CS105 Problem Solving

2 Dimensional Image Manipulation

Wholeness

- Using nested loops we can do 2 dimensional image manipulation, in essence an image is just a 2 dimensional array of pixels.

Two Dimensional Access

Manipulating Pixels Based on Where They Are

Window.getImageData()

- ImageData is An object that represents the pixels, and knows about their locations
  - The data type should be object

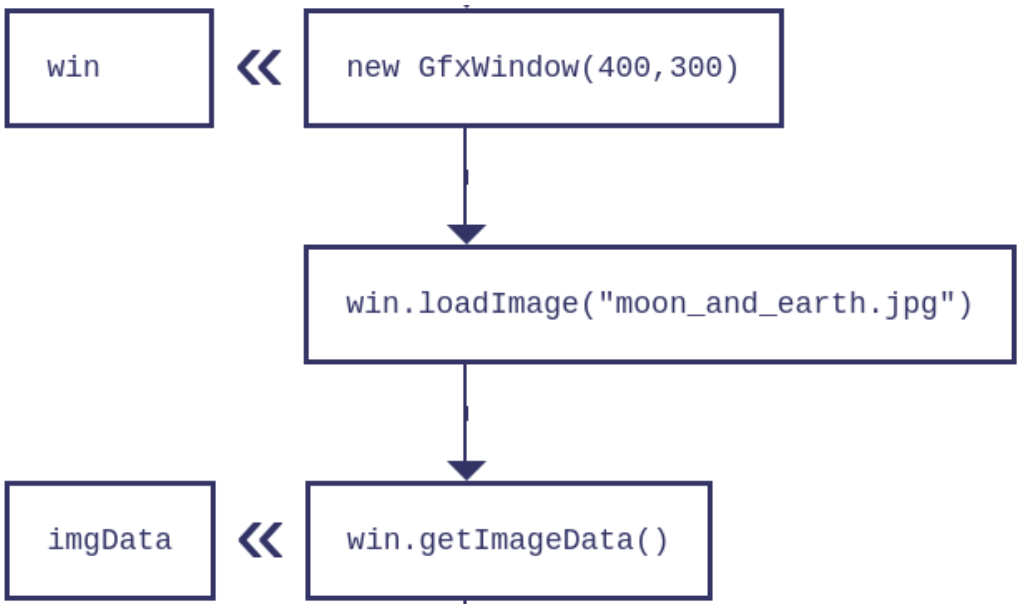
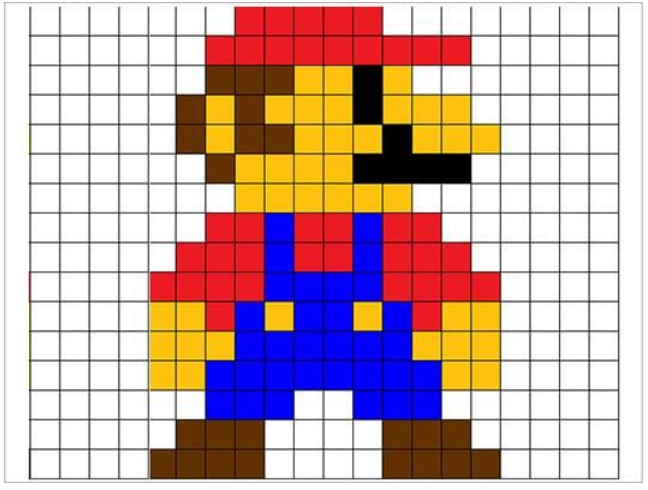


Image Data API

- ImageData objects have the following methods / data:
  - ImageData.getRedAt(x, y)
  - ImageData.getGreenAt(x, y)
  - ImageData.getBlueAt(x, y)
  - ImageData.getAlphaAt(x, y)
  - ImageData.setRedAt(x, y, val)
  - ImageData.setGreenAt(x, y, val)
  - ImageData.setBlueAt(x, y, val)
  - ImageData.setAlphaAt(x, y, val)
  - ImageData.width
  - ImageData.height

