noise()

Returns random numbers that can be tuned to feel organic.

Values returned by random() and randomGaussian() can change by large amounts between function calls. By contrast, values returned by `noise()` can be made "smooth". Calls to `noise()` with similar inputs will produce similar outputs. `noise()` is used to create textures, motion, shapes, terrains, and so on. Ken Perlin invented `noise()` while animating the original *Tron* film in the 1980s.

`noise()` always returns values between 0 and 1. It returns the same value for a given input while a sketch is running. `noise()` produces different results each time a sketch runs. The noiseSeed() function can be used to generate the same sequence of Perlin noise values each time a sketch runs.

The character of the noise can be adjusted in two ways. The first way is to scale the inputs. `noise()` interprets inputs as coordinates. The sequence of noise values will be smoother when the input coordinates are closer. The second way is to use the noiseDetail() function.

The version of `noise()` with one parameter computes noise values in one dimension. This dimension can be thought of as space, as in `noise(x)`, or time, as in `noise(t)`.

The version of `noise()` with two parameters computes noise values in two dimensions. These dimensions can be thought of as space, as in `noise(x, y)`, or space and time, as in `noise(x, t)`.

The version of `noise()` with three parameters computes noise values in three dimensions. These dimensions can be thought of as space, as in `noise(x, y, z)`, or space and time, as in `noise(x, y, t)`.

Examples

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

  describe('A black dot moves randomly on a gray square.');
}

function draw() {
  background(200);

  // Calculate the coordinates.
  let x = 100 * noise(0.005 * frameCount);
  let y = 100 * noise(0.005 * frameCount + 10000);

  // Draw the point.
  strokeWeight(5);
  point(x, y);
}
```

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

  describe('A black dot moves randomly on a gray square.');
}

function draw() {
  background(200);

  // Set the noise level and scale.
  let noiseLevel = 100;
  let noiseScale = 0.005;

  // Scale the input coordinate.
  let nt = noiseScale * frameCount;

  // Compute the noise values.
  let x = noiseLevel * noise(nt);
  let y = noiseLevel * noise(nt + 10000);

  // Draw the point.
  strokeWeight(5);
  point(x, y);
}
```

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

  describe('A hilly terrain drawn in gray against a black
sky.');
}

function draw() {
  // Set the noise level and scale.
  let noiseLevel = 100;
  let noiseScale = 0.02;

  // Scale the input coordinate.
  let x = frameCount;
  let nx = noiseScale * x;

  // Compute the noise value.
  let y = noiseLevel * noise(nx);

  // Draw the line.
  line(x, 0, x, y);
}
```

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

  describe('A calm sea drawn in gray against a black sky.');
}

function draw() {
  background(200);

  // Set the noise level and scale.
  let noiseLevel = 100;
  let noiseScale = 0.002;

  // Iterate from left to right.
  for (let x = 0; x < width; x += 1) {
    // Scale the input coordinates.
    let nx = noiseScale * x;
    let nt = noiseScale * frameCount;

    // Compute the noise value.
    let y = noiseLevel * noise(nx, nt);

    // Draw the line.
    line(x, 0, x, y);
  }
}
```

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

  background(200);

  // Set the noise level and scale.
  let noiseLevel = 255;
  let noiseScale = 0.01;

  // Iterate from top to bottom.
  for (let y = 0; y < height; y += 1) {
    // Iterate from left to right.
    for (let x = 0; x < width; x += 1) {
      // Scale the input coordinates.
      let nx = noiseScale * x;
      let ny = noiseScale * y;

      // Compute the noise value.
      let c = noiseLevel * noise(nx, ny);

      // Draw the point.
      stroke(c);
      point(x, y);
    }
  }

  describe('A gray cloudy pattern.');
}
```

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

  describe('A gray cloudy pattern that changes.');
}

function draw() {
  // Set the noise level and scale.
  let noiseLevel = 255;
  let noiseScale = 0.009;

  // Iterate from top to bottom.
  for (let y = 0; y < height; y += 1) {
    // Iterate from left to right.
    for (let x = 0; x < width; x += 1) {
      // Scale the input coordinates.
      let nx = noiseScale * x;
      let ny = noiseScale * y;
      let nt = noiseScale * frameCount;

      // Compute the noise value.
      let c = noiseLevel * noise(nx, ny, nt);

      // Draw the point.
      stroke(c);
      point(x, y);
    }
  }
}
```

Syntax

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

Parameters

x	Number: x-coordinate in noise space.
y	Number: y-coordinate in noise space.
z	Number: z-coordinate in noise space.

Returns

Number: Perlin noise value at specified coordinates.

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

Related References

noise
Returns random numbers that can be tuned to feel organic.

noiseDetail
Adjusts the character of the noise produced by the noise() function.

noiseSeed
Sets the seed value for the noise() function.