
pythreejs Documentation

Release 1.1.0

PyThreejs Development Team

Oct 22, 2018

1	Quickstart	3
2	Contents	5
2.1	Installation	5
2.2	Upgrading to 1.x	5
2.3	Introduction	6
2.4	Examples	7
2.5	API Reference	18
2.6	Extending pythreejs	162
2.7	Developer install	165
	Python Module Index	167

Version: 1.1.0

pythreejs is a [Jupyter widgets](#) based [notebook](#) extension that allows Jupyter to leverage the WebGL capabilities of modern browsers by creating bindings to the javascript library [three.js](#).

By being based on top of the jupyter-widgets infrastructure, it allows for eased integration with other interactive tools for notebooks.

CHAPTER 1

Quickstart

To get started with pythreejs, install with pip:

```
pip install pythreejs
```

If you are using a notebook version older than 5.3, or if your kernel is in another environment than the notebook server, you will also need to register the front-end extensions.

For the notebook front-end:

```
jupyter nbextension install [--sys-prefix | --user | --system] --py pythreejs  
jupyter nbextension enable [--sys-prefix | --user | --system] --py pythreejs
```

For jupyterlab:

```
jupyter labextension install jupyter-threejs
```

Note: If you are installing an older version of pythreejs, you might have to add a version specifier for the labextension to match the Python package, e.g. *jupyter-threejs@1.0.0*.

2.1 Installation

The simplest way to install pythreejs is via pip:

```
pip install pythreejs
```

or via conda:

```
conda install pythreejs
```

With jupyter notebook version ≥ 5.3 , this should also install and enable the relevant front-end extensions. If for some reason this did not happen (e.g. if the notebook server is in a different environment than the kernel), you can install / configure the front-end extensions manually. If you are using classic notebook (as opposed to Jupyterlab), run:

```
jupyter nbextension install [--sys-prefix / --user / --system] --py pythreejs
jupyter nbextension enable [--sys-prefix / --user / --system] --py pythreejs
```

with the appropriate flag. If you are using Jupyterlab, install the extension with:

```
jupyter labextension install jupyter-threejs
```

2.2 Upgrading to 1.x

If you are upgrading to version 1.x from a version prior to 1.0, there are certain backwards-incompatible changes that you should note:

- `Plain[Buffer]Geometry` was renamed to `[Buffer]Geometry`. This was done in order to be more consistent with the names used in three.js. The base classes for geometry are now called `Base[Buffer]Geometry`. This also avoids the confusion with `Plane[Buffer]Geometry`.

- `LambertMaterial` -> `MeshLambertMaterial`, and other similar material class renames were done. Again, this was to more closely match the names used in three.js itself.

2.3 Introduction

The pythreejs API attempts to mimic the three.js API as closely as possible, so any resource on its API should also be helpful for understanding pythreejs. See for example the [official three.js documentation](#).

The major difference between the two is the render loop. As we normally do not want to call back to the kernel for every rendered frame, some helper classes have been created to allow for user interaction with the scene with minimal overhead:

2.3.1 Renderer classes

While the `WebGLRenderer` class mimics its three.js counterpart in only rendering frames on demand (one frame per call to its `render()` method), the `Renderer` class sets up an interactive render loop allowing for *Interactive controls* and *Animation views*. Similarly, a *Preview* widget allows for a quick visualization of various threejs objects.

2.3.2 Interactive controls

These are classes for managing user interaction with the WebGL canvas, and translating that into actions. One example is the *OrbitControls* class, which allows the user to control the camera by zooming, panning, and orbital rotation around a target. Another example is the *Picker* widget, which allows for getting the objects and surface coordinates underneath the mouse cursor.

To use controls, pass them to the renderer, e.g.:

```
Renderer (controls=[OrbitControls (...), ...], ...)
```

2.3.3 Animation views

The view widgets for the *AnimationAction* class gives interactive controls to the user for controlling a *threejs animation*.

Other notable deviations from the threejs API are listed below:

- Buffers are based on [numpy arrays](#), with their inbuilt knowledge of shape and dtype. As such, most threejs APIs that take a buffer are slightly modified (fewer options need to be specified explicitly).
- The generative geometry objects (e.g. *SphereGeometry* and *BoxBufferGeometry*) do not sync their vertices or similar data by default. To gain access to the generated data, convert them to either the *Geometry* or *BufferGeometry* type with the *from_geometry()* factory method.
- Methods are often not mirrored to the Python side. However, they can be executed with the *exec_three_obj_method()* method. Consider contributing to make methods directly available. Possibly, these can be auto-generated as well.

2.4 Examples

This section contains several examples generated from Jupyter notebooks. The widgets have been embedded into the page.

2.4.1 Geometry types

```
In [1]: from pythreejs import *
        from IPython.display import display
        from math import pi
```

```
In [2]: BoxGeometry(
        width=5,
        height=10,
        depth=15,
        widthSegments=5,
        heightSegments=10,
        depthSegments=15)
```

```
Preview(child=BoxGeometry(depth=15.0, depthSegments=15, height=10.0, heightSegments=10, width=5.0, w
```

```
In [3]: BoxBufferGeometry(
        width=5,
        height=10,
        depth=15,
        widthSegments=5,
        heightSegments=10,
        depthSegments=15)
```

```
Preview(child=BoxBufferGeometry(depth=15.0, depthSegments=15, height=10.0, heightSegments=10, width=5
```

```
In [4]: CircleGeometry(
        radius=10,
        segments=10,
        thetaStart=0.25,
        thetaLength=5.0)
```

```
Preview(child=CircleGeometry(radius=10.0, segments=10, thetaLength=5.0, thetaStart=0.25), shadowMap=
```

```
In [5]: CircleBufferGeometry(
        radius=10,
        segments=10,
        thetaStart=0.25,
        thetaLength=5.0)
```

```
Preview(child=CircleBufferGeometry(radius=10.0, segments=10, thetaLength=5.0, thetaStart=0.25), sha
```

```
In [6]: CylinderGeometry(
        radiusTop=5,
        radiusBottom=10,
        height=15,
        radialSegments=6,
        heightSegments=10,
        openEnded=False,
        thetaStart=0,
        thetaLength=2.0*pi)
```

```
Preview(child=CylinderGeometry(height=15.0, heightSegments=10, radiusBottom=10.0, radiusTop=5.0), sha
```

```
In [7]: CylinderBufferGeometry(
        radiusTop=5,
        radiusBottom=10,
```

```

    height=15,
    radialSegments=6,
    heightSegments=10,
    openEnded=False,
    thetaStart=0,
    thetaLength=2.0*pi)

```

Preview(child=CylinderBufferGeometry(height=15.0, heightSegments=10, radiusBottom=10.0, radiusTop=5.0))

```
In [8]: DodecahedronGeometry(radius=10, detail=0, _flat=True)
```

Preview(child=DodecahedronGeometry(radius=10.0), shadowMap=WebGLShadowMap())

```
In [ ]: # TODO:
        # EdgesGeometry(...)
```

```
In [ ]: # TODO:
        # ExtrudeGeometry(...)
```

```
In [9]: IcosahedronGeometry(radius=10, _flat=True)
```

Preview(child=IcosahedronGeometry(radius=10.0), shadowMap=WebGLShadowMap())

```
In [10]: LatheBufferGeometry(
    points=[
        [ 0, -10, 0 ],
        [ 10, -5, 0 ],
        [ 5, 5, 0 ],
        [ 0, 10, 0 ]
    ],
    segments=16,
    phiStart=0.0,
    phiLength=2.0*pi, _flat=True)
```

Preview(child=LatheBufferGeometry(points=[[0, -10, 0], [10, -5, 0], [5, 5, 0], [0, 10, 0]], segments=16, phiStart=0.0, phiLength=2.0*pi))

```
In [11]: OctahedronGeometry(radius=10, detail=0, _flat=True)
```

Preview(child=OctahedronGeometry(radius=10.0), shadowMap=WebGLShadowMap())

```
In [12]: ParametricGeometry(
    func=""function(u,v) {
        var x = 5 * (0.5 - u);
        var y = 5 * (0.5 - v);
        return new THREE.Vector3(10 * x, 10 * y, x*x - y*y);
    }"",
    slices=5,
    stacks=10, _flat=True)
```

Preview(child=ParametricGeometry(func='function(u,v) { \n var x = 5 * (0.5 - u);\n var y = 5 * (0.5 - v);\n return new THREE.Vector3(10 * x, 10 * y, x*x - y*y);\n }', slices=5, stacks=10))

```
In [13]: PlaneGeometry(
    width=10,
    height=15,
    widthSegments=5,
    heightSegments=10)
```

Preview(child=PlaneGeometry(height=15.0, heightSegments=10, width=10.0, widthSegments=5), shadowMap=WebGLShadowMap())

```
In [14]: PlaneBufferGeometry(
    width=10,
    height=15,
    widthSegments=5,
    heightSegments=10)
```

Preview(child=PlaneBufferGeometry(height=15.0, heightSegments=10, width=10.0, widthSegments=5), shadowMap=WebGLShadowMap())

```
In [ ]: # TODO
        # PolyhedronGeometry(...)
```

```
In [15]: # TODO: issues when radius is 0...
         RingGeometry(
             innerRadius=10,
             outerRadius=25,
             thetaSegments=8,
             phiSegments=12,
             thetaStart=0,
             thetaLength=6.283185307179586)
```

```
Preview(child=RingGeometry(innerRadius=10.0, outerRadius=25.0, phiSegments=12), shadowMap=WebGLShadowMap)
```

```
In [16]: # TODO: issues when radius is 0...
         RingBufferGeometry(
             innerRadius=10,
             outerRadius=25,
             thetaSegments=8,
             phiSegments=12,
             thetaStart=0,
             thetaLength=6.283185307179586)
```

```
Preview(child=RingBufferGeometry(innerRadius=10.0, outerRadius=25.0, phiSegments=12), shadowMap=WebGLShadowMap)
```

```
In [ ]: # TODO
        # ShapeGeometry(...)
```

```
In [17]: SphereGeometry(
         radius=20,
         widthSegments=8,
         heightSegments=6,
         phiStart=0,
         phiLength=1.5*pi,
         thetaStart=0,
         thetaLength=2.0*pi/3.0)
```

```
Preview(child=SphereGeometry(phiLength=4.71238898038469, radius=20.0, thetaLength=2.0943951023931953), shadowMap=WebGLShadowMap)
```

```
In [18]: SphereBufferGeometry(
         radius=20,
         widthSegments=8,
         heightSegments=6,
         phiStart=0,
         phiLength=1.5*pi,
         thetaStart=0,
         thetaLength=2.0*pi/3.0)
```

```
Preview(child=SphereBufferGeometry(phiLength=4.71238898038469, radius=20.0, thetaLength=2.0943951023931953), shadowMap=WebGLShadowMap)
```

```
In [19]: TetrahedronGeometry(radius=10, detail=1, _flat=True)
```

```
Preview(child=TetrahedronGeometry(detail=1, radius=10.0), shadowMap=WebGLShadowMap())
```

```
In [ ]: # TODO: font loading
        # TextGeometry(...)
```

```
In [20]: TorusGeometry(
         radius=20,
         tube=5,
         radialSegments=20,
         tubularSegments=6,
         arc=1.5*pi)
```

```
Preview(child=TorusGeometry(arc=4.71238898038469, radialSegments=20, radius=20.0, tube=5.0), shadowMap=WebGLShadowMap)
```

```
In [21]: TorusBufferGeometry(radius=100)
```

```
Preview(child=TorusBufferGeometry(radius=100.0), shadowMap=WebGLShadowMap())
```

```
In [22]: TorusKnotGeometry(  
    radius=20,  
    tube=5,  
    tubularSegments=64,  
    radialSegments=8,  
    p=2,  
    q=3)
```

```
Preview(child=TorusKnotGeometry(radius=20.0, tube=5.0), shadowMap=WebGLShadowMap())
```

```
In [23]: TorusKnotBufferGeometry(  
    radius=20,  
    tube=5,  
    tubularSegments=64,  
    radialSegments=8,  
    p=2,  
    q=3)
```

```
Preview(child=TorusKnotBufferGeometry(radius=20.0, tube=5.0), shadowMap=WebGLShadowMap())
```

```
In [ ]: # TODO: handling THREE.Curve  
    TubeGeometry(  
        path=None,  
        segments=64,  
        radius=1,  
        radiusSegments=8,  
        close=False)
```

```
In [24]: WireframeGeometry(geometry=TorusBufferGeometry(  
    radius=20,  
    tube=5,  
    radialSegments=6,  
    tubularSegments=20,  
    arc=2.0*pi  
))
```

```
Preview(child=WireframeGeometry(geometry=TorusBufferGeometry(radialSegments=6, radius=20.0, tube=5.0),
```

```
In [ ]:
```

2.4.2 Animation

```
In [1]: from pythreejs import *  
import ipywidgets  
from IPython.display import display
```

```
In [2]: view_width = 600  
view_height = 400
```

Let's first set up a basic scene with a cube and a sphere,

```
In [3]: sphere = Mesh(  
    SphereBufferGeometry(1, 32, 16),  
    MeshStandardMaterial(color='red')  
)
```

```
In [4]: cube = Mesh(  
    BoxBufferGeometry(1, 1, 1),  
    MeshPhysicalMaterial(color='green'),
```

```
        position=[2, 0, 4]
    )
```

as well as lighting and camera:

```
In [5]: camera = PerspectiveCamera( position=[10, 6, 10], aspect=view_width/view_height)
        key_light = DirectionalLight(position=[0, 10, 10])
        ambient_light = AmbientLight()
```

Keyframe animation

The three.js animation system is built as a [keyframe](#) system. We'll demonstrate this by animating the position and rotation of our camera.

First, we set up the keyframes for the position and the rotation separately:

```
In [6]: positon_track = VectorKeyframeTrack(name='.position',
        times=[0, 2, 5],
        values=[10, 6, 10,
                6.3, 3.78, 6.3,
                -2.98, 0.84, 9.2,
                ])
        rotation_track = QuaternionKeyframeTrack(name='.quaternion',
        times=[0, 2, 5],
        values=[-0.184, 0.375, 0.0762, 0.905,
                -0.184, 0.375, 0.0762, 0.905,
                -0.0430, -0.156, -0.00681, 0.987,
                ])
```

Next, we create an animation clip combining the two tracks, and finally an animation action to control the animation. See the three.js docs for more details on the different responsibilities of the different classes.

```
In [7]: camera_clip = AnimationClip(tracks=[positon_track, rotation_track])
        camera_action = AnimationAction(AnimationMixer(camera), camera_clip, camera)
```

Now, let's see it in action:

```
In [8]: scene = Scene(children=[sphere, cube, camera, key_light, ambient_light])
        controller = OrbitControls(controlling=camera)
        renderer = Renderer(camera=camera, scene=scene, controls=[controller],
                             width=view_width, height=view_height)
```

```
In [9]: renderer
```

```
Renderer(camera=PerspectiveCamera(aspect=1.5, position=(10.0, 6.0, 10.0), projectionMatrix=(1.4296712
```

```
In [10]: camera_action
```

```
AnimationAction(clip=AnimationClip(duration=5.0, tracks=(VectorKeyframeTrack(name='.position', times=
dtype=float32)), QuaternionKeyframeTrack(name='.quaternion', times=array([0, 2, 5]), values=array(
0.0762, 0.905, -0.043, -0.156, -0.00681, 0.987),
dtype=float32))), localRoot=PerspectiveCamera(aspect=1.5, position=(10.0, 6.0, 10.0), project
```

Let's add another animation clip, this time animating the color of the sphere's material:

```
In [11]: color_track = ColorKeyframeTrack(name='.material.color',
        times=[0, 1], values=[1, 0, 0, 0, 0, 1]) # red to blue

        color_clip = AnimationClip(tracks=[color_track], duration=1.5)
        color_action = AnimationAction(AnimationMixer(sphere), color_clip, sphere)
```

```
In [12]: color_action
```

```
AnimationAction(clip=AnimationClip(duration=1.5, tracks=(ColorKeyframeTrack(name='.material.color', t
```

Note how the two animation clips can freely be combined since they affect different properties. It's also worth noting that the color animation can be combined with manual camera control, while the camera animation cannot. When animating the camera, you might want to consider disabling the manual controls.

Animating rotation

When animating the camera rotation above, we used the camera's quaternion. This is the most robust method for animating free-form rotations. For example, the animation above was created by first moving the camera manually, and then reading out its position and quaternion properties at the wanted views. If you want more intuitive axes control, it is possible to animate the rotation sub-attributes instead, as shown below.

```
In [13]: f = """
function f(origu,origv) {
  // scale u and v to the ranges I want: [0, 2*pi]
  var u = 2*Math.PI*origu;
  var v = 2*Math.PI*origv;

  var x = Math.sin(u);
  var y = Math.cos(v);
  var z = Math.cos(u+v);

  return new THREE.Vector3(x,y,z)
}
"""
surf_g = ParametricGeometry(func=f, slices=16, stacks=16);

surf1 = Mesh(geometry=surf_g,
              material=MeshLambertMaterial(color='green', side='FrontSide'))
surf2 = Mesh(geometry=surf_g,
              material=MeshLambertMaterial(color='yellow', side='BackSide'))
surf = Group(children=[surf1, surf2])

camera2 = PerspectiveCamera( position=[10, 6, 10], aspect=view_width/view_height)
scene2 = Scene(children=[surf, camera2,
                        DirectionalLight(position=[3, 5, 1], intensity=0.6),
                        AmbientLight(intensity=0.5)])
renderer2 = Renderer(camera=camera2, scene=scene2,
                    controls=[OrbitControls(controlling=camera2)],
                    width=view_width, height=view_height)
display(renderer2)

Renderer(camera=PerspectiveCamera(aspect=1.5, position=(10.0, 6.0, 10.0), quaternion=(0.0, 0.0, 0.0, 1.0)))

In [14]: spin_track = NumberKeyframeTrack(name='.rotation[y]', times=[0, 2], values=[0, 6.28])
spin_clip = AnimationClip(tracks=[spin_track])
spin_action = AnimationAction(AnimationMixer(surf), spin_clip, surf)
spin_action
```

```
AnimationAction(clip=AnimationClip(tracks=(NumberKeyframeTrack(name='.rotation[y]', times=array([0, 2]), values=array([0, 6.28]))), mixer=AnimationMixer(surf)))
```

Note that we are spinning the object itself, and that we are therefore free to manipulate the camera at will.

Morph targets

Set up a simple sphere geometry, and add a morph target that is an oblong pill shape:

```
In [15]: # This lets three.js create the geometry, then syncs back vertex positions etc.
# For this reason, you should allow for the sync to complete before executing
```



```
# the next cell.
morph = BufferGeometry.from_geometry(SphereBufferGeometry(1, 32, 16))
```

In [16]: `import numpy as np`

```
# Set up morph targets:
vertices = np.array(morph.attributes['position'].array)
for i in range(len(vertices)):
    if vertices[i, 0] > 0:
        vertices[i, 0] += 1
morph.morphAttributes = {'position': [
    BufferAttribute(vertices),
]}

morphMesh = Mesh(morph, MeshPhongMaterial(
    color='#ff3333', shininess=150, morphTargets=True))
```

Set up animation for going back and forth between the sphere and pill shape:

```
In [17]: pill_track = NumberKeyframeTrack(
    name='.morphTargetInfluences[0]', times=[0, 1.5, 3], values=[0, 2.5, 0])
pill_clip = AnimationClip(tracks=[pill_track])
pill_action = AnimationAction(AnimationMixer(morphMesh), pill_clip, morphMesh)
```

```
In [18]: camera3 = PerspectiveCamera( position=[5, 3, 5], aspect=view_width/view_height)
scene3 = Scene(children=[morphMesh, camera3,
    DirectionalLight(position=[3, 5, 1], intensity=0.6),
    AmbientLight(intensity=0.5)])
renderer3 = Renderer(camera=camera3, scene=scene3,
    controls=[OrbitControls(controlling=camera3)],
    width=view_width, height=view_height)
display(renderer3, pill_action)
```

```
Renderer(camera=PerspectiveCamera(aspect=1.5, position=(5.0, 3.0, 5.0), quaternion=(0.0, 0.0, 0.0, 1.0))
AnimationAction(clip=AnimationClip(duration=3.0, tracks=(NumberKeyframeTrack(name='.morphTargetInfluences[0]', times=[0, 1.5, 3], values=[0, 2.5, 0])))
```

Skeletal animation

First, set up a skinned mesh with some bones:

In [19]: `import numpy as np`

```
N_BONES = 3

ref_cylinder = CylinderBufferGeometry(5, 5, 50, 5, N_BONES * 5, True)
cylinder = BufferGeometry.from_geometry(ref_cylinder)
```

```
In [20]: skinIndices = []
skinWeights = []
vertices = cylinder.attributes['position'].array
boneHeight = ref_cylinder.height / (N_BONES - 1)
for i in range(vertices.shape[0]):

    y = vertices[i, 1] + 0.5 * ref_cylinder.height

    skinIndex = y // boneHeight
    skinWeight = ( y % boneHeight ) / boneHeight

    # Ease between each bone
    skinIndices.append([skinIndex, skinIndex + 1, 0, 0 ])
```

```
skinWeights.append([1 - skinWeight, skinWeight, 0, 0 ])

cylinder.attributes = dict(
    cylinder.attributes,
    skinIndex=BufferAttribute(skinIndices),
    skinWeight=BufferAttribute(skinWeights),
)

shoulder = Bone(position=(0, -25, 0))
elbow = Bone(position=(0, 25, 0))
hand = Bone(position=(0, 25, 0))

shoulder.add(elbow)
elbow.add(hand)
bones = [shoulder, elbow, hand]
skeleton = Skeleton(bones)

mesh = SkinnedMesh(cylinder, MeshPhongMaterial(side='DoubleSide', skinning=True))
mesh.add(bones[0])
mesh.skeleton = skeleton
```

```
In [21]: helper = SkeletonHelper(mesh)
```

Next, set up some simple rotation animations for the bones:

```
In [22]: # Rotate on x and z axes:
```

```
bend_tracks = [
    NumberKeyframeTrack(
        name='.bones[1].rotation[x]',
        times=[0, 0.5, 1.5, 2],
        values=[0, 0.3, -0.3, 0]),
    NumberKeyframeTrack(
        name='.bones[1].rotation[z]',
        times=[0, 0.5, 1.5, 2],
        values=[0, 0.3, -0.3, 0]),
    NumberKeyframeTrack(
        name='.bones[2].rotation[x]',
        times=[0, 0.5, 1.5, 2],
        values=[0, -0.3, 0.3, 0]),
    NumberKeyframeTrack(
        name='.bones[2].rotation[z]',
        times=[0, 0.5, 1.5, 2],
        values=[0, -0.3, 0.3, 0]),
]
bend_clip = AnimationClip(tracks=bend_tracks)
bend_action = AnimationAction(AnimationMixer(mesh), bend_clip, mesh)
```

```
# Rotate on y axis:
```

```
wring_tracks = [
    NumberKeyframeTrack(name='.bones[1].rotation[y]', times=[0, 0.5, 1.5, 2], values=[0, 0.7, -0.7, 0]),
    NumberKeyframeTrack(name='.bones[2].rotation[y]', times=[0, 0.5, 1.5, 2], values=[0, 0.7, -0.7, 0]),
]

wring_clip = AnimationClip(tracks=wring_tracks)
wring_action = AnimationAction(AnimationMixer(mesh), wring_clip, mesh)
```

```
In [23]: camera4 = PerspectiveCamera( position=[40, 24, 40], aspect=view_width/view_height)
scene4 = Scene(children=[mesh, helper, camera4,
    DirectionalLight(position=[3, 5, 1], intensity=0.6),
    AmbientLight(intensity=0.5)])
```

```

    renderer4 = Renderer(camera=camera4, scene=scene4,
                        controls=[OrbitControls(controlling=camera4)],
                        width=view_width, height=view_height)
    display(renderer4)

```

```
Renderer(camera=PerspectiveCamera(aspect=1.5, position=(40.0, 24.0, 40.0), quaternion=(0.0, 0.0, 0.0,
```

```
In [24]: bend_action
```

```
AnimationAction(clip=AnimationClip(duration=2.0, tracks=(NumberKeyframeTrack(name='.bones[1].rotation
```

```
In [25]: wring_action
```

```
AnimationAction(clip=AnimationClip(duration=2.0, tracks=(NumberKeyframeTrack(name='.bones[1].rotation
```

```
In [ ]:
```

2.4.3 Textures

```
In [1]: from pythreejs import *
        from IPython.display import display
        from math import pi
```

```
In [2]: checker_tex = ImageTexture(imageUri='img/checkerboard.png')
        earth_tex = ImageTexture(imageUri='img/earth.jpg')
```

```
In [3]: checker_tex
```

```
Preview(child=ImageTexture(imageUri='img/checkerboard.png', repeat=(1.0, 1.0), version=1), shadowMap=WebGLSL
```

```
In [4]: earth_tex
```

```
Preview(child=ImageTexture(imageUri='img/earth.jpg', repeat=(1.0, 1.0), version=1), shadowMap=WebGLSL
```

```
In [5]: #
        # Create checkerboard pattern
        #
        # tex dims need to be power of two.
        arr_w = 256
        arr_h = 256
        import numpy as np
        def gen_checkers(width, height, n_checkers_x, n_checkers_y):
            array = np.ones((width, height, 3), dtype='float32')
            # width in texels of each checker
            checker_w = width / n_checkers_x
            checker_h = height / n_checkers_y
            for y in range(arr_h):
                for x in range(arr_w):
                    color_key = int(x / checker_w) + int(y / checker_h)
                    if color_key % 2 == 0:
                        array[x, y, :] = [ 0, 0, 0 ]
                    else:
                        array[x, y, :] = [ 1, 1, 1 ]
            return array

```

```
data_tex = DataTexture(
```

```

        data=gen_checkers(arr_w, arr_h, 4, 4),
        format="RGBFormat",
        type="FloatType"
    )

```

In [6]: data_tex

```

Preview(child=DataTexture(data=array([[0., 0., 0.],
    [0., 0., 0.],
    [0., 0., 0.],
    ...,
    [1., 1., 1.],
    [1., 1., 1.],
    [1., 1., 1.]],

    [[0., 0., 0.],
    [0., 0., 0.],
    [0., 0., 0.],
    ...,
    [1., 1., 1.],
    [1., 1., 1.],
    [1., 1., 1.]],

    [[0., 0., 0.],
    [0., 0., 0.],
    [0., 0., 0.],
    ...,
    [1., 1., 1.],
    [1., 1., 1.],
    [1., 1., 1.]],

    ...,

    [[1., 1., 1.],
    [1., 1., 1.],
    [1., 1., 1.],
    ...,
    [0., 0., 0.],
    [0., 0., 0.],
    [0., 0., 0.]],

    [[1., 1., 1.],
    [1., 1., 1.],
    [1., 1., 1.],
    ...,
    [0., 0., 0.],
    [0., 0., 0.],
    [0., 0., 0.]],

    [[1., 1., 1.],
    [1., 1., 1.],
    [1., 1., 1.],
    ...,
    [0., 0., 0.],
    [0., 0., 0.],
    [0., 0., 0.]]], dtype=float32), format='RGBFormat', repeat=(1.0, 1.0), type='FloatType', vers

```

In [7]: data_tex.data = gen_checkers(arr_w, arr_h, 12, 20)

2.4.4 Renderer properties

```
In [1]: from pythreejs import *
        from IPython.display import display
        import ipywidgets
```

Transparent background

To have the render view use a transparent background, there are three steps you need to do: 1. Ensure that the background property of the Scene object is set to None. 2. Ensure that alpha=True is passed to the constructor of the Renderer object. This ensures that an alpha channel is used by the renderer. 3. Ensure that the clearOpacity property of the Renderer object is set to 0. For more details about this, see below.

```
In [2]: ball = Mesh(geometry=SphereGeometry(),
                    material=MeshLambertMaterial(color='red'))
        key_light = DirectionalLight(color='white', position=[3, 5, 1], intensity=0.5)

        c = PerspectiveCamera(position=[0, 5, 5], up=[0, 1, 0], children=[key_light])

        scene = Scene(children=[ball, c, AmbientLight(color='#777777')], background=None)

        renderer = Renderer(camera=c,
                             scene=scene,
                             alpha=True,
                             clearOpacity=0,
                             controls=[OrbitControls(controlling=c)])

        display(renderer)
```

```
Renderer(camera=PerspectiveCamera(children=(DirectionalLight(color='white', intensity=0.5, position=
```

The use of clear color/opacity is explained in more detailed in the docs of three.js, but in short: - If autoClear is true the renderer output is cleared on each rendered frame. - If autoClearColor is true the background color is cleared on each frame. - When the background color is cleared, it is reset to Renderer.clearColor, with an opacity of Renderer.clearOpacity.

```
In [3]: # Let's set up some controls for the clear color/opacity:
```

```
opacity = ipywidgets.FloatSlider(min=0., max=1.)
ipywidgets.jslink((opacity, 'value'), (renderer, 'clearOpacity'))

color = ipywidgets.ColorPicker()
ipywidgets.jslink((color, 'value'), (renderer, 'clearColor'))

display(ipywidgets.HBox(children=[
    ipywidgets.Label('Clear color:'), color, ipywidgets.Label('Clear opacity:'), opacity]))
```

```
HBox(children=(Label(value='Clear color:'), ColorPicker(value='black'), Label(value='Clear opacity:')
```

Scene background

If we set the background property of the scene, it will be filled in on top of whatever clear color is there, basically making the clear color ineffective.

```
In [4]: scene_background = ipywidgets.ColorPicker()
        _background_link = None

        def toggle_scene_background(change):
            global _background_link
```

```

    if change['new']:
        _background_link = ipywidgets.jslink((scene_background, 'value'), (scene, 'background'))
    else:
        _background_link.close()
        _background_link = None
        scene.background = None

scene_background_toggle = ipywidgets.ToggleButton(False, description='Scene Color')
scene_background_toggle.observe(toggle_scene_background, 'value')

display(ipywidgets.HBox(children=[
    ipywidgets.Label('Scene background color:'), scene_background, scene_background_toggle]))
HBox(children=(Label(value='Scene background color:'), ColorPicker(value='black'), ToggleButton(value='Scene Color')))
In [ ]:

```

2.5 API Reference

The pythreejs API attempts to mimic the [three.js API](#) as closely as possible. This API reference therefore does not attempt to explain the purpose of any forwarded objects or attributes, but can still be useful for:

- The trait signatures of various properties.
- Classes, properties and methods custom to pythreejs.
- Variations from the three.js API, e.g. for *BufferAttribute*.

