FAQ

Sistemas Gráficos

Grado en Ingeniería Informática

Curso 2019/2020

Pregunta

Si yo he implementado una clase que representa un cilindro, ¿puedo usar esa clase para construir un nodo BSP y hacer operaciones booleanas?

Respuesta

La respuesta es sí, pero **hay que tener claro cómo están estructurados los datos** de lo que construimos para poder luego referenciarlos adecuadamente. En general, hay que tener claro:

- De qué tipo es cada cosa
- De qué tipo es cada parámetro que espera un método o un constructor.
- De qué tipo es lo que devuelve cada método.
- De qué tipo es cada atributo que añadimos o usamos.
- Qué significa que una clase hereda de otra.
- **Y sobre todo**, cómo es la estructura de lo que se está construyendo, qué objetos hay y cómo se relacionan.

Supongamos que tenemos una clase con esta declaración y un constructor como el que se muestra.

```
class MiCilindro extends THREE.Object3D {
    constructor () {
        var geometria = new THREE.CylinderGeometry( ... );
        var unMaterial = new THREE.MeshNormalMaterial();

        this.cilindro = new THREE.Mesh(geometria, unMaterial);

        this.add (this.cilindro);  // el Mesh this.cilindro lo hacemos hijo del nodo this
    }
}
```

Si desde otra clase intento usar una instancia de MiCilindro para construir un ThreeBSP, ¿cómo lo hago?

```
var unCilindro = new MiCilindro();
var unNodoBSP = new ThreeBSP ( /* ¿qué ponemos aquí? */ );
```

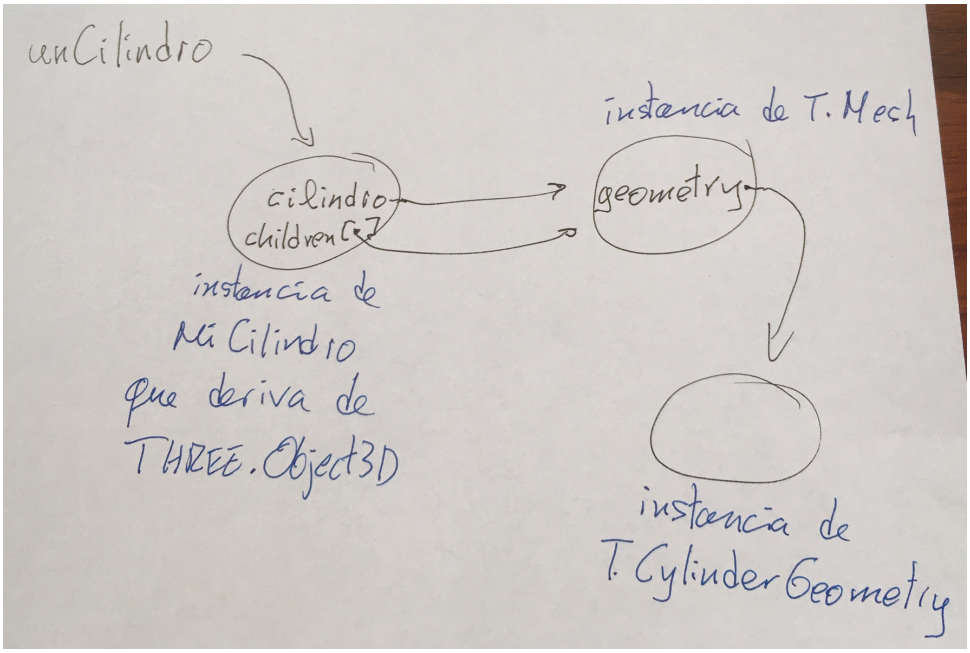
Figura 1:

Desde luego, no podemos usar unCilindro como parámetro, ya que unCilindro es una instancia de mi clase MiCilindro que deriva de THREE.Object3D, y eso no es una geometría, ni de cilindro ni de nada. Y el constructor de ThreeBSP lo que espera es una geometría.

Lo principal es saber que objetos se han creado cuando he instanciado MiCilindro y cómo se relacionan. Y cuando eso lo tenga claro, sabré qué debo poner como parámetro en ese constructor.

La variable unCilindro es una instancia de MiCilindro, que por herencia es un Object3D, que tiene un hijo (que está referenciado por un atributo, llamado cilindro) que es un Mesh, y dicho Mesh tiene a su vez un atributo, llamado geometry, que sí referencia a una geometría (un CylinderGeometry en este caso).

Tal vez se vea mejor en esta imagen. Las burbujas son objetos, las flechas son referencias, lo que hay dentro de las burbujas son atributos y en azul se indica a qué clase pertenece cada objeto.



Conociendo eso, ya sabemos como acceder a la geometría que hace falta para construir el objeto ThreeBSP.

Aunque alguno estará pensando en llegar a la geometría mediante una cadena casi interminable de objetos y atributos separados por puntos. Algo del tipo unCilindro.algo.otraCosa.otroAtributoMas.geometry, **no lo recomiendo de ninguna forma**. Esa forma de programar solo es una fuente de errores futuros ... y presentes también. Por ser poco legible y no fácilmente modificable.

¿Y ese quién es? El primo de la cuñada del sobrino del amigo de tu tío Juan. Clarísimo, ¿no?

Lo que recomiendo es añadir un consultor a la clase MiCilindro y usarlo.

A la clase MiCilindro se le añadiría

```
getGeometria () {
    return this.cilindro.geometry;
}
```

Con esto el código de la figura 1 quedaría tal que así

```
var unCilindro = new MiCilindro();
var unNodoBSP = new ThreeBSP (unCilindro.getGeometria());
```