

Reference > beginGeometry()

beginGeometry()

Begins adding shapes to a new **p5.Geometry** object.

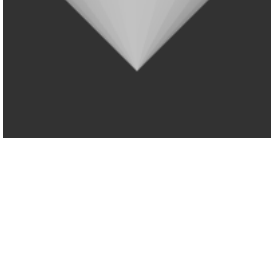
The `beginGeometry()` and `endGeometry()` functions help with creating complex 3D shapes from simpler ones such as `sphere()`. `beginGeometry()` begins adding shapes to a custom **p5.Geometry** object and `endGeometry()` stops adding them.

`beginGeometry()` and `endGeometry()` can help to make sketches more performant. For example, if a complex 3D shape doesn't change while a sketch runs, then it can be created with `beginGeometry()` and `endGeometry()`. Creating a **p5.Geometry** object once and then drawing it will run faster than repeatedly drawing the individual pieces.

See `buildGeometry()` for another way to build 3D shapes.

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

Examples



▶

■

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

let shape;

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

  // Start building the p5.Geometry object.
  beginGeometry();

  // Add a cone.
  cone();

  // Stop building the p5.Geometry object.
  shape = endGeometry();

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



▶

■

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

let shape;

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

  // Create the p5.Geometry object.
  createArrow();

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



▶

■

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

let blueArrow;
let redArrow;

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

  // Create the arrows.
  redArrow = createArrow('red');
  blueArrow = createArrow('blue');

  describe('A red arrow and a blue arrow drawn on a gray background. The blue arrow rotates slowly.');
```



▶

■

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

let button;
let particles;

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

  // Create a button to reset the particle system.
  button = createButton('Reset');

  // Call resetModel() when the user presses the button.
  button.mousePressed(resetModel);

  // Add the original set of particles.
  resetModel();
}

function draw() {
  background(50);

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

  // Turn on the lights.
  lights();

  // Style the particles.
  noStroke();

  // Draw the particles.
  model(particles);
}

function resetModel() {
```

This page is generated from the comments in [src/webgl/3d_primitives.js](#). Please feel free to edit it and submit a pull request!

Related References

calculateBoundingBox Calculates the position and size of the smallest box that contains the geometry.	clearColors Removes the geometry's internal colors.	computeFaces Computes the geometry's faces using its vertices.	computeNormals Calculates the normal vector for each vertex on the geometry.
---	---	--	--