2.5.1 `_base`

Preview

```

class pythreejs.Preview(child, **kwargs)
    Bases: pythreejs._base.renderable.RenderableWidget

    child = Instance()
        a ThreeWidget

```

RenderableWidget

```

class pythreejs.RenderableWidget(**kwargs)
    Bases: ipywidgets.widgets.domwidget.DOMWidget

    autoClear = Bool(True)
        A boolean (True, False) trait.

    autoClearColor = Bool(True)
        A boolean (True, False) trait.

    autoClearDepth = Bool(True)
        A boolean (True, False) trait.

    autoClearStencil = Bool(True)
        A boolean (True, False) trait.

    clearColor = Unicode('#000000')
        A trait for unicode strings.

```

clearOpacity = CFloat(1.0)
A casting version of the float trait.

clippingPlanes = List()
An instance of a Python list.

freeze()

gammaFactor = CFloat(2.0)
A casting version of the float trait.

gammaInput = Bool(False)
A boolean (True, False) trait.

gammaOutput = Bool(False)
A boolean (True, False) trait.

localClippingEnabled = Bool(False)
A boolean (True, False) trait.

log(msg)
A trait whose value must be an instance of a specified class.
The value can also be an instance of a subclass of the specified class.
Subclasses can declare default classes by overriding the class attribute

maxMorphNormals = CInt(4)
A casting version of the int trait.

maxMorphTargets = CInt(8)
A casting version of the int trait.

physicallyCorrectLights = Bool(False)
A boolean (True, False) trait.

send_msg(message_type, payload=None)

shadowMap = Instance()
A trait whose value must be an instance of a specified class.
The value can also be an instance of a subclass of the specified class.
Subclasses can declare default classes by overriding the class attribute

sortObject = Bool(True)
A boolean (True, False) trait.

toneMapping = Enum('LinearToneMapping')
An enum whose value must be in a given sequence.

toneMappingExposure = CFloat(1.0)
A casting version of the float trait.

toneMappingWhitePoint = CFloat(1.0)
A casting version of the float trait.

ThreeWidget

```
class pythreejs.ThreeWidget(**kwargs)
    Bases: ipywidgets.widgets.widget.Widget
    Base widget type for all pythreejs widgets
```

exec_three_obj_method (*method_name*, **args*, ***kwargs*)

Execute a method on the three object.

Excute the method specified by *method_name* on the three object, with arguments *args*. *kwargs* is currently ignored.

2.5.2 animation

tracks

BooleanKeyframeTrack

class pythreejs.**BooleanKeyframeTrack** (*name*="", *times*=None, *values*=None, *interpolation*="InterpolateLinear")

BooleanKeyframeTrack

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/animation/tracks/BooleanKeyframeTrack>

Inherits *KeyframeTrack*.

Three.js docs: <https://threejs.org/docs/#api/animation/tracks/BooleanKeyframeTrack>

ColorKeyframeTrack

class pythreejs.**ColorKeyframeTrack** (*name*="", *times*=None, *values*=None, *interpolation*="InterpolateLinear")

ColorKeyframeTrack

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/animation/tracks/ColorKeyframeTrack>

Inherits *KeyframeTrack*.

Three.js docs: <https://threejs.org/docs/#api/animation/tracks/ColorKeyframeTrack>

NumberKeyframeTrack

class pythreejs.**NumberKeyframeTrack** (*name*="", *times*=None, *values*=None, *interpolation*="InterpolateLinear")

NumberKeyframeTrack

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/animation/tracks/NumberKeyframeTrack>

Inherits *KeyframeTrack*.

Three.js docs: <https://threejs.org/docs/#api/animation/tracks/NumberKeyframeTrack>

QuaternionKeyframeTrack

class pythreejs.**QuaternionKeyframeTrack** (*name*="", *times*=None, *values*=None, *interpolation*="InterpolateLinear")

QuaternionKeyframeTrack

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/animation/tracks/QuaternionKeyframeTrack>

Inherits *KeyframeTrack*.

Three.js docs: <https://threejs.org/docs/#api/animation/tracks/QuaternionKeyframeTrack>

StringKeyframeTrack

```
class pythreejs.StringKeyframeTrack (name="", times=None, values=None, interpolation="InterpolateLinear")
```

StringKeyframeTrack

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/animation/tracks/StringKeyframeTrack>

Inherits *KeyframeTrack*.

Three.js docs: <https://threejs.org/docs/#api/animation/tracks/StringKeyframeTrack>

VectorKeyframeTrack

```
class pythreejs.VectorKeyframeTrack (name="", times=None, values=None, interpolation="InterpolateLinear")
```

VectorKeyframeTrack

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/animation/tracks/VectorKeyframeTrack>

Inherits *KeyframeTrack*.

Three.js docs: <https://threejs.org/docs/#api/animation/tracks/VectorKeyframeTrack>

AnimationAction

```
class pythreejs.AnimationAction (mixer=None, clip=None, localRoot=None)
```

AnimationAction is a three widget that also has its own view.

The view offers animation action controls.

This widget has some manual overrides on the Python side.

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/animation/AnimationAction>

mixer

```
Instance(AnimationMixer, allow_none=True).tag(sync=True, **widget_
↪ serialization)
```

clip

```
Instance(AnimationClip, allow_none=True).tag(sync=True, **widget_
↪ serialization)
```

localRoot

```
Instance(ThreeWidget, allow_none=True).tag(sync=True, **widget_serialization)
```

clampWhenFinished

```
Bool(False, allow_none=False).tag(sync=True)
```

enabled

```
Bool(True, allow_none=False).tag(sync=True)
```

loop

```
Enum(LoopModes, "LoopRepeat", allow_none=False).tag(sync=True)
```

paused

```
Bool(False, allow_none=False).tag(sync=True)
```

reiterations

```
CInt(float('inf'), allow_none=False).tag(sync=True)
```

time

```
CFloat(0, allow_none=False).tag(sync=True)
```

timeScale

```
CFloat(1, allow_none=False).tag(sync=True)
```

weight

```
CFloat(1, allow_none=False).tag(sync=True)
```

zeroSlopeAtEnd

```
Bool(True, allow_none=False).tag(sync=True)
```

zeroSlopeAtStart

```
Bool(True, allow_none=False).tag(sync=True)
```

reiterations = Union(int)

an int or a float

AnimationClip

class pythreejs.**AnimationClip** (*name=None, duration=-1, tracks=[]*)
 AnimationClip

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/animation/AnimationClip>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/animation/AnimationClip>

name

```
Unicode(None, allow_none=True).tag(sync=True)
```

duration

```
CFloat(-1, allow_none=False).tag(sync=True)
```

tracks

```
Tuple().tag(sync=True, **widget_serialization)
```

duration = CFloat(-1)
 a float

name = Unicode(None)
 a unicode string

tracks = Tuple()
 a tuple of any type

AnimationMixer

class pythreejs.**AnimationMixer** (*rootObject=None, time=0, timeScale=1*)
 AnimationMixer

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/animation/AnimationMixer>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/animation/AnimationMixer>

rootObject

```
Instance(ThreeWidget, allow_none=True).tag(sync=True, **widget_serialization)
```

time

```
CFloat(0, allow_none=False).tag(sync=True)
```

timeScale

```
CFloat(1, allow_none=False).tag(sync=True)
```

rootObject = Instance()

a ThreeWidget or None

time = CFloat(0)

a float

timeScale = CFloat(1)

a float

AnimationObjectGroup

class pythreejs.AnimationObjectGroup

AnimationObjectGroup

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/animation/AnimationObjectGroup>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/animation/AnimationObjectGroup>

AnimationUtils

class pythreejs.AnimationUtils

AnimationUtils

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/animation/AnimationUtils>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/animation/AnimationUtils>

KeyframeTrack

class pythreejs.KeyframeTrack (*name=""*, *times=None*, *values=None*, *interpolation="InterpolateLinear"*)

KeyframeTrack

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/animation/KeyframeTrack>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/animation/KeyframeTrack>

name

```
Unicode('', allow_none=False).tag(sync=True)
```

times

```
WebGLDataUnion().tag(sync=True)
```

values

```
WebGLDataUnion().tag(sync=True)
```

interpolation

```
Enum(InterpolationModes, "InterpolateLinear", allow_none=False).tag(sync=True)
```

```
interpolation = Enum('InterpolateLinear')
    any of ['InterpolateDiscrete', 'InterpolateLinear', 'InterpolateSmooth']
```

```
name = Unicode('')
    a unicode string
```

```
times = WebGLDataUnion()
    a numpy array or a NDArrayBase
```

```
values = WebGLDataUnion()
    a numpy array or a NDArrayBase
```

PropertyBinding

```
class pythreejs.PropertyBinding
    PropertyBinding
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/animation/PropertyBinding>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/animation/PropertyBinding>

PropertyMixer

```
class pythreejs.PropertyMixer
    PropertyMixer
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/animation/PropertyMixer>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/animation/PropertyMixer>

2.5.3 audio

AudioAnalyser

```
class pythreejs.AudioAnalyser
    AudioAnalyser
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/audio/AudioAnalyser>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/audio/AudioAnalyser>

AudioListener

class pythreejs.**AudioListener**
AudioListener

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/audio/AudioListener>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/audio/AudioListener>

Audio

class pythreejs.**Audio**
Audio

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/audio/Audio>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/audio/Audio>

PositionalAudio

class pythreejs.**PositionalAudio**
PositionalAudio

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/audio/PositionalAudio>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/audio/PositionalAudio>

2.5.4 cameras

ArrayCamera

class pythreejs.**ArrayCamera** (*fov=50, aspect=1, near=0.1, far=2000*)
ArrayCamera

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/cameras/ArrayCamera>

Inherits *PerspectiveCamera*.

Three.js docs: <https://threejs.org/docs/#api/cameras/ArrayCamera>

type

```
Unicode("ArrayCamera", allow_none=False).tag(sync=True)
```

type = Unicode('ArrayCamera')
a unicode string

Camera

class pythreejs.Camera

Camera

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/cameras/Camera>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/cameras/Camera>

matrixWorldInverse

```
Matrix4(default_value=[1,0,0,0,0,1,0,0,0,0,1,0,0,0,0,1]).tag(sync=True)
```

projectionMatrix

```
Matrix4(default_value=[1,0,0,0,0,1,0,0,0,0,1,0,0,0,0,1]).tag(sync=True)
```

type

```
Unicode("Camera", allow_none=False).tag(sync=True)
```

matrixWorldInverse = **Matrix4**((1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1))
a tuple of any type

projectionMatrix = **Matrix4**((1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1))
a tuple of any type

type = **Unicode**('Camera')
a unicode string

CombinedCamera

class pythreejs.CombinedCamera (*width=0, height=0, fov=50, near=0.1, far=2000, orthoNear=0.1, orthoFar=2000*)

CombinedCamera

Autogenerated by generate-wrappers.js This class is a custom class for pythreejs, with no direct corresponding class in three.js.

Inherits *Camera*.

Three.js docs: <https://threejs.org/docs/#api/cameras/CombinedCamera>

fov

```
CFloat(50, allow_none=False).tag(sync=True)
```

zoom

```
CFloat(1, allow_none=False).tag(sync=True)
```

near

```
CFloat(0.1, allow_none=False).tag(sync=True)
```

far

```
CFloat(2000, allow_none=False).tag(sync=True)
```

orthoNear

```
CFloat(0.1, allow_none=False).tag(sync=True)
```

orthoFar

```
CFloat(2000, allow_none=False).tag(sync=True)
```

width

```
CFloat(0, allow_none=False).tag(sync=True)
```

height

```
CFloat(0, allow_none=False).tag(sync=True)
```

mode

```
Enum(['perspective', 'orthographic'], "perspective", allow_none=False).  
tag(sync=True)
```

impersonate

```
Bool(True, allow_none=False).tag(sync=True)
```

type

```
Unicode("CombinedCamera", allow_none=False).tag(sync=True)
```

far = CFloat(2000)
a float

fov = CFloat(50)
a float

height = CFloat(0)
a float

impersonate = Bool(True)
a boolean

mode = Enum('perspective')
any of ['perspective', 'orthographic']


```

near = CFloat(0.1)
    a float

orthoFar = CFloat(2000)
    a float

orthoNear = CFloat(0.1)
    a float

type = Unicode('CombinedCamera')
    a unicode string

width = CFloat(0)
    a float

zoom = CFloat(1)
    a float

```

CubeCamera

```

class pythreejs.CubeCamera
    CubeCamera

```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/cameras/CubeCamera>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/cameras/CubeCamera>

type

```
Unicode("CubeCamera", allow_none=False).tag(sync=True)
```

```

type = Unicode('CubeCamera')
    a unicode string

```

OrthographicCamera

```

class pythreejs.OrthographicCamera (left=0, right=0, top=0, bottom=0, near=0.1, far=2000)
    OrthographicCamera

```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/cameras/OrthographicCamera>

Inherits *Camera*.

Three.js docs: <https://threejs.org/docs/#api/cameras/OrthographicCamera>

zoom

```
CFloat(1, allow_none=False).tag(sync=True)
```

left

```
CFloat(0, allow_none=False).tag(sync=True)
```

right

```
CFloat(0, allow_none=False).tag(sync=True)
```

top

```
CFloat(0, allow_none=False).tag(sync=True)
```

bottom

```
CFloat(0, allow_none=False).tag(sync=True)
```

near

```
CFloat(0.1, allow_none=False).tag(sync=True)
```

far

```
CFloat(2000, allow_none=False).tag(sync=True)
```

type

```
Unicode("OrthographicCamera", allow_none=False).tag(sync=True)
```

bottom = CFloat(0)
a float

far = CFloat(2000)
a float

left = CFloat(0)
a float

near = CFloat(0.1)
a float

right = CFloat(0)
a float

top = CFloat(0)
a float

type = Unicode('OrthographicCamera')
a unicode string

zoom = CFloat(1)
a float

PerspectiveCamera

class pythreejs.**PerspectiveCamera** (*fov=50, aspect=1, near=0.1, far=2000*)
PerspectiveCamera

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/cameras/PerspectiveCamera>

Inherits *Camera*.

Three.js docs: <https://threejs.org/docs/#api/cameras/PerspectiveCamera>

fov

```
CFloat(50, allow_none=False).tag(sync=True)
```

zoom

```
CFloat(1, allow_none=False).tag(sync=True)
```

near

```
CFloat(0.1, allow_none=False).tag(sync=True)
```

far

```
CFloat(2000, allow_none=False).tag(sync=True)
```

focus

```
CFloat(10, allow_none=False).tag(sync=True)
```

aspect

```
CFloat(1, allow_none=False).tag(sync=True)
```

type

```
Unicode("PerspectiveCamera", allow_none=False).tag(sync=True)
```

```
aspect = CFloat(1)  
a float
```

```
far = CFloat(2000)  
a float
```

```
focus = CFloat(10)  
a float
```

```
fov = CFloat(50)  
a float
```

```
near = CFloat(0.1)  
a float
```

```
type = Unicode('PerspectiveCamera')  
a unicode string
```

```
zoom = CFloat(1)  
a float
```

StereoCamera

class pythreejs.StereoCamera
StereoCamera

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/cameras/StereoCamera>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/cameras/StereoCamera>

aspect

```
CFloat(1, allow_none=False).tag(sync=True)
```

eyeSep

```
CFloat(0.064, allow_none=False).tag(sync=True)
```

cameraL

```
Instance(PerspectiveCamera, allow_none=True).tag(sync=True, **widget_
↪serialization)
```

cameraR

```
Instance(PerspectiveCamera, allow_none=True).tag(sync=True, **widget_
↪serialization)
```

aspect = CFloat(1)
a float

cameraL = Instance()
a PerspectiveCamera or None

cameraR = Instance()
a PerspectiveCamera or None

eyeSep = CFloat(0.064)
a float

2.5.5 controls

Controls

class pythreejs.Controls
Controls

Autogenerated by generate-wrappers.js This class is a custom class for pythreejs, with no direct corresponding class in three.js.

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/controls/Controls>

controlling

```
Instance(Object3D, allow_none=False).tag(sync=True, **widget_serialization)
```

```
controlling = Instance()
    an Object3D
```

FlyControls

```
class pythreejs.FlyControls (controlling=None)
```

```
    FlyControls
```

Autogenerated by generate-wrappers.js This class is a custom class for pythreejs, with no direct corresponding class in three.js.

Inherits *Controls*.

Three.js docs: <https://threejs.org/docs/#api/controls/FlyControls>

moveVector

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

rotationVector

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

movementSpeed

```
CFloat(1, allow_none=False).tag(sync=True)
```

rollSpeed

```
CFloat(0.05, allow_none=False).tag(sync=True)
```

syncRate

```
CFloat(1, allow_none=False).tag(sync=True)
```

```
moveVector = Vector3((0, 0, 0))
    a tuple of any type
```

```
movementSpeed = CFloat(1)
    a float
```

```
rollSpeed = CFloat(0.05)
    a float
```

```
rotationVector = Vector3((0, 0, 0))
    a tuple of any type
```

```
syncRate = CFloat(1)
    a float
```

OrbitControls

class pythreejs.OrbitControls (*controlling=None*)
OrbitControls

Autogenerated by generate-wrappers.js This class is a custom class for pythreejs, with no direct corresponding class in three.js.

Inherits *Controls*.

Three.js docs: <https://threejs.org/docs/#api/controls/OrbitControls>

enabled

```
Bool(True, allow_none=False).tag(sync=True)
```

minDistance

```
CFloat(0, allow_none=False).tag(sync=True)
```

maxDistance

```
CFloat(float('inf'), allow_none=False).tag(sync=True)
```

minZoom

```
CFloat(0, allow_none=False).tag(sync=True)
```

maxZoom

```
CFloat(float('inf'), allow_none=False).tag(sync=True)
```

minPolarAngle

```
CFloat(0, allow_none=False).tag(sync=True)
```

maxPolarAngle

```
CFloat(3.141592653589793, allow_none=False).tag(sync=True)
```

minAzimuthAngle

```
CFloat(-float('inf'), allow_none=False).tag(sync=True)
```

maxAzimuthAngle

```
CFloat(float('inf'), allow_none=False).tag(sync=True)
```

enableDamping

```
Bool(False, allow_none=False).tag(sync=True)
```

dampingFactor

```
CFloat(0.25, allow_none=False).tag(sync=True)
```

enableZoom

```
Bool(True, allow_none=False).tag(sync=True)
```

zoomSpeed

```
CFloat(1, allow_none=False).tag(sync=True)
```

enableRotate

```
Bool(True, allow_none=False).tag(sync=True)
```

rotateSpeed

```
CFloat(1, allow_none=False).tag(sync=True)
```

enablePan

```
Bool(True, allow_none=False).tag(sync=True)
```

keyPanSpeed

```
CFloat(7, allow_none=False).tag(sync=True)
```

autoRotate

```
Bool(False, allow_none=False).tag(sync=True)
```

autoRotateSpeed

```
CFloat(2, allow_none=False).tag(sync=True)
```

enableKeys

```
Bool(True, allow_none=False).tag(sync=True)
```

target

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

autoRotate = Bool(False)
a boolean

autoRotateSpeed = CFloat(2)
a float

dampingFactor = CFloat(0.25)
a float

enableDamping = Bool(False)
a boolean

enableKeys = Bool(True)
a boolean

enablePan = Bool(True)
a boolean

enableRotate = Bool(True)
a boolean

enableZoom = Bool(True)
a boolean

enabled = Bool(True)
a boolean

keyPanSpeed = CFloat(7)
a float

maxAzimuthAngle = CFloat(inf)
a float

maxDistance = CFloat(inf)
a float

maxPolarAngle = CFloat(3.141592653589793)
a float

maxZoom = CFloat(inf)
a float

minAzimuthAngle = CFloat(-inf)
a float

minDistance = CFloat(0)
a float

minPolarAngle = CFloat(0)
a float

minZoom = CFloat(0)
a float

rotateSpeed = CFloat(1)
a float

target = Vector3((0, 0, 0))
a tuple of any type


```
zoomSpeed = CFloat(1)
    a float
```

Picker

```
class pythreejs.Picker (controlling=None)
    Picker
```

Autogenerated by generate-wrappers.js This class is a custom class for pythreejs, with no direct corresponding class in three.js.

Inherits *Controls*.

Three.js docs: <https://threejs.org/docs/#api/controls/Picker>

event

The DOM MouseEvent type to trigger the pick

```
Unicode("click", allow_none=False).tag(sync=True)
```

all

Wether to send info on all object intersections beneath the picked point, or only the first one. See *picked*.

```
Bool(False, allow_none=False).tag(sync=True)
```

distance

The distance from the camera of the picked point (null if no object picked)

```
CFloat(None, allow_none=True).tag(sync=True)
```

point

The coordinates of the picked point (all zero if no object picked)

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

face

The vertex indices of the picked face (all zero if no face picked)

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

faceNormal

The normal vector of the picked face (all zero if no face picked)

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

faceVertices

The three vertices that make up the picked face, as vectors (empty if no face picked)

```
List(trait=List()).tag(sync=True)
```

faceIndex

```
CInt(0, allow_none=False).tag(sync=True)
```

modifiers

The keyboard modifiers held at the pick event in the following order: [SHIFT, CTRL, ALT, META]

```
List().tag(sync=True)
```

object

The picked object (null if no object picked)

```
Instance(Object3D, allow_none=True).tag(sync=True, **widget_serialization)
```

picked

The other fields on the picker will always be for the first object intersection. If `all` is set true, this field will be an array containing the same information for all intersections.

```
List().tag(sync=True)
```

uv

The UV coordinate picked (all zero if invalid pick)

```
Vector2(default_value=[0,0]).tag(sync=True)
```

indices

The vertex indices of the picked face (empty if no face picked)

```
List().tag(sync=True)
```

all = Bool(False)

a boolean

distance = CFloat(None)

a float

event = Unicode('click')

a unicode string

face = Vector3((0, 0, 0))

a tuple of any type

faceIndex = CInt(0)

an int

faceNormal = Vector3((0, 0, 0))

a tuple of any type

faceVertices = List()

a list with values that are: a list

indices = List()

a list of any type

modifiers = List()

a list of any type

object = Instance()

an Object3D or None

picked = List()

a list of any type

point = Vector3((0, 0, 0))

a tuple of any type

uv = Vector2((0, 0))

a tuple of any type

TrackballControls

class pythreejs.**TrackballControls** (*controlling=None*)
TrackballControls

Autogenerated by generate-wrappers.js This class is a custom class for pythreejs, with no direct corresponding class in three.js.

Inherits *Controls*.

Three.js docs: <https://threejs.org/docs/#api/controls/TrackballControls>

enabled

```
Bool(True, allow_none=False).tag(sync=True)
```

minDistance

```
CFloat(0, allow_none=False).tag(sync=True)
```

maxDistance

```
CFloat(float('inf'), allow_none=False).tag(sync=True)
```

rotateSpeed

```
CFloat(1, allow_none=False).tag(sync=True)
```

zoomSpeed

```
CFloat(1.2, allow_none=False).tag(sync=True)
```

panSpeed

```
CFloat(0.3, allow_none=False).tag(sync=True)
```

staticMoving

```
Bool(False, allow_none=False).tag(sync=True)
```

dynamicDampingFactor

```
CFloat(0.2, allow_none=False).tag(sync=True)
```

noRotate

```
Bool(False, allow_none=False).tag(sync=True)
```

noZoom

```
Bool(False, allow_none=False).tag(sync=True)
```

noPan

```
Bool(False, allow_none=False).tag(sync=True)
```

noRoll

```
Bool(False, allow_none=False).tag(sync=True)
```

target

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

dynamicDampingFactor = CFloat(0.2)

a float

enabled = Bool(True)

a boolean

maxDistance = CFloat(inf)

a float

minDistance = CFloat(0)

a float

noPan = Bool(False)

a boolean

noRoll = Bool(False)

a boolean

noRotate = Bool(False)

a boolean

noZoom = Bool(False)

a boolean

panSpeed = CFloat(0.3)

a float

rotateSpeed = CFloat(1)

a float

staticMoving = Bool(False)

a boolean

target = Vector3((0, 0, 0))

a tuple of any type

zoomSpeed = CFloat(1.2)

a float

2.5.6 core

BaseBufferGeometry

class pythreejs.**BaseBufferGeometry**

BaseBufferGeometry

Autogenerated by generate-wrappers.js This class is a custom class for pythreejs, with no direct corresponding class in three.js.

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/core/BaseBufferGeometry>

name

```
Unicode('', allow_none=False).tag(sync=True)
```

type

```
Unicode("BaseBufferGeometry", allow_none=False).tag(sync=True)
```

name = Unicode('')

a unicode string

type = Unicode('BaseBufferGeometry')

a unicode string

BaseGeometry

class pythreejs.**BaseGeometry**

BaseGeometry

Autogenerated by generate-wrappers.js This class is a custom class for pythreejs, with no direct corresponding class in three.js.

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/core/BaseGeometry>

name

```
Unicode('', allow_none=False).tag(sync=True)
```

type

```
Unicode("BaseGeometry", allow_none=False).tag(sync=True)
```

name = Unicode('')

a unicode string

type = Unicode('BaseGeometry')

a unicode string

BufferAttribute

class pythreejs.**BufferAttribute** (*array=None, normalized=True*)

This widget has some manual overrides on the Python side.

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/core/BufferAttribute>

array

```
WebGLDataUnion().tag(sync=True)
```

dynamic

```
Bool(False, allow_none=False).tag(sync=True)
```

needsUpdate

```
Bool(False, allow_none=False).tag(sync=True)
```

normalized

```
Bool(True, allow_none=False).tag(sync=True)
```

version

```
CInt(-1, allow_none=False).tag(sync=True)
```

BufferGeometry

class pythreejs.**BufferGeometry**

This widget has some manual overrides on the Python side.

Inherits *BaseBufferGeometry*.

Three.js docs: <https://threejs.org/docs/#api/core/BufferGeometry>

index

```
Union([
    Instance(BufferAttribute, allow_none=True),
    Instance(InterleavedBufferAttribute, allow_none=True)
]).tag(sync=True, **widget_serialization)
```

attributes

```
Dict(Union([
    Instance(BufferAttribute),
    Instance(InterleavedBufferAttribute)
])).tag(sync=True, **widget_serialization)
```

morphAttributes

```
Dict(Tuple(Union([
    Instance(BufferAttribute),
    Instance(InterleavedBufferAttribute)
]))) .tag(sync=True, **widget_serialization)
```

MaxIndex

```
CInt(65535, allow_none=False) .tag(sync=True)
```

_ref_geometry

```
Union([
    Instance(BaseGeometry, allow_none=True),
    Instance(BaseBufferGeometry, allow_none=True)
]) .tag(sync=True, **widget_serialization)
```

_store_ref

```
Bool(False, allow_none=False) .tag(sync=True)
```

type

```
Unicode("BufferGeometry", allow_none=False) .tag(sync=True)
```

classmethod from_geometry (*geometry*, *store_ref=False*)

Creates a PlainBufferGeometry of another geometry.

store_ref determines if the reference is stored after initialization. If it is, it will be used for future embedding.

validate**Clock****class** pythreejs.Clock

Clock

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/core/Clock>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/core/Clock>

DirectGeometry**class** pythreejs.DirectGeometry

DirectGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/core/DirectGeometry>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/core/DirectGeometry>

EventDispatcher

class pythreejs.**EventDispatcher**
EventDispatcher

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/core/EventDispatcher>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/core/EventDispatcher>

Geometry

class pythreejs.**Geometry**

This widget has some manual overrides on the Python side.

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/core/Geometry>

vertices

```
List(trait=List()).tag(sync=True)
```

colors

```
List(trait=Unicode(), default_value=["#ffffff"]).tag(sync=True)
```

faces

```
Tuple(trait=Face3()).tag(sync=True)
```

faceVertexUvs

```
List().tag(sync=True)
```

lineDistances

```
List().tag(sync=True)
```

morphTargets

```
List().tag(sync=True)
```

morphNormals

```
List().tag(sync=True)
```

skinWeights


```
List (trait=List ()) .tag (sync=True)
```

skinIndices

```
List (trait=List ()) .tag (sync=True)
```

_ref_geometry

```
Instance (BaseGeometry, allow_none=True) .tag (sync=True, **widget_serialization)
```

_store_ref

```
Bool (False, allow_none=False) .tag (sync=True)
```

type

```
Unicode ("Geometry", allow_none=False) .tag (sync=True)
```

classmethod from_geometry (geometry, store_ref=False)

Creates a PlainGeometry of another geometry.

store_ref determines if the reference is stored after initialization. If it is, it will be used for future embedding.

NOTE: The PlainGeometry will copy the arrays from the source geometry. To avoid this, use PlainBufferGeometry.

InstancedBufferAttribute

class pythreejs.**InstancedBufferAttribute** (array=None, meshPerAttribute=1)

InstancedBufferAttribute

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/core/InstancedBufferAttribute>

Inherits *BufferAttribute*.

Three.js docs: <https://threejs.org/docs/#api/core/InstancedBufferAttribute>

meshPerAttribute

```
CInt (1, allow_none=False) .tag (sync=True)
```

meshPerAttribute = CInt (1)

an int

InstancedBufferGeometry

class pythreejs.**InstancedBufferGeometry**

InstancedBufferGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/core/InstancedBufferGeometry>

Inherits *BufferGeometry*.

Three.js docs: <https://threejs.org/docs/#api/core/InstancedBufferGeometry>

maxInstancedCount