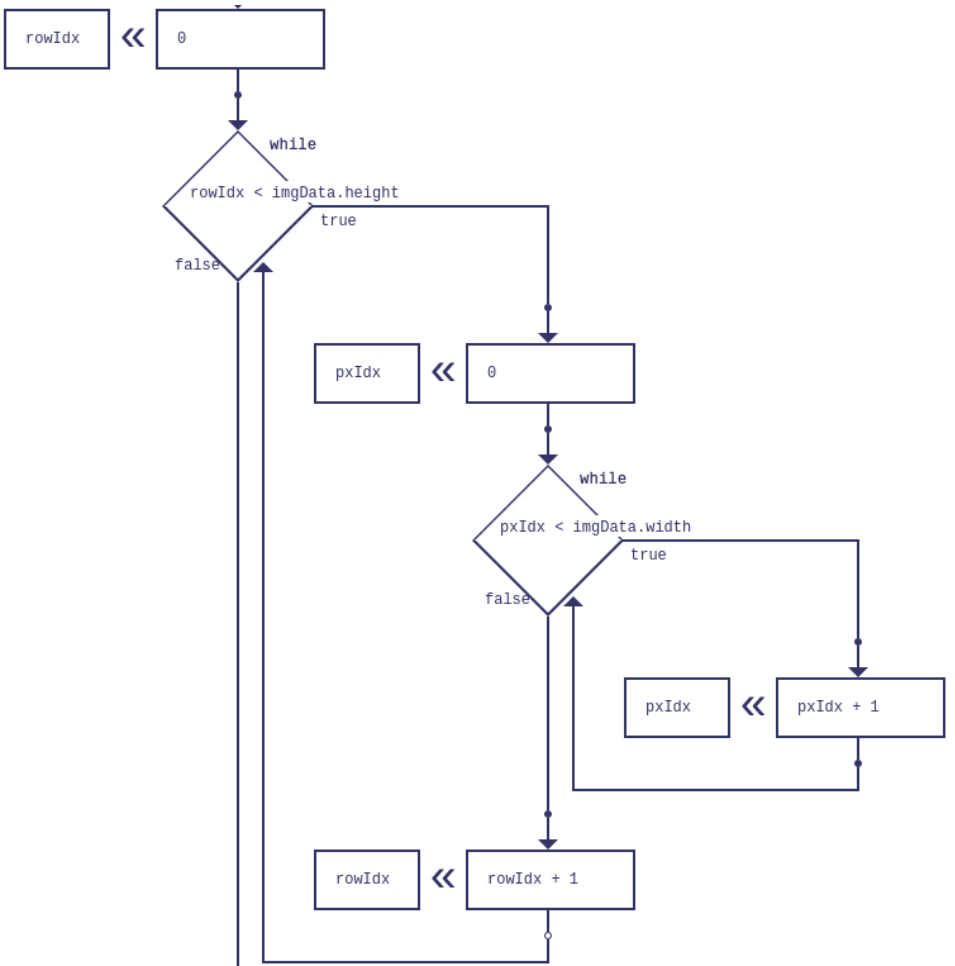
Images



	column 1	column 2	column 3	column 4	column 5
row1	arr[0][0]	arr[0][1]	arr[0][2]	arr[0][3]	arr[0][4]
row2	arr[1][0]	arr[1][1]	arr[1][2]	arr[1][3]	arr[1][4]
row3	arr[2][0]	arr[2][1]	arr[2][2]	arr[2][3]	arr[2][4]

