郭隆邦 2023-06-29

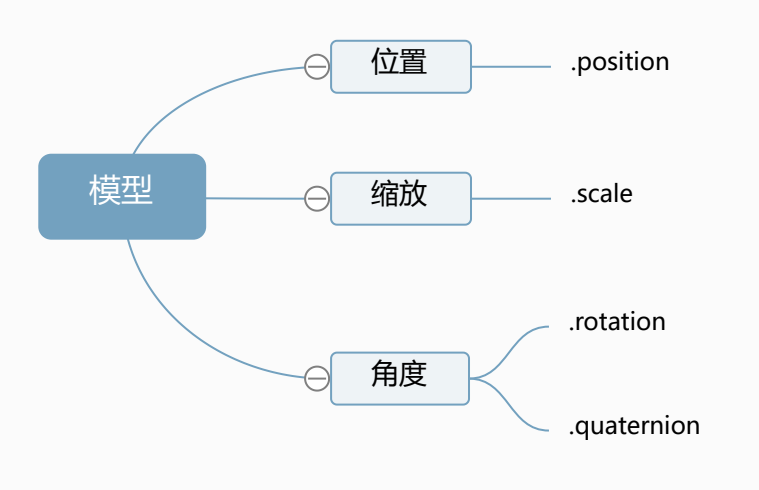
5. 模型本地矩阵、世界矩阵

本地矩阵 `.matrix` 、世界矩阵 `.matrixWorld`

查看Three.js文档，模型对象的父类 `Object3D` ，你可以看到本地矩阵 `.matrix` 和世界矩阵 `.matrixWorld` 两个属性。

知识回顾：模型位置、缩放、角度属性

通过基础部分学习，大家知道，通过位置属性 `.position` 可以平移模型，通过 `.scale` 属性可以缩放物体，通过四元数 `.quaternion` 或角度 `.rotation` 属性旋转改变物体的姿态角度。四元数 `.quaternion` 或角度 `.rotation` 属性都是用来表示物体姿态角度的，一个变化，另一个也会跟着变化。

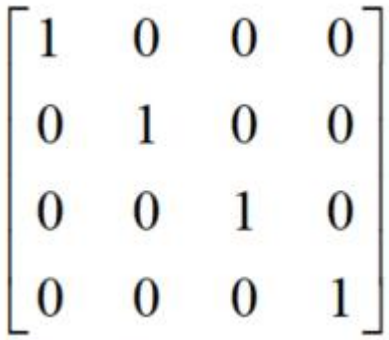


模型本地矩阵属性 `.matrix`

模型本地矩阵属性 `.matrix` 的属性值是一个4x4矩阵 `Matrix4` 。

```js
console.log('mesh.matrix',mesh.matrix);
```

当你没有对模型进行旋转、缩放、平移的时候，模型本地矩阵属性 `.matrix` 的默认值是一个单位矩阵。

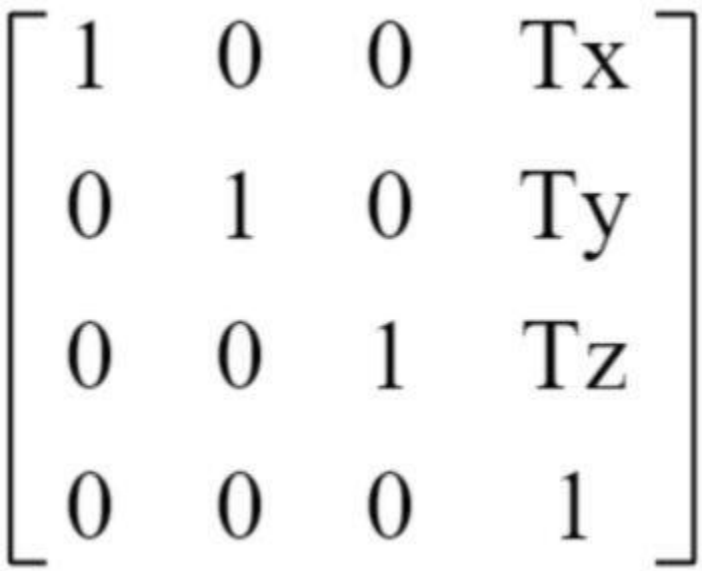


平移、旋转、缩放，查看 `.matrix` 变化

你可以尝试改变three.js模型对象的 `.position` 、 `.scale` 或 `.rotation` (`.quaternion`)任何一个属性，查看查看 `mesh.matrix` 值的变化。