```
CInt(0, allow_none=False).tag(sync=True)
```

type

```
Unicode("InstancedBufferGeometry", allow_none=False).tag(sync=True)
```

maxInstancedCount = CInt(0)

an int

type = Unicode('InstancedBufferGeometry')

a unicode string

InstancedInterleavedBuffer

class pythreejs.**InstancedInterleavedBuffer** (*array=None, meshPerAttribute=1*)

InstancedInterleavedBuffer

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/core/InstancedInterleavedBuffer>

Inherits *InterleavedBuffer*.

Three.js docs: <https://threejs.org/docs/#api/core/InstancedInterleavedBuffer>

meshPerAttribute

```
CInt(1, allow_none=False).tag(sync=True)
```

meshPerAttribute = CInt(1)

an int

InterleavedBufferAttribute

class pythreejs.**InterleavedBufferAttribute** (*data=None, itemSize=0, offset=0, normalized=True*)

InterleavedBufferAttribute

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/core/InterleavedBufferAttribute>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/core/InterleavedBufferAttribute>

data

```
Instance(InterleavedBuffer, allow_none=True).tag(sync=True, **widget_
↪ serialization)
```

itemSize

```
CInt(0, allow_none=False).tag(sync=True)
```

offset

```
CInt(0, allow_none=False).tag(sync=True)
```

normalized

```
Bool(True, allow_none=False).tag(sync=True)
```

data = Instance()
an InterleavedBuffer or None

itemSize = CInt(0)
an int

normalized = Bool(True)
a boolean

offset = CInt(0)
an int

InterleavedBuffer

class pythreejs.**InterleavedBuffer** (*array=None, stride=0*)
InterleavedBuffer

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/core/InterleavedBuffer>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/core/InterleavedBuffer>

array

```
WebGLDataUnion().tag(sync=True)
```

stride

```
CInt(0, allow_none=False).tag(sync=True)
```

dynamic

```
Bool(False, allow_none=False).tag(sync=True)
```

version

```
CInt(0, allow_none=False).tag(sync=True)
```

needsUpdate

```
Bool(False, allow_none=False).tag(sync=True)
```

array = WebGLDataUnion()
a numpy array or a NDArrayBase

dynamic = Bool(False)
a boolean

needsUpdate = Bool(False)
a boolean

stride = CInt(0)
an int

version = CInt(0)
an int

Layers

class pythreejs.Layers

Layers

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/core/Layers>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/core/Layers>

Object3D

class pythreejs.Object3D

This widget has some manual overrides on the Python side.

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/core/Object3D>

name

```
Unicode('', allow_none=False).tag(sync=True)
```

type

```
Unicode("Object3D", allow_none=False).tag(sync=True)
```

children

```
Tuple().tag(sync=True, **widget_serialization)
```

up

```
Vector3(default_value=[0,1,0]).tag(sync=True)
```

position

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

rotation

```
Euler(default_value=[0,0,0,"XYZ"]).tag(sync=True)
```

quaternion

```
Vector4(default_value=[0,0,0,1]).tag(sync=True)
```

scale

```
Vector3(default_value=[1,1,1]).tag(sync=True)
```

modelViewMatrix

```
Matrix4(default_value=[1,0,0,0,0,1,0,0,0,0,1,0,0,0,0,1]).tag(sync=True)
```

normalMatrix

```
Matrix3(default_value=[1,0,0,0,1,0,0,0,1]).tag(sync=True)
```

matrix

```
Matrix4(default_value=[1,0,0,0,0,1,0,0,0,0,1,0,0,0,0,1]).tag(sync=True)
```

matrixWorld

```
Matrix4(default_value=[1,0,0,0,0,1,0,0,0,0,1,0,0,0,0,1]).tag(sync=True)
```

matrixAutoUpdate

```
Bool(True, allow_none=False).tag(sync=True)
```

matrixWorldNeedsUpdate

```
Bool(False, allow_none=False).tag(sync=True)
```

visible

```
Bool(True, allow_none=False).tag(sync=True)
```

castShadow

```
Bool(False, allow_none=False).tag(sync=True)
```

receiveShadow

```
Bool(False, allow_none=False).tag(sync=True)
```

frustumCulled

```
Bool(True, allow_none=False).tag(sync=True)
```

renderOrder

```
CInt(0, allow_none=False).tag(sync=True)
```

add (*children*)

lookAt (*vector*)

remove (*children*)

rotateX (*rad*)

rotateY (*rad*)

rotateZ (*rad*)

setRotationFromMatrix (*m*)

m is a 3 by 3 matrix, as a list of rows. The columns of this matrix are the vectors x, y, and z

Raycaster

class pythreejs.**Raycaster** (*origin=[0,0,0], direction=[0,0,0], near=0, far=1000000,*)
Raycaster

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/core/Raycaster>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/core/Raycaster>

origin

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

direction

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

near

```
CFloat(0, allow_none=False).tag(sync=True)
```

far

```
CFloat(1000000, allow_none=False).tag(sync=True)
```

ray

```
Instance(Ray, allow_none=True).tag(sync=True, **widget_serialization)
```

linePrecision

```
CFloat(1, allow_none=False).tag(sync=True)
```

direction = Vector3((0, 0, 0))

a tuple of any type

far = CFloat(1000000)

a float

linePrecision = CFloat(1)

a float

near = CFloat(0)

a float

origin = Vector3((0, 0, 0))

a tuple of any type

ray = Instance()

a Ray or None

Uniform

class pythreejs.Uniform

Uniform

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/core/Uniform>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/core/Uniform>

2.5.7 extras**core****CurvePath**

class pythreejs.CurvePath

CurvePath

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/core/CurvePath>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/core/CurvePath>

Curve

class pythreejs.**Curve**

Curve

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/core/Curve>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/core/Curve>

Font

class pythreejs.**Font**

Font

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/core/Font>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/core/Font>

Path

class pythreejs.**Path**

Path

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/core/Path>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/core/Path>

ShapePath

class pythreejs.**ShapePath**

ShapePath

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/core/ShapePath>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/core/ShapePath>

Shape

class pythreejs.**Shape**

Shape

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/core/Shape>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/core/Shape>

curves

ArcCurve

class pythreejs.**ArcCurve**

ArcCurve

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/curves/ArcCurve>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/curves/ArcCurve>

CatmullRomCurve3

class pythreejs.**CatmullRomCurve3**

CatmullRomCurve3

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/curves/CatmullRomCurve3>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/curves/CatmullRomCurve3>

CubicBezierCurve3

class pythreejs.**CubicBezierCurve3**

CubicBezierCurve3

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/curves/CubicBezierCurve3>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/curves/CubicBezierCurve3>

CubicBezierCurve

class pythreejs.**CubicBezierCurve**

CubicBezierCurve

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/curves/CubicBezierCurve>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/curves/CubicBezierCurve>

EllipseCurve

class pythreejs.**EllipseCurve**

EllipseCurve

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/curves/EllipseCurve>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/curves/EllipseCurve>

LineCurve3

class pythreejs.**LineCurve3**

LineCurve3

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/curves/LineCurve3>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/curves/LineCurve3>

LineCurve

class pythreejs.**LineCurve**

LineCurve

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/curves/LineCurve>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/curves/LineCurve>

QuadraticBezierCurve3

class pythreejs.**QuadraticBezierCurve3**

QuadraticBezierCurve3

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/curves/QuadraticBezierCurve3>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/curves/QuadraticBezierCurve3>

QuadraticBezierCurve

class pythreejs.**QuadraticBezierCurve**

QuadraticBezierCurve

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/curves/QuadraticBezierCurve>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/curves/QuadraticBezierCurve>

SplineCurve

class pythreejs.**SplineCurve**

SplineCurve

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/curves/SplineCurve>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/curves/SplineCurve>

objects

ImmediateRenderObject

class pythreejs.ImmediateRenderObject

ImmediateRenderObject

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/extras/objects/ImmediateRenderObject>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/extras/objects/ImmediateRenderObject>

2.5.8 geometries

BoxBufferGeometry

class pythreejs.BoxBufferGeometry (*width=1, height=1, depth=1, widthSegments=1, heightSegments=1, depthSegments=1*)

BoxBufferGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/BoxGeometry>

Inherits *BaseBufferGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/BoxGeometry>

width

```
CFloat(1, allow_none=False).tag(sync=True)
```

height

```
CFloat(1, allow_none=False).tag(sync=True)
```

depth

```
CFloat(1, allow_none=False).tag(sync=True)
```

widthSegments

```
CInt(1, allow_none=False).tag(sync=True)
```

heightSegments

```
CInt(1, allow_none=False).tag(sync=True)
```

depthSegments

```
CInt(1, allow_none=False).tag(sync=True)
```

type

```
Unicode("BoxBufferGeometry", allow_none=False).tag(sync=True)
```

depth = CFloat(1)
a float

depthSegments = CInt(1)
an int

height = CFloat(1)
a float

heightSegments = CInt(1)
an int

type = Unicode('BoxBufferGeometry')
a unicode string

width = CFloat(1)
a float

widthSegments = CInt(1)
an int

BoxGeometry

class pythreejs.**BoxGeometry** (*width=1, height=1, depth=1, widthSegments=1, heightSegments=1, depthSegments=1*)

BoxGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/BoxGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/BoxGeometry>

width

```
CFloat(1, allow_none=False).tag(sync=True)
```

height

```
CFloat(1, allow_none=False).tag(sync=True)
```

depth

```
CFloat(1, allow_none=False).tag(sync=True)
```

widthSegments

```
CInt(1, allow_none=False).tag(sync=True)
```

heightSegments

```
CInt(1, allow_none=False).tag(sync=True)
```

depthSegments

```
CInt(1, allow_none=False).tag(sync=True)
```

type

```
Unicode("BoxGeometry", allow_none=False).tag(sync=True)
```

depth = CFloat(1)

a float

depthSegments = CInt(1)

an int

height = CFloat(1)

a float

heightSegments = CInt(1)

an int

type = Unicode('BoxGeometry')

a unicode string

width = CFloat(1)

a float

widthSegments = CInt(1)

an int

CircleBufferGeometry

```
class pythreejs.CircleBufferGeometry(radius=1, segments=8, thetaStart=0, thetaLength=6.283185307179586)
```

CircleBufferGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/CircleGeometry>

Inherits *BaseBufferGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/CircleGeometry>

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

segments

```
CInt(8, allow_none=False, min=3).tag(sync=True)
```

thetaStart

```
CFloat(0, allow_none=False).tag(sync=True)
```

thetaLength

```
CFloat(6.283185307179586, allow_none=False).tag(sync=True)
```

type

```
Unicode("CircleBufferGeometry", allow_none=False).tag(sync=True)
```

radius = CFloat(1)
a float

segments = CInt(8)
an int

thetaLength = CFloat(6.283185307179586)
a float

thetaStart = CFloat(0)
a float

type = Unicode('CircleBufferGeometry')
a unicode string

CircleGeometry

```
class pythreejs.CircleGeometry(radius=1, segments=8, thetaStart=0, thetaL-  
                               ength=6.283185307179586)
```

CircleGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/CircleGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/CircleGeometry>

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

segments

```
CInt(8, allow_none=False, min=3).tag(sync=True)
```

thetaStart

```
CFloat(0, allow_none=False).tag(sync=True)
```

thetaLength

```
CFloat(6.283185307179586, allow_none=False).tag(sync=True)
```

type

```
Unicode("CircleGeometry", allow_none=False).tag(sync=True)
```

radius = CFloat(1)

a float

segments = CInt(8)

an int

thetaLength = CFloat(6.283185307179586)

a float

thetaStart = CFloat(0)

a float

type = Unicode('CircleGeometry')

a unicode string

ConeGeometry

```
class pythreejs.ConeGeometry(radius=20, height=100, radialSegments=8, height-
                             segments=1, openEnded=False, thetaStart=0, thetaL-
                             ength=6.283185307179586)
```

ConeGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/ConeGeometry>Inherits *BaseGeometry*.Three.js docs: <https://threejs.org/docs/#api/geometries/ConeGeometry>**radius**

```
CFloat(20, allow_none=False).tag(sync=True)
```

height

```
CFloat(100, allow_none=False).tag(sync=True)
```

radialSegments

```
CInt(8, allow_none=False).tag(sync=True)
```

heightSegments

```
CInt(1, allow_none=False).tag(sync=True)
```

openEnded

```
Bool(False, allow_none=False).tag(sync=True)
```

thetaStart

```
CFloat(0, allow_none=False).tag(sync=True)
```

thetaLength

```
CFloat(6.283185307179586, allow_none=False).tag(sync=True)
```

type

```
Unicode("ConeGeometry", allow_none=False).tag(sync=True)
```

height = CFloat(100)

a float

heightSegments = CInt(1)

an int

openEnded = Bool(False)

a boolean

radialSegments = CInt(8)

an int

radius = CFloat(20)

a float

thetaLength = CFloat(6.283185307179586)

a float

thetaStart = CFloat(0)

a float

type = Unicode('ConeGeometry')

a unicode string

CylinderBufferGeometry

```
class pythreejs.CylinderBufferGeometry (radiusTop=1, radiusBottom=1, height=1,
                                         radiusSegments=8, heightSegments=1,
                                         openEnded=False, thetaStart=0, thetaLength=6.283185307179586)
```

CylinderBufferGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/CylinderGeometry>

Inherits *BaseBufferGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/CylinderGeometry>

radiusTop


```
CFloat(1, allow_none=False).tag(sync=True)
```

radiusBottom

```
CFloat(1, allow_none=False).tag(sync=True)
```

height

```
CFloat(1, allow_none=False).tag(sync=True)
```

radiusSegments

```
CInt(8, allow_none=False).tag(sync=True)
```

heightSegments

```
CInt(1, allow_none=False).tag(sync=True)
```

openEnded

```
Bool(False, allow_none=False).tag(sync=True)
```

thetaStart

```
CFloat(0, allow_none=False).tag(sync=True)
```

thetaLength

```
CFloat(6.283185307179586, allow_none=False).tag(sync=True)
```

type

```
Unicode("CylinderBufferGeometry", allow_none=False).tag(sync=True)
```

height = CFloat(1)

a float

heightSegments = CInt(1)

an int

openEnded = Bool(False)

a boolean

radiusBottom = CFloat(1)

a float

radiusSegments = CInt(8)

an int

```
radiusTop = CFloat(1)
    a float

thetaLength = CFloat(6.283185307179586)
    a float

thetaStart = CFloat(0)
    a float

type = Unicode('CylinderBufferGeometry')
    a unicode string
```

CylinderGeometry

```
class pythreejs.CylinderGeometry (radiusTop=1, radiusBottom=1, height=1, radiusSegments=8,
    heightSegments=1, openEnded=False, thetaStart=0, thetaLength=6.283185307179586)
```

CylinderGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/CylinderGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/CylinderGeometry>

radiusTop

```
CFloat(1, allow_none=False).tag(sync=True)
```

radiusBottom

```
CFloat(1, allow_none=False).tag(sync=True)
```

height

```
CFloat(1, allow_none=False).tag(sync=True)
```

radiusSegments

```
CInt(8, allow_none=False).tag(sync=True)
```

heightSegments

```
CInt(1, allow_none=False).tag(sync=True)
```

openEnded

```
Bool(False, allow_none=False).tag(sync=True)
```

thetaStart

```
CFloat(0, allow_none=False).tag(sync=True)
```

thetaLength

```
CFloat(6.283185307179586, allow_none=False).tag(sync=True)
```

type

```
Unicode("CylinderGeometry", allow_none=False).tag(sync=True)
```

height = CFloat(1)

a float

heightSegments = CInt(1)

an int

openEnded = Bool(False)

a boolean

radiusBottom = CFloat(1)

a float

radiusSegments = CInt(8)

an int

radiusTop = CFloat(1)

a float

thetaLength = CFloat(6.283185307179586)

a float

thetaStart = CFloat(0)

a float

type = Unicode('CylinderGeometry')

a unicode string

DodecahedronGeometry

class pythreejs.DodecahedronGeometry (*radius=1, detail=0*)

DodecahedronGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/DodecahedronGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/DodecahedronGeometry>

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

detail

```
CInt(0, allow_none=False).tag(sync=True)
```

type

```
Unicode("DodecahedronGeometry", allow_none=False).tag(sync=True)
```

detail = CInt(0)

an int

radius = CFloat(1)

a float

type = Unicode('DodecahedronGeometry')

a unicode string

EdgesGeometry

class pythreejs.**EdgesGeometry**

EdgesGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/EdgesGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/EdgesGeometry>

type

```
Unicode("EdgesGeometry", allow_none=False).tag(sync=True)
```

type = Unicode('EdgesGeometry')

a unicode string

ExtrudeGeometry

class pythreejs.**ExtrudeGeometry**

ExtrudeGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/ExtrudeGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/ExtrudeGeometry>

type

```
Unicode("ExtrudeGeometry", allow_none=False).tag(sync=True)
```

type = Unicode('ExtrudeGeometry')

a unicode string

IcosahedronGeometry

class pythreejs.**IcosahedronGeometry** (*radius=1, detail=0*)

IcosahedronGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/IcosahedronGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/IcosahedronGeometry>

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

detail

```
CInt(0, allow_none=False).tag(sync=True)
```

type

```
Unicode("IcosahedronGeometry", allow_none=False).tag(sync=True)
```

detail = CInt(0)

an int

radius = CFloat(1)

a float

type = Unicode('IcosahedronGeometry')

a unicode string

LatheBufferGeometry

```
class pythreejs.LatheBufferGeometry (points=[], segments=12, phiStart=0,
                                     phiLength=6.283185307179586)
```

LatheBufferGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/LatheGeometry>

Inherits *BaseBufferGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/LatheGeometry>

points

```
List(trait=List()).tag(sync=True)
```

segments

```
CInt(12, allow_none=False).tag(sync=True)
```

phiStart

```
CFloat(0, allow_none=False).tag(sync=True)
```

phiLength

```
CFloat(6.283185307179586, allow_none=False).tag(sync=True)
```

type

```
Unicode("LatheBufferGeometry", allow_none=False).tag(sync=True)
```

phiLength = CFloat(6.283185307179586)
a float

phiStart = CFloat(0)
a float

points = List()
a list with values that are: a list

segments = CInt(12)
an int

type = Unicode('LatheBufferGeometry')
a unicode string

LatheGeometry

class pythreejs.LatheGeometry(*points=[]*, *segments=12*, *phiStart=0*,
phiLength=6.283185307179586)

LatheGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/LatheGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/LatheGeometry>

points

```
List(trait=List()).tag(sync=True)
```

segments

```
CInt(12, allow_none=False).tag(sync=True)
```

phiStart

```
CFloat(0, allow_none=False).tag(sync=True)
```

phiLength

```
CFloat(6.283185307179586, allow_none=False).tag(sync=True)
```

type

```
Unicode("LatheGeometry", allow_none=False).tag(sync=True)
```

phiLength = CFloat(6.283185307179586)

a float

phiStart = CFloat(0)

a float

points = List()

a list with values that are: a list

segments = CInt(12)

an int

type = Unicode('LatheGeometry')

a unicode string

OctahedronGeometry

class pythreejs.OctahedronGeometry(radius=1, detail=0)

OctahedronGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/OctahedronGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/OctahedronGeometry>

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

detail

```
CInt(0, allow_none=False).tag(sync=True)
```

type

```
Unicode("OctahedronGeometry", allow_none=False).tag(sync=True)
```

detail = CInt(0)

an int

radius = CFloat(1)

a float

type = Unicode('OctahedronGeometry')

a unicode string

ParametricGeometry

class pythreejs.ParametricGeometry(func, slices=3, stacks=3)

ParametricGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/ParametricGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/ParametricGeometry>

func

```
Unicode('function(u,v) { return THREE.Vector3(); }').tag(sync=True)
```

slices

```
CInt(3, allow_none=False).tag(sync=True)
```

stacks

```
CInt(3, allow_none=False).tag(sync=True)
```

type

```
Unicode("ParametricGeometry", allow_none=False).tag(sync=True)
```

func = Unicode('function(u,v) { return THREE.Vector3(); }')
a unicode string

slices = CInt(3)
an int

stacks = CInt(3)
an int

type = Unicode('ParametricGeometry')
a unicode string

PlaneBufferGeometry

class pythreejs.PlaneBufferGeometry(*width=1, height=1, widthSegments=1, heightSegments=1*)

PlaneBufferGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/PlaneGeometry>

Inherits *BaseBufferGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/PlaneGeometry>

width

```
CFloat(1, allow_none=False).tag(sync=True)
```

height

```
CFloat(1, allow_none=False).tag(sync=True)
```

widthSegments


```
CInt(1, allow_none=False).tag(sync=True)
```

heightSegments

```
CInt(1, allow_none=False).tag(sync=True)
```

type

```
Unicode("PlaneBufferGeometry", allow_none=False).tag(sync=True)
```

height = CFloat(1)
a float

heightSegments = CInt(1)
an int

type = Unicode('PlaneBufferGeometry')
a unicode string

width = CFloat(1)
a float

widthSegments = CInt(1)
an int

PlaneGeometry

class pythreejs.PlaneGeometry (*width=1, height=1, widthSegments=1, heightSegments=1*)
PlaneGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/PlaneGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/PlaneGeometry>

width

```
CFloat(1, allow_none=False).tag(sync=True)
```

height

```
CFloat(1, allow_none=False).tag(sync=True)
```

widthSegments

```
CInt(1, allow_none=False).tag(sync=True)
```

heightSegments

```
CInt(1, allow_none=False).tag(sync=True)
```

type

```
Unicode("PlaneGeometry", allow_none=False).tag(sync=True)
```

height = CFloat(1)
a float

heightSegments = CInt(1)
an int

type = Unicode('PlaneGeometry')
a unicode string

width = CFloat(1)
a float

widthSegments = CInt(1)
an int

PolyhedronGeometry

class pythreejs.PolyhedronGeometry (vertices=[], faces=[], radius=1, detail=0)
PolyhedronGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/PolyhedronGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/PolyhedronGeometry>

vertices

```
List().tag(sync=True)
```

indices

```
List().tag(sync=True)
```

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

detail

```
CFloat(0, allow_none=False).tag(sync=True)
```

faces

```
List().tag(sync=True)
```

type

```
Unicode("PolyhedronGeometry", allow_none=False).tag(sync=True)
```

detail = **CFloat**(0)
a float

faces = **List**()
a list of any type

indices = **List**()
a list of any type

radius = **CFloat**(1)
a float

type = **Unicode**('PolyhedronGeometry')
a unicode string

vertices = **List**()
a list of any type

RingBufferGeometry

```
class pythreejs.RingBufferGeometry(innerRadius=0.5, outerRadius=1, thetaSegments=8, phiSegments=8, thetaStart=0, thetaLength=6.283185307179586)
```

RingBufferGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/RingGeometry>

Inherits *BaseBufferGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/RingGeometry>

innerRadius

```
CFloat(0.5, allow_none=False).tag(sync=True)
```

outerRadius

```
CFloat(1, allow_none=False).tag(sync=True)
```

thetaSegments

```
CInt(8, allow_none=False, min=3).tag(sync=True)
```

phiSegments

```
CInt(8, allow_none=False, min=1).tag(sync=True)
```

thetaStart

```
CFloat(0, allow_none=False).tag(sync=True)
```

thetaLength

```
CFloat(6.283185307179586, allow_none=False).tag(sync=True)
```

type

```
Unicode("RingBufferGeometry", allow_none=False).tag(sync=True)
```

innerRadius = CFloat(0.5)
a float

outerRadius = CFloat(1)
a float

phiSegments = CInt(8)
an int

thetaLength = CFloat(6.283185307179586)
a float

thetaSegments = CInt(8)
an int

thetaStart = CFloat(0)
a float

type = Unicode('RingBufferGeometry')
a unicode string

RingGeometry

class pythreejs.RingGeometry(*innerRadius=0.5, outerRadius=1, thetaSegments=8, phiSegments=8, thetaStart=0, thetaLength=6.283185307179586*)

RingGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/RingGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/RingGeometry>

innerRadius

```
CFloat(0.5, allow_none=False).tag(sync=True)
```

outerRadius

```
CFloat(1, allow_none=False).tag(sync=True)
```

thetaSegments

```
CInt(8, allow_none=False, min=3).tag(sync=True)
```

phiSegments

```
CInt(8, allow_none=False, min=1).tag(sync=True)
```

thetaStart

```
CFloat(0, allow_none=False).tag(sync=True)
```

thetaLength

```
CFloat(6.283185307179586, allow_none=False).tag(sync=True)
```

type

```
Unicode("RingGeometry", allow_none=False).tag(sync=True)
```

innerRadius = CFloat(0.5)

a float

outerRadius = CFloat(1)

a float

phiSegments = CInt(8)

an int

thetaLength = CFloat(6.283185307179586)

a float

thetaSegments = CInt(8)

an int

thetaStart = CFloat(0)

a float

type = Unicode('RingGeometry')

a unicode string

ShapeGeometry

class pythreejs.**ShapeGeometry** (*shapes=[]*)

ShapeGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/ShapeGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/ShapeGeometry>

shapes

```
Tuple().tag(sync=True, **widget_serialization)
```

curveSegments

```
CInt(12, allow_none=False).tag(sync=True)
```

material

```
CInt(0, allow_none=False).tag(sync=True)
```

type

```
Unicode("ShapeGeometry", allow_none=False).tag(sync=True)
```

curveSegments = CInt(12)

an int

material = CInt(0)

an int

shapes = Tuple()

a tuple of any type

type = Unicode('ShapeGeometry')

a unicode string

SphereBufferGeometry

class pythreejs.SphereBufferGeometry (*radius=1, widthSegments=8, heightSegments=6, phiStart=0, phiLength=6.283185307179586, thetaStart=0, thetaLength=3.141592653589793*)

SphereBufferGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/SphereGeometry>

Inherits *BaseBufferGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/SphereGeometry>

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

widthSegments

```
CInt(8, allow_none=False).tag(sync=True)
```

heightSegments

```
CInt(6, allow_none=False).tag(sync=True)
```

phiStart

```
CFloat(0, allow_none=False).tag(sync=True)
```

phiLength

```
CFloat(6.283185307179586, allow_none=False).tag(sync=True)
```

thetaStart

```
CFloat(0, allow_none=False).tag(sync=True)
```

thetaLength

```
CFloat(3.141592653589793, allow_none=False).tag(sync=True)
```

type

```
Unicode("SphereBufferGeometry", allow_none=False).tag(sync=True)
```

heightSegments = CInt(6)

an int

phiLength = CFloat(6.283185307179586)

a float

phiStart = CFloat(0)

a float

radius = CFloat(1)

a float

thetaLength = CFloat(3.141592653589793)

a float

thetaStart = CFloat(0)

a float

type = Unicode('SphereBufferGeometry')

a unicode string

widthSegments = CInt(8)

an int

SphereGeometry

```
class pythreejs.SphereGeometry(radius=1, widthSegments=8, heightSegments=6, phiStart=0,
                               phiLength=6.283185307179586, thetaStart=0, thetaLength=3.141592653589793)
```

SphereGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/SphereGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/SphereGeometry>

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

widthSegments

```
CInt(8, allow_none=False).tag(sync=True)
```

heightSegments

```
CInt(6, allow_none=False).tag(sync=True)
```

phiStart

```
CFloat(0, allow_none=False).tag(sync=True)
```

phiLength

```
CFloat(6.283185307179586, allow_none=False).tag(sync=True)
```

thetaStart

```
CFloat(0, allow_none=False).tag(sync=True)
```

thetaLength

```
CFloat(3.141592653589793, allow_none=False).tag(sync=True)
```

type

```
Unicode("SphereGeometry", allow_none=False).tag(sync=True)
```

heightSegments = CInt(6)

an int

phiLength = CFloat(6.283185307179586)

a float

phiStart = CFloat(0)

a float

radius = CFloat(1)

a float

thetaLength = CFloat(3.141592653589793)

a float

thetaStart = CFloat(0)

a float

type = Unicode('SphereGeometry')

a unicode string

widthSegments = CInt(8)

an int

TetrahedronGeometry

class pythreejs.**TetrahedronGeometry** (*radius=1, detail=0*)
TetrahedronGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/TetrahedronGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/TetrahedronGeometry>

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

detail

```
CInt(0, allow_none=False).tag(sync=True)
```

type

```
Unicode("TetrahedronGeometry", allow_none=False).tag(sync=True)
```

detail = CInt(0)
an int

radius = CFloat(1)
a float

type = Unicode('TetrahedronGeometry')
a unicode string

TextGeometry

class pythreejs.**TextGeometry**
TextGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/TextGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/TextGeometry>

type

```
Unicode("TextGeometry", allow_none=False).tag(sync=True)
```

type = Unicode('TextGeometry')
a unicode string

TorusBufferGeometry

class pythreejs.**TorusBufferGeometry** (*radius=1, tube=0.4, radialSegments=8, tubularSegments=6, arc=6.283185307179586*)
TorusBufferGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/TorusGeometry>

Inherits *BaseBufferGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/TorusGeometry>

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

tube

```
CFloat(0.4, allow_none=False).tag(sync=True)
```

radialSegments

```
CInt(8, allow_none=False).tag(sync=True)
```

tubularSegments

```
CInt(6, allow_none=False).tag(sync=True)
```

arc

```
CFloat(6.283185307179586, allow_none=False).tag(sync=True)
```

type

```
Unicode("TorusBufferGeometry", allow_none=False).tag(sync=True)
```

```
arc = CFloat(6.283185307179586)  
a float
```

```
radialSegments = CInt(8)  
an int
```

```
radius = CFloat(1)  
a float
```

```
tube = CFloat(0.4)  
a float
```

```
tubularSegments = CInt(6)  
an int
```

```
type = Unicode('TorusBufferGeometry')  
a unicode string
```

TorusGeometry

```
class pythreejs.TorusGeometry(radius=1, tube=0.4, radialSegments=8, tubularSegments=6,  
                                arc=6.283185307179586)  
    TorusGeometry
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/TorusGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/TorusGeometry>

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

tube

```
CFloat(0.4, allow_none=False).tag(sync=True)
```

radialSegments

```
CInt(8, allow_none=False).tag(sync=True)
```

tubularSegments

```
CInt(6, allow_none=False).tag(sync=True)
```

arc

```
CFloat(6.283185307179586, allow_none=False).tag(sync=True)
```

type

```
Unicode("TorusGeometry", allow_none=False).tag(sync=True)
```

```
arc = CFloat(6.283185307179586)
    a float
```

```
radialSegments = CInt(8)
    an int
```

```
radius = CFloat(1)
    a float
```

```
tube = CFloat(0.4)
    a float
```

```
tubularSegments = CInt(6)
    an int
```

```
type = Unicode('TorusGeometry')
    a unicode string
```

TorusKnotBufferGeometry

```
class pythreejs.TorusKnotBufferGeometry(radius=1, tube=0.4, tubularSegments=64, radialSegments=8, p=2, q=3)
    TorusKnotBufferGeometry
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/TorusKnotGeometry>

Inherits *BaseBufferGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/TorusKnotGeometry>

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

tube

```
CFloat(0.4, allow_none=False).tag(sync=True)
```

tubularSegments

```
CInt(64, allow_none=False).tag(sync=True)
```

radialSegments

```
CInt(8, allow_none=False).tag(sync=True)
```

p

```
CInt(2, allow_none=False).tag(sync=True)
```

q

```
CInt(3, allow_none=False).tag(sync=True)
```

type

```
Unicode("TorusKnotBufferGeometry", allow_none=False).tag(sync=True)
```

p = CInt(2)
an int

q = CInt(3)
an int

radialSegments = CInt(8)
an int

radius = CFloat(1)
a float

tube = CFloat(0.4)
a float

tubularSegments = CInt(64)
an int

type = Unicode('TorusKnotBufferGeometry')
a unicode string

TorusKnotGeometry

class pythreejs.**TorusKnotGeometry** (*radius=1, tube=0.4, tubularSegments=64, radialSegments=8, p=2, q=3*)

TorusKnotGeometry

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/TorusKnotGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/TorusKnotGeometry>

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

tube

```
CFloat(0.4, allow_none=False).tag(sync=True)
```

tubularSegments

```
CInt(64, allow_none=False).tag(sync=True)
```

radialSegments

```
CInt(8, allow_none=False).tag(sync=True)
```

p

```
CInt(2, allow_none=False).tag(sync=True)
```

q

```
CInt(3, allow_none=False).tag(sync=True)
```

type

```
Unicode("TorusKnotGeometry", allow_none=False).tag(sync=True)
```

p = CInt(2)

an int

q = CInt(3)

an int

radialSegments = CInt(8)

an int

radius = CFloat(1)

a float

```

tube = CFloat(0.4)
    a float

tubularSegments = CInt(64)
    an int

type = Unicode('TorusKnotGeometry')
    a unicode string
    
```

TubeGeometry

```

class pythreejs.TubeGeometry (path=None, segments=64, radius=1, radiusSegments=8,
                                close=False)
    TubeGeometry
    
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/TubeGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/TubeGeometry>

path

```

Instance(Curve, allow_none=True).tag(sync=True, **widget_serialization)
    
```

segments

```

CInt(64, allow_none=False).tag(sync=True)
    
```

radius

```

CFloat(1, allow_none=False).tag(sync=True)
    
```

radiusSegments

```

CInt(8, allow_none=False).tag(sync=True)
    
```

close

```

Bool(False, allow_none=False).tag(sync=True)
    
```

type

```

Unicode("TubeGeometry", allow_none=False).tag(sync=True)
    
```

```

close = Bool(False)
    a boolean
    
```

```

path = Instance()
    a Curve or None
    
```

```

radius = CFloat(1)
    a float
    
```

```

radiusSegments = CInt(8)
    an int

segments = CInt(64)
    an int

type = Unicode('TubeGeometry')
    a unicode string

```

WireframeGeometry

```

class pythreejs.WireframeGeometry(geometry=None)
    WireframeGeometry

```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/geometries/WireframeGeometry>

Inherits *BaseGeometry*.

