

Organize image information using list of lists

While loops

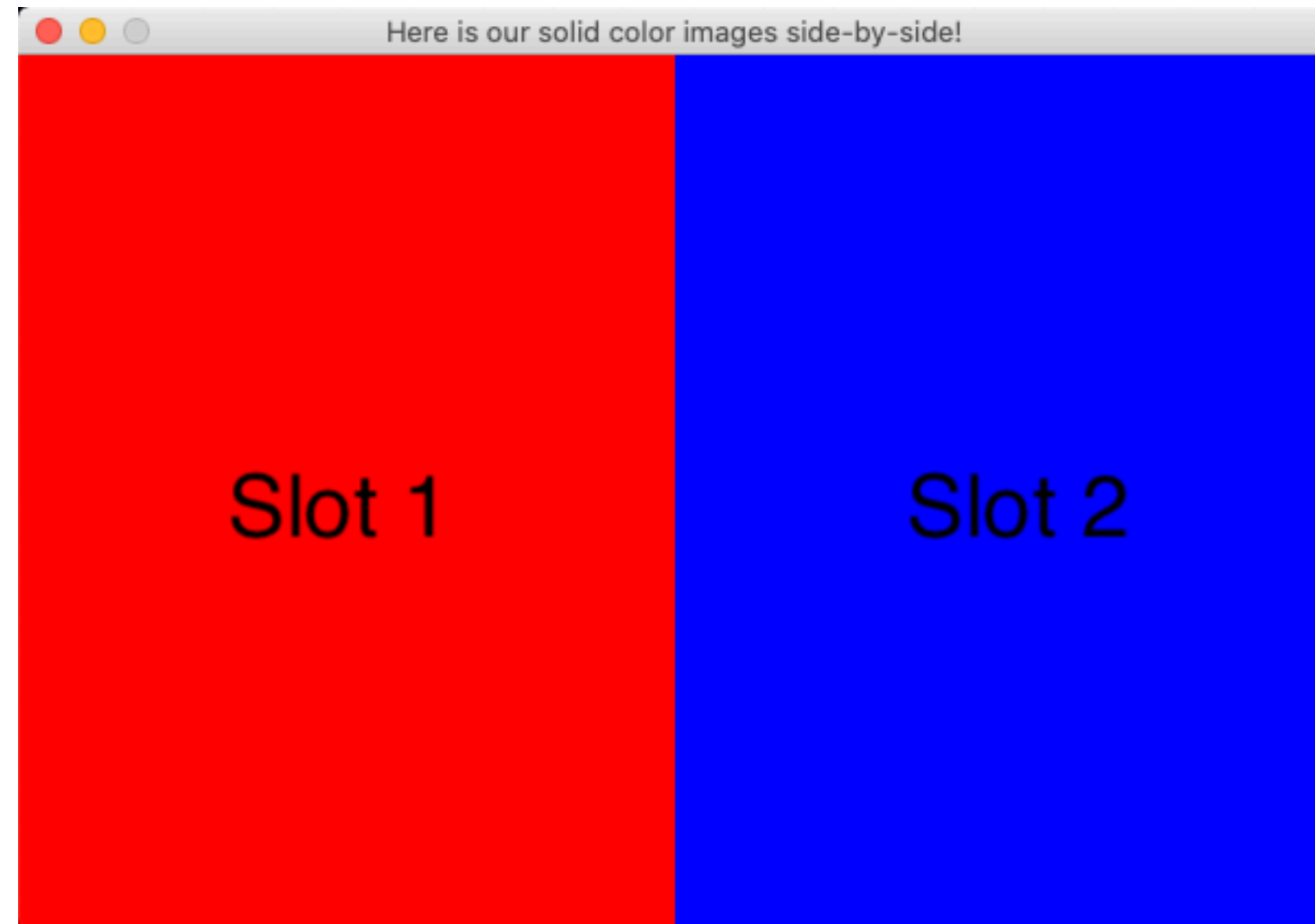
Oliver W. Layton

CS151: Computational Thinking: Visual Media

Lecture 20, Spring 2021

Monday March 29

Review of canvas layout



Two Image objects. Each has solid color. Placed in Slot 1 or Slot 2 in larger Image canvas.

