push()

Begins a drawing group that contains its own styles and transformations.

By default, styles such as fill() and transformations such as rotate() are applied to all drawing that follows. The `push()` and pop() functions can limit the effect of styles and transformations to a specific group of shapes, images, and text. For example, a group of shapes could be translated to follow the mouse without affecting the rest of the sketch:

```
// Begin the drawing group.
push();

// Translate the origin to the mouse's position.
translate(mouseX, mouseY);

// Style the face.
noStroke();
fill('green');

// Draw the face.
circle(0, 0, 60);

// Style the eyes.
fill('white');

// Draw the left eye.
ellipse(-20, -20, 30, 20);

// Draw the right eye.
ellipse(20, -20, 30, 20);

// End the drawing group.
pop();

// Draw a bug.
let x = random(0, 100);
let y = random(0, 100);
text('🐞', x, y);
```

In the code snippet above, the bug's position isn't affected by `translate(mouseX, mouseY)` because that transformation is contained between `push()` and pop(). The bug moves around the entire canvas as expected.

Note: `push()` and pop() are always called as a pair. Both functions are required to begin and end a drawing group.

`push()` and pop() can also be nested to create subgroups. For example, the code snippet above could be changed to give more detail to the frog's eyes:

```
// Begin the drawing group.
push();

// Translate the origin to the mouse's position.
translate(mouseX, mouseY);

// Style the face.
noStroke();
fill('green');

// Draw a face.
circle(0, 0, 60);

// Style the eyes.
fill('white');

// Draw the left eye.
push();
translate(-20, -20);
ellipse(0, 0, 30, 20);
fill('black');
circle(0, 0, 8);
pop();

// Draw the right eye.
push();
translate(20, -20);
ellipse(0, 0, 30, 20);
fill('black');
circle(0, 0, 8);
pop();

// End the drawing group.
pop();

// Draw a bug.
let x = random(0, 100);
let y = random(0, 100);
text('🐞', x, y);
```

In this version, the code to draw each eye is contained between its own `push()` and pop() functions. Doing so makes it easier to add details in the correct part of a drawing.

`push()` and pop() contain the effects of the following functions:

- fill()
- noFill()
- noStroke()
- stroke()
- tint()
- noTint()
- strokeWeight()
- strokeCap()
- strokeJoin()
- imageMode()
- rectMode()
- ellipseMode()
- colorMode()
- textAlign()
- textFont()
- textSize()
- textLeading()
- applyMatrix()
- resetMatrix()
- rotate()
- scale()
- shearX()
- shearY()
- translate()

In WebGL mode, `push()` and pop() contain the effects of a few additional styles:

- setCamera()
- ambientLight()
- directionalLight()
- pointLight() texture()
- specularMaterial()
- shininess()
- normalMaterial()
- shader()

Examples

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

  background(200);

  // Draw the left circle.
  circle(25, 50, 20);

  // Begin the drawing group.
  push();

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

  // Style the circle.
  strokeWeight(5);
  stroke('royalblue');
  fill('orange');

  // Draw the circle.
  circle(0, 0, 20);

  // End the drawing group.
  pop();

  // Draw the right circle.
  circle(75, 50, 20);

  describe(
    'Three circles drawn in a row on a gray background. The left and right circles are white with thin, black borders. The middle circle is orange with a thick, blue border.'
  );
}
```

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

  // Slow the frame rate.
  frameRate(24);

  describe('A mosquito buzzes in front of a green frog. The frog follows the mouse as the user moves.');
}

function draw() {
  background(200);

  // Begin the drawing group.
  push();

  // Translate the origin to the mouse's position.
  translate(mouseX, mouseY);

  // Style the face.
  noStroke();
  fill('green');

  // Draw a face.
  circle(0, 0, 60);

  // Style the eyes.
  fill('white');

  // Draw the left eye.
  push();
  translate(-20, -20);
  ellipse(0, 0, 30, 20);

  // Draw the right eye.
  push();
  translate(20, -20);
  ellipse(0, 0, 30, 20);
```

```
// Click and drag the mouse to view the scene from different angles.

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

  describe(
    'Two spheres drawn on a gray background. The sphere on the left is red and lit from the front. The sphere on the right is a blue wireframe.'
  );
}

function draw() {
  background(200);

  // Enable orbiting with the mouse.
  orbitControl();

  // Draw the red sphere.
  push();
  translate(-25, 0, 0);
  noStroke();
  directionalLight(255, 0, 0, 0, 0, -1);
  sphere(20);
  pop();

  // Draw the blue sphere.
  push();
  translate(25, 0, 0);
  strokeWeight(0.3);
  stroke(0, 0, 255);
```

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

Related References

disableFriendlyErrors
Turns off the parts of the Friendly Error System (FES) that impact performance.

draw
A function that's called repeatedly while the sketch runs.

isLooping
Returns true if the draw loop is running and false if not.

loop
Resumes the draw loop after noLoop() has been called.