# Cesium

Librería basada en javaScript para la creación de globos terrestres en 3D y mapas en 2D en el navegador sin la necesidad de algún plugin, es de libre acceso (open source) bajo licencia Apache 2.0.

A diferencia de los otros frameworks basados en javaScript este no posee *cdn*, por lo tanto, para su uso debemos descargarnos el framework desde la página oficial ( <http://cesiumjs.org/downloads.html> ).

Esencialmente utilizaremos la librería para acceder a los datos de las elevaciones del terreno. Resumiendo el uso de cesium:

* Con las coordenadas de la ruta accedemos a los tiles que nos ofrece cesium. Los tiles son rectángulos que guardan información acerca de los datos del terreno.
* La información obtenida en lo tiles la graficaremos usando *threeJS*.

## Uso de Cesium

En este ejemplo de Cesium usaremos las coordenadas de una zona de Pamplona, en específico latitud *42.8* y longitud *-1.634*.

Para obtener los datos de cesium debemos seguir los siguientes pasos:

* Declarar un proveedor de datos de Cesium, haremos un bucle hasta que el proveedor esté listo.
* Con las coordenadas obtenemos la posición del tile y de este obtenemos la información. Para obtener la información del tile usaremos una llamada asíncrona que nos proporciona Cesium, para mayor comodidad almacenaré cada petición en un arreglo y usare una función que ofrece Cesium que se ejecutará cuando las peticiones hayan acabado.

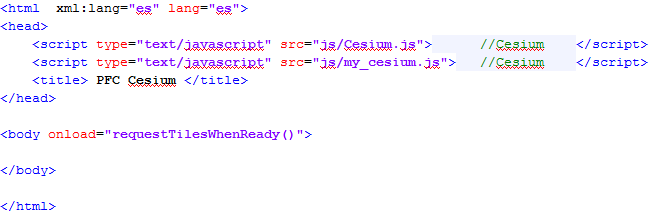


Figura x.x – PFCCesium.html

Se llama a la función *requestTileWhenReady* que se encargará de establecer la conexión con Cesium usando un objeto *CesiumTerrainProvider*. Se creará una función recursiva que llamará tantas veces sea necesario a la función hasta obtener la conexión, todas las funciones y código javaScript estará en el fichero */js/my\_cesium.js*

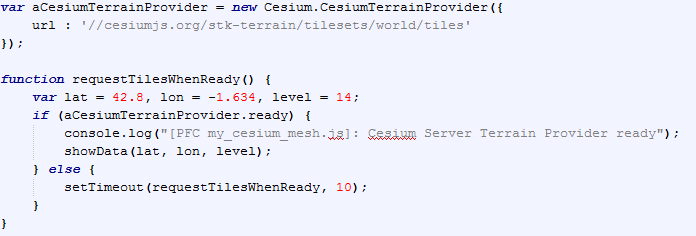


Figura x.x – Cesium función recursiva.

Cuando se haya establecida la conexión llamaremos a la función *showData* pasando como argumentos la latitud, longitud y el nivel. El nivel es la aproximación real de los datos, es decir, mientras mayor sea el nivel más específico son los datos y dependiendo de la ubicación geográfica hay más o menos niveles.

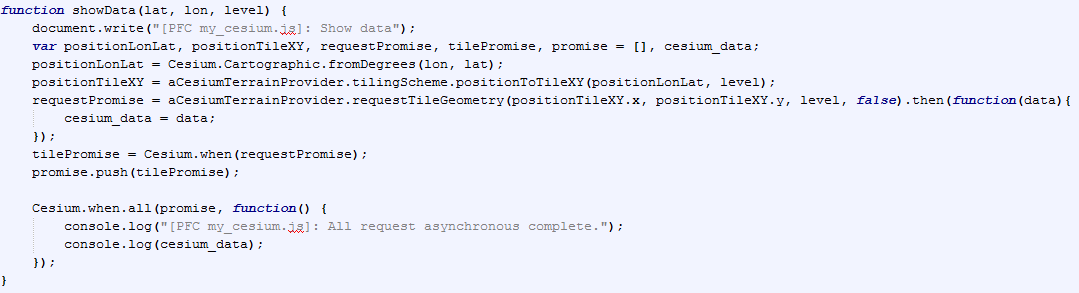


Figura x.x – Cesium función obtención de datos.

En la función *showData* realizamos lo siguiente:

* *Cesium.Cartographic.fromDegrees(longitude, latitude, height optional, reult optional)*: Creamos una instancia del tipo *Cartographic* (objeto Cesium) a partir de una longitud y latitud en grados, el objeto resultante está en radianes.
* *Cesium.CesiumTerrainProvider.tilingScheme.positionToTileXY(position, level, result optional)*: Calculamos las coordenadas x e y del tile según la posición de las coordenadas.
* *Cesium.CesiumTerrainProvider.requestTileGeometry(x, y, leve, throttleRequests optional)*: Obtenemos la geometría a partir del tile indicado.
* Almacenaremos la información en la variable *cesium\_data*.
* Crearemos por cada petición una variable usando la función *Cesium.when(petición)* y la almacenaremos en el arreglo *promise*. Implementamos el método *Cesium.when.all()* que nos indicará que las peticiones se han realizado y mostraremos la información por la consola del navegador.



Figura x.x – Datos Cesium.

De los datos ofrecidos por cesium usaremos los siguientes:

* *maximumHeight* y *minimumHeight* como los valores mínimos y máximos de alturas del terreno.
* *uValues*, *vValues* y *heightValues* almacenados en el objeto *quantizedVertices* que contiene tripletes donde guardamos la longitud, latitud y altura.
* Con estos datos crearemos las geometrías de los terrenos, esto se explicará más adelante en el desarrollo del proyecto.