The Plan

- Arrays of Colors
- Random Selection of Colors
- Custom Colors
- Random Generation of Colors

Coloring 4 Cars; more ...

- Create Array of Colors to Use
  
  ```java
  Color[] colors = {Color.GREEN, Color.BLUE, ... };
  ```

- When using loop to create an array of sprites, assign color using same array index
  
  ```java
  car[k].setColor(colors[k]);
  ```

- What if we have more cars than colors in our array?
  
  ```java
  car[k].setColor(colors[k % colors.length]);
  ```

  - Wraps around and re-uses the colors

Random Assignment of Colors

- Can select from our array of colors, or palette, at random
  
  ```java
  Math.random() gives us a random value between 0 and (but not including) 1.0.
  ```

- `(int) (Math.random() * n)` gives us a random integer in the range 0 thru n-1

- Use
  
  ```java
  car[k].setColor(
    colors[(int) (Math.random() * colors.length)];
  ```
Custom Colors

- You do not have to limit yourself to the standard pre-defined Java colors.
- Use `new Color(r, g, b)` to generate any colors where `r`, `g`, and `b` are each ints in the range 0-255
- They specify the mixture of red, green, and blue that you want.
  - `new Color(255, 0, 0)` give us pure red
  - `new Color(0, 255, 0)` give us pure green
  - `new Color(0, 0, 255)` give us pure blue
- Mix and stir at will

Random Colors

- We can use our `Math.random` function to help us generate random colors so that each time you run your program, the colors are different
- `(int)(Math.random()*255)` gives us a random int in the required range.
- Can then write
  ```java
car[k].setColor(new Color((int)(Math.random()*255),
               (int)(Math.random()*255),
               (int)(Math.random()*255))
  ```

Transparency

- The Color class allows you to specify a transparency to the color as well as the three color components.
- Feel free to try this out and to experiment.
- Use your Java API to look at what is available (and alternate ways of specifying colors).