Image Manipulation

Reviewing Pixels

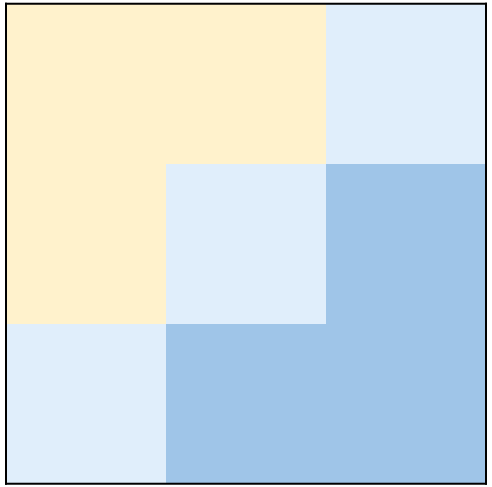
Last class, we discussed how we can use pixels to store all of the information about a digital image.

R: 255 G: 242 B: 204	R: 255 G: 242 B: 204	R: 224 G: 238 B: 251
R: 255 G: 242 B: 204	R: 224 G: 238 B: 251	R: 159 G: 197 B: 232
R: 224 G: 238 B: 251	R: 159 G: 197 B: 232	R: 159 G: 197 B: 232

Reviewing Pixels

Last class, we discussed how we can use pixels to store all of the information about a digital image.

R: 255 G: 242 B: 204	R: 255 G: 242 B: 204	R: 224 G: 238 B: 251
R: 255 G: 242 B: 204	R: 224 G: 238 B: 251	R: 159 G: 197 B: 232
R: 224 G: 238 B: 251	R: 159 G: 197 B: 232	R: 159 G: 197 B: 232

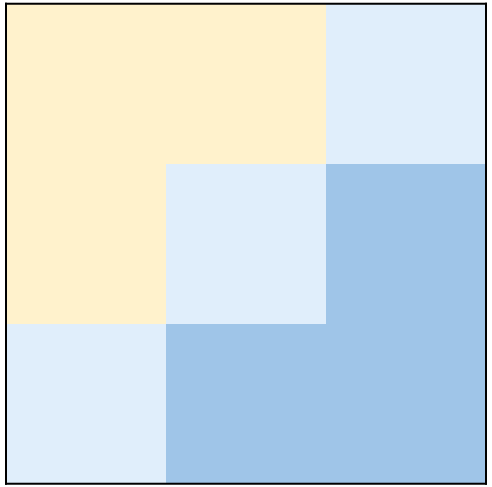


Reviewing Pixels

Last class, we discussed how we can use pixels to store all of the information about a digital image.

R: 255 G: 242 B: 204	R: 255 G: 242 B: 204	R: 224 G: 238 B: 251
R: 255 G: 242 B: 204	R: 224 G: 238 B: 251	R: 159 G: 197 B: 232
R: 224 G: 238 B: 251	R: 159 G: 197 B: 232	R: 159 G: 197 B: 232

We can manipulate this information to modify the image!



Modifying 1 Pixel

(120, 20, 220)

BRIGHTNESS FILTER

Modifying 1 Pixel

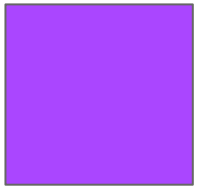
Modifying 1 Pixel


```
R = R + 50
G = G + 50
B = B + 50
```

Modifying 1 Pixel

(120, 20, 220)

BRIGHTNESS FILTER



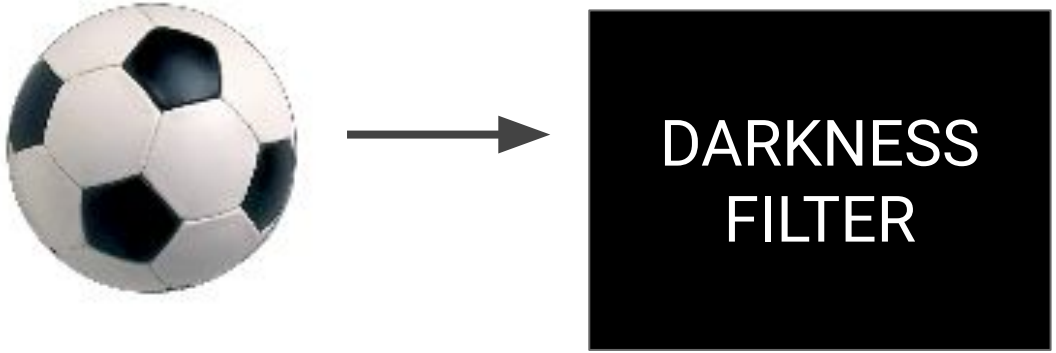
(170, 70, 255)

```
R = min(R + 50, 255)
G = min(G + 50, 255)
B = min(B + 50, 255)
```