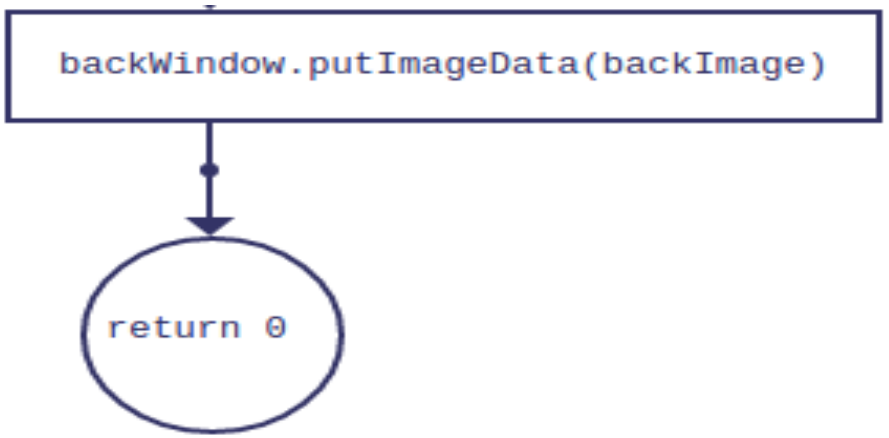
Nested Loop

- To get both x and y we need a loop in a loop.
- Where the outer loop selects a row of pixels (the y value)
- And the inner loop selects the pixel in that row (the x value)
- You don’t have to select all the pixels if don’t need!!!



Window.putImageData()

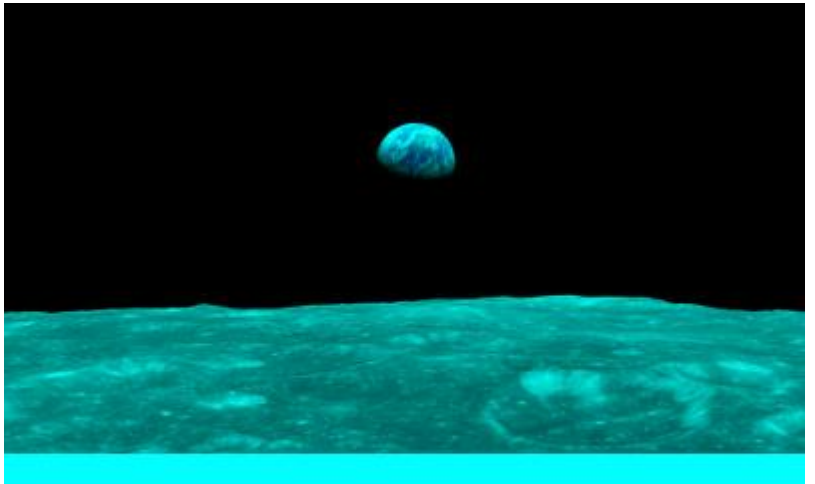
- Once you've changed imageData you can put it back into the window so as to display it
  - window.putImageData(imageData)



Removing Red

```
function main() {
    var win; // object
    var imgData; // object
    var rowIdx; // number
    var pxIdx; // number
    win = new GfxWindow(400,300);
    win.loadImage("moon_and_earth.jpg");
    imgData = win.getImageData();
    rowIdx = 0;
    while (rowIdx < imgData.height) {
        pxIdx = 0;
        while (pxIdx < imgData.width) {
            imgData.setRedAt(pxIdx, rowIdx, 0);
            pxIdx = pxIdx + 1;
        }
        rowIdx = rowIdx + 1;
    }
    win.putImageData(imgData);
    return 0;
}

main(); // start executing main
```



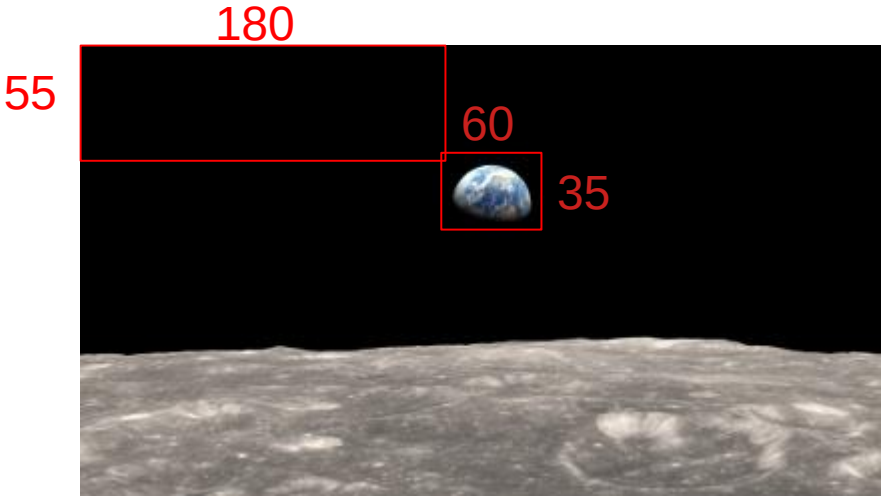
Reason to use X / Y

- The previous example could have been done with getPixels()
  - Why would you use getImageData() ?
  - Nested loop is more work / code (more difficult)

