Ejercicio: Consulta y visualización de resultados

Introducción

Las consultas a servicios GIS son unas de las tareas que más veces se van a realizar con los visores cartográficos. Una vez realizada una consulta, se plantean varias cuestiones para el desarrollador del visor:

- ¿De qué manera o maneras debo mostrar el resultado?
- ¿Voy a pintar el resultado de la consulta? ¿Cómo voy a pintar el resultado?
- ¿Solo quiero mostrar la información alfanumérica en una tabla?

En esta práctica se van a combinar varias clases del API de JavaScript de ArcGIS como, por ejemplo:

Query; QueryTask; Graphic; GraphicsLayer; Renderer;...

HTML y el DOM

Para esta práctica vamos a necesitar dos elementos principales. El primer elemento será un contenedor en el que vamos a dibujar nuestro mapa. El otro elemento será una tabla que crearemos y que poblaremos con datos cuando nos llegue el resultado de la consulta.

El segundo elemento será de tipo tabla. En esta tabla se tendrá que crear con un primer elemento que será la cabecera de la tabla (**thead**). En esta cabecerá se pondrán las columnas que se quieran de la tabla de atributos de las entidades (una para el **objectID**, otra para la **región** y otra para la **población**).

También habrá que crear el elemento "**tbody**" de la tabla, aunque estará vacío ya que aún no se tienen los resultados de la consulta.

En la tabla, cada fila corresponderá a una entidad resultado de la consulta y se crearán estos elementos de HTML de manera dinámica mediante JavaScript.

Empezando… ¿desde cero?

Como en la práctica o ejercicio anterior, no es recomendable comenzar una aplicación usando un editor y una página en blanco. Se recomienda basar el desarrollo en algún ejemplo que ya exista y que sea sencillo. Por ejemplo:
https://developers.arcgis.com/javascript/3/sandbox/sandbox.html?sample=map_simple

Descarga el código fuente, guárdalo en tu servidor IIS y comienza a editarlo.

Mapa y capas del mapa

Para esta aplicación únicamente será necesario crear un mapa con un BaseMap. Además, se creará una capa dinámica y se añadirá al mapa. El servicio de mapas que se va a cargar es:

http://sampleserver6.arcgisonline.com/arcgis/rest/services/USA/MapServer

Consulta

Se quiere consultar la capa de ciudades (id = 0) del servicio anterior y se quieren obtener las ciudades con una población mayor de 500000 habitantes y que intersecten en un radio de 600 km.

Para realizar la consulta se necesitarán dos objetos: el objeto Query y el objeto QueryTask. En el objeto Query se establecerán los parámetros de la consulta:

- población mayor de 500000 habitantes
- que devuelva la geometría
- que el sistema de referencia sea el mismo que el del mapa
- que los únicos campos que devuelva la consulta sean: objectId, el areaname y el pop2000
- que la relación espacial sea la intersección
- la geometría para la relación espacial tendrá centro en el punto que se haga clic y de radio 600km

El objeto QueryTask será el encargado de lanzar la consulta al servicio, con lo que se creará con su correspondiente URL.

En qué momento se ejecutará la consulta

La consulta se ejecutará en el momento en que se haga clic en el mapa, con lo que se creará una función para el evento clic del mapa que ejecute la consulta con el objeto query completo y también estableciendo una función de callback.

La función de callback

En la función de callback se va a recibir un objeto de tipo featureSet. En este objeto hay una propiedad features con un array de objetos graphic. Con estos objetos graphic se deberán realizar diversas acciones:

1. Poblar la tabla de entidades con los atributos de las entidades.
2. Dibujar los gráficos en el mapa.

Poblar la tabla

Antes de poblar la tabla con los atributos de las entidades, se deben de borrar los elementos que existen, con lo que tenemos que usar esta. Con lo que usaremos la siguiente función:

```
//función que borra todas las filas de una tabla
function borrarTabla(){
	//se recoje la tabla
	var tabla = document.getElementById("tbody");
	//se comprueba que la tabla tenga algún elemento
	if (tabla.firstChild){
	//mientras la tabla tenga una fila, se borrará este elemento
		while(tabla.firstChild){
			tabla.removeChild(tabla.firstChild);
		};
	}
};
```

Por otra parte, dentro de esta función de callback:

1. Recoger los atributos de cada uno de los gráficos resultado
2. Dibujar las entidades
3. Poblar la tabla

1-. Recoger los atributos de cada uno de los gráficos resultado

Con un bucle for, recorreremos el array de gráficos dentro del featureset.