Three.js docs: <https://threejs.org/docs/#api/geometries/WireframeGeometry>

geometry

```

Union([
    Instance(BaseGeometry, allow_none=True),
    Instance(BaseBufferGeometry, allow_none=True)
]).tag(sync=True, **widget_serialization)

```

type

```

Unicode("WireframeGeometry", allow_none=False).tag(sync=True)

```

```

geometry = Union()
    a BaseGeometry or None or a BaseBufferGeometry or None

```

```

type = Unicode('WireframeGeometry')
    a unicode string

```

2.5.9 helpers

ArrowHelper

```

class pythreejs.ArrowHelper
    ArrowHelper

```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/ArrowHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/ArrowHelper>

dir

```

Vector3(default_value=[1, 0, 0]).tag(sync=True)

```

origin

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

length

```
CFloat(1, allow_none=False).tag(sync=True)
```

hex

```
CInt(0, allow_none=False).tag(sync=True)
```

headLength

```
CFloat(None, allow_none=True).tag(sync=True)
```

headWidth

```
CFloat(None, allow_none=True).tag(sync=True)
```

type

```
Unicode("ArrowHelper", allow_none=False).tag(sync=True)
```

dir = Vector3((0, 0, 0))
a tuple of any type

headLength = CFloat(None)
a float

headWidth = CFloat(None)
a float

hex = CInt(0)
an int

length = CFloat(1)
a float

origin = Vector3((0, 0, 0))
a tuple of any type

type = Unicode('ArrowHelper')
a unicode string

AxesHelper

```
class pythreejs.AxesHelper(size=1)
    AxesHelper
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/AxesHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/AxesHelper>

size

```
CFloat(1, allow_none=False).tag(sync=True)
```

type

```
Unicode("AxesHelper", allow_none=False).tag(sync=True)
```

size = CFloat(1)

a float

type = Unicode('AxesHelper')

a unicode string

Box3Helper

class pythreejs.Box3Helper (*box=None, color="yellow"*)

Box3Helper

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/PlaneHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/PlaneHelper>

box

```
Instance(Box3, allow_none=True).tag(sync=True, **widget_serialization)
```

color

```
Unicode("yellow", allow_none=True).tag(sync=True)
```

type

```
Unicode("Box3Helper", allow_none=False).tag(sync=True)
```

box = Instance()

a Box3 or None

color = Unicode('yellow')

a unicode string

type = Unicode('Box3Helper')

a unicode string

BoxHelper

class pythreejs.BoxHelper (*object=None, color="#ffffff"*)

BoxHelper

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/BoxHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/BoxHelper>

object

```
Instance(Object3D, allow_none=True).tag(sync=True, **widget_serialization)
```

color

```
Unicode("#ffffff", allow_none=True).tag(sync=True)
```

type

```
Unicode("BoxHelper", allow_none=False).tag(sync=True)
```

color = **Unicode**(' #ffffff ')
a unicode string

object = **Instance**()
an Object3D or None

type = **Unicode**('BoxHelper ')
a unicode string

CameraHelper

class pythreejs.**CameraHelper** (*camera=None*)
CameraHelper

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/CameraHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/CameraHelper>

camera

```
Instance(Camera, allow_none=True).tag(sync=True, **widget_serialization)
```

type

```
Unicode("CameraHelper", allow_none=False).tag(sync=True)
```

camera = **Instance**()
a Camera or None

type = **Unicode**('CameraHelper ')
a unicode string

DirectionalLightHelper

class pythreejs.**DirectionalLightHelper** (*light=None, size=1, color="#ffffff"*)
DirectionalLightHelper

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/DirectionalLightHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/DirectionalLightHelper>

light

```
Instance(DirectionalLight, allow_none=True).tag(sync=True, **widget_
↪serialization)
```

size

```
CFloat(1, allow_none=False).tag(sync=True)
```

color

```
Unicode("#ffffff", allow_none=True).tag(sync=True)
```

type

```
Unicode("DirectionalLightHelper", allow_none=False).tag(sync=True)
```

color = Unicode('#ffffff')
a unicode string

light = Instance()
a DirectionalLight or None

size = CFloat(1)
a float

type = Unicode('DirectionalLightHelper')
a unicode string

FaceNormalsHelper

class pythreejs.FaceNormalsHelper (*object=None, size=1, color="0xffff00", linewidth=1*)
FaceNormalsHelper

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/FaceNormalsHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/FaceNormalsHelper>

object

```
Instance(Object3D, allow_none=True).tag(sync=True, **widget_serialization)
```

size

```
CFloat(1, allow_none=False).tag(sync=True)
```

color

```
Unicode("0xffff00", allow_none=False).tag(sync=True)
```

linewidth

```
CFloat(1, allow_none=False).tag(sync=True)
```

type

```
Unicode("FaceNormalsHelper", allow_none=False).tag(sync=True)
```

color = Unicode('0xffff00')
a unicode string

linewidth = CFloat(1)
a float

object = Instance()
an Object3D or None

size = CFloat(1)
a float

type = Unicode('FaceNormalsHelper')
a unicode string

GridHelper

class pythreejs.**GridHelper** (*size=10, divisions=10, colorCenterLine="0x444444", color-Grid="0x888888"*)

GridHelper

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/GridHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/GridHelper>

size

```
CFloat(10, allow_none=False).tag(sync=True)
```

divisions

```
CInt(10, allow_none=False).tag(sync=True)
```

colorCenterLine

```
Unicode("0x444444", allow_none=False).tag(sync=True)
```

colorGrid

```
Unicode("0x888888", allow_none=False).tag(sync=True)
```

type

```
Unicode("GridHelper", allow_none=False).tag(sync=True)
```

colorCenterLine = Unicode('0x444444')

a unicode string

colorGrid = Unicode('0x888888')

a unicode string

divisions = CInt(10)

an int

size = CFloat(10)

a float

type = Unicode('GridHelper')

a unicode string

HemisphereLightHelper

class pythreejs.HemisphereLightHelper (*light=None, size=1, color="#ffffff"*)

HemisphereLightHelper

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/HemisphereLightHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/HemisphereLightHelper>

light

```
Instance(HemisphereLight, allow_none=True).tag(sync=True, **widget_
↪serialization)
```

size

```
CFloat(1, allow_none=False).tag(sync=True)
```

color

```
Unicode("#ffffff", allow_none=True).tag(sync=True)
```

type

```
Unicode("HemisphereLightHelper", allow_none=False).tag(sync=True)
```

color = Unicode('#ffffff')

a unicode string

light = Instance()

a HemisphereLight or None

```
size = CFloat(1)
    a float
type = Unicode('HemisphereLightHelper')
    a unicode string
```

PlaneHelper

```
class pythreejs.PlaneHelper (plane=None, size=1, color="yellow")
    PlaneHelper
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/PlaneHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/PlaneHelper>

plane

```
Instance(Plane, allow_none=True).tag(sync=True, **widget_serialization)
```

size

```
CFloat(1, allow_none=False).tag(sync=True)
```

color

```
Unicode("yellow", allow_none=True).tag(sync=True)
```

type

```
Unicode("PlaneHelper", allow_none=False).tag(sync=True)
```

```
color = Unicode('yellow')
    a unicode string
```

```
plane = Instance()
    a Plane or None
```

```
size = CFloat(1)
    a float
```

```
type = Unicode('PlaneHelper')
    a unicode string
```

PointLightHelper

```
class pythreejs.PointLightHelper (light=None, sphereSize=1, color="#ffffff")
    PointLightHelper
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/PointLightHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/PointLightHelper>

light

```
Instance(PointLight, allow_none=True).tag(sync=True, **widget_serialization)
```

sphereSize

```
CFloat(1, allow_none=False).tag(sync=True)
```

color

```
Unicode("#ffffff", allow_none=True).tag(sync=True)
```

type

```
Unicode("PointLightHelper", allow_none=False).tag(sync=True)
```

color = Unicode('#ffffff')
a unicode string

light = Instance()
a PointLight or None

sphereSize = CFloat(1)
a float

type = Unicode('PointLightHelper')
a unicode string

PolarGridHelper

```
class pythreejs.PolarGridHelper (radius=10, radials=16, circles=8, divisions=64,
                                color1="0x444444", color2="0x888888")
```

PolarGridHelper

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/PolarGridHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/PolarGridHelper>

radius

```
CInt(10, allow_none=False).tag(sync=True)
```

radials

```
CInt(16, allow_none=False).tag(sync=True)
```

circles

```
CInt(8, allow_none=False).tag(sync=True)
```

divisions

```
CInt(64, allow_none=False).tag(sync=True)
```

color1

```
Unicode("0x444444", allow_none=False).tag(sync=True)
```

color2

```
Unicode("0x888888", allow_none=False).tag(sync=True)
```

type

```
Unicode("PolarGridHelper", allow_none=False).tag(sync=True)
```

circles = CInt(8)

an int

color1 = Unicode('0x444444')

a unicode string

color2 = Unicode('0x888888')

a unicode string

divisions = CInt(64)

an int

radials = CInt(16)

an int

radius = CInt(10)

an int

type = Unicode('PolarGridHelper')

a unicode string

RectAreaLightHelper

class pythreejs.RectAreaLightHelper (*light=None, color="#ffffff"*)

RectAreaLightHelper

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/RectAreaLightHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/RectAreaLightHelper>

light

```
Instance(RectAreaLight, allow_none=True).tag(sync=True, **widget_  
↪ serialization)
```

color


```
Unicode("#ffffff", allow_none=True).tag(sync=True)
```

type

```
Unicode("RectAreaLightHelper", allow_none=False).tag(sync=True)
```

color = Unicode('#ffffff')
a unicode string

light = Instance()
a RectAreaLight or None

type = Unicode('RectAreaLightHelper')
a unicode string

SkeletonHelper

class pythreejs.**SkeletonHelper** (*root=None*)
SkeletonHelper

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/SkeletonHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/SkeletonHelper>

root

```
Instance(Object3D, allow_none=True).tag(sync=True, **widget_serialization)
```

type

```
Unicode("SkeletonHelper", allow_none=False).tag(sync=True)
```

root = Instance()
an Object3D or None

type = Unicode('SkeletonHelper')
a unicode string

SpotLightHelper

class pythreejs.**SpotLightHelper** (*light=None, color="#ffffff"*)
SpotLightHelper

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/SpotLightHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/SpotLightHelper>

light

```
Instance(SpotLight, allow_none=True).tag(sync=True, **widget_serialization)
```

color

```
Unicode("#ffffff", allow_none=True).tag(sync=True)
```

type

```
Unicode("SpotLightHelper", allow_none=False).tag(sync=True)
```

color = Unicode('#ffffff')
a unicode string

light = Instance()
a SpotLight or None

type = Unicode('SpotLightHelper')
a unicode string

VertexNormalsHelper

class pythreejs.**VertexNormalsHelper** (*object=None, size=1, color="0xffff00", linewidth=1*)
VertexNormalsHelper

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/helpers/VertexNormalsHelper>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/helpers/VertexNormalsHelper>

object

```
Instance(Object3D, allow_none=True).tag(sync=True, **widget_serialization)
```

size

```
CFloat(1, allow_none=False).tag(sync=True)
```

color

```
Unicode("0xffff00", allow_none=False).tag(sync=True)
```

linewidth

```
CFloat(1, allow_none=False).tag(sync=True)
```

type

```
Unicode("VertexNormalsHelper", allow_none=False).tag(sync=True)
```

color = Unicode('0xffff00')
a unicode string

```

linewidth = CFloat(1)
    a float

object = Instance()
    an Object3D or None

size = CFloat(1)
    a float

type = Unicode('VertexNormalsHelper')
    a unicode string

```

2.5.10 lights

AmbientLight

```

class pythreejs.AmbientLight (color="#ffffff", intensity=1)
    AmbientLight

```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/lights/AmbientLight>

Inherits *Light*.

Three.js docs: <https://threejs.org/docs/#api/lights/AmbientLight>

type

```
Unicode("AmbientLight", allow_none=False).tag(sync=True)
```

```

type = Unicode('AmbientLight')
    a unicode string

```

DirectionalLightShadow

```

class pythreejs.DirectionalLightShadow
    DirectionalLightShadow

```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/lights/DirectionalLightShadow>

Inherits *LightShadow*.

Three.js docs: <https://threejs.org/docs/#api/lights/DirectionalLightShadow>

DirectionalLight

```

class pythreejs.DirectionalLight (color="#ffffff", intensity=1)
    DirectionalLight

```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/lights/DirectionalLight>

Inherits *Light*.

Three.js docs: <https://threejs.org/docs/#api/lights/DirectionalLight>

target

```
Union([
    Instance(Uninitialized),
    Instance(Object3D),
    ], default_value=UninitializedSentinel, allow_none=False).tag(sync=True,
↳**uninitialized_serialization)
```

shadow

```
Union([
    Instance(Uninitialized),
    Instance(LightShadow),
    ], default_value=UninitializedSentinel, allow_none=False).tag(sync=True,
↳**uninitialized_serialization)
```

type

```
Unicode("DirectionalLight", allow_none=False).tag(sync=True)
```

shadow = Union(<pythreejs.traits.Uninitialized object at 0x7f8cf52246a0>
an Uninitialized or a LightShadow)

target = Union(<pythreejs.traits.Uninitialized object at 0x7f8cf52246a0>
an Uninitialized or an Object3D)

type = Unicode('DirectionalLight')
a unicode string

HemisphereLight

class pythreejs.HemisphereLight (color="#ffffff", groundColor="#000000", intensity=1)
HemisphereLight

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/lights/HemisphereLight>

Inherits *Light*.

Three.js docs: <https://threejs.org/docs/#api/lights/HemisphereLight>

groundColor

```
Unicode("#000000", allow_none=False).tag(sync=True)
```

type

```
Unicode("HemisphereLight", allow_none=False).tag(sync=True)
```

groundColor = Unicode('#000000')
a unicode string

type = Unicode('HemisphereLight')
a unicode string

LightShadow

class pythreejs.**LightShadow** (*camera=UninitializedSentinel*)
LightShadow

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/lights/LightShadow>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/lights/LightShadow>

camera

```
Union([
  Instance(Uninitialized),
  Instance(Camera),
], default_value=UninitializedSentinel, allow_none=False).tag(sync=True,
↳**uninitialized_serialization)
```

bias

```
CFloat(0, allow_none=False).tag(sync=True)
```

mapSize

```
Vector2(default_value=[512, 512]).tag(sync=True)
```

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

bias = CFloat(0)

a float

camera = Union(<pythreejs.traits.Uninitialized object at 0x7f8cf52246a0>)

an Uninitialized or a Camera

mapSize = Vector2((0, 0))

a tuple of any type

radius = CFloat(1)

a float

Light

class pythreejs.**Light** (*color="#ffffff", intensity=1*)
Light

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/lights/Light>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/lights/Light>

color

```
Unicode("#ffffff", allow_none=False).tag(sync=True)
```

intensity

```
CFloat(1, allow_none=False).tag(sync=True)
```

type

```
Unicode("Light", allow_none=False).tag(sync=True)
```

color = Unicode('#ffffff')
a unicode string

intensity = CFloat(1)
a float

type = Unicode('Light')
a unicode string

PointLight

class pythreejs.**PointLight** (*color="#ffffff", intensity=1, distance=0, decay=1*)
PointLight

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/lights/PointLight>

Inherits *Light*.

Three.js docs: <https://threejs.org/docs/#api/lights/PointLight>

power

```
CFloat(12.566370614359172, allow_none=False).tag(sync=True)
```

distance

```
CFloat(0, allow_none=False).tag(sync=True)
```

decay

```
CFloat(1, allow_none=False).tag(sync=True)
```

shadow

```
Union([
    Instance(Uninitialized),
    Instance(LightShadow),
], default_value=UninitializedSentinel, allow_none=False).tag(sync=True,
↪**uninitialized_serialization)
```

type

```
Unicode("PointLight", allow_none=False).tag(sync=True)
```

decay = CFloat(1)

a float

distance = CFloat(0)

a float

power = CFloat(12.566370614359172)

a float

shadow = Union(<pythreejs.traits.Uninitialized object at 0x7f8cf52246a0>)

an Uninitialized or a LightShadow

type = Unicode('PointLight')

a unicode string

RectAreaLight

class pythreejs.RectAreaLight (*color="#ffffff", intensity=1, width=10, height=10*)

RectAreaLight

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/lights/RectAreaLight>

Inherits *Light*.

Three.js docs: <https://threejs.org/docs/#api/lights/RectAreaLight>

width

```
CFloat(10, allow_none=False).tag(sync=True)
```

height

```
CFloat(10, allow_none=False).tag(sync=True)
```

type

```
Unicode("RectAreaLight", allow_none=False).tag(sync=True)
```

height = CFloat(10)

a float

type = Unicode('RectAreaLight')

a unicode string

width = CFloat(10)

a float

SpotLightShadow

class pythreejs.SpotLightShadow

SpotLightShadow

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/lights/SpotLightShadow>

Inherits *LightShadow*.

Three.js docs: <https://threejs.org/docs/#api/lights/SpotLightShadow>

SpotLight

class pythreejs.**SpotLight** (*color="#ffffff", intensity=1, distance=0, angle=1.0471975511965976, penumbra=0, decay=1*)

SpotLight

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/lights/SpotLight>

Inherits *Light*.

Three.js docs: <https://threejs.org/docs/#api/lights/SpotLight>

target

```
Union([
  Instance(Uninitialized),
  Instance(Object3D),
], default_value=UninitializedSentinel, allow_none=False).tag(sync=True,
↳**uninitialized_serialization)
```

distance

```
CFloat(0, allow_none=False).tag(sync=True)
```

angle

```
CFloat(1.0471975511965976, allow_none=False).tag(sync=True)
```

penumbra

```
CFloat(0, allow_none=False).tag(sync=True)
```

decay

```
CFloat(1, allow_none=False).tag(sync=True)
```

shadow

```
Union([
  Instance(Uninitialized),
  Instance(LightShadow),
], default_value=UninitializedSentinel, allow_none=False).tag(sync=True,
↳**uninitialized_serialization)
```

type


```
Unicode("SpotLight", allow_none=False).tag(sync=True)
```

```
angle = CFloat(1.0471975511965976)
    a float
```

```
decay = CFloat(1)
    a float
```

```
distance = CFloat(0)
    a float
```

```
penumbra = CFloat(0)
    a float
```

```
shadow = Union(<pythreejs.traits.Uninitialized object at 0x7f8cf52246a0>)
    an Uninitialized or a LightShadow
```

```
target = Union(<pythreejs.traits.Uninitialized object at 0x7f8cf52246a0>)
    an Uninitialized or an Object3D
```

```
type = Unicode('SpotLight')
    a unicode string
```

2.5.11 loaders

AnimationLoader

```
class pythreejs.AnimationLoader
    AnimationLoader
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/AnimationLoader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/AnimationLoader>

AudioLoader

```
class pythreejs.AudioLoader
    AudioLoader
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/AudioLoader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/AudioLoader>

BufferGeometryLoader

```
class pythreejs.BufferGeometryLoader
    BufferGeometryLoader
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/BufferGeometryLoader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/BufferGeometryLoader>

Cache

class pythreejs.Cache
Cache

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/Cache>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/Cache>

CompressedTextureLoader

class pythreejs.CompressedTextureLoader
CompressedTextureLoader

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/CompressedTextureLoader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/CompressedTextureLoader>

CubeTextureLoader

class pythreejs.CubeTextureLoader
CubeTextureLoader

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/CubeTextureLoader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/CubeTextureLoader>

DataTextureLoader

class pythreejs.DataTextureLoader
DataTextureLoader

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/DataTextureLoader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/DataTextureLoader>

FileLoader

class pythreejs.FileLoader
FileLoader

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/FileLoader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/FileLoader>

FontLoader

class pythreejs.**FontLoader**
FontLoader

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/FontLoader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/FontLoader>

ImageBitmapLoader

class pythreejs.**ImageBitmapLoader**
ImageBitmapLoader

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/ImageBitmapLoader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/ImageBitmapLoader>

ImageLoader

class pythreejs.**ImageLoader**
ImageLoader

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/ImageLoader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/ImageLoader>

JSONLoader

class pythreejs.**JSONLoader**
JSONLoader

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/JSONLoader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/JSONLoader>

Loader

class pythreejs.**Loader**
Loader

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/Loader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/Loader>

LoadingManager

class pythreejs.**LoadingManager**
LoadingManager

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/LoadingManager>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/LoadingManager>

MaterialLoader

class pythreejs.**MaterialLoader**
MaterialLoader

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/MaterialLoader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/MaterialLoader>

ObjectLoader

class pythreejs.**ObjectLoader**
ObjectLoader

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/ObjectLoader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/ObjectLoader>

TextureLoader

class pythreejs.**TextureLoader**
TextureLoader

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/loaders/TextureLoader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/loaders/TextureLoader>

2.5.12 materials

LineBasicMaterial

class pythreejs.**LineBasicMaterial**
LineBasicMaterial

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/LineBasicMaterial>

Inherits *Material*.

Three.js docs: <https://threejs.org/docs/#api/materials/LineBasicMaterial>

color

```
Unicode("#ffffff", allow_none=False).tag(sync=True)
```

lights

```
Bool(False, allow_none=False).tag(sync=True)
```

linewidth

```
CFloat(1, allow_none=False).tag(sync=True)
```

linecap

```
Unicode("round", allow_none=False).tag(sync=True)
```

linejoin

```
Unicode("round", allow_none=False).tag(sync=True)
```

type

```
Unicode("LineBasicMaterial", allow_none=False).tag(sync=True)
```

color = Unicode('#ffffff')
a unicode string

lights = Bool(False)
a boolean

linecap = Unicode('round')
a unicode string

linejoin = Unicode('round')
a unicode string

linewidth = CFloat(1)
a float

type = Unicode('LineBasicMaterial')
a unicode string

LineDashedMaterial

class pythreejs.LineDashedMaterial
LineDashedMaterial

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/LineDashedMaterial>

Inherits *Material*.

Three.js docs: <https://threejs.org/docs/#api/materials/LineDashedMaterial>

color

```
Unicode("#ffffff", allow_none=False).tag(sync=True)
```

lights

```
Bool(False, allow_none=False).tag(sync=True)
```

linewidth

```
CFloat(1, allow_none=False).tag(sync=True)
```

scale

```
CFloat(1, allow_none=False).tag(sync=True)
```

dashSize

```
CFloat(3, allow_none=False).tag(sync=True)
```

gapSize

```
CFloat(1, allow_none=False).tag(sync=True)
```

type

```
Unicode("LineDashedMaterial", allow_none=False).tag(sync=True)
```

color = Unicode('#ffffff')
a unicode string

dashSize = CFloat(3)
a float

gapSize = CFloat(1)
a float

lights = Bool(False)
a boolean

linewidth = CFloat(1)
a float

scale = CFloat(1)
a float

type = Unicode('LineDashedMaterial')
a unicode string

Material

class pythreejs.Material

This widget has some manual overrides on the Python side.

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/materials/Material>

alphaTest

```
CFloat(0, allow_none=False).tag(sync=True)
```

blendDst

```
Enum(BlendFactors, "OneMinusSrcAlphaFactor", allow_none=False).tag(sync=True)
```

blendDstAlpha

```
CFloat(0, allow_none=False).tag(sync=True)
```

blending

```
Enum(BlendingMode, "NormalBlending", allow_none=False).tag(sync=True)
```

blendSrc

```
Enum(BlendFactors, "SrcAlphaFactor", allow_none=False).tag(sync=True)
```

blendSrcAlpha

```
CFloat(0, allow_none=False).tag(sync=True)
```

blendEquation

```
Enum(Equations, "AddEquation", allow_none=False).tag(sync=True)
```

blendEquationAlpha

```
CFloat(0, allow_none=False).tag(sync=True)
```

clipIntersection

```
Bool(False, allow_none=False).tag(sync=True)
```

clippingPlanes

```
Tuple().tag(sync=True, **widget_serialization)
```

clipShadows

```
Bool(False, allow_none=False).tag(sync=True)
```

colorWrite

```
Bool(True, allow_none=False).tag(sync=True)
```

defines

```
Dict(default_value=None, allow_none=True).tag(sync=True)
```

depthFunc

```
Enum(DepthMode, "LessEqualDepth", allow_none=False).tag(sync=True)
```

depthTest

```
Bool(True, allow_none=False).tag(sync=True)
```

depthWrite

```
Bool(True, allow_none=False).tag(sync=True)
```

dithering

```
Bool(False, allow_none=False).tag(sync=True)
```

flatShading

```
Bool(False, allow_none=False).tag(sync=True)
```

fog

```
Bool(True, allow_none=False).tag(sync=True)
```

lights

```
Bool(True, allow_none=False).tag(sync=True)
```

name

```
Unicode('', allow_none=False).tag(sync=True)
```

opacity


```
CFloat(1, allow_none=False).tag(sync=True)
```

overdraw

```
CFloat(0, allow_none=False).tag(sync=True)
```

polygonOffset

```
Bool(False, allow_none=False).tag(sync=True)
```

polygonOffsetFactor

```
CFloat(0, allow_none=False).tag(sync=True)
```

polygonOffsetUnits

```
CFloat(0, allow_none=False).tag(sync=True)
```

precision

```
Unicode(None, allow_none=True).tag(sync=True)
```

premultipliedAlpha

```
Bool(False, allow_none=False).tag(sync=True)
```

shadowSide

```
Enum(Side, None, allow_none=True).tag(sync=True)
```

side

```
Enum(Side, "FrontSide", allow_none=False).tag(sync=True)
```

transparent

```
Bool(False, allow_none=False).tag(sync=True)
```

type

```
Unicode("Material", allow_none=False).tag(sync=True)
```

vertexColors

```
Enum(Colors, "NoColors", allow_none=False).tag(sync=True)
```

visible

```
Bool(True, allow_none=False).tag(sync=True)
```

needsUpdate = Bool(False)
a boolean

onNeedsUpdate

MeshBasicMaterial

class pythreejs.**MeshBasicMaterial**
MeshBasicMaterial

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/MeshBasicMaterial>

Inherits *Material*.

