

translate()

Translates the coordinate system.

By default, the origin `(0, 0)` is at the sketch's top-left corner in 2D mode and center in WebGL mode. The `translate()` function shifts the origin to a different position. Everything drawn after `translate()` is called will appear to be shifted. There are two ways to call `translate()` with parameters that set the origin's position.

The first way to call `translate()` uses numbers to set the amount of translation. The first two parameters, `x` and `y`, set the amount to translate along the positive x- and y-axes. For example, calling `translate(20, 30)` translates the origin 20 pixels along the x-axis and 30 pixels along the y-axis. The third parameter, `z`, is optional. It sets the amount to translate along the positive z-axis. For example, calling `translate(20, 30, 40)` translates the origin 20 pixels along the x-axis, 30 pixels along the y-axis, and 40 pixels along the z-axis.

The second way to call `translate()` uses a `p5.Vector` object to set the amount of translation. For example, calling `translate(myVector)` uses the x-, y-, and z-components of `myVector` to set the amount to translate along the x-, y-, and z-axes. Doing so is the same as calling `translate(myVector.x, myVector.y, myVector.z)`.

By default, transformations accumulate. For example, calling `translate(10, 0)` twice has the same effect as calling `translate(20, 0)` once. The `push()` and `pop()` functions can be used to isolate transformations within distinct drawing groups.

Note: Transformations are reset at the beginning of the draw loop. Calling `translate(10, 0)` inside the `draw()` function won't cause shapes to move continuously.

Examples

▶

■

```
function setup() {
  createCanvas(100, 100);

  describe('A white circle on a gray background.');
```

```
function draw() {
  background(200);

  // Translate the origin to the center.
  translate(50, 50);

  // Draw a circle at coordinates (0, 0).
  circle(0, 0, 40);
}
```

▶

■

```
function setup() {
  createCanvas(100, 100);

  describe(
    'Two circles drawn on a gray background. The blue circle
    on the right overlaps the red circle at the center.'
  );
}
```

```
function draw() {
  background(200);

  // Translate the origin to the center.
  translate(50, 50);

  // Draw the red circle.
  fill('red');
  circle(0, 0, 40);

  // Translate the origin to the right.
  translate(25, 0);

  // Draw the blue circle.
  fill('blue');
  circle(0, 0, 40);
}
```

▶

■

```
function setup() {
  createCanvas(100, 100);

  describe('A white circle moves slowly from left to right on
a gray background.');
```

```
function draw() {
  background(200);

  // Calculate the x-coordinate.
  let x = frameCount * 0.2;

  // Translate the origin.
  translate(x, 50);

  // Draw a circle at coordinates (0, 0).
  circle(0, 0, 40);
}
```

▶

■

```
function setup() {
  createCanvas(100, 100);

  describe('A white circle on a gray background.');
```

```
function draw() {
  background(200);

  // Create a p5.Vector object.
  let v = createVector(50, 50);

  // Translate the origin by the vector.
  translate(v);

  // Draw a circle at coordinates (0, 0).
  circle(0, 0, 40);
}
```

▶

■

```
function setup() {
  createCanvas(100, 100, WEBGL);

  describe(
    'Two spheres sitting side-by-side on gray background. The
    sphere at the center is red. The sphere on the right is blue.'
  );
}
```

```
function draw() {
  background(200);

  // Turn on the lights.
  lights();

  // Style the spheres.
  noStroke();

  // Draw the red sphere.
  fill('red');
  sphere(10);

  // Translate the origin to the right.
  translate(30, 0, 0);

  // Draw the blue sphere.
  fill('blue');
  sphere(10);
}
```

Syntax

```
translate(x, y, [z])
```

```
translate(vector)
```

Parameters

x	Number: amount to translate along the positive x-axis.
y	Number: amount to translate along the positive y-axis.
z	Number: amount to translate along the positive z-axis.
vector	p5.Vector: vector by which to translate.

This page is generated from the comments in `src/core/transform.js`. Please feel free to edit it and submit a pull request!

Related References

applyMatrix Applies a transformation matrix to the coordinate system.	resetMatrix Clears all transformations applied to the coordinate system.	rotate Rotates the coordinate system.	rotateX Rotates the coordinate system about the x-axis in WebGL mode.
---	--	---	---