1.仅改变 `mesh.position` ,你会发现 `mesh.matrix` 的值会从单位矩阵变成一个平移矩阵(沿着xyz轴分别平移2、3、4)😭)

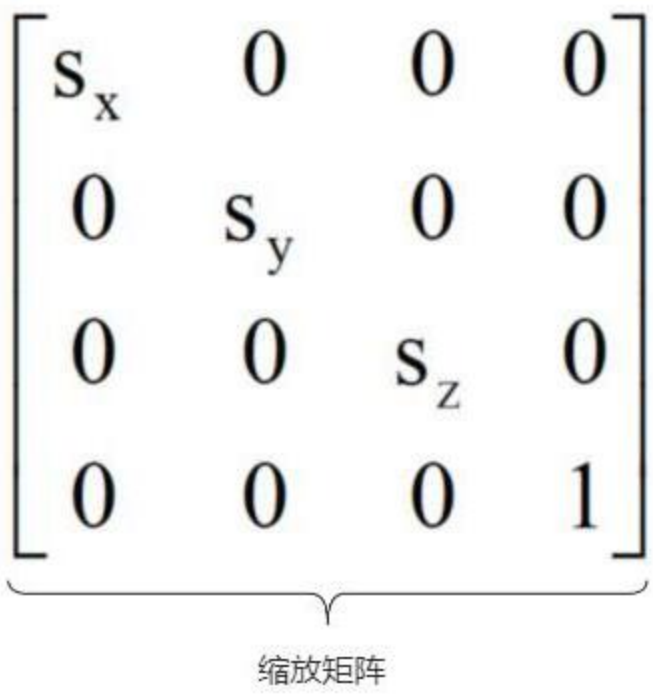
```js
// 不执行renderer.render(scene, camera);情况下测试
mesh.position.set(2,3,4);
mesh.updateMatrix();//更新矩阵，.matrix会变化
console.log('本地矩阵',mesh.matrix);
```



2.仅改变 `mesh.scale` ,你会发现 `mesh.matrix` 的值会从单位矩阵变成一个缩放矩阵(沿着xyz轴三个方向分别缩放6、6、6倍);

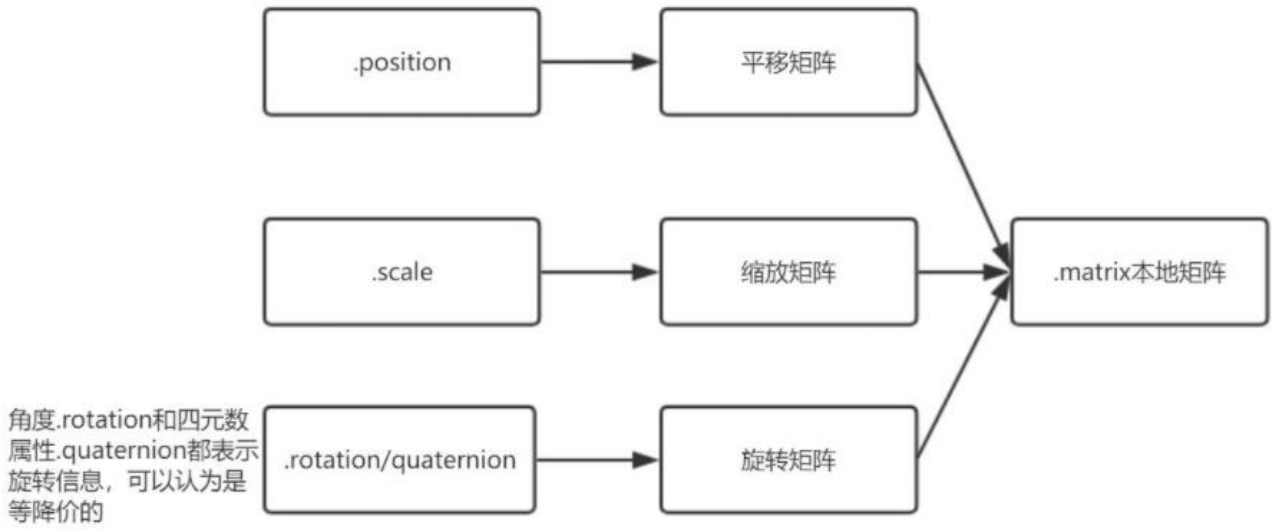
```js
mesh.scale.set(6,6,6);
mesh.updateMatrix();
```

```
console.log('本地矩阵',mesh.matrix);
```



