Translate

6 min

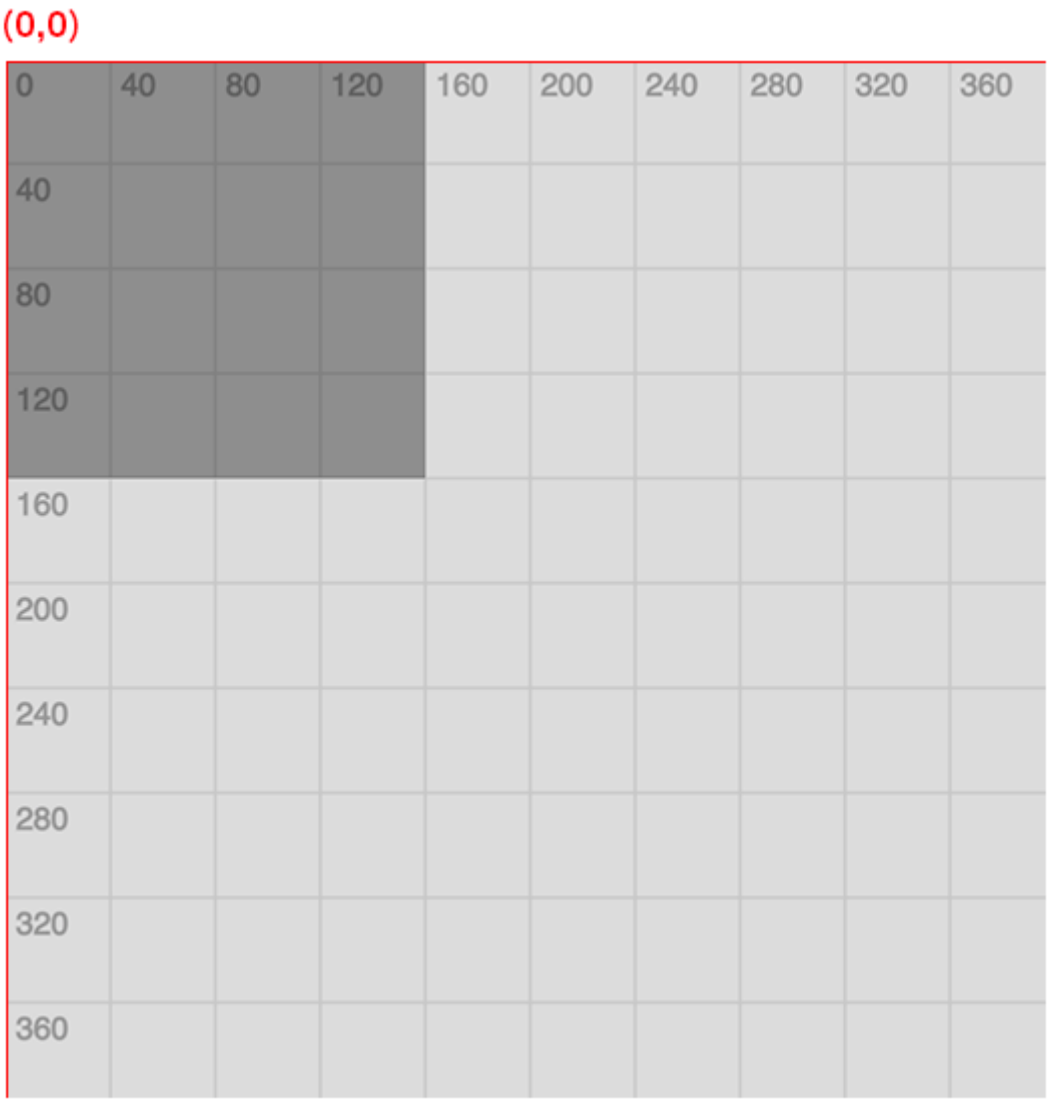
There are two ways to move a shape across your canvas. Let's say you wanted to move your rectangle 60 pixels right and down, you can add 60 pixels to the x and y coordinates given to the `rect()` function. For example, `rect(0, 0, 100, 100)` would change to `rect(60, 60, 100, 100)`. This is how we've been drawing and positioning shapes so far.

Another method is to move the p5.js coordinate system itself instead of moving the shape. The `translate()` function does exactly this—it changes the (0, 0) origin of the p5.js coordinate system to be the location specified as the function's arguments.

For example, `translate(60, 60)` will move the coordinate system 60 pixels to the right and 60 pixels to the bottom. The first argument in the parentheses represents the amount of pixels to move along the x-axis. The second argument represents the amount of pixels to move along the y-axis.