Modifying an entire image

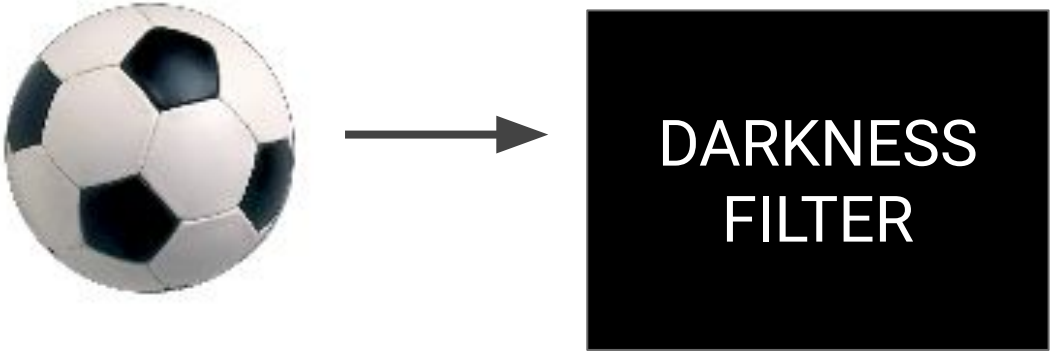

```
For each pixel:
    R = min(R + 50, 255)
    G = min(G + 50, 255)
    B = min(B + 50, 255)
```

Modifying an entire image




```
For each pixel:
    R =
    G =
    B =
```

Modifying an entire image




```
For each pixel:
    R = R - 50
    G = G - 50
    B = B - 50
```

Modifying an entire image

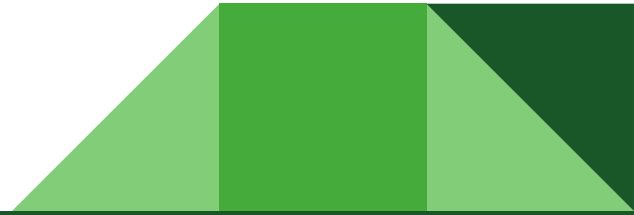

```
For each pixel:
    R = max(R - 50, 0)
    G = max(G - 50, 0)
    B = max(B - 50, 0)
```

Writing the code to do that

In order to manipulate an image in our programs, we first need to actually get an image into our program!

We can do this using the `Image()` function, built into the graphics system we've been working with.

We need to provide this function with the URL of the image we want, and give it a variable to save the image into!

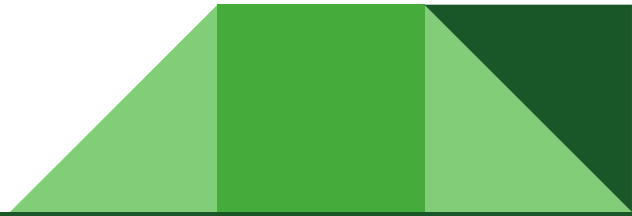


Example Image Import Code

```
IMAGE_URL = "https://codehs.com/uploads/e07cd01271cac589cc9ef1bf012c6a0c "

my_image = Image(IMAGE_URL)

add(my_image)
```



Example Image Import Code

```
IMAGE_URL = "https://codehs.com/uploads/e07cd01271cac589cc9ef1bf012c6a0c "

my_image = Image(IMAGE_URL)

my_image.set_size(280, 200)

my_image.set_position(70, 70)

add(my_image)
```

Example Image Import Code

```
IMAGE_URL = "https://codehs.com/uploads/e07cd01271cac589cc9ef1bf012c6a0c "

my_image = Image(IMAGE_URL)

my_image.set_size(280, 200)

my_image.set_position(70, 70)

add(my_image)
```

Width, Height

x, y coordinate of **top left**

Getting Pixel Data

Once we have a variable to represent our image, we can access information about any given pixel using the `get_pixel()` function.

This function needs to know which x, y coordinate within the image we want to access, and it will return a list containing the R, G, and B values for that pixel.

```
pix = my_image.get_pixel(x, y)
print(pix[0]) # Prints the Red value
print(pix[1]) # Prints the Green value
print(pix[2]) # Prints the Blue value
```

Getting Pixel Data

Our pixels are going to follow the same coordinate convention as normal coordinates in our graphics system.

```
(0, 0)
```

```
(my_image.get_width(), 0)
```

```
(0, my_image.get_height())
```

```
(my_image.get_width(),
 my_image.get_height())
```

Changing Pixel Data

In order to actually change the color values for a pixel, we need to call the `set_red()`, `set_green()`, and/or `set_blue()` functions!

Each of these functions needs to know the x, y coordinate of the pixel being updated, as well as the new value to give to that color channel.

```
my_image.set_red(x, y, new_value)
my_image.set_green(x, y, new_value)
my_image.set_blue(x, y, new_value)
```

General Image Filter Pseudocode

```
def filter(pixel):
    Modify the R, G, and B values of the provided pixel
      according to some function
    Return the updated R, G, and B values

Loop over every (x, y) coordinate in image
    Get the current pixel
    Get new color by calling filter() on pixel
    Update image at (x, y) with new color
```