Images

loading and displaying images
image filters
image pixels
image processing

Chapter 7, Examples 7-1, 7-2, 7-3, 7-4, 7-5, 7-6, Robot 5 (p. 117-118)
Today & Next Week

- We will learn how to “use” images
Today & Next Week

- We will learn how to “use” images
  - load & display images
Today & Next Week

- We will learn how to “use” images
  - load & display images
  - apply filters to images
Today & Next Week

- We will learn how to “use” images
  - load & display images
  - apply filters to images
  - create our own filters by manipulating pixels of an image
Today & Next Week

- We will learn how to “use” images
  - load & display images
  - apply filters to images
  - create our own filters by manipulating pixels of image
  - explore some image processing techniques
Today & Next Week

- We will learn how to “use” images
  - load & display images
  - apply filters to images
  - create our own filters by manipulating pixels of image
  - explore some image processing techniques
Today & Next Week

- We will learn how to “use” images
  - load & display images
  - apply filters to images
  - create our own filters by manipulating pixels of image
  - explore some image processing techniques

Next week
Today & Next Week

- We will learn how to “use” images
  - load & display images
But first

- You need to add the images to your project in ...
  - p5 web editor
  - processing IDE
Add Image to the p5 Web Editor

1. **Add Folder** called “data” to store images

2. Add image to that folder using **Add File**, then click on upload area or drag image into it.

https://editor.p5js.org/
Add Image to the Processing IDE

1. Add the image file to the project using Sketch/Add File ... (will automatically store it in a “data” subfolder).
Then...

- Once files have been added
1. Create a global variable to load the image into
   
   ```javascript
   let img;
   ```
Loading and Drawing Images

1. Create a global variable to load the image into
   ```javascript
   let img;
   ```

2. Load the image into the variable
   ```javascript
   img = loadImage("data/bird.jpg");
   ```

   **NOTE:** image must already be added to project
Loading and Drawing Images

1. Create a global variable to load the image into
   let img;

2. Load the image into the variable
   img = loadImage("data/bird.jpg");

| img | undefined |
Loading and Drawing Images

1. Create a global variable to load the image into
   let img;

2. Load the image into the variable
   ```javascript
   img = loadImage("data/bird.jpg");
   ```

<table>
<thead>
<tr>
<th>img</th>
</tr>
</thead>
<tbody>
<tr>
<td>{</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>width: 250,</td>
</tr>
<tr>
<td>height: 250,</td>
</tr>
<tr>
<td>canvas: HTMLCanvasElement,</td>
</tr>
<tr>
<td>drawingContext: ...</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>
Loading and Drawing Images

1. Create a global variable to load the image into
   ```javascript
   let img;
   ```

2. Load the image into the variable
   ```javascript
   img = loadImage("data/bird.jpg");
   ```

3. Draw the image by passing the variable as an argument
   ```javascript
   image(img, 25, 25);
   ```
Loading and Drawing Images

1. Create a global variable to load the image into
   ```javascript
   let img;
   ```

2. Load the image into the variable
   ```javascript
   img = loadImage("data/bird.jpg");
   ```

3. Draw the image by passing the variable as an argument
   ```javascript
   image(img, 25, 25);
   ```
Loading and **Drawing Images**

1. Create a global variable to load the image into
   ```javascript
   let img;
   ```

2. Load the image into the variable
   ```javascript
   img = loadImage("data/bird.jpg");
   ```

3. Draw the image by passing the variable as an argument
   ```javascript
   image(img, 25, 25);
   ```
let img;

function setup() {
    createCanvas(300, 300);
    img = loadImage("data/bird.jpg");
}

function draw() {
    background(220);
    image(img, 25, 25);
}
Runtime Error when Loading Images

```javascript
let img;

function setup() {
  createCanvas(400, 400);
  img = loadImage("data/burd.jpg");
}

function draw() {
  background(220);
  image(img, 0, 0);
}
```