Three.js docs: <https://threejs.org/docs/#api/materials/MeshBasicMaterial>

alphaMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

aoMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

aoMapIntensity

```
CFloat(1, allow_none=False).tag(sync=True)
```

color

```
Unicode("#ffffff", allow_none=False).tag(sync=True)
```

combine

```
Enum(Operations, "MultiplyOperation", allow_none=False).tag(sync=True)
```

envMap

```
Instance(CubeTexture, allow_none=True).tag(sync=True, **widget_serialization)
```

lightMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

lightMapIntensity

```
CFloat(1, allow_none=False).tag(sync=True)
```

lights

```
Bool(False, allow_none=False).tag(sync=True)
```

map

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

morphTargets

```
Bool(False, allow_none=False).tag(sync=True)
```

reflectivity

```
CFloat(1, allow_none=False).tag(sync=True)
```

refractionRatio

```
CFloat(0.98, allow_none=False).tag(sync=True)
```

skinning

```
Bool(False, allow_none=False).tag(sync=True)
```

specularMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

wireframe

```
Bool(False, allow_none=False).tag(sync=True)
```

wireframeLinewidth

```
CFloat(1, allow_none=False).tag(sync=True)
```

wireframeLinecap

```
Unicode("round", allow_none=False).tag(sync=True)
```

wireframeLinejoin

```
Unicode("round", allow_none=False).tag(sync=True)
```

type

```
Unicode("MeshBasicMaterial", allow_none=False).tag(sync=True)
```

alphaMap = Instance()

a Texture or None

aoMap = Instance()

a Texture or None

aoMapIntensity = CFloat(1)

a float

color = Unicode('#ffffff')

a unicode string

combine = Enum('MultiplyOperation')

any of ['AddOperation', 'MixOperation', 'MultiplyOperation']

envMap = Instance()

a CubeTexture or None

lightMap = Instance()

a Texture or None

lightMapIntensity = CFloat(1)

a float

lights = Bool(False)

a boolean

map = Instance()

a Texture or None

morphTargets = Bool(False)

a boolean

reflectivity = CFloat(1)

a float

refractionRatio = CFloat(0.98)

a float

skinning = Bool(False)

a boolean

specularMap = Instance()

a Texture or None

type = Unicode('MeshBasicMaterial')

a unicode string

wireframe = Bool(False)

a boolean

```

wireframeLinecap = Unicode('round')
    a unicode string

wireframeLinejoin = Unicode('round')
    a unicode string

wireframeLinewidth = CFloat(1)
    a float

```

MeshDepthMaterial

```

class pythreejs.MeshDepthMaterial
    MeshDepthMaterial

```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/MeshDepthMaterial>

Inherits *Material*.

Three.js docs: <https://threejs.org/docs/#api/materials/MeshDepthMaterial>

alphaMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

displacementMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

displacementScale

```
CFloat(1, allow_none=False).tag(sync=True)
```

displacementBias

```
CFloat(0, allow_none=False).tag(sync=True)
```

fog

```
Bool(False, allow_none=False).tag(sync=True)
```

lights

```
Bool(False, allow_none=False).tag(sync=True)
```

map

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

morphTargets

```
Bool(False, allow_none=False).tag(sync=True)
```

skinning

```
Bool(False, allow_none=False).tag(sync=True)
```

wireframe

```
Bool(False, allow_none=False).tag(sync=True)
```

wireframeLinewidth

```
CFloat(1, allow_none=False).tag(sync=True)
```

type

```
Unicode("MeshDepthMaterial", allow_none=False).tag(sync=True)
```

alphaMap = Instance()
a Texture or None

displacementBias = CFloat(0)
a float

displacementMap = Instance()
a Texture or None

displacementScale = CFloat(1)
a float

fog = Bool(False)
a boolean

lights = Bool(False)
a boolean

map = Instance()
a Texture or None

morphTargets = Bool(False)
a boolean

skinning = Bool(False)
a boolean

type = Unicode('MeshDepthMaterial')
a unicode string

wireframe = Bool(False)
a boolean

wireframeLinewidth = CFloat(1)
a float

MeshLambertMaterial

class pythreejs.MeshLambertMaterial

MeshLambertMaterial

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/MeshLambertMaterial>

Inherits *Material*.

Three.js docs: <https://threejs.org/docs/#api/materials/MeshLambertMaterial>

alphaMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

aoMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

aoMapIntensity

```
CFloat(1, allow_none=False).tag(sync=True)
```

color

```
Unicode("#ffffff", allow_none=False).tag(sync=True)
```

combine

```
Enum(Operations, "MultiplyOperation", allow_none=False).tag(sync=True)
```

emissive

```
Unicode("#000000", allow_none=False).tag(sync=True)
```

emissiveMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

emissiveIntensity

```
CFloat(1, allow_none=False).tag(sync=True)
```

envMap

```
Instance(CubeTexture, allow_none=True).tag(sync=True, **widget_serialization)
```

lightMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

lightMapIntensity

```
CFloat(1, allow_none=False).tag(sync=True)
```

map

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

morphNormals

```
Bool(False, allow_none=False).tag(sync=True)
```

morphTargets

```
Bool(False, allow_none=False).tag(sync=True)
```

reflectivity

```
CFloat(1, allow_none=False).tag(sync=True)
```

refractionRatio

```
CFloat(0.98, allow_none=False).tag(sync=True)
```

skinning

```
Bool(False, allow_none=False).tag(sync=True)
```

specularMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

wireframe

```
Bool(False, allow_none=False).tag(sync=True)
```

wireframeLinecap

```
Unicode("round", allow_none=False).tag(sync=True)
```

wireframeLinejoin


```
Unicode("round", allow_none=False).tag(sync=True)
```

wireframeLinewidth

```
CFloat(1, allow_none=False).tag(sync=True)
```

type

```
Unicode("MeshLambertMaterial", allow_none=False).tag(sync=True)
```

alphaMap = Instance()

a Texture or None

aoMap = Instance()

a Texture or None

aoMapIntensity = CFloat(1)

a float

color = Unicode('#ffffff')

a unicode string

combine = Enum('MultiplyOperation')

any of ['AddOperation', 'MixOperation', 'MultiplyOperation']

emissive = Unicode('#000000')

a unicode string

emissiveIntensity = CFloat(1)

a float

emissiveMap = Instance()

a Texture or None

envMap = Instance()

a CubeTexture or None

lightMap = Instance()

a Texture or None

lightMapIntensity = CFloat(1)

a float

map = Instance()

a Texture or None

morphNormals = Bool(False)

a boolean

morphTargets = Bool(False)

a boolean

reflectivity = CFloat(1)

a float

refractionRatio = CFloat(0.98)

a float

skinning = Bool(False)

a boolean

```

specularMap = Instance()
    a Texture or None

type = Unicode('MeshLambertMaterial')
    a unicode string

wireframe = Bool(False)
    a boolean

wireframeLinecap = Unicode('round')
    a unicode string

wireframeLinejoin = Unicode('round')
    a unicode string

wireframeLinewidth = CFloat(1)
    a float
    
```

MeshNormalMaterial

```

class pythreejs.MeshNormalMaterial
    MeshNormalMaterial
    
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/MeshNormalMaterial>

Inherits *Material*.

Three.js docs: <https://threejs.org/docs/#api/materials/MeshNormalMaterial>

fog

```

Bool(False, allow_none=False).tag(sync=True)
    
```

lights

```

Bool(False, allow_none=False).tag(sync=True)
    
```

morphTargets

```

Bool(False, allow_none=False).tag(sync=True)
    
```

wireframe

```

Bool(False, allow_none=False).tag(sync=True)
    
```

wireframeLinewidth

```

CFloat(1, allow_none=False).tag(sync=True)
    
```

type

```

Unicode("MeshNormalMaterial", allow_none=False).tag(sync=True)
    
```

```

fog = Bool(False)
    a boolean

lights = Bool(False)
    a boolean

morphTargets = Bool(False)
    a boolean

type = Unicode('MeshNormalMaterial')
    a unicode string

wireframe = Bool(False)
    a boolean

wireframeLinewidth = CFloat(1)
    a float

```

MeshPhongMaterial

class pythreejs.**MeshPhongMaterial**

MeshPhongMaterial

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/MeshPhongMaterial>

Inherits *Material*.

Three.js docs: <https://threejs.org/docs/#api/materials/MeshPhongMaterial>

alphaMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

aoMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

aoMapIntensity

```
CFloat(1, allow_none=False).tag(sync=True)
```

bumpMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

bumpScale

```
CFloat(1, allow_none=False).tag(sync=True)
```

color

```
Unicode("#ffffff", allow_none=False).tag(sync=True)
```

combine

```
Enum(Operations, "MultiplyOperation", allow_none=False).tag(sync=True)
```

displacementMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

displacementScale

```
CFloat(1, allow_none=False).tag(sync=True)
```

displacementBias

```
CFloat(0, allow_none=False).tag(sync=True)
```

emissive

```
Unicode("#000000", allow_none=False).tag(sync=True)
```

emissiveMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

emissiveIntensity

```
CFloat(1, allow_none=False).tag(sync=True)
```

envMap

```
Instance(CubeTexture, allow_none=True).tag(sync=True, **widget_serialization)
```

lightMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

lightMapIntensity

```
CFloat(1, allow_none=False).tag(sync=True)
```

map

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

morphNormals

```
Bool(False, allow_none=False).tag(sync=True)
```

morphTargets

```
Bool(False, allow_none=False).tag(sync=True)
```

normalMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

normalScale

```
Vector2(default_value=[1,1]).tag(sync=True)
```

reflectivity

```
CFloat(1, allow_none=False).tag(sync=True)
```

refractionRatio

```
CFloat(0.98, allow_none=False).tag(sync=True)
```

shininess

```
CFloat(30, allow_none=False).tag(sync=True)
```

skinning

```
Bool(False, allow_none=False).tag(sync=True)
```

specular

```
Unicode("#111111", allow_none=False).tag(sync=True)
```

specularMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

wireframe

```
Bool(False, allow_none=False).tag(sync=True)
```

wireframeLinewidth

```
CFloat(1, allow_none=False).tag(sync=True)
```

wireframeLinecap

```
Unicode("round", allow_none=False).tag(sync=True)
```

wireframeLinejoin

```
Unicode("round", allow_none=False).tag(sync=True)
```

type

```
Unicode("MeshPhongMaterial", allow_none=False).tag(sync=True)
```

alphaMap = Instance()

a Texture or None

aoMap = Instance()

a Texture or None

aoMapIntensity = CFloat(1)

a float

bumpMap = Instance()

a Texture or None

bumpScale = CFloat(1)

a float

color = Unicode('#ffffff')

a unicode string

combine = Enum('MultiplyOperation')

any of ['AddOperation', 'MixOperation', 'MultiplyOperation']

displacementBias = CFloat(0)

a float

displacementMap = Instance()

a Texture or None

displacementScale = CFloat(1)

a float

emissive = Unicode('#000000')

a unicode string

emissiveIntensity = CFloat(1)

a float

emissiveMap = Instance()

a Texture or None

envMap = Instance()

a CubeTexture or None

lightMap = Instance()

a Texture or None

```

lightMapIntensity = CFloat(1)
    a float

map = Instance()
    a Texture or None

morphNormals = Bool(False)
    a boolean

morphTargets = Bool(False)
    a boolean

normalMap = Instance()
    a Texture or None

normalScale = Vector2((0, 0))
    a tuple of any type

reflectivity = CFloat(1)
    a float

refractionRatio = CFloat(0.98)
    a float

shininess = CFloat(30)
    a float

skinning = Bool(False)
    a boolean

specular = Unicode('#111111')
    a unicode string

specularMap = Instance()
    a Texture or None

type = Unicode('MeshPhongMaterial')
    a unicode string

wireframe = Bool(False)
    a boolean

wireframeLinecap = Unicode('round')
    a unicode string

wireframeLinejoin = Unicode('round')
    a unicode string

wireframeLinewidth = CFloat(1)
    a float

```

MeshPhysicalMaterial

```

class pythreejs.MeshPhysicalMaterial
    MeshPhysicalMaterial

```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/MeshPhysicalMaterial>

Inherits *MeshStandardMaterial*.

Three.js docs: <https://threejs.org/docs/#api/materials/MeshPhysicalMaterial>

clearCoat

```
CFloat(0, allow_none=False).tag(sync=True)
```

clearCoatRoughness

```
CFloat(0, allow_none=False).tag(sync=True)
```

defines

```
Dict(default_value={"PHYSICAL":""}, allow_none=True).tag(sync=True)
```

reflectivity

```
CFloat(0.5, allow_none=False).tag(sync=True)
```

type

```
Unicode("MeshPhysicalMaterial", allow_none=False).tag(sync=True)
```

clearCoat = CFloat(0)

a float

clearCoatRoughness = CFloat(0)

a float

defines = Dict()

a dict or None with elements of any type

reflectivity = CFloat(0.5)

a float

type = Unicode('MeshPhysicalMaterial')

a unicode string

MeshStandardMaterial

class pythreejs.**MeshStandardMaterial**

MeshStandardMaterial

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/MeshStandardMaterial>

Inherits *Material*.

Three.js docs: <https://threejs.org/docs/#api/materials/MeshStandardMaterial>

alphaMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

aoMap


```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

aoMapIntensity

```
CFloat(1, allow_none=False).tag(sync=True)
```

bumpMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

bumpScale

```
CFloat(1, allow_none=False).tag(sync=True)
```

color

```
Unicode("#ffffff", allow_none=False).tag(sync=True)
```

defines

```
Dict(default_value={"STANDARD":""}, allow_none=True).tag(sync=True)
```

displacementMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

displacementScale

```
CFloat(1, allow_none=False).tag(sync=True)
```

displacementBias

```
CFloat(0, allow_none=False).tag(sync=True)
```

emissive

```
Unicode("#000000", allow_none=False).tag(sync=True)
```

emissiveMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

emissiveIntensity

```
CFloat(1, allow_none=False).tag(sync=True)
```

envMap

```
Instance(CubeTexture, allow_none=True).tag(sync=True, **widget_serialization)
```

envMapIntensity

```
CFloat(1, allow_none=False).tag(sync=True)
```

lightMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

lightMapIntensity

```
CFloat(1, allow_none=False).tag(sync=True)
```

map

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

metalness

```
CFloat(0.5, allow_none=False).tag(sync=True)
```

metalnessMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

morphTargets

```
Bool(False, allow_none=False).tag(sync=True)
```

morphNormals

```
Bool(False, allow_none=False).tag(sync=True)
```

normalMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

normalScale

```
Vector2(default_value=[1,1]).tag(sync=True)
```

refractionRatio

```
CFloat(0.98, allow_none=False).tag(sync=True)
```

roughness

```
CFloat(0.5, allow_none=False).tag(sync=True)
```

roughnessMap

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

skinning

```
Bool(False, allow_none=False).tag(sync=True)
```

wireframe

```
Bool(False, allow_none=False).tag(sync=True)
```

wireframeLinecap

```
Unicode("round", allow_none=False).tag(sync=True)
```

wireframeLinejoin

```
Unicode("round", allow_none=False).tag(sync=True)
```

wireframeLinewidth

```
CFloat(1, allow_none=False).tag(sync=True)
```

type

```
Unicode("MeshStandardMaterial", allow_none=False).tag(sync=True)
```

alphaMap = Instance()

a Texture or None

aoMap = Instance()

a Texture or None

aoMapIntensity = CFloat(1)

a float

bumpMap = Instance()

a Texture or None

bumpScale = **CFloat**(1)
a float

color = **Unicode**('#ffffff')
a unicode string

defines = **Dict**()
a dict or None with elements of any type

displacementBias = **CFloat**(0)
a float

displacementMap = **Instance**()
a Texture or None

displacementScale = **CFloat**(1)
a float

emissive = **Unicode**('#000000')
a unicode string

emissiveIntensity = **CFloat**(1)
a float

emissiveMap = **Instance**()
a Texture or None

envMap = **Instance**()
a CubeTexture or None

envMapIntensity = **CFloat**(1)
a float

lightMap = **Instance**()
a Texture or None

lightMapIntensity = **CFloat**(1)
a float

map = **Instance**()
a Texture or None

metalness = **CFloat**(0.5)
a float

metalnessMap = **Instance**()
a Texture or None

morphNormals = **Bool**(False)
a boolean

morphTargets = **Bool**(False)
a boolean

normalMap = **Instance**()
a Texture or None

normalScale = **Vector2**((0, 0))
a tuple of any type

refractionRatio = **CFloat**(0.98)
a float

```

roughness = CFloat(0.5)
    a float
roughnessMap = Instance()
    a Texture or None
skinning = Bool(False)
    a boolean
type = Unicode('MeshStandardMaterial')
    a unicode string
wireframe = Bool(False)
    a boolean
wireframeLinecap = Unicode('round')
    a unicode string
wireframeLinejoin = Unicode('round')
    a unicode string
wireframeLinewidth = CFloat(1)
    a float

```

MeshToonMaterial

```

class pythreejs.MeshToonMaterial
    MeshToonMaterial

```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/MeshToonMaterial>

Inherits *MeshPhongMaterial*.

Three.js docs: <https://threejs.org/docs/#api/materials/MeshToonMaterial>

```

gradientMap

```

```

Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)

```

```

type

```

```

Unicode("MeshToonMaterial", allow_none=False).tag(sync=True)

```

```

gradientMap = Instance()
    a Texture or None

```

```

type = Unicode('MeshToonMaterial')
    a unicode string

```

PointsMaterial

```

class pythreejs.PointsMaterial
    PointsMaterial

```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/PointsMaterial>

Inherits *Material*.

Three.js docs: <https://threejs.org/docs/#api/materials/PointsMaterial>

color

```
Unicode("#ffffff", allow_none=False).tag(sync=True)
```

lights

```
Bool(False, allow_none=False).tag(sync=True)
```

map

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

size

```
CFloat(1, allow_none=False).tag(sync=True)
```

sizeAttenuation

```
Bool(True, allow_none=False).tag(sync=True)
```

type

```
Unicode("PointsMaterial", allow_none=False).tag(sync=True)
```

color = Unicode('#ffffff')
a unicode string

lights = Bool(False)
a boolean

map = Instance()
a Texture or None

size = CFloat(1)
a float

sizeAttenuation = Bool(True)
a boolean

type = Unicode('PointsMaterial')
a unicode string

RawShaderMaterial

class pythreejs.RawShaderMaterial
RawShaderMaterial

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/RawShaderMaterial>

Inherits *ShaderMaterial*.

Three.js docs: <https://threejs.org/docs/#api/materials/RawShaderMaterial>

type

```
Unicode("RawShaderMaterial", allow_none=False).tag(sync=True)
```

```
type = Unicode('RawShaderMaterial')
a unicode string
```

ShaderMaterial

```
class pythreejs.ShaderMaterial
```

```
ShaderMaterial
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/ShaderMaterial>

Inherits *Material*.

Three.js docs: <https://threejs.org/docs/#api/materials/ShaderMaterial>

uniforms

```
Dict(default_value={}, allow_none=False).tag(sync=True)
```

clipping

```
Bool(False, allow_none=False).tag(sync=True)
```

extensions

```
Dict(default_value={}, allow_none=False).tag(sync=True)
```

fog

```
Bool(False, allow_none=False).tag(sync=True)
```

fragmentShader

```
Unicode('', allow_none=False).tag(sync=True)
```

lights

```
Bool(False, allow_none=False).tag(sync=True)
```

linewidth

```
CFloat(1, allow_none=False).tag(sync=True)
```

morphNormals

```
Bool(False, allow_none=False).tag(sync=True)
```

morphTargets

```
Bool(False, allow_none=False).tag(sync=True)
```

flatShading

```
Bool(False, allow_none=False).tag(sync=True)
```

skinning

```
Bool(False, allow_none=False).tag(sync=True)
```

uniformsNeedUpdate

```
Bool(False, allow_none=False).tag(sync=True)
```

vertexShader

```
Unicode('', allow_none=False).tag(sync=True)
```

wireframe

```
Bool(False, allow_none=False).tag(sync=True)
```

wireframeLinewidth

```
CFloat(1, allow_none=False).tag(sync=True)
```

type

```
Unicode("ShaderMaterial", allow_none=False).tag(sync=True)
```

clipping = Bool(False)

a boolean

extensions = Dict()

a dict with elements of any type

flatShading = Bool(False)

a boolean

fog = Bool(False)

a boolean

fragmentShader = Unicode('')

a unicode string


```

lights = Bool(False)
    a boolean

linewidth = CFloat(1)
    a float

morphNormals = Bool(False)
    a boolean

morphTargets = Bool(False)
    a boolean

skinning = Bool(False)
    a boolean

type = Unicode('ShaderMaterial')
    a unicode string

uniforms = Dict()
    a dict with elements of any type

uniformsNeedUpdate = Bool(False)
    a boolean

vertexShader = Unicode('')
    a unicode string

wireframe = Bool(False)
    a boolean

wireframeLinewidth = CFloat(1)
    a float

```

ShadowMaterial

```

class pythreejs.ShadowMaterial
    ShadowMaterial

```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/ShadowMaterial>

Inherits *ShaderMaterial*.

Three.js docs: <https://threejs.org/docs/#api/materials/ShadowMaterial>

lights

```

Bool(True, allow_none=False).tag(sync=True)

```

transparent

```

Bool(True, allow_none=False).tag(sync=True)

```

type

```

Unicode("ShadowMaterial", allow_none=False).tag(sync=True)

```

```

lights = Bool(True)
    a boolean

```

```
transparent = Bool(True)
    a boolean

type = Unicode('ShadowMaterial')
    a unicode string
```

SpriteMaterial

```
class pythreejs.SpriteMaterial
    SpriteMaterial
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/materials/SpriteMaterial>

Inherits *Material*.

Three.js docs: <https://threejs.org/docs/#api/materials/SpriteMaterial>

color

```
Unicode("#ffffff", allow_none=False).tag(sync=True)
```

fog

```
Bool(False, allow_none=False).tag(sync=True)
```

lights

```
Bool(False, allow_none=False).tag(sync=True)
```

map

```
Instance(Texture, allow_none=True).tag(sync=True, **widget_serialization)
```

rotation

```
CFloat(0, allow_none=False).tag(sync=True)
```

type

```
Unicode("SpriteMaterial", allow_none=False).tag(sync=True)
```

```
color = Unicode('#ffffff')
    a unicode string
```

```
fog = Bool(False)
    a boolean
```

```
lights = Bool(False)
    a boolean
```

```
map = Instance()
    a Texture or None
```

```
rotation = CFloat(0)
    a float

type = Unicode('SpriteMaterial')
    a unicode string
```

2.5.13 math

interpolants

CubicInterpolant

```
class pythreejs.CubicInterpolant
    CubicInterpolant
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/interpolants/CubicInterpolant>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/interpolants/CubicInterpolant>

DiscreteInterpolant

```
class pythreejs.DiscreteInterpolant
    DiscreteInterpolant
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/interpolants/DiscreteInterpolant>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/interpolants/DiscreteInterpolant>

LinearInterpolant

```
class pythreejs.LinearInterpolant
    LinearInterpolant
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/interpolants/LinearInterpolant>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/interpolants/LinearInterpolant>

QuaternionLinearInterpolant

```
class pythreejs.QuaternionLinearInterpolant
    QuaternionLinearInterpolant
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/interpolants/QuaternionLinearInterpolant>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/interpolants/QuaternionLinearInterpolant>

Box2

class pythreejs.Box2 (min=[0,0], max=[0,0],)
Box2

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/Box2>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/Box2>

min

```
Vector2 (default_value=[0,0]) .tag (sync=True)
```

max

```
Vector2 (default_value=[0,0]) .tag (sync=True)
```

max = Vector2 ((0, 0))
a tuple of any type

min = Vector2 ((0, 0))
a tuple of any type

Box3

class pythreejs.Box3 (min=[0,0,0], max=[0,0,0],)
Box3

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/Box3>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/Box3>

min

```
Vector3 (default_value=[0,0,0]) .tag (sync=True)
```

max

```
Vector3 (default_value=[0,0,0]) .tag (sync=True)
```

max = Vector3 ((0, 0, 0))
a tuple of any type

min = Vector3 ((0, 0, 0))
a tuple of any type

Cylindrical

class pythreejs.Cylindrical (radius=1, theta=0, y=0)
Cylindrical

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/Cylindrical>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/Cylindrical>

radius

```
CFloat(1, allow_none=False).tag(sync=True)
```

theta

```
CFloat(0, allow_none=False).tag(sync=True)
```

y

```
CFloat(0, allow_none=False).tag(sync=True)
```

radius = CFloat(1)

a float

theta = CFloat(0)

a float

y = CFloat(0)

a float

Frustum

class pythreejs.**Frustum** (*p0=None, p1=None, p2=None, p3=None, p4=None, p5=None*)

Frustum

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/Frustum>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/Frustum>

p0

```
Instance(Plane, allow_none=True).tag(sync=True, **widget_serialization)
```

p1

```
Instance(Plane, allow_none=True).tag(sync=True, **widget_serialization)
```

p2

```
Instance(Plane, allow_none=True).tag(sync=True, **widget_serialization)
```

p3

```
Instance(Plane, allow_none=True).tag(sync=True, **widget_serialization)
```

p4

```
Instance(Plane, allow_none=True).tag(sync=True, **widget_serialization)
```

p5

```
Instance(Plane, allow_none=True).tag(sync=True, **widget_serialization)
```

p0 = Instance()
a Plane or None

p1 = Instance()
a Plane or None

p2 = Instance()
a Plane or None

p3 = Instance()
a Plane or None

p4 = Instance()
a Plane or None

p5 = Instance()
a Plane or None

Interpolant

class pythreejs.**Interpolant**

Interpolant

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/Interpolant>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/Interpolant>

Line3

class pythreejs.**Line3** (*start=[0,0,0], end=[0,0,0],*)

Line3

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/Line3>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/Line3>

start

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

end

```
Vector3(default_value=[0, 0, 0]).tag(sync=True)
```

end = **Vector3**((0, 0, 0))

a tuple of any type

start = **Vector3**((0, 0, 0))

a tuple of any type

Math

class pythreejs.**Math**

Math

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/Math>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/Math>

Plane

class pythreejs.**Plane** (*normal*=[0,0,0], *constant*=0,)

Plane

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/Plane>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/Plane>

normal

```
Vector3(default_value=[0, 0, 0]).tag(sync=True)
```

constant

```
CFloat(0, allow_none=False).tag(sync=True)
```

constant = **CFloat**(0)

a float

normal = **Vector3**((0, 0, 0))

a tuple of any type

Quaternion

class pythreejs.**Quaternion** (*x*=0, *y*=0, *z*=0, *w*=1)

Quaternion

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/Quaternion>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/Quaternion>

x

```
CFloat(0, allow_none=False).tag(sync=True)
```

y

```
CFloat(0, allow_none=False).tag(sync=True)
```

z

```
CFloat(0, allow_none=False).tag(sync=True)
```

w

```
CFloat(1, allow_none=False).tag(sync=True)
```

w = CFloat(1)
a float

x = CFloat(0)
a float

y = CFloat(0)
a float

z = CFloat(0)
a float

Ray

class pythreejs.**Ray** (*origin*=[0,0,0], *direction*=[0,0,0],)

Ray

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/Ray>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/Ray>

origin

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

direction

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

direction = Vector3((0, 0, 0))
a tuple of any type

origin = Vector3((0, 0, 0))
a tuple of any type

Sphere

class pythreejs.**Sphere** (*center*=[0,0,0], *radius*=0,)
Sphere

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/Sphere>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/Sphere>

center

```
Vector3 (default_value=[0,0,0]) .tag (sync=True)
```

radius

```
CFloat (0, allow_none=False) .tag (sync=True)
```

center = **Vector3**((0, 0, 0))
a tuple of any type

radius = **CFloat**(0)
a float

Spherical

class pythreejs.**Spherical**
Spherical

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/Spherical>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/Spherical>

Triangle

class pythreejs.**Triangle** (*a*=[0,0,0], *b*=[0,0,0], *c*=[0,0,0],)
Triangle

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/math/Triangle>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/math/Triangle>

a

```
Vector3 (default_value=[0,0,0]) .tag (sync=True)
```

b

```
Vector3 (default_value=[0,0,0]) .tag (sync=True)
```

c

```
Vector3(default_value=[0,0,0]).tag(sync=True)
```

a = Vector3((0, 0, 0))
a tuple of any type

b = Vector3((0, 0, 0))
a tuple of any type

c = Vector3((0, 0, 0))
a tuple of any type

2.5.14 objects

Blackbox

class pythreejs.**Blackbox**

A widget with unsynced children.

This widget allows extension authors to expose scene control of a given three object, without attempting to sync its children. This makes it possible for a library to give access to an outer object, without exposing the full object three, and can be useful in avoiding possibly heavy sync operations.

This widget has some manual overrides on the Python side.

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/objects/Blackbox>

type

```
Unicode("Blackbox", allow_none=False).tag(sync=True)
```

children = None

Bone

class pythreejs.**Bone**

Bone

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/objects/Bone>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/objects/Bone>

type

```
Unicode("Bone", allow_none=False).tag(sync=True)
```

type = Unicode('Bone')
a unicode string

CloneArray

class pythreejs.**CloneArray** (*original=None, positions=[], merge=False*)
CloneArray

Autogenerated by generate-wrappers.js This class is a custom class for pythreejs, with no direct corresponding class in three.js.

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/objects/CloneArray>

original

```
Instance(Object3D, allow_none=True).tag(sync=True, **widget_serialization)
```

positions

```
List(trait=List()).tag(sync=True)
```

merge

```
Bool(False, allow_none=False).tag(sync=True)
```

type

```
Unicode("CloneArray", allow_none=False).tag(sync=True)
```

merge = Bool(False)
a boolean

original = Instance()
an Object3D or None

positions = List()
a list with values that are: a list

type = Unicode('CloneArray')
a unicode string

Group

class pythreejs.**Group**
Group

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/objects/Group>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/objects/Group>

type

```
Unicode("Group", allow_none=False).tag(sync=True)
```

```
type = Unicode('Group')
    a unicode string
```

LOD

```
class pythreejs.LOD
    LOD
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/objects/LOD>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/objects/LOD>

LineLoop

```
class pythreejs.LineLoop (geometry=None, material=None)
    LineLoop
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/objects/LineLoop>

Inherits *Line*.