Goal

Organize the Image objects and their placement information to easily control:

- where each of the two images go (*left or right*)
- what color to make them (e.g. red, blue, yellow, etc.)
- Can use **list of lists** to store information about each image in the canvas. This information includes:
 - the x-coordinate of the top-left corner in the large canvas
 - the y-coordinate of the top-left corner in the large canvas
 - the solid color of the image.
 - the Image object

Example: placing two images on the canvas

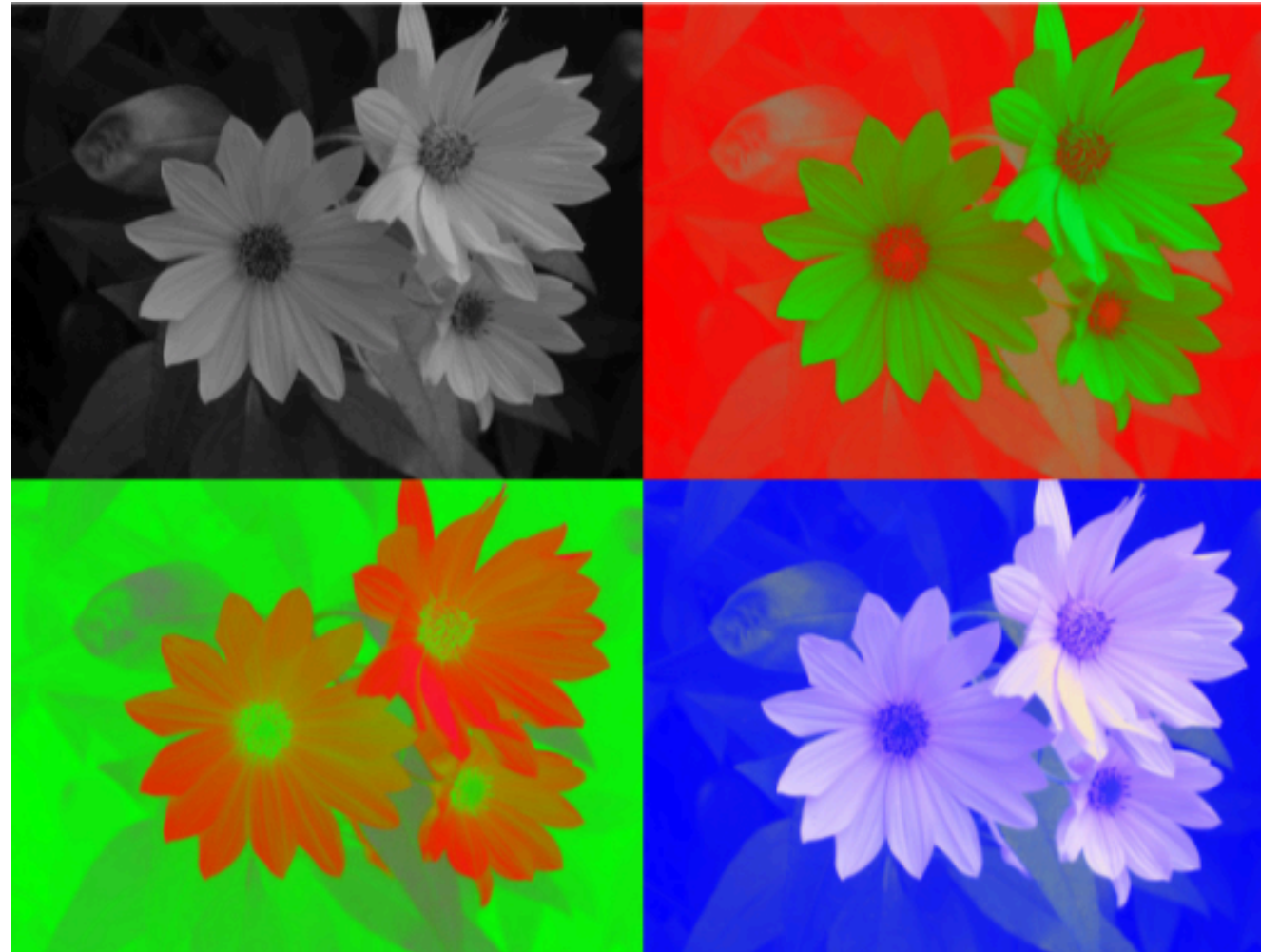
Use one sublist per image in the canvas (i.e. two sublists total).

```
imageList = [[0, 0, 'red', img], # Slot 1  
             [<x offset>, <y offset>, 'blue', img] # Slot 2  
            ]
```