- It allows us to manipulate a part of an image
  - Specify an exact square where we want to make changes

“Cooling” the Earth

- To only remove red from the Earth
  - Find where it is in the image
  - X starts at 180 goes to 230
  - Y starts at 55 goes to 90



Code & Result

```
function main() {
    var win; // object
    var imgData; // object
    var rowIdx; // number
    var pxIdx; // number
    win = new GfxWindow(400,300);
    win.loadImage("moon_and_earth.jpg");
    imgData = win.getImageData();
    rowIdx = 55;
    while (rowIdx < 90) {
        pxIdx = 180;
        while (pxIdx < 230) {
            imgData.setRedAt(pxIdx, rowIdx, 0);
            pxIdx = pxIdx + 1;
        }
        rowIdx = rowIdx + 1;
    }
    win.putImageData(imgData);
    return 0;
}

main(); // start executing main
```

Exercise

- Image Touch-Up
  - The right border of saucer.jpg is not properly green
  - Write a program that loads the image and sets all pixels whose X value is greater than 300 to 0 red, 0 blue, 255 green
  - You can simply take all Y values, from 0 to 180 (or imgData.height)

Main Point

- Manipulating based on x, y locations give us greater power and flexibility
- Deeper levels of reality have greater power and flexibility

Green Screen

Taking non-green pixels Or replacing green pixels

Green Screen

- An image with an all green background
  - Allows us to replace the green pixels
  - Allows us to copy non-green pixels
- To qualify as “green” a pixel must have:
  - Red < 150
  - Green > 200
  - Blue < 150

Result of Replacing Green Pixels

Fun, but the minions aren't
in the location we want

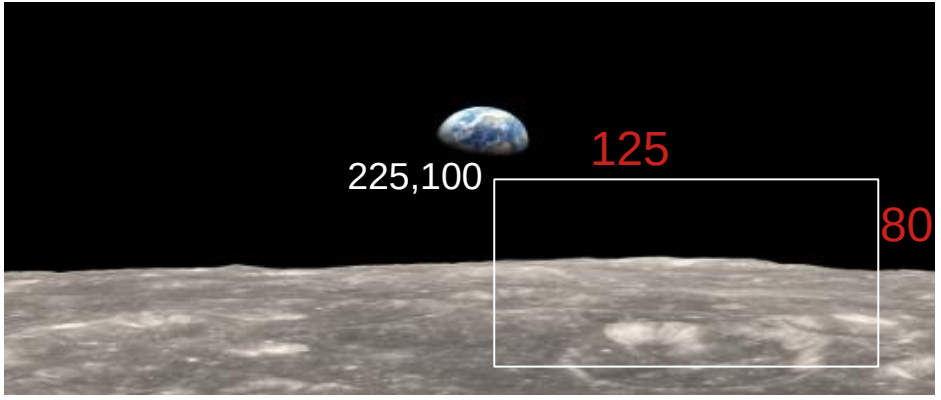
Code

```
function main() {
    var bg_win; // object
    var fg_win; // object
    var bg_pixels; // array
    var fg_pixels; // array
    var n; // number
    bg_win = new GfxWindow();
    bg_win.loadImage('moon_and_earth.jpg');
    bg_pixels = bg_win.getPixels();
    fg_win = new GfxWindow();
    fg_win.loadImage('minions.jpg');
    fg_pixels = fg_win.getPixels();
    n = 0;
    while (n < fg_pixels.length) {
        if (fg_pixels[n].getRed() < 150 && fg_pixels[n].getGreen() > 200 && fg_pixels[n].getBlue() < 150) {
            fg_pixels[n].setRed(bg_pixels[n].getRed());
            fg_pixels[n].setGreen(bg_pixels[n].getGreen());
            fg_pixels[n].setBlue(bg_pixels[n].getBlue());
        } else {
        }
        n = n + 1;
    }
    fg_pixels.show();
    return 0;
}
```

To replace pixels you do not need 2D access. Just check for correct green and replace with pixels from another image!