Three.js docs: <https://threejs.org/docs/#api/objects/LineLoop>

```
type
```

```
Unicode("LineLoop", allow_none=False).tag(sync=True)
```

```
type = Unicode('LineLoop')
    a unicode string
```

LineSegments

```
class pythreejs.LineSegments (geometry=None, material=None)
    LineSegments
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/objects/LineSegments>

Inherits *Line*.

Three.js docs: <https://threejs.org/docs/#api/objects/LineSegments>

```
type
```

```
Unicode("LineSegments", allow_none=False).tag(sync=True)
```

```
type = Unicode('LineSegments')
    a unicode string
```

Line

```
class pythreejs.Line (geometry=None, material=None)
    Line
```

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/objects/Line>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/objects/Line>

material

```
Instance(Material, allow_none=True).tag(sync=True, **widget_serialization)
```

geometry

```
Union([
    Instance(BaseGeometry, allow_none=True),
    Instance(BaseBufferGeometry, allow_none=True)
]).tag(sync=True, **widget_serialization)
```

type

```
Unicode("Line", allow_none=False).tag(sync=True)
```

geometry = Union()

a BaseGeometry or None or a BaseBufferGeometry or None

material = Instance()

a Material or None

type = Unicode('Line')

a unicode string

Mesh

class pythreejs.**Mesh**(*geometry=None, material=[]*)

Mesh

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/objects/Mesh>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/objects/Mesh>

material

```
Union([Instance(Material), Tuple()]).tag(sync=True, **widget_serialization)
```

geometry

```
Union([
    Instance(BaseGeometry, allow_none=False),
    Instance(BaseBufferGeometry, allow_none=False)
]).tag(sync=True, **widget_serialization)
```

drawMode

```
Enum(DrawModes, "TrianglesDrawMode", allow_none=False).tag(sync=True)
```

morphTargetInfluences

```
List().tag(sync=True)
```

type

```
Unicode("Mesh", allow_none=False).tag(sync=True)
```

```
drawMode = Enum('TrianglesDrawMode')
    any of ['TriangleFanDrawMode', 'TriangleStripDrawMode', 'TrianglesDrawMode']

geometry = Union()
    a BaseGeometry or a BaseBufferGeometry

material = Union()
    a Material or a tuple

morphTargetInfluences = List()
    a list of any type

type = Unicode('Mesh')
    a unicode string
```

Points

```
class pythreejs.Points(geometry=None, material=None)
    Points

    Autogenerated by generate-wrappers.js See https://threejs.org/docs/#api/objects/Points

    Inherits Object3D.

    Three.js docs: https://threejs.org/docs/#api/objects/Points
```

material

```
Instance(Material, allow_none=False).tag(sync=True, **widget_serialization)
```

geometry

```
Union([
    Instance(BaseGeometry, allow_none=False),
    Instance(BaseBufferGeometry, allow_none=False)
]).tag(sync=True, **widget_serialization)
```

type

```
Unicode("Points", allow_none=False).tag(sync=True)
```

```
geometry = Union()
    a BaseGeometry or a BaseBufferGeometry
```

```
material = Instance()
    a Material

type = Unicode('Points')
    a unicode string
```

Skeleton

```
class pythreejs.Skeleton(bones=[])
    Skeleton

    Autogenerated by generate-wrappers.js See https://threejs.org/docs/#api/objects/Skeleton

    Inherits ThreeWidget.

    Three.js docs: https://threejs.org/docs/#api/objects/Skeleton

bones
```

```
Tuple().tag(sync=True, **widget_serialization)
```

```
bones = Tuple()
    a tuple of any type
```

SkinnedMesh

```
class pythreejs.SkinnedMesh(geometry=None, material=[])
    SkinnedMesh

    Autogenerated by generate-wrappers.js See https://threejs.org/docs/#api/objects/SkinnedMesh

    Inherits Mesh.

    Three.js docs: https://threejs.org/docs/#api/objects/SkinnedMesh

bindMode
```

```
Unicode("attached", allow_none=False).tag(sync=True)
```

bindMatrix

```
Matrix4(default_value=[1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1]).tag(sync=True)
```

skeleton

```
Instance(Skeleton, allow_none=True).tag(sync=True, **widget_serialization)
```

type

```
Unicode("SkinnedMesh", allow_none=False).tag(sync=True)
```

```
bindMatrix = Matrix4((1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1))
    a tuple of any type
```

bindMode = Unicode('attached')
 a unicode string

skeleton = Instance()
 a Skeleton or None

type = Unicode('SkinnedMesh')
 a unicode string

Sprite

class pythreejs.**Sprite** (*material=None*)
 Sprite

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/objects/Sprite>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/objects/Sprite>

material

```
Instance(SpriteMaterial, allow_none=True).tag(sync=True, **widget_
↪serialization)
```

center

```
Vector2(default_value=[0.5, 0.5]).tag(sync=True)
```

type

```
Unicode("Sprite", allow_none=False).tag(sync=True)
```

center = Vector2((0, 0))
 a tuple of any type

material = Instance()
 a SpriteMaterial or None

type = Unicode('Sprite')
 a unicode string

2.5.15 renderers

webgl

WebGLBufferRenderer

class pythreejs.**WebGLBufferRenderer**
 WebGLBufferRenderer

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/renderers/webgl/WebGLBufferRenderer>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/renderers/webgl/WebGLBufferRenderer>

WebGLCapabilities

class `pythreejs.WebGLCapabilities`

`WebGLCapabilities`

Autogenerated by `generate-wrappers.js` See <https://threejs.org/docs/#api/renderers/webgl/WebGLCapabilities>

Inherits `ThreeWidget`.

Three.js docs: <https://threejs.org/docs/#api/renderers/webgl/WebGLCapabilities>

WebGLExtensions

class `pythreejs.WebGLExtensions`

`WebGLExtensions`

Autogenerated by `generate-wrappers.js` See <https://threejs.org/docs/#api/renderers/webgl/WebGLExtensions>

Inherits `ThreeWidget`.

Three.js docs: <https://threejs.org/docs/#api/renderers/webgl/WebGLExtensions>

WebGLGeometries

class `pythreejs.WebGLGeometries`

`WebGLGeometries`

Autogenerated by `generate-wrappers.js` See <https://threejs.org/docs/#api/renderers/webgl/WebGLGeometries>

Inherits `ThreeWidget`.

Three.js docs: <https://threejs.org/docs/#api/renderers/webgl/WebGLGeometries>

WebGLIndexedBufferRenderer

class `pythreejs.WebGLIndexedBufferRenderer`

`WebGLIndexedBufferRenderer`

Autogenerated by `generate-wrappers.js` See <https://threejs.org/docs/#api/renderers/webgl/WebGLIndexedBufferRenderer>

Inherits `ThreeWidget`.

Three.js docs: <https://threejs.org/docs/#api/renderers/webgl/WebGLIndexedBufferRenderer>

WebGLLights

class `pythreejs.WebGLLights`

`WebGLLights`

Autogenerated by `generate-wrappers.js` See <https://threejs.org/docs/#api/renderers/webgl/WebGLLights>

Inherits `ThreeWidget`.

Three.js docs: <https://threejs.org/docs/#api/renderers/webgl/WebGLLights>

WebGLObjects

class pythreejs.**WebGLObjects**

WebGLObjects

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/renderers/webgl/WebGLObjects>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/renderers/webgl/WebGLObjects>

WebGLProgram

class pythreejs.**WebGLProgram**

WebGLProgram

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/renderers/webgl/WebGLProgram>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/renderers/webgl/WebGLProgram>

WebGLPrograms

class pythreejs.**WebGLPrograms**

WebGLPrograms

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/renderers/webgl/WebGLPrograms>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/renderers/webgl/WebGLPrograms>

WebGLProperties

class pythreejs.**WebGLProperties**

WebGLProperties

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/renderers/webgl/WebGLProperties>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/renderers/webgl/WebGLProperties>

WebGLShader

class pythreejs.**WebGLShader**

WebGLShader

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/renderers/webgl/WebGLShader>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/renderers/webgl/WebGLShader>

WebGLShadowMap

class pythreejs.WebGLShadowMap

WebGLShadowMap

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/renderers/webgl/WebGLShadowMap>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/renderers/webgl/WebGLShadowMap>

enabled

```
Bool(False, allow_none=False).tag(sync=True)
```

type

```
Enum(ShadowTypes, "PCFShadowMap", allow_none=False).tag(sync=True)
```

enabled = Bool(False)

a boolean

type = Enum('PCFShadowMap')

any of ['BasicShadowMap', 'PCFShadowMap', 'PCFSofShadowMap']

WebGLState

class pythreejs.WebGLState

WebGLState

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/renderers/webgl/WebGLState>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/renderers/webgl/WebGLState>

WebGLRenderTargetCube

class pythreejs.WebGLRenderTargetCube

WebGLRenderTargetCube

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/renderers/WebGLRenderTargetCube>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/renderers/WebGLRenderTargetCube>

WebGLRenderTarget

class pythreejs.WebGLRenderTarget

WebGLRenderTarget

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/renderers/WebGLRenderTarget>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/renderers/WebGLRenderTarget>

2.5.16 scenes

FogExp2

class pythreejs.**FogExp2** (*color="white", density=0.00025*)

FogExp2

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/scenes/FogExp2>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/scenes/FogExp2>

name

```
Unicode('', allow_none=False).tag(sync=True)
```

color

```
Unicode("white", allow_none=False).tag(sync=True)
```

density

```
CFloat(0.00025, allow_none=False).tag(sync=True)
```

color = Unicode('white')

a unicode string

density = CFloat(0.00025)

a float

name = Unicode('')

a unicode string

Fog

class pythreejs.**Fog** (*color="white", near=1, far=1000*)

Fog

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/scenes/Fog>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/scenes/Fog>

name

```
Unicode('', allow_none=False).tag(sync=True)
```

color

```
Unicode("white", allow_none=False).tag(sync=True)
```

near

```
CFloat(1, allow_none=False).tag(sync=True)
```

far

```
CFloat(1000, allow_none=False).tag(sync=True)
```

color = Unicode('white')
a unicode string

far = CFloat(1000)
a float

name = Unicode('')
a unicode string

near = CFloat(1)
a float

Scene

class pythreejs.**Scene**

Scene

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/scenes/Scene>

Inherits *Object3D*.

Three.js docs: <https://threejs.org/docs/#api/scenes/Scene>

fog

```
Union([
    Instance(Fog, allow_none=True),
    Instance(FogExp2, allow_none=True)
]).tag(sync=True, **widget_serialization)
```

overrideMaterial

```
Instance(Material, allow_none=True).tag(sync=True, **widget_serialization)
```

autoUpdate

```
Bool(True, allow_none=False).tag(sync=True)
```

background

```
Unicode("#ffffff", allow_none=True).tag(sync=True)
```

type

```
Unicode("Scene", allow_none=False).tag(sync=True)
```

autoUpdate = Bool(True)
a boolean

background = Unicode('#ffffff')
a unicode string

fog = Union()
a Fog or None or a FogExp2 or None

overrideMaterial = Instance()
a Material or None

type = Unicode('Scene')
a unicode string

2.5.17 textures

CompressedTexture

class pythreejs.CompressedTexture
CompressedTexture

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/textures/CompressedTexture>

Inherits *Texture*.

Three.js docs: <https://threejs.org/docs/#api/textures/CompressedTexture>

CubeTexture

class pythreejs.CubeTexture (*images=[]*, *mapping="UVMapping"*,
wrapS="ClampToEdgeWrapping", *wrapT="ClampToEdgeWrapping"*,
magFilter="LinearFilter", *minFilter="LinearMipMapLinearFilter"*,
format="RGBAFormat", *type="UnsignedByteType"*, *anisotropy=1*)

CubeTexture

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/textures/CubeTexture>

Inherits *Texture*.

Three.js docs: <https://threejs.org/docs/#api/textures/CubeTexture>

images

```
List().tag(sync=True)
```

images = List()
a list of any type

DataTexture

class pythreejs.DataTexture (*data=None*, *format="RGBAFormat"*, *type="UnsignedByteType"*,
mapping="UVMapping", *wrapS="ClampToEdgeWrapping"*,
wrapT="ClampToEdgeWrapping", *magFilter="NearestFilter"*,
minFilter="NearestFilter", *anisotropy=1*)

This widget has some manual overrides on the Python side.

Inherits *Texture*.

Three.js docs: <https://threejs.org/docs/#api/textures/DataTexture>

data

```
WebGLDataUnion().tag(sync=True)
```

minFilter

```
Enum(Filters, "NearestFilter", allow_none=False).tag(sync=True)
```

magFilter

```
Enum(Filters, "NearestFilter", allow_none=False).tag(sync=True)
```

flipY

```
Bool(False, allow_none=False).tag(sync=True)
```

generateMipmaps

```
Bool(False, allow_none=False).tag(sync=True)
```

DepthTexture

```
class pythreejs.DepthTexture (width=0, height=0, type="UnsignedShortType",
                               wrapS="ClampToEdgeWrapping", wrapT="ClampToEdgeWrapping",
                               magFilter="NearestFilter", minFilter="NearestFilter",
                               anisotropy=1, format="DepthFormat")
```

DepthTexture

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/textures/DepthTexture>

Inherits *Texture*.

Three.js docs: <https://threejs.org/docs/#api/textures/DepthTexture>

width

```
CInt(0, allow_none=False).tag(sync=True)
```

height

```
CInt(0, allow_none=False).tag(sync=True)
```

format

```
Enum(DepthFormats, "DepthFormat", allow_none=False).tag(sync=True)
```

type

```
Enum(DataTypes, "UnsignedShortType", allow_none=False).tag(sync=True)
```

minFilter

```
Enum(Filters, "NearestFilter", allow_none=False).tag(sync=True)
```

magFilter

```
Enum(Filters, "NearestFilter", allow_none=False).tag(sync=True)
```

flipY

```
Bool(False, allow_none=False).tag(sync=True)
```

generateMipmaps

```
Bool(False, allow_none=False).tag(sync=True)
```

flipY = Bool(False)

a boolean

format = Enum('DepthFormat')

any of ['DepthFormat', 'DepthStencilFormat']

generateMipmaps = Bool(False)

a boolean

height = CInt(0)

an int

magFilter = Enum('NearestFilter')

any of ['LinearFilter', 'LinearMipMapLinearFilter', 'LinearMipMapNearestFilter', 'NearestFilter', 'NearestMipMapLinearFilter', 'NearestMipMapNearestFilter']

minFilter = Enum('NearestFilter')

any of ['LinearFilter', 'LinearMipMapLinearFilter', 'LinearMipMapNearestFilter', 'NearestFilter', 'NearestMipMapLinearFilter', 'NearestMipMapNearestFilter']

type = Enum('UnsignedShortType')

any of ['ByteType', 'FloatType', 'HalfFloatType', 'IntType', 'ShortType', 'UnsignedByteType', 'UnsignedIntType', 'UnsignedShortType']

width = CInt(0)

an int

ImageTexture

```
class pythreejs.ImageTexture (imageUri="", mapping="UVMapping",
                               wrapS="ClampToEdgeWrapping", wrapT="ClampToEdgeWrapping",
                               magFilter="LinearFilter", minFilter="LinearMipMapLinearFilter",
                               format="RGBAFormat",
                               type="UnsignedByteType", anisotropy=1)
```

ImageTexture

Autogenerated by generate-wrappers.js This class is a custom class for pythreejs, with no direct corresponding class in three.js.

Inherits *Texture*.

Three.js docs: <https://threejs.org/docs/#api/textures/ImageTexture>

imageUri

```
Unicode('', allow_none=False).tag(sync=True)
```

imageUri = Unicode('')

a unicode string

TextTexture

```
class pythreejs.TextTexture (string="")
```

TextTexture

Autogenerated by generate-wrappers.js This class is a custom class for pythreejs, with no direct corresponding class in three.js.

Inherits *Texture*.

Three.js docs: <https://threejs.org/docs/#api/textures/TextTexture>

color

```
Unicode("white", allow_none=False).tag(sync=True)
```

fontFace

```
Unicode("Arial", allow_none=False).tag(sync=True)
```

size

```
CInt(12, allow_none=False).tag(sync=True)
```

string

```
Unicode('', allow_none=False).tag(sync=True)
```

squareTexture

```
Bool(True, allow_none=False).tag(sync=True)
```

```
color = Unicode('white')
    a unicode string

fontFace = Unicode('Arial')
    a unicode string

size = CInt(12)
    an int

squareTexture = Bool(True)
    a boolean

string = Unicode('')
    a unicode string
```

Texture

class pythreejs.Texture

Texture

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/textures/Texture>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/textures/Texture>

name

```
Unicode('', allow_none=False).tag(sync=True)
```

mapping

```
Enum(MappingModes, "UVMapping", allow_none=False).tag(sync=True)
```

wrapS

```
Enum(WrappingModes, "ClampToEdgeWrapping", allow_none=False).tag(sync=True)
```

wrapT

```
Enum(WrappingModes, "ClampToEdgeWrapping", allow_none=False).tag(sync=True)
```

magFilter

```
Enum(Filters, "LinearFilter", allow_none=False).tag(sync=True)
```

minFilter

```
Enum(Filters, "LinearMipMapLinearFilter", allow_none=False).tag(sync=True)
```

format

```
Enum(PixelFormats, "RGBAFormat", allow_none=False).tag(sync=True)
```

type

```
Enum(DataTypes, "UnsignedByteType", allow_none=False).tag(sync=True)
```

anisotropy

```
CFloat(1, allow_none=False).tag(sync=True)
```

repeat

```
Vector2(default_value=[1,1]).tag(sync=True)
```

offset

```
Vector2(default_value=[0,0]).tag(sync=True)
```

generateMipmaps

```
Bool(True, allow_none=False).tag(sync=True)
```

premultiplyAlpha

```
Bool(False, allow_none=False).tag(sync=True)
```

flipY

```
Bool(True, allow_none=False).tag(sync=True)
```

unpackAlignment

```
CInt(4, allow_none=False).tag(sync=True)
```

encoding

```
Enum(TextureEncodings, "LinearEncoding", allow_none=False).tag(sync=True)
```

version

```
CInt(0, allow_none=False).tag(sync=True)
```

rotation

```
CFloat(0, allow_none=False).tag(sync=True)
```

anisotropy = CFloat(1)

a float

encoding = Enum('LinearEncoding')

any of ['GammaEncoding', 'LinearEncoding', 'LogLuvEncoding', 'RGBDEncoding', 'RGBEEncoding', 'RGBM16Encoding', 'RGBM7Encoding', 'sRGBEncoding']

flipY = Bool(True)

a boolean

format = Enum('RGBAFormat')

any of ['AlphaFormat', 'DepthFormat', 'DepthStencilFormat', 'LuminanceAlphaFormat', 'LuminanceFormat', 'RGBAFormat', 'RGBEFormat', 'RGBFormat']

generateMipmaps = Bool(True)

a boolean

magFilter = Enum('LinearFilter')

any of ['LinearFilter', 'LinearMipMapLinearFilter', 'LinearMipMapNearestFilter', 'NearestFilter', 'NearestMipMapLinearFilter', 'NearestMipMapNearestFilter']

mapping = Enum('UVMapping')

any of ['CubeReflectionMapping', 'CubeRefractionMapping', 'CubeUVReflectionMapping', 'CubeUVRefractionMapping', 'EquirectangularReflectionMapping', 'EquirectangularRefractionMapping', 'SphericalReflectionMapping', 'UVMapping']

minFilter = Enum('LinearMipMapLinearFilter')

any of ['LinearFilter', 'LinearMipMapLinearFilter', 'LinearMipMapNearestFilter', 'NearestFilter', 'NearestMipMapLinearFilter', 'NearestMipMapNearestFilter']

name = Unicode('')

a unicode string

offset = Vector2((0, 0))

a tuple of any type

premultiplyAlpha = Bool(False)

a boolean

repeat = Vector2((0, 0))

a tuple of any type

rotation = CFloat(0)

a float

type = Enum('UnsignedByteType')

any of ['ByteType', 'FloatType', 'HalfFloatType', 'IntType', 'ShortType', 'UnsignedByteType', 'UnsignedIntType', 'UnsignedShortType']

unpackAlignment = CInt(4)

an int

version = CInt(0)

an int

wrapS = Enum('ClampToEdgeWrapping')

any of ['ClampToEdgeWrapping', 'MirroredRepeatWrapping', 'RepeatWrapping']

wrapT = Enum('ClampToEdgeWrapping')

any of ['ClampToEdgeWrapping', 'MirroredRepeatWrapping', 'RepeatWrapping']

VideoTexture

class pythreejs.VideoTexture

VideoTexture

Autogenerated by generate-wrappers.js See <https://threejs.org/docs/#api/textures/VideoTexture>

Inherits *ThreeWidget*.

Three.js docs: <https://threejs.org/docs/#api/textures/VideoTexture>

2.5.18 traits

class pythreejs.traits.Euler (*default_value=traitlets.Undefined, **kwargs*)

A trait for a set of Euler angles.

Expressed as a tuple of tree floats (the angles), and the order as a string. See the three.js docs for futher details.

default_value = (0, 0, 0, 'XYZ')

info_text = 'a set of Euler angles'

class pythreejs.traits.Face3 (***kwargs*)

A trait for a named tuple corresponding to a three.js Face3.

Accepts named tuples with the field names: ('a', 'b', 'c', 'normal', 'color', 'materialIndex')

info_text = 'a named tuple representing a Face3'

class

alias of *Face3*

class pythreejs.traits.Matrix3 (*trait=<class traitlets.traitlets.CFloat>, de-
fault_value=traitlets.Undefined, **kwargs*)

A trait for a 9-tuple corresponding to a three.js Matrix3.

default_value = (1, 0, 0, 0, 1, 0, 0, 0, 1)

info_text = 'a three-by-three matrix (9 element tuple)'

class pythreejs.traits.Matrix4 (*trait=<class traitlets.traitlets.CFloat>, de-
fault_value=traitlets.Undefined, **kwargs*)

A trait for a 16-tuple corresponding to a three.js Matrix4.

default_value = (1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1)

info_text = 'a four-by-four matrix (16 element tuple)'

class pythreejs.traits.Uninitialized

Placeholder sentinel used while waiting for a initialization via sync

class pythreejs.traits.Vector2 (*trait=<class traitlets.traitlets.CFloat>, de-
fault_value=traitlets.Undefined, **kwargs*)

A trait for a 2-tuple corresponding to a three.js Vector2.

default_value = (0, 0)

info_text = 'a two-element vector'

class pythreejs.traits.Vector3 (*trait=<class traitlets.traitlets.CFloat>, de-
fault_value=traitlets.Undefined, **kwargs*)

A trait for a 3-tuple corresponding to a three.js Vector3.

default_value = (0, 0, 0)

```

    info_text = 'a three-element vector'
class pythreejs.traits.Vector4 (trait=<class      'traitlets.traitlets.CFloat'>,      de-
                                fault_value=traitlets.Undefined, **kwargs)
    A trait for a 4-tuple corresponding to a three.js Vector4.
    default_value = (0, 0, 0, 0)
    info_text = 'a four-element vector'
class pythreejs.traits.WebGLDataUnion (default_value=traitlets.Undefined,      dtype=None,
                                        shape_constraint=None,      kw_array=None,
                                        kw_widget=None, **kwargs)
    A trait that accepts either a numpy array, or an NDArarrayWidget reference.
    Also constrains the use of 64-bit arrays, as this is not supported by WebGL.
    validate (obj, value)

```

2.6 Extending pythreejs

While you can do a lot with pythreejs out of the box, you might have some custom rendering you want to do, that would be more efficient to configure as a separate widget. To be able to integrate such objects with pythreejs, the following extension guide can be helpful.

2.6.1 Blackbox object

Pythreejs exports a *Blackbox* Widget, which inherits *Object3D*. The intention is for third-party widget libraries to inherit from it on both the Python and JS side. You would add the traits needed to set up your object, and have the JS side set up the corresponding three.js object. The three.js object itself would not be synced across the wire, which is why it is called a blackbox, but you can still manipulate it in a scene (transforming, putting it as a child, etc.). This can be very efficient e.g. for complex, generated objects, where the final three.js data would be prohibitively expensive to synchronize.

Example implementation

Below is an example implementation for rendering a crystal lattice. It takes a basis structure, and then tiles copies of this basis in $x/y/z$, potentially generating thousands of spheres.

Note: This example is not a good/optimized crystal structure viewer. It is merely used to convey the concept of a widget with a few parameters translating to something with potentially huge amounts of data/objects.

Python:

```

import traitlets
import pythreejs

class CubicLattice (pythreejs.Blackbox):
    _model_name: traitlets.Unicode('CubicLatticeModel').tag(sync=True)
    _model_module = traitlets.Unicode('my_module_name').tag(sync=True)

    basis = traitlets.List(
        trait=pythreejs.Vector3(),

```

(continues on next page)

(continued from previous page)

```

        default_value=[[0, 0, 0]],
        max_length=5
    ).tag(sync=True)

    repetitions = traitlets.List(
        trait=traitlets.Int(),
        default_value=[5, 5, 5],
        min_length=3,
        max_length=3
    ).tag(sync=True)

```

JavaScript:

```

import * as THREE from "three";

import {
    BlackboxModel
} from 'jupyter-threejs';

const atomGeometry = new THREE.SphereBufferGeometry(0.2, 16, 8);
const atomMaterials = [
    new THREE.MeshLambertMaterial({color: 'red'}),
    new THREE.MeshLambertMaterial({color: 'green'}),
    new THREE.MeshLambertMaterial({color: 'yellow'}),
    new THREE.MeshLambertMaterial({color: 'blue'}),
    new THREE.MeshLambertMaterial({color: 'cyan'}),
];

export class CubicLatticeModel extends BlackboxModel {
    defaults() {
        return {...super.defaults(), ...{
            _model_name: 'CubicLatticeModel',
            _model_module: 'my_module_name',
            basis: [[0, 0, 0]],
            repetitions: [5, 5, 5],
        }};
    }

    // This method is called to create the three.js object of the model:
    constructThreeObject() {
        const root = new THREE.Group();
        // Create the children of this group:
        // This is the part that is specific to this example
        this.createLattice(root);
        return root;
    }

    // This method is called whenever the model changes:
    onChange(model, options) {
        super.onChange(model, options);
        // If any of the parameters change, simply rebuild children:
        this.createLattice();
    }

    // Our custom method to build the lattice:
    createLattice(obj) {

```

(continues on next page)

(continued from previous page)

```

obj = obj || this.obj;

// Set up the basis to tile:
const basisInput = this.get('basis');
const basis = new THREE.Group();
for (let i=0; i < basisInput.length; ++i) {
    let mesh = new THREE.Mesh(atomGeometry, atomMaterials[i]);
    mesh.position.fromArray(basisInput[i]);
    basis.add(mesh);
}

// Tile in x, y, z:
const [nx, ny, nz] = this.get('repetitions');
const children = [];
for (let x = 0; x < nx; ++x) {
    for (let y = 0; y < ny; ++y) {
        for (let z = 0; z < nz; ++z) {
            let copy = basis.clone();
            copy.position.set(x, y, z);
            children.push(copy);
        }
    }
}

obj.remove(...obj.children);
obj.add(...children);
}
}

```

This code should then be wrapped up in a widget extension (see documentation from ipywidgets on how to do this).

Usage:

```

import pythreejs
from IPython.display import display
from my_module import CubicLattice

lattice = CubicLattice(basis=[[0,0,0], [0.5, 0.5, 0.5]])

# Preview the lattice directly:
display(lattice)

# Or put it in a scene:
width=600
height=400
key_light = pythreejs.DirectionalLight(position=[-5, 5, 3], intensity=0.7)
ambient_light = pythreejs.AmbientLight(color='#777777')

camera = pythreejs.PerspectiveCamera(
    position=[-5, 0, -5],
    children=[
        # Have the key light follow the camera:
        key_light
    ],
    aspect=width/height,
)
control = pythreejs.OrbitControls(controlling=camera)

```

(continues on next page)

(continued from previous page)

```
scene = pythreejs.Scene(children=[lattice, camera, ambient_light])

renderer = pythreejs.Renderer(camera=camera,
                              scene=scene,
                              controls=[control],
                              width=width, height=height)

display(renderer)
```

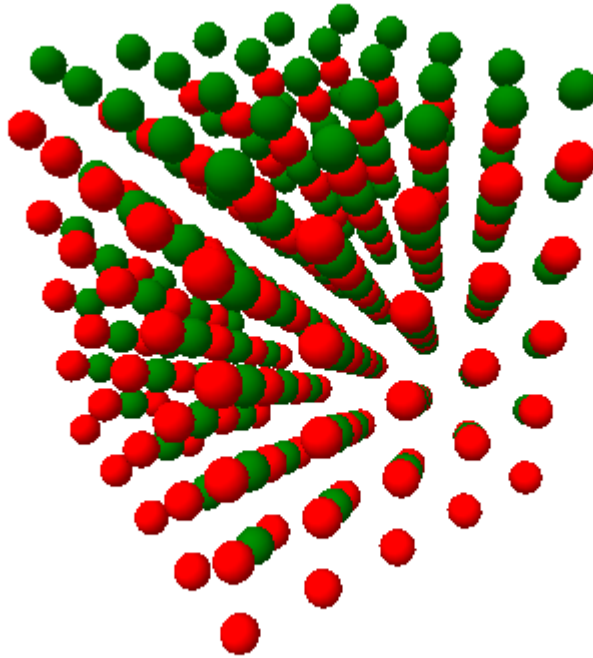


Fig. 1: Figure: Example view of the rendered lattice object.

2.7 Developer install

To install a developer version of pythreejs, you will first need to clone the repository:

```
git clone https://github.com/jupyter-widgets/pythreejs.git
cd pythreejs
```

Next, install it with a develop install using pip:

```
pip install -e .
```

If you are not planning on working on the JS/frontend code, you can simply install the extensions as you would for a *normal install*. For a JS develop install, you should link your extensions:

