage("natura-morta.jpg");
  image(img, 50, 40);
}
```
### Conditionals: if-statement

```java
if (boolean_expression) {
    statements;
}
```

What does this do?

```java
void draw() {
    if (mouseX > 50 && mouseY > 50) {
        ellipse( mouseX, mouseY, 10, 10 );
    }
}
```

### Logical Expressions

#### 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

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;</code></td>
<td>less than</td>
</tr>
<tr>
<td><code>&gt;</code></td>
<td>is greater than</td>
</tr>
<tr>
<td><code>&lt;=</code></td>
<td>is less than or equal to</td>
</tr>
<tr>
<td><code>&gt;=</code></td>
<td>is greater than or equal to</td>
</tr>
<tr>
<td><code>==</code></td>
<td>is equal</td>
</tr>
<tr>
<td><code>!=</code></td>
<td>is not equal</td>
</tr>
</tbody>
</table>

### Logical Expression Examples

1. if ((true) && (false)) { ... }
2. if ((false) || false) { ... }
3. if ((false) && (false)) { ... }
4. if ((false) && (false)) { ... }
5. if ((false) && (false)) { ... }

### Relational Expressions: Examples

1. if (true) { ... }
2. if (false) { ... }
3. if (false) { ... }
4. if (false) { ... }
5. if (false) { ... }
6. if (false) { ... }

### Conditionals: if-else-statement

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

What does this do?

```java
void draw() {
    if (mouseY < 50) {
        println("the sky");
    } else {
        println("the ground");
    }
}
```
Conditionals: if-else-if-statement

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

What does this do?

```java
void setup() {
    size(500, 500);
}
void draw() {
    if (mouseX < width/2) {
        fill(0, 255, 0);
    } else if (mouseY < height/2) {
        fill(0, 0, 255);
    } else {
        fill(0, 0, 255);
    }
elipse(mouseX, mouseY, 50, 30);
}
```

And this?

```java
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?

```java
void setup() {
    size(500, 500);
}
void draw() {
    if (mouseX > 200) {
        background(0, 0, 255);
    } else if (mouseX > 100) {
        background(255, 0, 0);
    }
}
```

Simulated Motion (balldrop)

- **p** = position
- **v** = velocity
- **a** = acceleration

- Constant acceleration (a)
  - assuming small time intervals (t=1)
  ```
  \[
  p_{i+1} = p_i + v_i \\
  v_{i+1} = v_i + a 
  \]
  ```