with actual values:

```
imageList = [[0, 0, 'red', img], # Slot 1  
             [img.getWidth(), 0, 'blue', img] # Slot 2  
            ]
```

Where we're going with this



In Project 6: The Warhol Project you will extend the same idea to make a 2x2 grid of images in the style of the artist Andy Warhol (*the original image + 3 filtered versions of it*).

Coloring and placing images on the larger canvas

Each sublist contains **all** the info needed to color and place an image in one of the canvas slots.

```
imageList = [[0, 0, 'red', img],  
             [img.getWidth(), 0, 'blue', img]]
```

- Loop thru list, deal with one sublist at a time.
- Clone the image object (*blank image initially*).
- "Paint" img the appropriate solid color, depending on the name in the sublist (*3rd item*).
- Place it on the canvas.
- Display the canvas.

Let's write code to place a pair of solid color
images side-by-side using a list of lists
(lecture_20_sidebysidelist.py)

Differences in Project 6

```
imageList = [[0, 0, 'red', img],  
             [img.getWidth(), 0, 'blue', img]]
```

1. `img` should be an original PPM image loaded in from your project folder (e.g. `millер.ppm`), not a blank one.
2. 3rd item in each sublist is a string to indicate which filter to apply (e.g. `swapBlueGreen`).
3. You will have more sublists (Warhol is a 2x2 grid, 4 images total).

While loops

For loops are not the only kind of loops

For loops execute a sequence of computations for a **fixed number of iterations**. Not always what we want to do!

- If you're hammering a nail, you can't know for sure how many times to hit it to get it into a block of wood... You hammer the nail **while it's sticking out**.
- A **while loop** executes code *while a condition is true*.

What is a while loop?

```
while <condition is True>:  
    <body>
```

- In Python, the **while loop** runs commands in the body over-and-over-again until the condition evaluates to `False`.
- The `<condition is True>` is called the **loop sentinel**. It tells us whether we continue looping.

Example use: Entering car trip data by hand

Loop sentinel: Entering a negative number (invalid data)

Summary: Increment and decrement operators

- There is an **increment operator**: `+=`

The line `i += 10` is the same as `i = i + 10`.

- Translation: *Assign to `i` its current value, plus 10*

- There's also the **decrement operator** `-=`

The line `i -= 10` is the same as `i = i - 10`

- Translation: *Assign to `i` its current value, minus 10*

Can we make a while loop that mimics a for loop?

Convert this into a while loop:

```
for i in range(10):  
    print(i)
```

while loops can be dangerous!

What does the following loop do?

```
i = 0
while i < 99:
    print(i)
```

- We forgot to increment `i`!
- It would run on forever, which leads to a dreaded **infinite loop**. This means you'd have to force-quit out of your Python program...it would otherwise only stop running if you shut off your computer!
- For loops are sometimes called **definite loops**. By contrast, while loops are sometimes call **indefinite loops**.