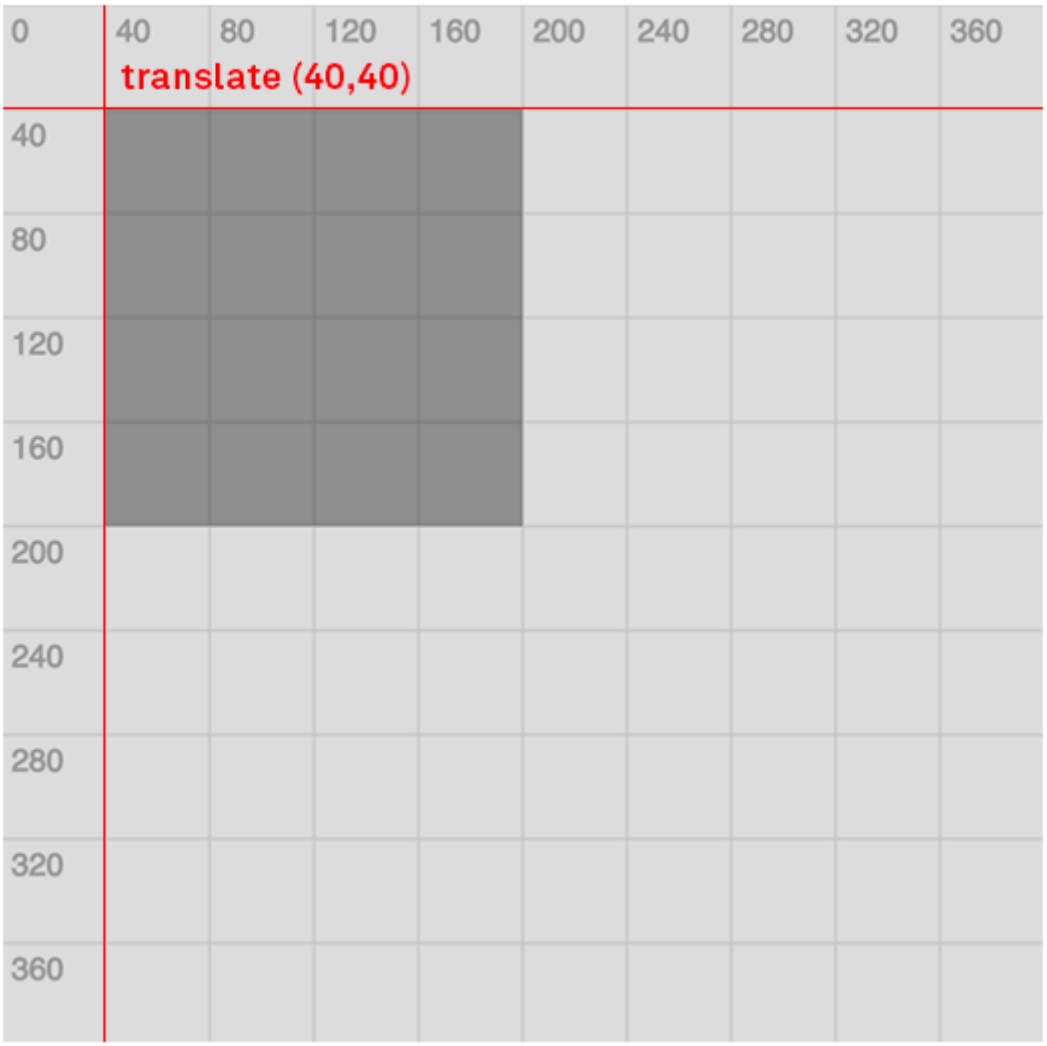
Both ways would change the position of your shape, but the main difference is that instead of moving the shape itself, the `translate()` function moves the entire sketch's coordinate system to the new position specified within the parentheses.

The diagram below shows the rendering of `rect(0, 0, 140, 140)`. As you can see, the shape's origin point is at (0, 0).



Notice what happens to the rectangle when you apply `translate(40, 40)` to it. The rectangle moves `40` pixels right and down without having to change the x and y arguments for the `rect()` function—the first and second arguments of the `rect()` function remain at (0, 0).

```
translate(40,40);
rect(0,0,140,140);
```