```
jupyter nbextension install [--sys-prefix / --user / --system] --symlink --py_
↳pythreejs

jupyter nbextension enable [--sys-prefix / --user / --system] --py pythreejs
```

with the appropriate flag. Or, if you are using Jupyterlab:

```
jupyter labextension link ./js
```

p

`pythreejs`, 18

`pythreejs.traits`, 161

Symbols

`_ref_geometry` (pythreejs.BufferGeometry attribute), 43
`_ref_geometry` (pythreejs.Geometry attribute), 45
`_store_ref` (pythreejs.BufferGeometry attribute), 43
`_store_ref` (pythreejs.Geometry attribute), 45

A

`a` (pythreejs.Triangle attribute), 141
`add()` (pythreejs.Object3D method), 50
`all` (pythreejs.Picker attribute), 37
`alphaMap` (pythreejs.MeshBasicMaterial attribute), 110
`alphaMap` (pythreejs.MeshDepthMaterial attribute), 113
`alphaMap` (pythreejs.MeshLambertMaterial attribute), 115
`alphaMap` (pythreejs.MeshPhongMaterial attribute), 119
`alphaMap` (pythreejs.MeshStandardMaterial attribute), 124
`alphaTest` (pythreejs.Material attribute), 107
`AmbientLight` (class in pythreejs), 95
`angle` (pythreejs.SpotLight attribute), 100
`AnimationAction` (class in pythreejs), 21
`AnimationClip` (class in pythreejs), 23
`AnimationLoader` (class in pythreejs), 101
`AnimationMixer` (class in pythreejs), 23
`AnimationObjectGroup` (class in pythreejs), 24
`AnimationUtils` (class in pythreejs), 24
`anisotropy` (pythreejs.Texture attribute), 159
`aoMap` (pythreejs.MeshBasicMaterial attribute), 110
`aoMap` (pythreejs.MeshLambertMaterial attribute), 115
`aoMap` (pythreejs.MeshPhongMaterial attribute), 119
`aoMap` (pythreejs.MeshStandardMaterial attribute), 124
`aoMapIntensity` (pythreejs.MeshBasicMaterial attribute), 110
`aoMapIntensity` (pythreejs.MeshLambertMaterial attribute), 115
`aoMapIntensity` (pythreejs.MeshPhongMaterial attribute), 119
`aoMapIntensity` (pythreejs.MeshStandardMaterial attribute), 125

`arc` (pythreejs.TorusBufferGeometry attribute), 78
`arc` (pythreejs.TorusGeometry attribute), 79
`ArcCurve` (class in pythreejs), 53
`array` (pythreejs.BufferAttribute attribute), 42
`array` (pythreejs.InterleavedBuffer attribute), 47
`ArrayCamera` (class in pythreejs), 26
`ArrowHelper` (class in pythreejs), 83
`aspect` (pythreejs.PerspectiveCamera attribute), 31
`aspect` (pythreejs.StereoCamera attribute), 32
`attributes` (pythreejs.BufferGeometry attribute), 42
`Audio` (class in pythreejs), 26
`AudioAnalyser` (class in pythreejs), 25
`AudioListener` (class in pythreejs), 26
`AudioLoader` (class in pythreejs), 101
`autoRotate` (pythreejs.OrbitControls attribute), 35
`autoRotateSpeed` (pythreejs.OrbitControls attribute), 35
`autoUpdate` (pythreejs.Scene attribute), 153
`AxesHelper` (class in pythreejs), 84

B

`b` (pythreejs.Triangle attribute), 141
`background` (pythreejs.Scene attribute), 153
`BaseBufferGeometry` (class in pythreejs), 41
`BaseGeometry` (class in pythreejs), 41
`bias` (pythreejs.LightShadow attribute), 97
`bindMatrix` (pythreejs.SkinnedMesh attribute), 147
`bindMode` (pythreejs.SkinnedMesh attribute), 147
`Blackbox` (class in pythreejs), 142
`blendDst` (pythreejs.Material attribute), 107
`blendDstAlpha` (pythreejs.Material attribute), 107
`blendEquation` (pythreejs.Material attribute), 107
`blendEquationAlpha` (pythreejs.Material attribute), 107
`blending` (pythreejs.Material attribute), 107
`blendSrc` (pythreejs.Material attribute), 107
`blendSrcAlpha` (pythreejs.Material attribute), 107
`Bone` (class in pythreejs), 142
`bones` (pythreejs.Skeleton attribute), 147
`BooleanKeyframeTrack` (class in pythreejs), 20
`bottom` (pythreejs.OrthographicCamera attribute), 30
`box` (pythreejs.Box3Helper attribute), 85

Box2 (class in pythreejs), 136
 Box3 (class in pythreejs), 136
 Box3Helper (class in pythreejs), 85
 BoxBufferGeometry (class in pythreejs), 55
 BoxGeometry (class in pythreejs), 56
 BoxHelper (class in pythreejs), 85
 BufferAttribute (class in pythreejs), 42
 BufferGeometry (class in pythreejs), 42
 BufferGeometryLoader (class in pythreejs), 101
 bumpMap (pythreejs.MeshPhongMaterial attribute), 119
 bumpMap (pythreejs.MeshStandardMaterial attribute), 125
 bumpScale (pythreejs.MeshPhongMaterial attribute), 119
 bumpScale (pythreejs.MeshStandardMaterial attribute), 125

C

c (pythreejs.Triangle attribute), 141
 Cache (class in pythreejs), 102
 Camera (class in pythreejs), 27
 camera (pythreejs.CameraHelper attribute), 86
 camera (pythreejs.LightShadow attribute), 97
 CameraHelper (class in pythreejs), 86
 cameraL (pythreejs.StereoCamera attribute), 32
 cameraR (pythreejs.StereoCamera attribute), 32
 castShadow (pythreejs.Object3D attribute), 49
 CatmullRomCurve3 (class in pythreejs), 53
 center (pythreejs.Sphere attribute), 141
 center (pythreejs.Sprite attribute), 148
 children (pythreejs.Blackbox attribute), 142
 children (pythreejs.Object3D attribute), 48
 CircleBufferGeometry (class in pythreejs), 57
 CircleGeometry (class in pythreejs), 58
 circles (pythreejs.PolarGridHelper attribute), 91
 clampWhenFinished (pythreejs.AnimationAction attribute), 22
 clearCoat (pythreejs.MeshPhysicalMaterial attribute), 123
 clearCoatRoughness (pythreejs.MeshPhysicalMaterial attribute), 124
 clip (pythreejs.AnimationAction attribute), 21
 clipIntersection (pythreejs.Material attribute), 107
 clipping (pythreejs.ShaderMaterial attribute), 131
 clippingPlanes (pythreejs.Material attribute), 107
 clipShadows (pythreejs.Material attribute), 107
 Clock (class in pythreejs), 43
 CloneArray (class in pythreejs), 143
 close (pythreejs.TubeGeometry attribute), 82
 color (pythreejs.Box3Helper attribute), 85
 color (pythreejs.BoxHelper attribute), 86
 color (pythreejs.DirectionallLightHelper attribute), 87
 color (pythreejs.FaceNormalsHelper attribute), 87
 color (pythreejs.Fog attribute), 152
 color (pythreejs.FogExp2 attribute), 152

color (pythreejs.HemisphereLightHelper attribute), 89
 color (pythreejs.Light attribute), 97
 color (pythreejs.LineBasicMaterial attribute), 104
 color (pythreejs.LineDashedMaterial attribute), 105
 color (pythreejs.MeshBasicMaterial attribute), 110
 color (pythreejs.MeshLambertMaterial attribute), 115
 color (pythreejs.MeshPhongMaterial attribute), 119
 color (pythreejs.MeshStandardMaterial attribute), 125
 color (pythreejs.PlaneHelper attribute), 90
 color (pythreejs.PointLightHelper attribute), 91
 color (pythreejs.PointsMaterial attribute), 129
 color (pythreejs.RectAreaLightHelper attribute), 92
 color (pythreejs.SpotLightHelper attribute), 93
 color (pythreejs.SpriteMaterial attribute), 134
 color (pythreejs.TextTexture attribute), 157
 color (pythreejs.VertexNormalsHelper attribute), 94
 color1 (pythreejs.PolarGridHelper attribute), 92
 color2 (pythreejs.PolarGridHelper attribute), 92
 colorCenterLine (pythreejs.GridHelper attribute), 88
 colorGrid (pythreejs.GridHelper attribute), 88
 ColorKeyframeTrack (class in pythreejs), 20
 colors (pythreejs.Geometry attribute), 44
 colorWrite (pythreejs.Material attribute), 108
 combine (pythreejs.MeshBasicMaterial attribute), 110
 combine (pythreejs.MeshLambertMaterial attribute), 115
 combine (pythreejs.MeshPhongMaterial attribute), 119
 CombinedCamera (class in pythreejs), 27
 CompressedTexture (class in pythreejs), 154
 CompressedTextureLoader (class in pythreejs), 102
 ConeGeometry (class in pythreejs), 59
 constant (pythreejs.Plane attribute), 139
 controlling (pythreejs.Controls attribute), 32
 Controls (class in pythreejs), 32
 CubeCamera (class in pythreejs), 29
 CubeTexture (class in pythreejs), 154
 CubeTextureLoader (class in pythreejs), 102
 CubicBezierCurve (class in pythreejs), 53
 CubicBezierCurve3 (class in pythreejs), 53
 CubicInterpolant (class in pythreejs), 135
 Curve (class in pythreejs), 52
 CurvePath (class in pythreejs), 51
 curveSegments (pythreejs.ShapeGeometry attribute), 73
 CylinderBufferGeometry (class in pythreejs), 60
 CylinderGeometry (class in pythreejs), 62
 Cylindrical (class in pythreejs), 136

D

dampingFactor (pythreejs.OrbitControls attribute), 35
 dashSize (pythreejs.LineDashedMaterial attribute), 106
 data (pythreejs.DataTexture attribute), 155
 data (pythreejs.InterleavedBufferAttribute attribute), 46
 DataTexture (class in pythreejs), 154
 DataTextureLoader (class in pythreejs), 102
 decay (pythreejs.PointLight attribute), 98

- decay (pythreejs.SpotLight attribute), 100
 - default_value (pythreejs.traits.Euler attribute), 161
 - default_value (pythreejs.traits.Matrix3 attribute), 161
 - default_value (pythreejs.traits.Matrix4 attribute), 161
 - default_value (pythreejs.traits.Vector2 attribute), 161
 - default_value (pythreejs.traits.Vector3 attribute), 161
 - default_value (pythreejs.traits.Vector4 attribute), 162
 - defines (pythreejs.Material attribute), 108
 - defines (pythreejs.MeshPhysicalMaterial attribute), 124
 - defines (pythreejs.MeshStandardMaterial attribute), 125
 - density (pythreejs.FogExp2 attribute), 152
 - depth (pythreejs.BoxBufferGeometry attribute), 55
 - depth (pythreejs.BoxGeometry attribute), 56
 - depthFunc (pythreejs.Material attribute), 108
 - depthSegments (pythreejs.BoxBufferGeometry attribute), 55
 - depthSegments (pythreejs.BoxGeometry attribute), 57
 - depthTest (pythreejs.Material attribute), 108
 - DepthTexture (class in pythreejs), 155
 - depthWrite (pythreejs.Material attribute), 108
 - detail (pythreejs.DodecahedronGeometry attribute), 63
 - detail (pythreejs.IcosahedronGeometry attribute), 65
 - detail (pythreejs.OctahedronGeometry attribute), 67
 - detail (pythreejs.PolyhedronGeometry attribute), 70
 - detail (pythreejs.TetrahedronGeometry attribute), 77
 - dir (pythreejs.ArrowHelper attribute), 83
 - DirectGeometry (class in pythreejs), 43
 - direction (pythreejs.Ray attribute), 140
 - direction (pythreejs.Raycaster attribute), 50
 - DirectionalLight (class in pythreejs), 95
 - DirectionalLightHelper (class in pythreejs), 86
 - DirectionalLightShadow (class in pythreejs), 95
 - DiscreteInterpolant (class in pythreejs), 135
 - displacementBias (pythreejs.MeshDepthMaterial attribute), 113
 - displacementBias (pythreejs.MeshPhongMaterial attribute), 120
 - displacementBias (pythreejs.MeshStandardMaterial attribute), 125
 - displacementMap (pythreejs.MeshDepthMaterial attribute), 113
 - displacementMap (pythreejs.MeshPhongMaterial attribute), 120
 - displacementMap (pythreejs.MeshStandardMaterial attribute), 125
 - displacementScale (pythreejs.MeshDepthMaterial attribute), 113
 - displacementScale (pythreejs.MeshPhongMaterial attribute), 120
 - displacementScale (pythreejs.MeshStandardMaterial attribute), 125
 - distance (pythreejs.Picker attribute), 37
 - distance (pythreejs.PointLight attribute), 98
 - distance (pythreejs.SpotLight attribute), 100
 - dithering (pythreejs.Material attribute), 108
 - divisions (pythreejs.GridHelper attribute), 88
 - divisions (pythreejs.PolarGridHelper attribute), 91
 - DodecahedronGeometry (class in pythreejs), 63
 - drawMode (pythreejs.Mesh attribute), 145
 - duration (pythreejs.AnimationClip attribute), 23
 - dynamic (pythreejs.BufferAttribute attribute), 42
 - dynamic (pythreejs.InterleavedBuffer attribute), 47
 - dynamicDampingFactor (pythreejs.TrackballControls attribute), 39
- ## E
- EdgesGeometry (class in pythreejs), 64
 - EllipseCurve (class in pythreejs), 53
 - emissive (pythreejs.MeshLambertMaterial attribute), 115
 - emissive (pythreejs.MeshPhongMaterial attribute), 120
 - emissive (pythreejs.MeshStandardMaterial attribute), 125
 - emissiveIntensity (pythreejs.MeshLambertMaterial attribute), 115
 - emissiveIntensity (pythreejs.MeshPhongMaterial attribute), 120
 - emissiveIntensity (pythreejs.MeshStandardMaterial attribute), 125
 - emissiveMap (pythreejs.MeshLambertMaterial attribute), 115
 - emissiveMap (pythreejs.MeshPhongMaterial attribute), 120
 - emissiveMap (pythreejs.MeshStandardMaterial attribute), 125
 - enabled (pythreejs.AnimationAction attribute), 22
 - enabled (pythreejs.OrbitControls attribute), 34
 - enabled (pythreejs.TrackballControls attribute), 39
 - enabled (pythreejs.WebGLShadowMap attribute), 151
 - enableDamping (pythreejs.OrbitControls attribute), 34
 - enableKeys (pythreejs.OrbitControls attribute), 35
 - enablePan (pythreejs.OrbitControls attribute), 35
 - enableRotate (pythreejs.OrbitControls attribute), 35
 - enableZoom (pythreejs.OrbitControls attribute), 35
 - encoding (pythreejs.Texture attribute), 159
 - end (pythreejs.Line3 attribute), 138
 - envMap (pythreejs.MeshBasicMaterial attribute), 110
 - envMap (pythreejs.MeshLambertMaterial attribute), 115
 - envMap (pythreejs.MeshPhongMaterial attribute), 120
 - envMap (pythreejs.MeshStandardMaterial attribute), 126
 - envMapIntensity (pythreejs.MeshStandardMaterial attribute), 126
 - Euler (class in pythreejs.traits), 161
 - event (pythreejs.Picker attribute), 37
 - EventDispatcher (class in pythreejs), 44
 - exec_three_obj_method() (pythreejs.ThreeWidget method), 19
 - extensions (pythreejs.ShaderMaterial attribute), 131
 - ExtrudeGeometry (class in pythreejs), 64
 - eyeSep (pythreejs.StereoCamera attribute), 32

F

face (pythreejs.Picker attribute), 37
 Face3 (class in pythreejs.traits), 161
 faceIndex (pythreejs.Picker attribute), 37
 faceNormal (pythreejs.Picker attribute), 37
 FaceNormalsHelper (class in pythreejs), 87
 faces (pythreejs.Geometry attribute), 44
 faces (pythreejs.PolyhedronGeometry attribute), 70
 faceVertexUvs (pythreejs.Geometry attribute), 44
 faceVertices (pythreejs.Picker attribute), 37
 far (pythreejs.CombinedCamera attribute), 28
 far (pythreejs.Fog attribute), 153
 far (pythreejs.OrthographicCamera attribute), 30
 far (pythreejs.PerspectiveCamera attribute), 31
 far (pythreejs.Raycaster attribute), 50
 FileLoader (class in pythreejs), 102
 flatShading (pythreejs.Material attribute), 108
 flatShading (pythreejs.ShaderMaterial attribute), 132
 flipY (pythreejs.DataTexture attribute), 155
 flipY (pythreejs.DepthTexture attribute), 156
 flipY (pythreejs.Texture attribute), 159
 FlyControls (class in pythreejs), 33
 focus (pythreejs.PerspectiveCamera attribute), 31
 Fog (class in pythreejs), 152
 fog (pythreejs.Material attribute), 108
 fog (pythreejs.MeshDepthMaterial attribute), 113
 fog (pythreejs.MeshNormalMaterial attribute), 118
 fog (pythreejs.Scene attribute), 153
 fog (pythreejs.ShaderMaterial attribute), 131
 fog (pythreejs.SpriteMaterial attribute), 134
 FogExp2 (class in pythreejs), 152
 Font (class in pythreejs), 52
 fontFace (pythreejs.TextTexture attribute), 157
 FontLoader (class in pythreejs), 103
 format (pythreejs.DepthTexture attribute), 155
 format (pythreejs.Texture attribute), 158
 fov (pythreejs.CombinedCamera attribute), 27
 fov (pythreejs.PerspectiveCamera attribute), 31
 fragmentShader (pythreejs.ShaderMaterial attribute), 131
 freeze() (pythreejs.RenderableWidget method), 19
 from_geometry() (pythreejs.BufferGeometry class method), 43
 from_geometry() (pythreejs.Geometry class method), 45
 Frustum (class in pythreejs), 137
 frustumCulled (pythreejs.Object3D attribute), 50
 func (pythreejs.ParametricGeometry attribute), 68

G

gapSize (pythreejs.LineDashedMaterial attribute), 106
 generateMipmaps (pythreejs.DataTexture attribute), 155
 generateMipmaps (pythreejs.DepthTexture attribute), 156
 generateMipmaps (pythreejs.Texture attribute), 159
 Geometry (class in pythreejs), 44
 geometry (pythreejs.Line attribute), 145

geometry (pythreejs.Mesh attribute), 145
 geometry (pythreejs.Points attribute), 146
 geometry (pythreejs.WireframeGeometry attribute), 83
 gradientMap (pythreejs.MeshToonMaterial attribute), 129
 GridHelper (class in pythreejs), 88
 groundColor (pythreejs.HemisphereLight attribute), 96
 Group (class in pythreejs), 143

H

headLength (pythreejs.ArrowHelper attribute), 84
 headWidth (pythreejs.ArrowHelper attribute), 84
 height (pythreejs.BoxBufferGeometry attribute), 55
 height (pythreejs.BoxGeometry attribute), 56
 height (pythreejs.CombinedCamera attribute), 28
 height (pythreejs.ConeGeometry attribute), 59
 height (pythreejs.CylinderBufferGeometry attribute), 61
 height (pythreejs.CylinderGeometry attribute), 62
 height (pythreejs.DepthTexture attribute), 155
 height (pythreejs.PlaneBufferGeometry attribute), 68
 height (pythreejs.PlaneGeometry attribute), 69
 height (pythreejs.RectAreaLight attribute), 99
 heightSegments (pythreejs.BoxBufferGeometry attribute), 55
 heightSegments (pythreejs.BoxGeometry attribute), 56
 heightSegments (pythreejs.ConeGeometry attribute), 59
 heightSegments (pythreejs.CylinderBufferGeometry attribute), 61
 heightSegments (pythreejs.CylinderGeometry attribute), 62
 heightSegments (pythreejs.PlaneBufferGeometry attribute), 69
 heightSegments (pythreejs.PlaneGeometry attribute), 69
 heightSegments (pythreejs.SphereBufferGeometry attribute), 74
 heightSegments (pythreejs.SphereGeometry attribute), 76
 HemisphereLight (class in pythreejs), 96
 HemisphereLightHelper (class in pythreejs), 89
 hex (pythreejs.ArrowHelper attribute), 84

I

IcosahedronGeometry (class in pythreejs), 64
 ImageBitmapLoader (class in pythreejs), 103
 ImageLoader (class in pythreejs), 103
 images (pythreejs.CubeTexture attribute), 154
 ImageTexture (class in pythreejs), 157
 imageUri (pythreejs.ImageTexture attribute), 157
 ImmediateRenderObject (class in pythreejs), 55
 impersonate (pythreejs.CombinedCamera attribute), 28
 index (pythreejs.BufferGeometry attribute), 42
 indices (pythreejs.Picker attribute), 38
 indices (pythreejs.PolyhedronGeometry attribute), 70
 info_text (pythreejs.traits.Euler attribute), 161
 info_text (pythreejs.traits.Face3 attribute), 161
 info_text (pythreejs.traits.Matrix3 attribute), 161

info_text (pythreejs.traits.Matrix4 attribute), 161
 info_text (pythreejs.traits.Vector2 attribute), 161
 info_text (pythreejs.traits.Vector3 attribute), 161
 info_text (pythreejs.traits.Vector4 attribute), 162
 innerRadius (pythreejs.RingBufferGeometry attribute), 71
 innerRadius (pythreejs.RingGeometry attribute), 72
 InstancedBufferAttribute (class in pythreejs), 45
 InstancedBufferGeometry (class in pythreejs), 45
 InstancedInterleavedBuffer (class in pythreejs), 46
 intensity (pythreejs.Light attribute), 98
 InterleavedBuffer (class in pythreejs), 47
 InterleavedBufferAttribute (class in pythreejs), 46
 Interpolant (class in pythreejs), 138
 interpolation (pythreejs.KeyframeTrack attribute), 25
 itemSize (pythreejs.InterleavedBufferAttribute attribute), 46

J

JSONLoader (class in pythreejs), 103

K

KeyframeTrack (class in pythreejs), 24
 keyPanSpeed (pythreejs.OrbitControls attribute), 35
 klass (pythreejs.traits.Face3 attribute), 161

L

LatheBufferGeometry (class in pythreejs), 65
 LatheGeometry (class in pythreejs), 66
 Layers (class in pythreejs), 48
 left (pythreejs.OrthographicCamera attribute), 29
 length (pythreejs.ArrowHelper attribute), 84
 Light (class in pythreejs), 97
 light (pythreejs.DirectionallightHelper attribute), 87
 light (pythreejs.HemisphereLightHelper attribute), 89
 light (pythreejs.PointLightHelper attribute), 90
 light (pythreejs.RectAreaLightHelper attribute), 92
 light (pythreejs.SpotLightHelper attribute), 93
 lightMap (pythreejs.MeshBasicMaterial attribute), 110
 lightMap (pythreejs.MeshLambertMaterial attribute), 115
 lightMap (pythreejs.MeshPhongMaterial attribute), 120
 lightMap (pythreejs.MeshStandardMaterial attribute), 126
 lightMapIntensity (pythreejs.MeshBasicMaterial attribute), 111
 lightMapIntensity (pythreejs.MeshLambertMaterial attribute), 116
 lightMapIntensity (pythreejs.MeshPhongMaterial attribute), 120
 lightMapIntensity (pythreejs.MeshStandardMaterial attribute), 126
 lights (pythreejs.LineBasicMaterial attribute), 105
 lights (pythreejs.LineDashedMaterial attribute), 106
 lights (pythreejs.Material attribute), 108

lights (pythreejs.MeshBasicMaterial attribute), 111
 lights (pythreejs.MeshDepthMaterial attribute), 113
 lights (pythreejs.MeshNormalMaterial attribute), 118
 lights (pythreejs.PointsMaterial attribute), 130
 lights (pythreejs.ShaderMaterial attribute), 131
 lights (pythreejs.ShadowMaterial attribute), 133
 lights (pythreejs.SpriteMaterial attribute), 134
 LightShadow (class in pythreejs), 97
 Line (class in pythreejs), 144
 Line3 (class in pythreejs), 138
 LinearInterpolant (class in pythreejs), 135
 LineBasicMaterial (class in pythreejs), 104
 linecap (pythreejs.LineBasicMaterial attribute), 105
 LineCurve (class in pythreejs), 54
 LineCurve3 (class in pythreejs), 54
 LineDashedMaterial (class in pythreejs), 105
 lineDistances (pythreejs.Geometry attribute), 44
 linejoin (pythreejs.LineBasicMaterial attribute), 105
 LineLoop (class in pythreejs), 144
 linePrecision (pythreejs.Raycaster attribute), 51
 LineSegments (class in pythreejs), 144
 linewidth (pythreejs.FaceNormalsHelper attribute), 88
 linewidth (pythreejs.LineBasicMaterial attribute), 105
 linewidth (pythreejs.LineDashedMaterial attribute), 106
 linewidth (pythreejs.ShaderMaterial attribute), 131
 linewidth (pythreejs.VertexNormalsHelper attribute), 94
 Loader (class in pythreejs), 103
 LoadingManager (class in pythreejs), 104
 localRoot (pythreejs.AnimationAction attribute), 21
 LOD (class in pythreejs), 144
 log() (pythreejs.RenderableWidget method), 19
 lookAt() (pythreejs.Object3D method), 50
 loop (pythreejs.AnimationAction attribute), 22

M

magFilter (pythreejs.DataTexture attribute), 155
 magFilter (pythreejs.DepthTexture attribute), 156
 magFilter (pythreejs.Texture attribute), 158
 map (pythreejs.MeshBasicMaterial attribute), 111
 map (pythreejs.MeshDepthMaterial attribute), 113
 map (pythreejs.MeshLambertMaterial attribute), 116
 map (pythreejs.MeshPhongMaterial attribute), 120
 map (pythreejs.MeshStandardMaterial attribute), 126
 map (pythreejs.PointsMaterial attribute), 130
 map (pythreejs.SpriteMaterial attribute), 134
 mapping (pythreejs.Texture attribute), 158
 mapSize (pythreejs.LightShadow attribute), 97
 Material (class in pythreejs), 106
 material (pythreejs.Line attribute), 145
 material (pythreejs.Mesh attribute), 145
 material (pythreejs.Points attribute), 146
 material (pythreejs.ShapeGeometry attribute), 73
 material (pythreejs.Sprite attribute), 148
 MaterialLoader (class in pythreejs), 104

Math (class in pythreejs), 139
 matrix (pythreejs.Object3D attribute), 49
 Matrix3 (class in pythreejs.traits), 161
 Matrix4 (class in pythreejs.traits), 161
 matrixAutoUpdate (pythreejs.Object3D attribute), 49
 matrixWorld (pythreejs.Object3D attribute), 49
 matrixWorldInverse (pythreejs.Camera attribute), 27
 matrixWorldNeedsUpdate (pythreejs.Object3D attribute), 49
 max (pythreejs.Box2 attribute), 136
 max (pythreejs.Box3 attribute), 136
 maxAzimuthAngle (pythreejs.OrbitControls attribute), 34
 maxDistance (pythreejs.OrbitControls attribute), 34
 maxDistance (pythreejs.TrackballControls attribute), 39
 MaxIndex (pythreejs.BufferGeometry attribute), 43
 maxInstancedCount (pythreejs.InstancedBufferGeometry attribute), 46
 maxPolarAngle (pythreejs.OrbitControls attribute), 34
 maxZoom (pythreejs.OrbitControls attribute), 34
 merge (pythreejs.CloneArray attribute), 143
 Mesh (class in pythreejs), 145
 MeshBasicMaterial (class in pythreejs), 110
 MeshDepthMaterial (class in pythreejs), 113
 MeshLambertMaterial (class in pythreejs), 115
 MeshNormalMaterial (class in pythreejs), 118
 meshPerAttribute (pythreejs.InstancedBufferAttribute attribute), 45
 meshPerAttribute (pythreejs.InstancedInterleavedBuffer attribute), 46
 MeshPhongMaterial (class in pythreejs), 119
 MeshPhysicalMaterial (class in pythreejs), 123
 MeshStandardMaterial (class in pythreejs), 124
 MeshToonMaterial (class in pythreejs), 129
 metalness (pythreejs.MeshStandardMaterial attribute), 126
 metalnessMap (pythreejs.MeshStandardMaterial attribute), 126
 min (pythreejs.Box2 attribute), 136
 min (pythreejs.Box3 attribute), 136
 minAzimuthAngle (pythreejs.OrbitControls attribute), 34
 minDistance (pythreejs.OrbitControls attribute), 34
 minDistance (pythreejs.TrackballControls attribute), 39
 minFilter (pythreejs.DataTexture attribute), 155
 minFilter (pythreejs.DepthTexture attribute), 156
 minFilter (pythreejs.Texture attribute), 158
 minPolarAngle (pythreejs.OrbitControls attribute), 34
 minZoom (pythreejs.OrbitControls attribute), 34
 mixer (pythreejs.AnimationAction attribute), 21
 mode (pythreejs.CombinedCamera attribute), 28
 modelViewMatrix (pythreejs.Object3D attribute), 49
 modifiers (pythreejs.Picker attribute), 37
 morphAttributes (pythreejs.BufferGeometry attribute), 42
 morphNormals (pythreejs.Geometry attribute), 44

morphNormals (pythreejs.MeshLambertMaterial attribute), 116
 morphNormals (pythreejs.MeshPhongMaterial attribute), 120
 morphNormals (pythreejs.MeshStandardMaterial attribute), 126
 morphNormals (pythreejs.ShaderMaterial attribute), 131
 morphTargetInfluences (pythreejs.Mesh attribute), 146
 morphTargets (pythreejs.Geometry attribute), 44
 morphTargets (pythreejs.MeshBasicMaterial attribute), 111
 morphTargets (pythreejs.MeshDepthMaterial attribute), 113
 morphTargets (pythreejs.MeshLambertMaterial attribute), 116
 morphTargets (pythreejs.MeshNormalMaterial attribute), 118
 morphTargets (pythreejs.MeshPhongMaterial attribute), 121
 morphTargets (pythreejs.MeshStandardMaterial attribute), 126