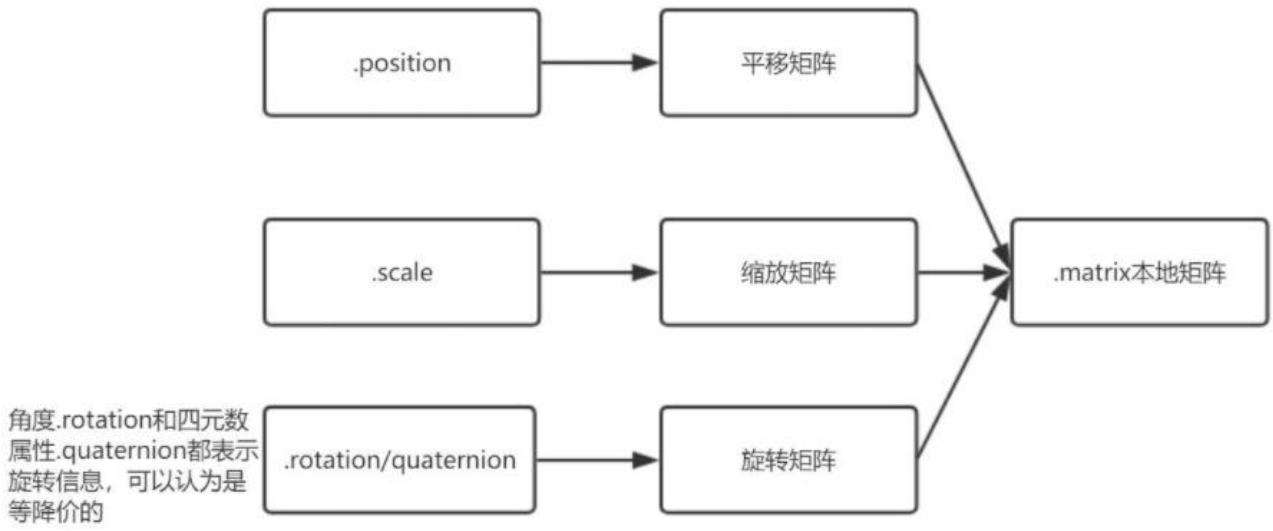
3.同时平移和缩放，查看`.matrix`变化，你可以看到`mesh.matrix`是平移矩阵和缩放矩阵的复合矩阵。

```js
mesh.position.set(2,3,4);
mesh.scale.set(6,6,6);
mesh.updateMatrix();
console.log('本地矩阵',mesh.matrix);
```



总结

当你改变模型位置`.position`、缩放`.scale`或角度`.rotation`(`.quaternion`)任何一个属性的时候，都会影响`.matrix`的值。`.matrix`就是本质上就是旋转矩阵、缩放矩阵、平移矩阵的复合矩阵。



世界矩阵`.matrixWorld`

了解世界矩阵属性`.matrixWorld`之前，先回顾下前面基础内容讲解的知识点本地坐标和世界坐标：http://www.webgl3d.cn/pages/00ddfa/

