modify()

This API is experimental

Its behavior may change in a future version of p5.js.

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

Each shader may let you override bits of its behavior. Each bit is called a *hook*. A hook is either for the *vertex* shader, if it affects the position of vertices, or in the *fragment* shader, if it affects the pixel color. You can inspect the different hooks available by calling `yourShader.inspectHooks()`. You can also read the reference for the default material, normal material, color, line, and point shaders to see what hooks they have available.

`modify()` takes one parameter, `hooks`, an object with the hooks you want to override. Each key of the `hooks` object is the name of a hook, and the value is a string with the GLSL code for your hook.

If you supply functions that aren't existing hooks, they will get added at the start of the shader as helper functions so that you can use them in your hooks.

To add new uniforms to your shader, you can pass in a `uniforms` object containing the type and name of the uniform as the key, and a default value or function returning a default value as its value. These will be automatically set when the shader is set with `shader(yourShader)`.

You can also add a `declarations` key, where the value is a GLSL string declaring custom uniform variables, globals, and functions shared between hooks. To add declarations just in a vertex or fragment shader, add `vertexDeclarations` and `fragmentDeclarations` keys.

Examples

```
let myShader;

function setup() {
  createCanvas(200, 200, WEBGL);
  myShader = baseMaterialShader().modify({
    uniforms: {
      'float time': () => millis()
    },
    'vec3 getWorldPosition': `(vec3 pos) {
      pos.y += 20. * sin(time * 0.001 + pos.x * 0.05);
      return pos;
    }`
  });
}

function draw() {
  background(255);
  shader(myShader);
  lights();
  noStroke();
  fill('red');
  sphere(50);
}
```

```
let myShader;

function setup() {
  createCanvas(200, 200, WEBGL);
  myShader = baseMaterialShader().modify({
    // Manually specifying a uniform
    declarations: 'uniform float time;',
    'vec3 getWorldPosition': `(vec3 pos) {
      pos.y += 20. * sin(time * 0.001 + pos.x * 0.05);
      return pos;
    }`
  });
}

function draw() {
  background(255);
  shader(myShader);
  myShader.setUniform('time', millis());
  lights();
  noStroke();
  fill('red');
  sphere(50);
}
```

Syntax

```
modify([hooks])
```

Parameters

hooks	Object: The hooks in the shader to replace.

Returns

p5.Shader:

This page is generated from the comments in src/webgl/p5.Shader.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.