To a Different Location

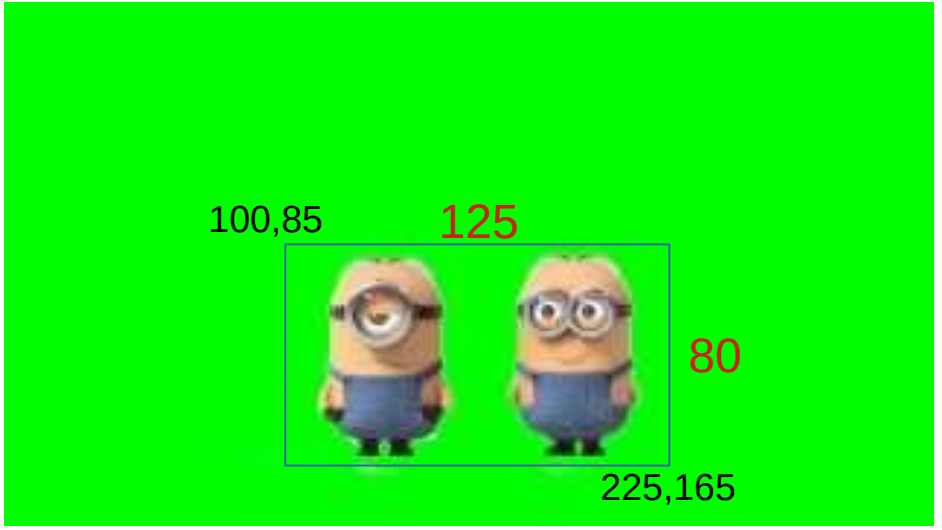
Destination



Source and destination areas need to have the same width and height (125x80).

But can be at different locations inside their image

Source



From top left of source (100,85) to top left of destination (225,100)

The offset is: +125 X and +15 Y

Copying an Area Without Offset

```
function main() {
    var win1; // object
    var win2; // object
    var bg; // object
    var fg; // object
    var x; // number
    var y; // number
    win1 = new GfxWindow();
    win2 = new GfxWindow();
    win1.loadImage("moon_and_earth.jpg");
    win2.loadImage("minions.jpg");
    bg = win1.getImageData();
    fg = win2.getImageData();
    y = 85;
    while (y < 165) {
        x = 100;
        while (x < 225) {
            if (!(fg.getRedAt(x,y) < 150 && fg.getGreenAt(x,y) > 200 && fg.getBlueAt(x,y) < 150)) {
                bg.setRedAt(x, y, fg.getRedAt(x,y));
                bg.setGreenAt(x, y, fg.getGreenAt(x,y));
                bg.setBlueAt(x, y, fg.getBlueAt(x,y));
            } else {
            }
            x = x + 1;
        }
        y = y + 1;
    }
    win1.putImageData(bg);
    return 0;
}
```

Less pixels are copied, but minions are in the same place as before

With Offset

```
function main() {
    var win1; // object
    var win2; // object
    var bg; // object
    var fg; // object
    var x; // number
    var y; // number
    win1 = new GfxWindow();
    win2 = new GfxWindow();
    win1.loadImage("moon_and_earth.jpg");
    win2.loadImage("minions.jpg");
    bg = win1.getImageData();
    fg = win2.getImageData();
    y = 85;
    while (y < 165) {
        x = 100;
        while (x < 225) {
            if (!(fg.getRedAt(x,y) < 150 && fg.getGreenAt(x,y) > 200 && fg.getBlueAt(x,y) < 150)) {
                bg.setRedAt(x + 125, y + 15, fg.getRedAt(x,y));
                bg.setGreenAt(x + 125, y + 15, fg.getGreenAt(x,y));
                bg.setBlueAt(x + 125, y + 15, fg.getBlueAt(x,y));
            } else {
            }
            x = x + 1;
        }
        y = y + 1;
    }
    win1.putImageData(bg);
    return 0;
}
```

Main Point

- Using green screen we can select (green) pixels that should be replaced or select (not green) pixels that should be copied.

Summary

- Using 2 Dimensional Access we can update a part of an image
- Using green screen we can selectively copy pixels
- We can copy pixels to a different location with an offset