Proyecto YuppGIS

Plugin jQuery para OpenLayers

Página Web

http://yuppgis.googlecode.com/

Estudiantes

Emilia Rosa

Jorge Davison

Martin Taruselli

German Schnyder

Para representar un mapa a nivel de interfaz de usuario, y poder interactuar con él, se implementó un plugin jQuery. Esto permite manejar el objeto independientemente del HTML visualizado y de la ejecución de lógica en el servidor, lo que aliviana la dependencia de datos entre ambas capas y permite una mayor fluidez a la hora del desarrollo.

En la instalación de YuppGIS se copia el archivo .js que contiene el plugin, y el framework se encarga de importarlo cuando se decida incluir un mapa mediante el helper correspondiente. Este complemento queda integrado a la implementación de OpenLayers que se incluye como parte del componente.

El plugin se divide en dos grandes partes, la inicialización y los métodos para modificar su estado. La primera parte consiste en la inicialización de variables, parámetros por defecto y por último, pero no menos importante, la obtención de las capas que van a representarse en el mapa.

La obtención de las capas se realiza a través de pedidos AJAX al controlador proporcionado para el manejo de las acciones geográficas base, por lo que no es necesario implementar nada en este sentido (los diferentes pedidos para la inicialización están resueltos en base a convenciones y a métodos ya existentes).

Para ejecutar la inicialización se dispone de un UI helper que invoca al constructor del mapa sobre los elementos que se indiquen de acuerdo a la sintaxis clásica de jQuery:

```
/*muestra un mapa en el elemento del DOM con id=mapid*/
$("#mapid").YuppGISMap(<params>)
```

Cabe destacar que el plugin puede ser instanciado sobre el resultado de cualquier selector de jQuery, por lo que sería posible instanciar varios mapas a la vez si el selector devolviera más de un resultado:

```
/*muestra un mapa en cada elemento del DOM con class=map*/
$(".map").YuppGISMap(<params>)
```

A su vez, este constructor es el que habilita la operativa sobre el mapa a través de diferentes métodos, por lo que si se invoca el constructor del plugin sobre un elemento que ya contiene un mapa es posible acceder a las propiedades del mismo a la vez que es posible invocar los diferentes métodos disponibles:

```
/*creo el mapa*/
$("#mapid").YuppGISMap(<params>);
….
….
/*accedo al mapa representado en el elemento con id=mapid*/
/*e invoco el método method1() sobre él*/
$("#mapid").YuppGISMap().method1();
```

Las operaciones que se brindan en el plugin son:

```
/*Agrega un handler de click al mapa*/
addClickHandler

/*Agrega un handler de select al mapa*/
addSelectHandler

/*Elimina el handler de click del mapa*/
removeClickHandler
```

/*Elimina el handler de select del mapa*/
removeSelectHandler

/*Obtiene todos los handlers registrados*/
getHandlers

/*Obtiene las capas visibles en el mapa*/
getVisibleLayers

/*Oculta los elementos especificados por id*/
hideFeatures

/*Vuelve visibles los elementos especificados por id*/
showFeatures

/*Obtiene un objeto json que contiene toda la información de visualización asociada al mapa*/
getVisualizationState

/*Restaura la información de visualización asociada al mapa especificada por parámetro*/
loadVisualizationState

Es importante aclarar que esta lista es extensiva de lo que se brinda en el plugin, pero no es para nada definitiva. Se pueden incorporar todas las funcionalidades que se desee, teniendo en cuenta lo siguiente:

- Respetar la declaración de los métodos como públicos y relativos al plugin, como por ejemplo:

  ```
  this.removeClickHandler = function (handler) {
          return _removeHandler(handler, "click");
  };
  ```

- Respetar la estructura del plugin en cuanto a la posibilidad de ir componiendo operaciones al estilo pipe-filters. Es decir, poder concatenar acciones a aplicar sobre el mapa donde cada una aplica sobre la anterior, por ejemplo:

  ```
  var handler1 = …
  var handler2 = …
  $(“#map”).addClickHandler(handler1).addClickHandler(handler2);
  ```

- La implementación de la funcionalidad depende de la versión de OpenLayers que se maneje, dado que para realizar acciones sobre el visualizador es necesario realizarlo a través de la API brindada por el mismo.