 morphTargets (pythreejs.ShaderMaterial attribute), 132
 movementSpeed (pythreejs.FlyControls attribute), 33
 moveVector (pythreejs.FlyControls attribute), 33

N

name (pythreejs.AnimationClip attribute), 23
 name (pythreejs.BaseBufferGeometry attribute), 41
 name (pythreejs.BaseGeometry attribute), 41
 name (pythreejs.Fog attribute), 152
 name (pythreejs.FogExp2 attribute), 152
 name (pythreejs.KeyframeTrack attribute), 24
 name (pythreejs.Material attribute), 108
 name (pythreejs.Object3D attribute), 48
 name (pythreejs.Texture attribute), 158
 near (pythreejs.CombinedCamera attribute), 27
 near (pythreejs.Fog attribute), 152
 near (pythreejs.OrthographicCamera attribute), 30
 near (pythreejs.PerspectiveCamera attribute), 31
 near (pythreejs.Raycaster attribute), 50
 needsUpdate (pythreejs.BufferAttribute attribute), 42
 needsUpdate (pythreejs.InterleavedBuffer attribute), 47
 noPan (pythreejs.TrackballControls attribute), 40
 normal (pythreejs.Plane attribute), 139
 normalized (pythreejs.BufferAttribute attribute), 42
 normalized (pythreejs.InterleavedBufferAttribute attribute), 47
 normalMap (pythreejs.MeshPhongMaterial attribute), 121
 normalMap (pythreejs.MeshStandardMaterial attribute), 126
 normalMatrix (pythreejs.Object3D attribute), 49
 normalScale (pythreejs.MeshPhongMaterial attribute), 121

normalScale (pythreejs.MeshStandardMaterial attribute), 126
 noRoll (pythreejs.TrackballControls attribute), 40
 noRotate (pythreejs.TrackballControls attribute), 39
 noZoom (pythreejs.TrackballControls attribute), 39
 NumberKeyframeTrack (class in pythreejs), 20

O

object (pythreejs.BoxHelper attribute), 86
 object (pythreejs.FaceNormalsHelper attribute), 87
 object (pythreejs.Picker attribute), 38
 object (pythreejs.VertexNormalsHelper attribute), 94
 Object3D (class in pythreejs), 48
 ObjectLoader (class in pythreejs), 104
 OctahedronGeometry (class in pythreejs), 67
 offset (pythreejs.InterleavedBufferAttribute attribute), 47
 offset (pythreejs.Texture attribute), 159
 onNeedsUpdate (pythreejs.Material attribute), 110
 opacity (pythreejs.Material attribute), 108
 openEnded (pythreejs.ConeGeometry attribute), 59
 openEnded (pythreejs.CylinderBufferGeometry attribute), 61
 openEnded (pythreejs.CylinderGeometry attribute), 62
 OrbitControls (class in pythreejs), 34
 origin (pythreejs.ArrowHelper attribute), 83
 origin (pythreejs.Ray attribute), 140
 origin (pythreejs.Raycaster attribute), 50
 original (pythreejs.CloneArray attribute), 143
 orthoFar (pythreejs.CombinedCamera attribute), 28
 OrthographicCamera (class in pythreejs), 29
 orthoNear (pythreejs.CombinedCamera attribute), 28
 outerRadius (pythreejs.RingBufferGeometry attribute), 71
 outerRadius (pythreejs.RingGeometry attribute), 72
 overdraw (pythreejs.Material attribute), 109
 overrideMaterial (pythreejs.Scene attribute), 153

P

p (pythreejs.TorusKnotBufferGeometry attribute), 80
 p (pythreejs.TorusKnotGeometry attribute), 81
 p0 (pythreejs.Frustum attribute), 137
 p1 (pythreejs.Frustum attribute), 137
 p2 (pythreejs.Frustum attribute), 137
 p3 (pythreejs.Frustum attribute), 137
 p4 (pythreejs.Frustum attribute), 138
 p5 (pythreejs.Frustum attribute), 138
 panSpeed (pythreejs.TrackballControls attribute), 39
 ParametricGeometry (class in pythreejs), 67
 Path (class in pythreejs), 52
 path (pythreejs.TubeGeometry attribute), 82
 paused (pythreejs.AnimationAction attribute), 22
 penumbra (pythreejs.SpotLight attribute), 100
 PerspectiveCamera (class in pythreejs), 30
 phiLength (pythreejs.LatheBufferGeometry attribute), 65

phiLength (pythreejs.LatheGeometry attribute), 66
 phiLength (pythreejs.SphereBufferGeometry attribute), 74
 phiLength (pythreejs.SphereGeometry attribute), 76
 phiSegments (pythreejs.RingBufferGeometry attribute), 71
 phiSegments (pythreejs.RingGeometry attribute), 72
 phiStart (pythreejs.LatheBufferGeometry attribute), 65
 phiStart (pythreejs.LatheGeometry attribute), 66
 phiStart (pythreejs.SphereBufferGeometry attribute), 74
 phiStart (pythreejs.SphereGeometry attribute), 76
 picked (pythreejs.Picker attribute), 38
 Picker (class in pythreejs), 37
 Plane (class in pythreejs), 139
 plane (pythreejs.PlaneHelper attribute), 90
 PlaneBufferGeometry (class in pythreejs), 68
 PlaneGeometry (class in pythreejs), 69
 PlaneHelper (class in pythreejs), 90
 point (pythreejs.Picker attribute), 37
 PointLight (class in pythreejs), 98
 PointLightHelper (class in pythreejs), 90
 Points (class in pythreejs), 146
 points (pythreejs.LatheBufferGeometry attribute), 65
 points (pythreejs.LatheGeometry attribute), 66
 PointsMaterial (class in pythreejs), 129
 PolarGridHelper (class in pythreejs), 91
 polygonOffset (pythreejs.Material attribute), 109
 polygonOffsetFactor (pythreejs.Material attribute), 109
 polygonOffsetUnits (pythreejs.Material attribute), 109
 PolyhedronGeometry (class in pythreejs), 70
 position (pythreejs.Object3D attribute), 48
 PositionalAudio (class in pythreejs), 26
 positions (pythreejs.CloneArray attribute), 143
 power (pythreejs.PointLight attribute), 98
 precision (pythreejs.Material attribute), 109
 premultipliedAlpha (pythreejs.Material attribute), 109
 premultiplyAlpha (pythreejs.Texture attribute), 159
 Preview (class in pythreejs), 18
 projectionMatrix (pythreejs.Camera attribute), 27
 PropertyBinding (class in pythreejs), 25
 PropertyMixer (class in pythreejs), 25
 pythreejs (module), 18
 pythreejs.traits (module), 161

Q

q (pythreejs.TorusKnotBufferGeometry attribute), 80
 q (pythreejs.TorusKnotGeometry attribute), 81
 QuadraticBezierCurve (class in pythreejs), 54
 QuadraticBezierCurve3 (class in pythreejs), 54
 Quaternion (class in pythreejs), 139
 quaternion (pythreejs.Object3D attribute), 49
 QuaternionKeyframeTrack (class in pythreejs), 20
 QuaternionLinearInterpolant (class in pythreejs), 135

R

- radials (pythreejs.PolarGridHelper attribute), 91
 - radialSegments (pythreejs.ConeGeometry attribute), 59
 - radialSegments (pythreejs.TorusBufferGeometry attribute), 78
 - radialSegments (pythreejs.TorusGeometry attribute), 79
 - radialSegments (pythreejs.TorusKnotBufferGeometry attribute), 80
 - radialSegments (pythreejs.TorusKnotGeometry attribute), 81
 - radius (pythreejs.CircleBufferGeometry attribute), 57
 - radius (pythreejs.CircleGeometry attribute), 58
 - radius (pythreejs.ConeGeometry attribute), 59
 - radius (pythreejs.Cylindrical attribute), 137
 - radius (pythreejs.DodecahedronGeometry attribute), 63
 - radius (pythreejs.IcosahedronGeometry attribute), 65
 - radius (pythreejs.LightShadow attribute), 97
 - radius (pythreejs.OctahedronGeometry attribute), 67
 - radius (pythreejs.PolarGridHelper attribute), 91
 - radius (pythreejs.PolyhedronGeometry attribute), 70
 - radius (pythreejs.Sphere attribute), 141
 - radius (pythreejs.SphereBufferGeometry attribute), 74
 - radius (pythreejs.SphereGeometry attribute), 75
 - radius (pythreejs.TetrahedronGeometry attribute), 77
 - radius (pythreejs.TorusBufferGeometry attribute), 78
 - radius (pythreejs.TorusGeometry attribute), 79
 - radius (pythreejs.TorusKnotBufferGeometry attribute), 80
 - radius (pythreejs.TorusKnotGeometry attribute), 81
 - radius (pythreejs.TubeGeometry attribute), 82
 - radiusBottom (pythreejs.CylinderBufferGeometry attribute), 61
 - radiusBottom (pythreejs.CylinderGeometry attribute), 62
 - radiusSegments (pythreejs.CylinderBufferGeometry attribute), 61
 - radiusSegments (pythreejs.CylinderGeometry attribute), 62
 - radiusSegments (pythreejs.TubeGeometry attribute), 82
 - radiusTop (pythreejs.CylinderBufferGeometry attribute), 60
 - radiusTop (pythreejs.CylinderGeometry attribute), 62
 - RawShaderMaterial (class in pythreejs), 130
 - Ray (class in pythreejs), 140
 - ray (pythreejs.Raycaster attribute), 51
 - Raycaster (class in pythreejs), 50
 - receiveShadow (pythreejs.Object3D attribute), 50
 - RectAreaLight (class in pythreejs), 99
 - RectAreaLightHelper (class in pythreejs), 92
 - reflectivity (pythreejs.MeshBasicMaterial attribute), 111
 - reflectivity (pythreejs.MeshLambertMaterial attribute), 116
 - reflectivity (pythreejs.MeshPhongMaterial attribute), 121
 - reflectivity (pythreejs.MeshPhysicalMaterial attribute), 124
 - refractionRatio (pythreejs.MeshBasicMaterial attribute), 111
 - refractionRatio (pythreejs.MeshLambertMaterial attribute), 116
 - refractionRatio (pythreejs.MeshPhongMaterial attribute), 121
 - refractionRatio (pythreejs.MeshStandardMaterial attribute), 127
 - remove() (pythreejs.Object3D method), 50
 - RenderableWidget (class in pythreejs), 18
 - renderOrder (pythreejs.Object3D attribute), 50
 - repeat (pythreejs.Texture attribute), 159
 - repetitions (pythreejs.AnimationAction attribute), 22
 - right (pythreejs.OrthographicCamera attribute), 29
 - RingBufferGeometry (class in pythreejs), 71
 - RingGeometry (class in pythreejs), 72
 - rollSpeed (pythreejs.FlyControls attribute), 33
 - root (pythreejs.SkeletonHelper attribute), 93
 - rootObject (pythreejs.AnimationMixer attribute), 23
 - rotateSpeed (pythreejs.OrbitControls attribute), 35
 - rotateSpeed (pythreejs.TrackballControls attribute), 39
 - rotateX() (pythreejs.Object3D method), 50
 - rotateY() (pythreejs.Object3D method), 50
 - rotateZ() (pythreejs.Object3D method), 50
 - rotation (pythreejs.Object3D attribute), 49
 - rotation (pythreejs.SpriteMaterial attribute), 134
 - rotation (pythreejs.Texture attribute), 159
 - rotationVector (pythreejs.FlyControls attribute), 33
 - roughness (pythreejs.MeshStandardMaterial attribute), 127
 - roughnessMap (pythreejs.MeshStandardMaterial attribute), 127
- ## S
- scale (pythreejs.LineDashedMaterial attribute), 106
 - scale (pythreejs.Object3D attribute), 49
 - Scene (class in pythreejs), 153
 - segments (pythreejs.CircleBufferGeometry attribute), 57
 - segments (pythreejs.CircleGeometry attribute), 58
 - segments (pythreejs.LatheBufferGeometry attribute), 65
 - segments (pythreejs.LatheGeometry attribute), 66
 - segments (pythreejs.TubeGeometry attribute), 82
 - send_msg() (pythreejs.RenderableWidget method), 19
 - setRotationFromMatrix() (pythreejs.Object3D method), 50
 - ShaderMaterial (class in pythreejs), 131
 - shadow (pythreejs.DirectionallLight attribute), 96
 - shadow (pythreejs.PointLight attribute), 98
 - shadow (pythreejs.SpotLight attribute), 100
 - ShadowMaterial (class in pythreejs), 133
 - shadowSide (pythreejs.Material attribute), 109
 - Shape (class in pythreejs), 52
 - ShapeGeometry (class in pythreejs), 73
 - ShapePath (class in pythreejs), 52

shapes (pythreejs.ShapeGeometry attribute), 73
 shininess (pythreejs.MeshPhongMaterial attribute), 121
 side (pythreejs.Material attribute), 109
 size (pythreejs.AxesHelper attribute), 84
 size (pythreejs.DirectionLightHelper attribute), 87
 size (pythreejs.FaceNormalsHelper attribute), 87
 size (pythreejs.GridHelper attribute), 88
 size (pythreejs.HemisphereLightHelper attribute), 89
 size (pythreejs.PlaneHelper attribute), 90
 size (pythreejs.PointsMaterial attribute), 130
 size (pythreejs.TextTexture attribute), 157
 size (pythreejs.VertexNormalsHelper attribute), 94
 sizeAttenuation (pythreejs.PointsMaterial attribute), 130
 Skeleton (class in pythreejs), 147
 skeleton (pythreejs.SkinnedMesh attribute), 147
 SkeletonHelper (class in pythreejs), 93
 skinIndices (pythreejs.Geometry attribute), 45
 SkinnedMesh (class in pythreejs), 147
 skinning (pythreejs.MeshBasicMaterial attribute), 111
 skinning (pythreejs.MeshDepthMaterial attribute), 114
 skinning (pythreejs.MeshLambertMaterial attribute), 116
 skinning (pythreejs.MeshPhongMaterial attribute), 121
 skinning (pythreejs.MeshStandardMaterial attribute), 127
 skinning (pythreejs.ShaderMaterial attribute), 132
 skinWeights (pythreejs.Geometry attribute), 44
 slices (pythreejs.ParametricGeometry attribute), 68
 specular (pythreejs.MeshPhongMaterial attribute), 121
 specularMap (pythreejs.MeshBasicMaterial attribute), 111
 specularMap (pythreejs.MeshLambertMaterial attribute), 116
 specularMap (pythreejs.MeshPhongMaterial attribute), 121
 Sphere (class in pythreejs), 141
 SphereBufferGeometry (class in pythreejs), 74
 SphereGeometry (class in pythreejs), 75
 sphereSize (pythreejs.PointLightHelper attribute), 91
 Spherical (class in pythreejs), 141
 SplineCurve (class in pythreejs), 54
 SpotLight (class in pythreejs), 100
 SpotLightHelper (class in pythreejs), 93
 SpotLightShadow (class in pythreejs), 99
 Sprite (class in pythreejs), 148
 SpriteMaterial (class in pythreejs), 134
 squareTexture (pythreejs.TextTexture attribute), 157
 stacks (pythreejs.ParametricGeometry attribute), 68
 start (pythreejs.Line3 attribute), 138
 staticMoving (pythreejs.TrackballControls attribute), 39
 StereoCamera (class in pythreejs), 32
 stride (pythreejs.InterleavedBuffer attribute), 47
 string (pythreejs.TextTexture attribute), 157
 StringKeyframeTrack (class in pythreejs), 21
 syncRate (pythreejs.FlyControls attribute), 33

T

target (pythreejs.DirectionLight attribute), 95
 target (pythreejs.OrbitControls attribute), 35
 target (pythreejs.SpotLight attribute), 100
 target (pythreejs.TrackballControls attribute), 40
 TetrahedronGeometry (class in pythreejs), 77
 TextGeometry (class in pythreejs), 77
 TextTexture (class in pythreejs), 157
 Texture (class in pythreejs), 158
 TextureLoader (class in pythreejs), 104
 theta (pythreejs.Cylindrical attribute), 137
 thetaLength (pythreejs.CircleBufferGeometry attribute), 58
 thetaLength (pythreejs.CircleGeometry attribute), 58
 thetaLength (pythreejs.ConeGeometry attribute), 60
 thetaLength (pythreejs.CylinderBufferGeometry attribute), 61
 thetaLength (pythreejs.CylinderGeometry attribute), 63
 thetaLength (pythreejs.RingBufferGeometry attribute), 71
 thetaLength (pythreejs.RingGeometry attribute), 73
 thetaLength (pythreejs.SphereBufferGeometry attribute), 75
 thetaLength (pythreejs.SphereGeometry attribute), 76
 thetaSegments (pythreejs.RingBufferGeometry attribute), 71
 thetaSegments (pythreejs.RingGeometry attribute), 72
 thetaStart (pythreejs.CircleBufferGeometry attribute), 57
 thetaStart (pythreejs.CircleGeometry attribute), 58
 thetaStart (pythreejs.ConeGeometry attribute), 60
 thetaStart (pythreejs.CylinderBufferGeometry attribute), 61
 thetaStart (pythreejs.CylinderGeometry attribute), 62
 thetaStart (pythreejs.RingBufferGeometry attribute), 71
 thetaStart (pythreejs.RingGeometry attribute), 73
 thetaStart (pythreejs.SphereBufferGeometry attribute), 75
 thetaStart (pythreejs.SphereGeometry attribute), 76
 ThreeWidget (class in pythreejs), 19
 time (pythreejs.AnimationAction attribute), 22
 time (pythreejs.AnimationMixer attribute), 23
 times (pythreejs.KeyframeTrack attribute), 24
 timeScale (pythreejs.AnimationAction attribute), 22
 timeScale (pythreejs.AnimationMixer attribute), 23
 top (pythreejs.OrthographicCamera attribute), 30
 TorusBufferGeometry (class in pythreejs), 77
 TorusGeometry (class in pythreejs), 78
 TorusKnotBufferGeometry (class in pythreejs), 79
 TorusKnotGeometry (class in pythreejs), 81
 TrackballControls (class in pythreejs), 39
 tracks (pythreejs.AnimationClip attribute), 23
 transparent (pythreejs.Material attribute), 109
 transparent (pythreejs.ShadowMaterial attribute), 133
 Triangle (class in pythreejs), 141
 tube (pythreejs.TorusBufferGeometry attribute), 78

- tube (pythreejs.TorusGeometry attribute), 79
- tube (pythreejs.TorusKnotBufferGeometry attribute), 80
- tube (pythreejs.TorusKnotGeometry attribute), 81
- TubeGeometry (class in pythreejs), 82
- tubularSegments (pythreejs.TorusBufferGeometry attribute), 78
- tubularSegments (pythreejs.TorusGeometry attribute), 79
- tubularSegments (pythreejs.TorusKnotBufferGeometry attribute), 80
- tubularSegments (pythreejs.TorusKnotGeometry attribute), 81
- type (pythreejs.AmbientLight attribute), 95
- type (pythreejs.ArrayCamera attribute), 26
- type (pythreejs.ArrowHelper attribute), 84
- type (pythreejs.AxesHelper attribute), 85
- type (pythreejs.BaseBufferGeometry attribute), 41
- type (pythreejs.BaseGeometry attribute), 41
- type (pythreejs.Blackbox attribute), 142
- type (pythreejs.Bone attribute), 142
- type (pythreejs.Box3Helper attribute), 85
- type (pythreejs.BoxBufferGeometry attribute), 55
- type (pythreejs.BoxGeometry attribute), 57
- type (pythreejs.BoxHelper attribute), 86
- type (pythreejs.BufferGeometry attribute), 43
- type (pythreejs.Camera attribute), 27
- type (pythreejs.CameraHelper attribute), 86
- type (pythreejs.CircleBufferGeometry attribute), 58
- type (pythreejs.CircleGeometry attribute), 59
- type (pythreejs.CloneArray attribute), 143
- type (pythreejs.CombinedCamera attribute), 28
- type (pythreejs.ConeGeometry attribute), 60
- type (pythreejs.CubeCamera attribute), 29
- type (pythreejs.CylinderBufferGeometry attribute), 61
- type (pythreejs.CylinderGeometry attribute), 63
- type (pythreejs.DepthTexture attribute), 155
- type (pythreejs.DirectionallLight attribute), 96
- type (pythreejs.DirectionallLightHelper attribute), 87
- type (pythreejs.DodecahedronGeometry attribute), 63
- type (pythreejs.EdgesGeometry attribute), 64
- type (pythreejs.ExtrudeGeometry attribute), 64
- type (pythreejs.FaceNormalsHelper attribute), 88
- type (pythreejs.Geometry attribute), 45
- type (pythreejs.GridHelper attribute), 89
- type (pythreejs.Group attribute), 143
- type (pythreejs.HemisphereLight attribute), 96
- type (pythreejs.HemisphereLightHelper attribute), 89
- type (pythreejs.IcosahedronGeometry attribute), 65
- type (pythreejs.InstancedBufferGeometry attribute), 46
- type (pythreejs.LatheBufferGeometry attribute), 66
- type (pythreejs.LatheGeometry attribute), 66
- type (pythreejs.Light attribute), 98
- type (pythreejs.Line attribute), 145
- type (pythreejs.LineBasicMaterial attribute), 105
- type (pythreejs.LineDashedMaterial attribute), 106
- type (pythreejs.LineLoop attribute), 144
- type (pythreejs.LineSegments attribute), 144
- type (pythreejs.Material attribute), 109
- type (pythreejs.Mesh attribute), 146
- type (pythreejs.MeshBasicMaterial attribute), 112
- type (pythreejs.MeshDepthMaterial attribute), 114
- type (pythreejs.MeshLambertMaterial attribute), 117
- type (pythreejs.MeshNormalMaterial attribute), 118
- type (pythreejs.MeshPhongMaterial attribute), 122
- type (pythreejs.MeshPhysicalMaterial attribute), 124
- type (pythreejs.MeshStandardMaterial attribute), 127
- type (pythreejs.MeshToonMaterial attribute), 129
- type (pythreejs.Object3D attribute), 48
- type (pythreejs.OctahedronGeometry attribute), 67
- type (pythreejs.OrthographicCamera attribute), 30
- type (pythreejs.ParametricGeometry attribute), 68
- type (pythreejs.PerspectiveCamera attribute), 31
- type (pythreejs.PlaneBufferGeometry attribute), 69
- type (pythreejs.PlaneGeometry attribute), 69
- type (pythreejs.PlaneHelper attribute), 90
- type (pythreejs.PointLight attribute), 98
- type (pythreejs.PointLightHelper attribute), 91
- type (pythreejs.Points attribute), 146
- type (pythreejs.PointsMaterial attribute), 130
- type (pythreejs.PolarGridHelper attribute), 92
- type (pythreejs.PolyhedronGeometry attribute), 70
- type (pythreejs.RawShaderMaterial attribute), 130
- type (pythreejs.RectAreaLight attribute), 99
- type (pythreejs.RectAreaLightHelper attribute), 93
- type (pythreejs.RingBufferGeometry attribute), 72
- type (pythreejs.RingGeometry attribute), 73
- type (pythreejs.Scene attribute), 153
- type (pythreejs.ShaderMaterial attribute), 132
- type (pythreejs.ShadowMaterial attribute), 133
- type (pythreejs.ShapeGeometry attribute), 74
- type (pythreejs.SkeletonHelper attribute), 93
- type (pythreejs.SkinnedMesh attribute), 147
- type (pythreejs.SphereBufferGeometry attribute), 75
- type (pythreejs.SphereGeometry attribute), 76
- type (pythreejs.SpotLight attribute), 100
- type (pythreejs.SpotLightHelper attribute), 94
- type (pythreejs.Sprite attribute), 148
- type (pythreejs.SpriteMaterial attribute), 134
- type (pythreejs.TetrahedronGeometry attribute), 77
- type (pythreejs.TextGeometry attribute), 77
- type (pythreejs.Texture attribute), 159
- type (pythreejs.TorusBufferGeometry attribute), 78
- type (pythreejs.TorusGeometry attribute), 79
- type (pythreejs.TorusKnotBufferGeometry attribute), 80
- type (pythreejs.TorusKnotGeometry attribute), 81
- type (pythreejs.TubeGeometry attribute), 82
- type (pythreejs.VertexNormalsHelper attribute), 94
- type (pythreejs.WebGLShadowMap attribute), 151
- type (pythreejs.WireframeGeometry attribute), 83

U

Uniform (class in pythreejs), 51
 uniforms (pythreejs.ShaderMaterial attribute), 131
 uniformsNeedUpdate (pythreejs.ShaderMaterial attribute), 132
 Uninitialized (class in pythreejs.traits), 161
 unpackAlignment (pythreejs.Texture attribute), 159
 up (pythreejs.Object3D attribute), 48
 uv (pythreejs.Picker attribute), 38

V

validate (pythreejs.BufferGeometry attribute), 43
 validate() (pythreejs.traits.WebGLDataUnion method), 162
 values (pythreejs.KeyframeTrack attribute), 24
 Vector2 (class in pythreejs.traits), 161
 Vector3 (class in pythreejs.traits), 161
 Vector4 (class in pythreejs.traits), 162
 VectorKeyframeTrack (class in pythreejs), 21
 version (pythreejs.BufferAttribute attribute), 42
 version (pythreejs.InterleavedBuffer attribute), 47
 version (pythreejs.Texture attribute), 159
 vertexColors (pythreejs.Material attribute), 109
 VertexNormalsHelper (class in pythreejs), 94
 vertexShader (pythreejs.ShaderMaterial attribute), 132
 vertices (pythreejs.Geometry attribute), 44
 vertices (pythreejs.PolyhedronGeometry attribute), 70
 VideoTexture (class in pythreejs), 161
 visible (pythreejs.Material attribute), 110
 visible (pythreejs.Object3D attribute), 49

W

w (pythreejs.Quaternion attribute), 140
 WebGLBufferRenderer (class in pythreejs), 148
 WebGLCapabilities (class in pythreejs), 149
 WebGLDataUnion (class in pythreejs.traits), 162
 WebGLExtensions (class in pythreejs), 149
 WebGLGeometries (class in pythreejs), 149
 WebGLIndexedBufferRenderer (class in pythreejs), 149
 WebGLLights (class in pythreejs), 149
 WebGLObjects (class in pythreejs), 150
 WebGLProgram (class in pythreejs), 150
 WebGLPrograms (class in pythreejs), 150
 WebGLProperties (class in pythreejs), 150
 WebGLRenderTarget (class in pythreejs), 151
 WebGLRenderTargetCube (class in pythreejs), 151
 WebGLShader (class in pythreejs), 150
 WebGLShadowMap (class in pythreejs), 151
 WebGLState (class in pythreejs), 151
 weight (pythreejs.AnimationAction attribute), 22
 width (pythreejs.BoxBufferGeometry attribute), 55
 width (pythreejs.BoxGeometry attribute), 56
 width (pythreejs.CombinedCamera attribute), 28

width (pythreejs.DepthTexture attribute), 155
 width (pythreejs.PlaneBufferGeometry attribute), 68
 width (pythreejs.PlaneGeometry attribute), 69
 width (pythreejs.RectAreaLight attribute), 99
 widthSegments (pythreejs.BoxBufferGeometry attribute), 55
 widthSegments (pythreejs.BoxGeometry attribute), 56
 widthSegments (pythreejs.PlaneBufferGeometry attribute), 68
 widthSegments (pythreejs.PlaneGeometry attribute), 69
 widthSegments (pythreejs.SphereBufferGeometry attribute), 74
 widthSegments (pythreejs.SphereGeometry attribute), 75
 wireframe (pythreejs.MeshBasicMaterial attribute), 111
 wireframe (pythreejs.MeshDepthMaterial attribute), 114
 wireframe (pythreejs.MeshLambertMaterial attribute), 116
 wireframe (pythreejs.MeshNormalMaterial attribute), 118
 wireframe (pythreejs.MeshPhongMaterial attribute), 121
 wireframe (pythreejs.MeshStandardMaterial attribute), 127
 wireframe (pythreejs.ShaderMaterial attribute), 132
 WireframeGeometry (class in pythreejs), 83
 wireframeLinecap (pythreejs.MeshBasicMaterial attribute), 111
 wireframeLinecap (pythreejs.MeshLambertMaterial attribute), 116
 wireframeLinecap (pythreejs.MeshPhongMaterial attribute), 122
 wireframeLinecap (pythreejs.MeshStandardMaterial attribute), 127
 wireframeLinejoin (pythreejs.MeshBasicMaterial attribute), 112
 wireframeLinejoin (pythreejs.MeshLambertMaterial attribute), 116
 wireframeLinejoin (pythreejs.MeshPhongMaterial attribute), 122
 wireframeLinejoin (pythreejs.MeshStandardMaterial attribute), 127
 wireframeLinewidth (pythreejs.MeshBasicMaterial attribute), 111
 wireframeLinewidth (pythreejs.MeshDepthMaterial attribute), 114
 wireframeLinewidth (pythreejs.MeshLambertMaterial attribute), 117
 wireframeLinewidth (pythreejs.MeshNormalMaterial attribute), 118
 wireframeLinewidth (pythreejs.MeshPhongMaterial attribute), 121
 wireframeLinewidth (pythreejs.MeshStandardMaterial attribute), 127
 wireframeLinewidth (pythreejs.ShaderMaterial attribute), 132

wrapS (pythreejs.Texture attribute), 158
wrapT (pythreejs.Texture attribute), 158

X

x (pythreejs.Quaternion attribute), 139

Y

y (pythreejs.Cylindrical attribute), 137
y (pythreejs.Quaternion attribute), 140

Z

z (pythreejs.Quaternion attribute), 140
zeroSlopeAtEnd (pythreejs.AnimationAction attribute),
22
zeroSlopeAtStart (pythreejs.AnimationAction attribute),
22
zoom (pythreejs.CombinedCamera attribute), 27
zoom (pythreejs.OrthographicCamera attribute), 29
zoom (pythreejs.PerspectiveCamera attribute), 31
zoomSpeed (pythreejs.OrbitControls attribute), 35
zoomSpeed (pythreejs.TrackballControls attribute), 39