> p5.js says: It looks like there was a problem loading your image. Try checking if the file path [https://editor.p5js.org/sanghosuh/sketches/TXkT1iijd](https://editor.p5js.org/sanghosuh/sketches/TXkT1iijd) is correct, hosting the image online, or running a local server. [https://github.com/processing/p5.js/wiki/Local-server]

```
Event {is Trusted: true, constructor: Object}
```
image(img, 14, 53);

- variable storing the image object
- x and y location to draw image

imageMode(CENTER); // or CORNER, etc..

https://p5js.org/reference/#/p5/image
Drawing Images with Scaling

```javascript
image(img, 14, 53, 300, 400);
```

width and height to draw image

https://p5js.org/reference/#/p5/image
let imgBird;
let imgPerson;

function setup() {
  createCanvas(300, 300);
  imgBird = loadImage("data/bird.jpg");
  imgPerson = loadImage("data/person.png");
}

function draw() {
  background(220);
  image(imgBird, 25, 25);
  image(imgPerson, mouseX, mouseY, 100, 250);
}

Starter: [https://editor.p5js.org/sanghosuh/sketches/IyvunBOW2](https://editor.p5js.org/sanghosuh/sketches/IyvunBOW2)
[https://editor.p5js.org/sanghosuh/sketches/O-EsGFq5a](https://editor.p5js.org/sanghosuh/sketches/O-EsGFq5a)
Getting the Image Size

- You can access an image’s width and height using “dot syntax”:
  
  ```javascript
  print(img.width, img.height);
  ```

- However, sometimes this doesn’t work as expected:
  ```javascript
  imgBird = loadImage("data/bird.png");
  print(imgBird.width); // prints 1, not 250!
  ```

- Because `loadImage` is an **asynchronous** function
  - it returns an empty image before it’s done loading the image

[Link to the code](https://editor.p5js.org/sanghosuh/sketches/x-e8iz3Po)
We expect ...

```javascript
imgBird = loadImage("data/bird.jpg");
print(imgBird.width);
```

<table>
<thead>
<tr>
<th>imgBird</th>
<th>undefined</th>
</tr>
</thead>
</table>

```javascript
{
width: 250,
height: 250,
canvas: HTMLCanvasElement,
drawingContext: ...
.
.
.
}
```
We expect ...

```
imgBird = loadImage("data/bird.jpg");
print(imgBird.width);
```

<table>
<thead>
<tr>
<th>imgBird</th>
</tr>
</thead>
<tbody>
<tr>
<td>{</td>
</tr>
<tr>
<td>width: 250,</td>
</tr>
<tr>
<td>height: 250,</td>
</tr>
<tr>
<td>canvas: HTMLCanvasElement,</td>
</tr>
<tr>
<td>drawingContext: ...</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>
We expect ...

```javascript
imgBird = loadImage("data/bird.jpg");
print(imgBird.width);
```

<table>
<thead>
<tr>
<th>imgBird</th>
</tr>
</thead>
<tbody>
<tr>
<td>{</td>
</tr>
<tr>
<td>width: 250,</td>
</tr>
<tr>
<td>height: 250,</td>
</tr>
<tr>
<td>canvas: HTMLCanvasElement,</td>
</tr>
<tr>
<td>drawingContext: ...</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Console</th>
</tr>
</thead>
<tbody>
<tr>
<td>250</td>
</tr>
</tbody>
</table>
But we see this

```javascript
imgBird = loadImage("data/bird.jpg");
print(imgBird.width);
```

<table>
<thead>
<tr>
<th>imgBird</th>
</tr>
</thead>
<tbody>
<tr>
<td>{</td>
</tr>
<tr>
<td>width: 250,</td>
</tr>
<tr>
<td>height: 250,</td>
</tr>
<tr>
<td>canvas: HTMLCanvasElement,</td>
</tr>
<tr>
<td>drawingContext: ...</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Console</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
</tbody>
</table>
This is because ...

```javascript
const imgBird = loadImage("data/bird.jpg");
print(imgBird.width);
```