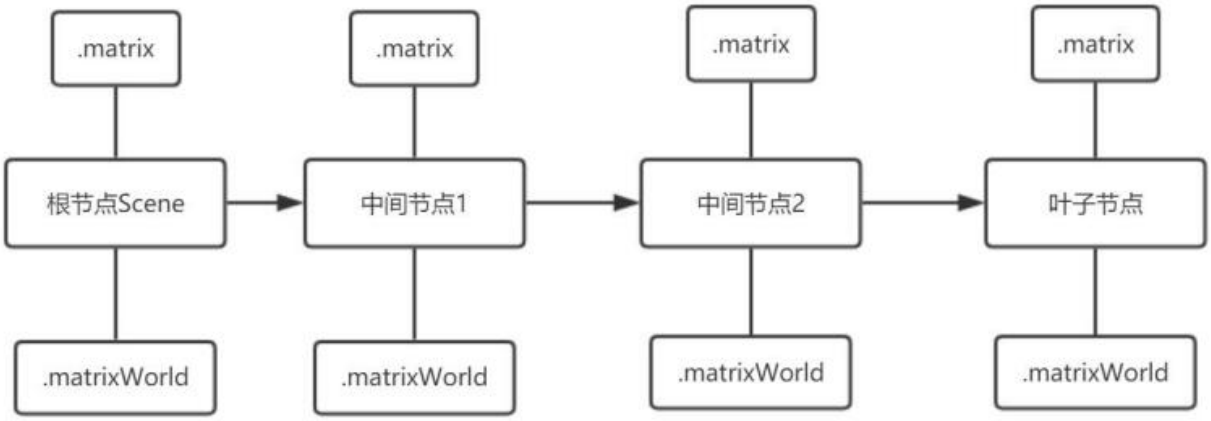
位置属性`.position`表示一个模型相对父对象的偏移，或者说相对本地坐标系的位置。

```
console.log('本地坐标',mesh.position);
```

通过`.getWorldPosition()`获取的世界坐标，是模型相对世界坐标系的坐标，也就是该对象及其父对象所有`.position`属性值的累加。

```
const worldPosition = new THREE.Vector3();
mesh.getWorldPosition(worldPosition)
console.log('世界坐标',worldPosition);
```

一个模型对象的世界矩阵属性`.matrixWorld`是自身及其所有父对象本地矩阵属性`.matrix`的复合矩阵。



根节点世界矩阵.matrixWorld = 根节点本地矩阵.matrix

中间节点1.matrixWorld = scene.matrixWorld X 中间节点1.matrix

中间节点2.matrixWorld= 中间节点1.matrixWorld X 中间节点2.matrix

叶子节点.matrixWorld= 中间节点2.matrixWorld X 叶子结点.matrix

对象世界矩阵=父对象世界矩阵 X 自身本地矩阵

```
// 你可以通过下面代码测试上面结论
mesh.position.set(2,3,4);
const group = new THREE.Group();
group.add(mesh);
group.position.set(2,3,4);

//执行updateMatrixWorld()方法，模型的本地矩阵和世界属性都会更新
mesh.updateMatrixWorld();
console.log('本地矩阵',mesh.matrix);
console.log('世界矩阵',mesh.matrixWorld);
```

不执行`.updateMatrixWorld()`，`.render()`之后查看矩阵

当你改变`.position`、`.scale`等属性，不执行`.updateMatrixWorld()`更新矩阵矩阵，在`.render`之后查看本地矩阵和世界矩阵的值，你会发现发生了变化。这说明three.js默认情况下，在执行`.render()`的时候，会自动获取`.position`、`.scale`等属性的值，更新模型的本地矩阵、世界矩阵属性。

```
const scene = new THREE.Scene();
mesh.position.set(2,3,4);
const group = new THREE.Group();
group.position.set(2,3,4);
```

```
group.add(mesh);
scene.add(group);

// render渲染时候，会获取模型`.position`等属性更新计算模型矩阵值
renderer.render(scene, camera);

console.log('本地矩阵',mesh.matrix);
console.log('世界矩阵',mesh.matrixWorld);
```

← 4. 矩阵乘法multiply

6. 视图矩阵、投影矩阵→