

box()

Draws a box (rectangular prism).

A box is a 3D shape with six faces. Each face makes a 90° with four neighboring faces.

The first parameter, `width`, is optional. If a `Number` is passed, as in `box(20)`, it sets the box's width and height. By default, `width` is 50.

The second parameter, `height`, is also optional. If a `Number` is passed, as in `box(20, 30)`, it sets the box's height. By default, `height` is set to the box's `width`.

The third parameter, `depth`, is also optional. If a `Number` is passed, as in `box(20, 30, 40)`, it sets the box's depth. By default, `depth` is set to the box's `height`.

The fourth parameter, `detailX`, is also optional. If a `Number` is passed, as in `box(20, 30, 40, 5)`, it sets the number of triangle subdivisions to use along the x-axis. All 3D shapes are made by connecting triangles to form their surfaces. By default, `detailX` is 1.

The fifth parameter, `detailY`, is also optional. If a number is passed, as in `box(20, 30, 40, 5, 7)`, it sets the number of triangle subdivisions to use along the y-axis. All 3D shapes are made by connecting triangles to form their surfaces. By default, `detailY` is 1.

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

Examples

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

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

  describe('A white box on a gray background.');
}

function draw() {
  background(200);

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

  // Draw the box.
  box();
}
```

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

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

  describe('A white box on a gray background.');
}

function draw() {
  background(200);

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

  // Draw the box.
  // Set its width and height to 30.
  box(30);
}
```

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

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

  describe('A white box on a gray background.');
}

function draw() {
  background(200);

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

  // Draw the box.
  // Set its width to 30, height to 50.
  box(30, 50);
}
```

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

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

  describe('A white box on a gray background.');
}

function draw() {
  background(200);

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

  // Draw the box.
  // Set its width to 30, height to 50, and depth to 10.
  box(30, 50, 10);
}
```

Syntax

`box([width], [height], [depth], [detailX], [detailY])`

Parameters

<code>width</code>	Number: width of the box.
<code>height</code>	Number: height of the box.
<code>depth</code>	Number: depth of the box.
<code>detailX</code>	Integer: number of triangle subdivisions along the x-axis.
<code>detailY</code>	Integer: number of triangle subdivisions along the y-axis.

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

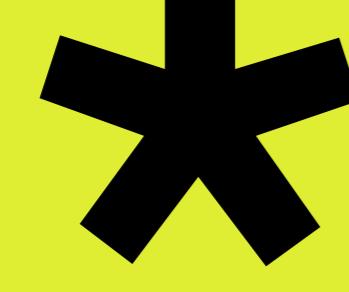
Related References

[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.



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

