

# 3D SHAPE

View online: <https://www.construct.net/en/make-games/manuals/construct-3/plugin-reference/3d-shape>

---

The **3D shape** object can add basic elements of 3D to your project, such as a 3D box. Six images can be drawn to cover each face of the shape.

Try out the [3D shape tour example](#) for a visual demonstration of what the 3D shape object can do. The Start Page has a number of other examples under the *3D* tag.

You can learn more about 3D shape and how it works with other 3D features in the tutorial [Using 3D in Construct](#).

## Scripting

When using JavaScript or TypeScript coding, the features of this object can be accessed via the [I3DShapeInstance script interface](#).

## Size and position on Z axis

Construct's normal 2D editing features cover the X and Y co-ordinates and size. To adjust the position on the Z axis, change the **Z elevation** property (which is the same property used to move other 2D objects like Sprite up and down on the Z axis). To adjust how tall the shape is (i.e. its depth), change the **Z height** property of the 3D shape object.

## Transparency

Due to the way 3D rendering works, transparency may not work as expected in 3D shape objects. To correctly render 3D features, Construct must use a [depth buffer](#), but this only fully supports opaque surfaces. Therefore **3D shape objects will work best with opaque images**.

Making individual faces invisible works as expected, as this means the face skips drawing entirely, rather than drawing content that is assumed to be opaque.

For more information, refer to the section on transparency in the tutorial [Using 3D features in Construct](#).

## 3D shape properties

---

### Images

Click the *Edit* link to edit the six images the object uses for face images, one for each face of a cube. Not all shapes use all six images. Some images may also be cropped according to the shape of the face it appears on. For example the *Prism* shape has triangular end faces, so the images at these ends are cropped to only show a triangular portion of the image.

## Shape

Select from one of a few pre-defined shapes that the object will use. Not all shapes use all six images, since some have fewer than six faces. The available shapes are:

- **Box:** a six-sided cube or cuboid shape.
- **Prism:** a five-sided prism shape, like a tent. The front face is not used.
- **Wedge:** a five-sided shape like a cuboid cut in half diagonally, sloping up to the right. The front face is not used.
- **Pyramid:** a five-sided square-based pyramid, with the peak of the pyramid centered over the base. The front face is not used.
- **Corner (out):** a five-sided shape similar to a pyramid with the peak aligned to the upper-right corner. The front face is not used. The name derives from the fact this shape can join rows of *Wedge* shapes at an outside corner.
- **Corner (in):** a six-sided shape similar to a box where the front and back faces join in the bottom-left. The front face is used, but slopes down in the bottom-left half of the image. The name derives from the fact this shape can join rows of *Wedge* shapes at an inside corner.

## Z height

Set the height of the shape on the Z axis, i.e. its depth. Note this must be positive. If you wish to display the object lower down, change its *Z elevation* instead.

## Initially visible

Set whether the object is shown (visible) or hidden (invisible) when the layout starts.

## Face visibility (Back/Front/Left/Right/Top/Bottom)

Set whether each of the six faces of the shape is initially visible.

*The Back face is hidden by default, as normally it cannot be seen and so may as well skip drawing. If you make another face invisible, you may wish to make the back face visible again.*

## Back face culling

Automatically hides faces of the shape that are pointing away from the camera. This helps optimize performance if the shape is fully opaque, as those faces will never be seen anyway. If the shape is semitransparent (e.g. the opacity is less than 100%) then this mode can be useful to stop the faces pointing away from the camera from still being visible. Whether or not that is desirable depends on your art style.

---

### Use object image for faces (Back/Front/Left/Right/Top/Bottom)

Optionally choose a Sprite, Tiled Background or 9-Patch object to display instead of the 3D shape's own face images for a given face of the shape. An instance of the object must be placed in the same layout for this to work. The properties of this instance can also be used to control the appearance of the face on the 3D shape. This allows using animated face images (via Sprite), or varieties of tiled/repeating images for face images (via Tiled Background and 9-Patch).

---

### Z tiling factor

By default the camera appears at  $Z=100$  and looks down to  $Z=0$ , meaning the camera is normally 100 units above the layout. However when using tiled images for 3D shape faces, such as Tiled Background or 9-Patch, this can result in unexpected tiling results. For example a 3D shape that has a Z height of 25 will display a Tiled Background as if it was 25 pixels tall, which may be too small for the displayed size of the object. The Z tiling factor is a multiple for the Z height to use when tiling images. For example the default of 8 means a Z height of 25 will actually tile as if it was 200 pixels tall, which usually produces better tiling results.

## 3D shape conditions

For conditions in common to other objects, see [Common conditions](#).

---

### Compare shape

Compare the current shape that is in use. This can be changed by the *Set shape* action.

---

### Compare Z height

Compare the current Z height (i.e. depth) of the shape.

---

### Is face visible

Check if one of the six faces of the shape is currently set to visible. Note this only checks whether the visibility is currently enabled, either in the object's properties or with the *Set face visible* action - it does not test whether the face is really showing on-screen.

## 3D shape actions

For actions common to other objects, see [Common actions](#).

---

### Set back face culling

Change the *Back face culling* property. See above for more details.

---

### Set face image

Change one of the shape faces to use one of the other face images. For example this allows swapping the front face image for the back face image. To restore the original image, use the same face for both parameters, e.g. set back face to use image of back face.

*This also undoes Set face object, restoring the 3D shape's own face image instead of another object's image.*

---

### Set face object

Replace the image used for a face of the shape with the image used by a Sprite, Tiled Background or 9-Patch object. An instance of the given object must exist on the current layout. See the property *Use object image for faces* for more information.

*This action can be undone with Set face image.*

---

### Set face visible

Enable or disable the visibility of one of the faces of the shape. See the *Face visibility* properties for more information.

---

### Set shape

Change the shape currently used by the object. See the *Shape* property for more information.

---

### Set Z height

Change the Z height, i.e. depth, of the 3D shape. This must be greater or equal to 0. See the *Z height* property for more information.

---

### Set Z tiling factor

Change the multiple used for tiling images along the Z height of the object. See the *Z tiling factor* property for more information.

---

## 3D shape expressions

For expressions common to other objects, see [common expressions](#).

---

### FacelImagePointCount(Face)

**FacelImagePointX(Face, ImagePoint)****FacelImagePointY(Face, ImagePoint)****FacelImagePointZ(Face, ImagePoint)**

Retrieve the 3D position of an image point on any of the 3D shape's faces. The face is a zero-based index of the face as shown in the image editor, i.e.:

- 0: Back
- 1: Front
- 2: Left
- 3: Right
- 4: Top
- 5: Bottom

The image point is specified by the name or number of the image point. When using a number, note that as per other image point expressions, the first image point is number 1, as 0 refers to the origin.

**ZHeight****ZTilingFactor**

Return the current Z height and Z tiling factor properties. See the documentation on the corresponding properties above for more information.