- Añadimos el gráfico a la capa de gráficos del mapa (map.graphics).
- Pasamos el gráfico a una función que nos crearemos para poblar la tabla.

3-. Poblar la tabla:

Esta función accederá a los valores de los atributos del gráfico que le llegue.

```
var atributos = feature.attributes;
var oid = atributos.objectid;
var nombre = atributos.areaname;
var poblacion = atributos.pop2000;
```

Creará un elemento HTML fila, y un elemento celda para cada uno de los atributos.

```
    //se crea un elemento fila para la tabla
var fila = document.createElement("tr");

    //se crea un elemento celda
var celda = document.createElement("td");
```

Se añadirán los datos a cada celda.

```
    //se crea un elemento de texto con el valor del atributo
var textoCelda = document.createTextNode(valorAttributu)
    //se añade el valor a la celda
celda.appendChild(textoCelda);
```

Se añadirán estos elementos columna al elemento fila.

```
    //se añade la celda a la fila
fila.appendChild(celda);
```

Se añadirá el elemento fila a la tabla con.

```
    //se añade la fila a la tabla
tabla.appendChild(fila);
```

Dibujar las entidades

Las entidades resultado (objetos graphic) se han añadido a la capa de gráphicos del mapa, pero de momento, no se visualizan ya que ninguno de estos elementos tiene definida la simbología. Para simbolizar todos los gráficos de esta capa, se debe de aplicar un objeto Renderer a esta capa de gráficos.

Crearemos un SimpleRenderer para visualizar los datos todos con el mismo símbolo. Una vez se cree el renderer se aplicará a la capa de gráficos y posteriormente se actualizará la capa para que se actualice la visualización.

Hay que tener en cuenta la naturaleza de la geometría de los gráficos que se van a pintar, ya que dependiendo de la geometría se creará un símbolo de un tipo o de otro.

¿Cuántas veces se va a querer que se aplique el renderer? En principio se va a aplicar una vez el renderer con lo que se puede aplicar el renderer solamente al principio de la aplicación.

Opcional: Modificar la visualización

- (Opcional) Se pueden crear diferentes tipos de renderer (classbreakrenderer).
- (Opcional) Se pueden editar varias propiedades del simple renderer como la propiedad size o usar el método setColorInfo para modificar la rampa de color…
- (Opcional) Usar una librería externa para aplicar estilos CSS a la tabla de atributos.
- (Opcional) Crear una interfaz de usuario para modificar de manera dinámica los valores de la cláusula where de la consulta.

Función poblar tabla

```
//función que añade una fila con los respectivos valores de los atributos a la tabla
function poblarTabla(feature){
    var tabla = document.getElementById("tbody");
        //se recuperan los atributos de la entidad que llega a la función
    var atributos = feature.attributes;
    var oid = atributos.objectid;
    var nombre = atributos.areaname;
    var poblacion = atributos.pop2000;
        //se crea un elemento fila para la tabla
    var fila = document.createElement("tr");
    var filaData = [oid, nombre, poblacion];
        //se recorren los datos de los atributos
    for(i in filaData){
            //se crea un elemento celda
        var celda = document.createElement("td");
            //se crea un elemento de texto con el valor del atributo
        var textoCelda = document.createTextNode(filaData[i])
            //se añade el valor a la celda
        celda.appendChild(textoCelda);
            //se añade la celda a la fila
        fila.appendChild(celda);
    }
        //se añade la fila a la tabla
    tabla.appendChild(fila);
};
```