Returns temporary placeholder object first and then waits for image to be loaded.
This is because ...

```javascript
imgBird = loadImage("data/bird.jpg");
print(imgBird.width);
```

<table>
<thead>
<tr>
<th>imgBird</th>
</tr>
</thead>
<tbody>
<tr>
<td>{</td>
</tr>
<tr>
<td>width: 1,</td>
</tr>
<tr>
<td>height: 1,</td>
</tr>
<tr>
<td>canvas: HTMLCanvasElement,</td>
</tr>
<tr>
<td>drawingContext: ...</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>
This is because ...

```javascript
let imgBird = loadImage("data/bird.jpg");
print(imgBird.width);
```

<table>
<thead>
<tr>
<th>imgBird</th>
</tr>
</thead>
<tbody>
<tr>
<td>{</td>
</tr>
<tr>
<td>width: 1,</td>
</tr>
<tr>
<td>height: 1,</td>
</tr>
<tr>
<td>canvas: HTMLCanvasElement,</td>
</tr>
<tr>
<td>drawingContext: ...</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>

```

<table>
<thead>
<tr>
<th>Console</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
</tbody>
</table>
```
So ...

- Use `preload()` function
  - it is an event function like `draw()` and `setup()`
  - it is called before `setup()` function
  - it forces `loadImage()` to finish loading the image and return an actual image object, not a temporary object
```javascript
let imgBird;
let imgPerson;

// preload is an event function
// that p5 calls before setup
function preload() {
  imgBird = loadImage("data/bird.jpg");
  imgPerson = loadImage("data/person.png");
}

function setup() {
  createCanvas(300, 300);
  print(imgPerson.width); // prints 93
  ...
}
```

for this course, always load images in `preload`
map

use preload for loading large image

https://editor.p5js.org/cs105/sketches/LViBiYg4-
an example that illustrates why it is useful to be able to know the image’s width and height

https://editor.p5js.org/sanghosuh/sketches/MVzKPqJPC
Calculate the Image Aspect Ratio

- **aspect ratio** is the ratio of width to height
  
  4:3 is 1.333333

  ![4:3 Aspect Ratio Diagram](Diagram)

  4

  3

  16:9 is 1.7777777

  ![16:9 Aspect Ratio Diagram](Diagram)

  16

  9

- You can calculate an image aspect ratio like this:
  
  `let ar = img.width / img.height;`

- Use the aspect ratio to scale images without distorting them
(using aspect ratio to scale image)

1. given height, find the width
2. given width, find the height

https://editor.p5js.org/sanghosuh.sketches/9rTCMGGfF
filters

filter(POSTERIZE, 3);
filter(INVERT);
filter(BLUR, 3);
...

p5 reference for filter:
https://p5js.org/reference/#/p5/filter

https://editor.p5js.org/cs105/sketches/dMZbJ2JBa
You can filter anything drawn in the canvas

<draw using p5 functions first>

let iterations = map(mouseX, 0, width, 0, 10);
filter(BLUR, iterations);

https://editor.p5js.org/cs105/sketches/1wnjwIGKr
Walk Cycle using Array of Images

- Keyframe animation with images
- Walk cycle explanation (requires Flash)

Credit: [http://www.angryanimator.com/](http://www.angryanimator.com/)
store the walkcycle image sequence
let walkcycle = [];
...

load all the walk cycle images
walkcycle[0] = loadImage("data/1.png");
walkcycle[1] = loadImage("data/2.png");
...

draw the image in one element of the array
image(walkcycle[frameToShow], x, 0);

Starter: https://editor.p5js.org/sanghosuh/sketches/euy7GWEKw
https://editor.p5js.org/cs105/sketches/j5Tmti_O4