Interacción a través del ratón.

El mecanismo estándar de interacción con los objetos de una escena 3D programada con Three.js es a través de las clases THREE.Projector y THREE.Raycaster (se recomienda ver su documentación en el manual oficial http://mrdoob.github.com/three.js/docs/57/).

Para comprender su funcionamiento, se propone el uso del ejemplo mostrado por el fichero "canvas_interactive_cubes_tween.html", de la batería de ejemplos oficiales de Three.js (el listado completo de ejemplos se puede descargar desde https://github.com/mrdoob/three.js/).

Analizad los siguientes aspectos para el ejemplo anterior:

- Uso de un renderer de tipo canvas para escenarios sencillos respecto del uso de un renderer WebGL para escenas más complejas como, por ejemplo, la dibujada en "webgl_interactive_cubes.html".
- Utilización de la librería Tween.js (https://github.com/sole/tween.js) para la creación de interpolaciones avanzadas. La integración de dicha librería con Three.js para la realización de animaciones queda muy clara en la siguiente entrada corta del blog Learning Three.js: http://learningthreejs.com/blog/2011/08/17/tweenjs-for-smooth-animation/

Sin embargo, se han desarrollado extensiones que permiten gestionar los eventos del ratón mediante un mecanismo similar a cómo se hacía en JavaScript para el resto de objetos pertenecientes al árbol DOM.

El repositorio threex (https://github.com/jeromeetienne/threex) proporciona un conjunto de librerías en JavaScript que facilitan la interacción con el usuario. De entre ellas, la librería "domevent" se encarga de los eventos del ratón.

La utilización de esta librería se pone de manifiesto en el ejemplo oficial: http://jeromeetienne.github.com/threex/examples/threex.domevent. Podéis leer una descripción de este código en un post del blog Learning Three.js: http://learningthreejs.com/blog/2012/01/17/dom-events-in-3d-space/.

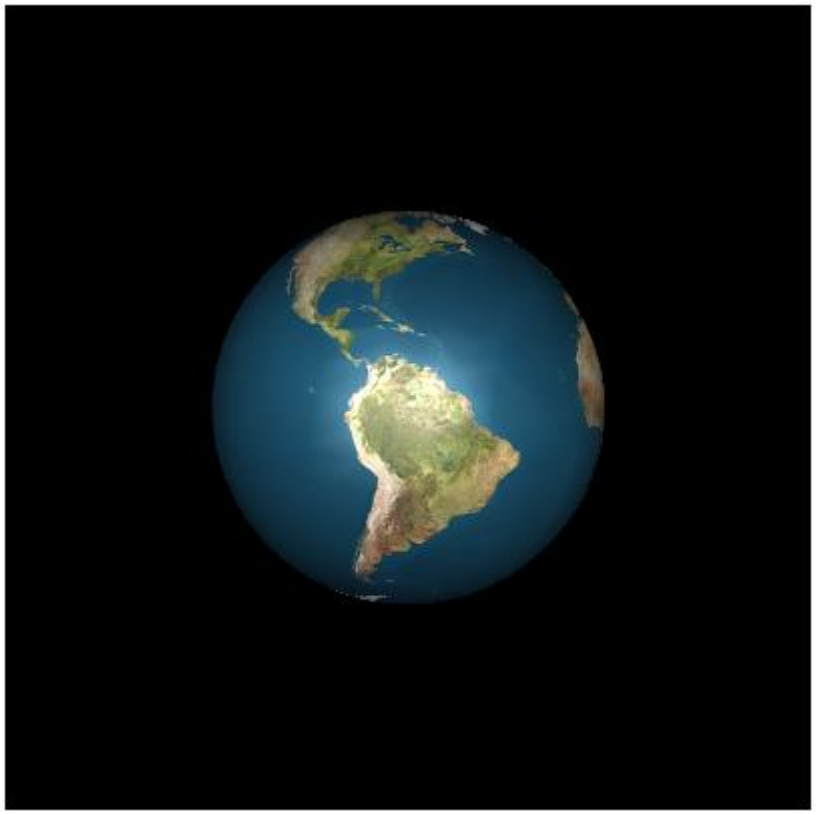
Habiendo consultado toda la información anterior, se pide:

- Modificad el ejemplo "canvas_interactive_cubes_tween.html" para que se comporte de la misma manera, pero utilizando la librería threex.
- *NOTA 1:* Dado que la librería "domevent" sustituye el objeto Object3D de Three.js, será necesario añadir la siguiente línea de código: `THREE.Object3D._threexDomEvent.camera(camera);`
- *NOTA 2:* Desafortunadamente, la librería "domevent" no está actualizada para el API de la última versión de la librería Three.js (r57, en el momento de realización de este boletín). Por ello, será necesario, utilizar la versión modificada de la librería "domevent" que se suministra en el aula virtual (fichero "threex.domevent.r57.js").

Esta versión es el resultado de reemplazar el método “subSelf” por el método “sub” cuando se usa la clase “Vector3” y de sustituir la clase “Ray” por “Raycaster”.

Globo terráqueo.

Utilizando Three.js, dibujad una escena 3D que muestre un globo terráqueo sobre fondo negro que gire sobre su eje (ligeramente inclinado) y tenga el siguiente aspecto:



Una vez conseguido lo anterior, se pide que al pinchar sobre la esfera el planeta invierta su sentido de giro.