faces

An array that lists which of the geometry's vertices form each of its faces.

All 3D shapes are made by connecting sets of points called *vertices*. A geometry's surface is formed by connecting vertices to form triangles that are stitched together. Each triangular patch on the geometry's surface is called a *face*.

The geometry's vertices are stored as p5.Vector objects in the myGeometry.vertices array. The geometry's first vertex is the p5.Vector object at `myGeometry.vertices[0]`, its second vertex is `myGeometry.vertices[1]`, its third vertex is `myGeometry.vertices[2]`, and so on.

For example, a geometry made from a rectangle has two faces because a rectangle is made by joining two triangles. `myGeometry.faces` for a rectangle would be the two-dimensional array `[[0, 1, 2], [2, 1, 3]]`. The first face, `myGeometry.faces[0]`, is the array `[0, 1, 2]` because it's formed by connecting `myGeometry.vertices[0]`, `myGeometry.vertices[1]`,and `myGeometry.vertices[2]`. The second face, `myGeometry.faces[1]`, is the array `[2, 1, 3]` because it's formed by connecting `myGeometry.vertices[2]`, `myGeometry.vertices[1]`,and `myGeometry.vertices[3]`.

Examples

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

let myGeometry;

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

  // Create a p5.Geometry object.
  beginGeometry();
  sphere();
  myGeometry = endGeometry();

  describe("A sphere drawn on a gray background. The sphere's
surface is a grayscale patchwork of triangles.");
}

function draw() {
  background(200);

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

  // Turn on the lights.
  lights();

  // Style the p5.Geometry object.
  noStroke();

  // Set a random seed.
  randomSeed(1234);

  // Iterate over the faces array.
```

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