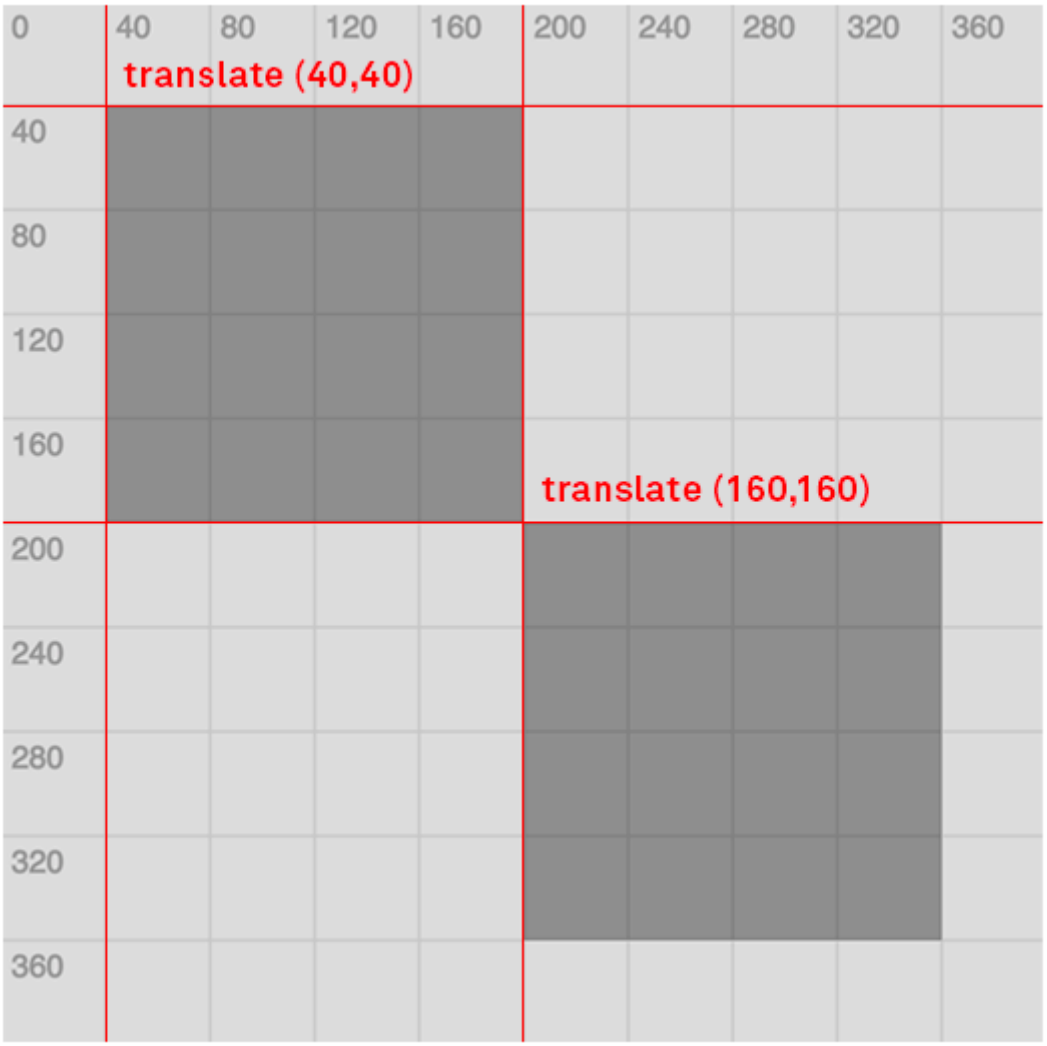


Keep in mind that transformation functions, like `translate()`, accumulate. This means that the x and y values given to a second `translate()` function will not represent the exact position of the new origin. Instead, the values inside the first and second `translate()` functions will add up to become the new position.

In the diagram below, you can see that while the second rectangle is translated by 160 pixels to the right and the bottom, the top-left corner of the shape is not at the x and y coordinate of (160, 160). This is because the second translation by 160 pixels to the right and the bottom is added onto the first translation by 40 pixels to the right and the bottom. Together, this adds up to set the new origin at the coordinates (200, 200).

```
//First rectangle is at (40, 40)
translate(40,40);
rect(0, 0, 80, 80);
```

```
//Second rectangle is at (200, 200) because translate() accumulates
translate(160,160);
rect(0, 0, 80, 80);
```



The main benefit of the `translate()` function is that the shapes become independent of the position. This means that you can reuse the code for a shape multiple times to be drawn in various locations without having to assign new x and y variables for the shape functions.

Instructions

1.

Above the first `rect()` function that has a width and height of 360 pixels, translate the shape by 60 pixels right and 60 pixels down.
Hint

You can translate a shape using the `translate()` function with the following syntax:

```
translate(x, y);
```

where `x` represents the number of pixels to move along the x-axis and `y` represents the number of pixels to move along the y-axis.

2.

Right above the second `rect()` function that has a width and height of 180 pixels, translate the shape to be 60 pixels right and 60 pixels down.

Notice how the second rectangle is not in the same location as the first rectangle, even though both `rect()` functions have the same x and y positions.
Hint
You can translate a shape using the `translate()` function with the following syntax:

```
translate(x, y);
```

Remember that the values for the `translate()` function accumulate.

sketch.js

```
function setup() {
  createCanvas(windowWidth, windowHeight);
}

function draw() {
  background(0);

  fill(255,0,0);
  // TODO: Translate the rectangle to be 60px right and down
  translate(60, 60);
  rect(0, 0, 360, 360);

  fill(0,0,255);
  // TODO: Translate the rectangle to be 60px right and down
  translate(60, 60);
  rect(0, 0, 180, 180);
}
```