

textureMode()

Changes the coordinate system used for textures when they're applied to custom shapes.

In order for `texture()` to work, a shape needs a way to map the points on its surface to the pixels in an image. Built-in shapes such as `rect()` and `box()` already have these texture mappings based on their vertices. Custom shapes created with `vertex()` require texture mappings to be passed as uv coordinates.

Each call to `vertex()` must include 5 arguments, as in `vertex(x, y, z, u, v)`, to map the vertex at coordinates `(x, y, z)` to the pixel at coordinates `(u, v)` within an image. For example, the corners of a rectangular image are mapped to the corners of a rectangle by default:

```
// Apply the image as a texture.
texture(img);

// Draw the rectangle.
rect(0, 0, 30, 50);
```

If the image in the code snippet above has dimensions of 300 x 500 pixels, the same result could be achieved as follows:

```
// Apply the image as a texture.
texture(img);

// Draw the rectangle.
beginShape();

// Top-left.
// u: 0, v: 0
vertex(0, 0, 0, 0, 0);

// Top-right.
// u: 300, v: 0
vertex(30, 0, 0, 300, 0);

// Bottom-right.
// u: 300, v: 500
vertex(30, 50, 0, 300, 500);

// Bottom-left.
// u: 0, v: 500
vertex(0, 50, 0, 0, 500);

endShape();
```

`textureMode()` changes the coordinate system for uv coordinates.

The parameter, `mode`, accepts two possible constants. If `NORMAL` is passed, as in `textureMode(NORMAL)`, then the texture's uv coordinates can be provided in the range 0 to 1 instead of the image's dimensions. This can be helpful for using the same code for multiple images of different sizes. For example, the code snippet above could be rewritten as follows:

```
// Set the texture mode to use normalized coordinates.
textureMode(NORMAL);

// Apply the image as a texture.
texture(img);

// Draw the rectangle.
beginShape();

// Top-left.
// u: 0, v: 0
vertex(0, 0, 0, 0, 0);

// Top-right.
// u: 1, v: 0
vertex(30, 0, 0, 1, 0);

// Bottom-right.
// u: 1, v: 1
vertex(30, 50, 0, 1, 1);

// Bottom-left.
// u: 0, v: 1
vertex(0, 50, 0, 0, 1);

endShape();
```

By default, `mode` is `IMAGE`, which scales uv coordinates to the dimensions of the image. Calling `textureMode(IMAGE)` applies the default.

Note: `textureMode()` can only be used in WebGL mode.

Examples

```
let img;

// Load an image and create a p5.Image object.
function preload() {
  img = loadImage('/assets/laDefense.jpg');
}

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

  describe('An image of a ceiling against a black background.');
}

function draw() {
  background(0);

  // Apply the image as a texture.
  texture(img);

  // Draw the custom shape.
  // Use the image's width and height as uv coordinates.
  beginShape();
  vertex(-30, -30, 0, 0);
  vertex(30, -30, img.width, 0);
  vertex(30, 30, img.width, img.height);
  vertex(-30, 30, 0, img.height);
  endShape();
}

let img;

// Load an image and create a p5.Image object.
function preload() {
  img = loadImage('/assets/laDefense.jpg');
}

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

  describe('An image of a ceiling against a black background.');
}

function draw() {
  background(0);

  // Set the texture mode.
  textureMode(NORMAL);

  // Apply the image as a texture.
  texture(img);

  // Draw the custom shape.
  // Use normalized uv coordinates.
  beginShape();
  vertex(-30, -30, 0, 0);
  vertex(30, -30, 1, 0);
  vertex(30, 30, 1, 1);
  vertex(-30, 30, 0, 1);
  endShape();
}
```

Syntax

```
textureMode(mode)
```

Parameters

`mode` Constant: either `IMAGE` or `NORMAL`.

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

Related References

`copyToContext` Copies the shader from one drawing context to another.

`inspectHooks` Logs the hooks available in this shader, and their current implementation.

`modify` Returns a new shader, based on the original, but with custom snippets of shader code replacing default behaviour.

`setUniform` Sets the shader's uniform (global) variables.

`p5.js`

`Resources`

`Information`

`Socials`

[Reference](#)

[Download](#)

[GitHub ↗](#)

[Tutorials](#)

[Contact](#)

[Instagram ↗](#)

[Examples](#)

[Copyright](#)

[X ↗](#)

[Contribute](#)

[Privacy Policy](#)

[YouTube ↗](#)

[Community](#)

[Terms of Use](#)

[Discord ↗](#)

[About](#)

[FAQ](#)

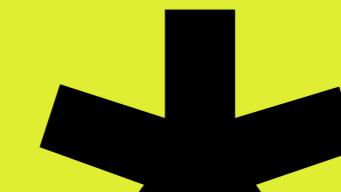
[Forum ↗](#)

[Start Coding](#)

[Help](#)

[Feedback ↗](#)

[Donate](#)

The p5.js logo, which is a stylized asterisk or star shape composed of thick black lines.

Donate Today! Support p5.js and the Processing Foundation.

X