The Draw Function

To animate in JavaScript, we use something called the Draw function. Essentially, it is a function that runs all the code within it multiple times every second. It looks like this:

**draw** = **function**() { };

Inside the curly brackets goes all the code you want to run every second (like a regular function). By default, it runs at 60 FPS (i.e. the function runs sixty times every second). You can change this using the frameRate function, like this:

**frameRate**(30);

This changes the draw function to run at 30 FPS.

Note: If you are using the online p5 editor, you can use their pre-made draw function function draw() { } for animation. Also, to use the frameRate function, you must put it inside the pre-made function setup() { } function.

How to Animate

Beyond the draw function, animating is simply logic using the concepts you’ve learned so far. For example:

* Use variables to slowly change shapes’ positions and sizes over time by increasing or decreasing the value of the variable and then using the variable as a parameter for various shapes
* Use functions with parameters to draw custom shapes while still being able to use variables as parameters to change the size and position
* Use if statements to set boundaries for shapes and have certain things happen while conditions are true

Randomization

Randomization creates a nice element of surprise with animations and allows each time the program is run to be different. To generate a random value, the following command is used:

**random**(**low\_value**, **high\_value**);

This generates a random value between the low and high values entered. Note that the lowest value is included, but the highest value is excluded (i.e. if the range is from 1 to 10, it is possible to get 1, but *not* possible to get 10).

Practice Exercises

1. Have a square move 3 pixels right and 2 pixels down every frame. Have the square change direction and color every time it hits the edge (i.e. if it hits the right edge, have it move 3 pixels left every frame). Once you finish this, try making the square move at a random speed whenever it changes direction.
2. Have a ball bounce across the screen, changing direction every time it hits the edge of the canvas (similar to the last exercise), but this time with realistic gravity as it bounces up and down (i.e. have it accelerate downwards over time).
3. For the exercise above, have 3 balls be created when the mouse is clicked (one ball per click), and have their colours be randomized every frame.
4. Have a circle move around a single point by constantly modifying its x and y position with speed variables and constantly modifying its x and y speed with acceleration variables. Once you get this working, try having the ball circle any point the user clicks on by resetting the program with a mouseClicked function.