Gobierno de Santa Fe

Ministerio de Gobierno e Innovación Pública

Secretaría de Tecnologías para la Gestión

IDESF

Manual de uso para desarrolladores

Módulo de DIBUJO

Fecha de creación: 02/07/2023

Fecha de última modificación: 21/05/2025

Versión del módulo: 1.1.0

Novedades:

12/12/2023	Modulo inicial	Permite búsquedas por domicilios y graficado de puntos líneas polígonos, y superposición de capas a demanda.
27/09/2024	Métodos nuevos	Agregado de marcado de calles, y exportación en nuevos formatos WKT y WKT_5347
21/05/2025	Métodos nuevos	• Se habilitaron los métodos de posicionamiento en coordenadas iniciales por URL. • Se agrego el método de carga de parcelas catastrales, con opción de snapping o autoensamblado. • Se configuraron los modos de tipo de servicio , para habilitar diferentes configuraciones de mapas base. • Se habilito la búsqueda por identificador de parcela en el buscador (solo 1 parcela por búsqueda)

Índice de contenido

1- Introducción

El Módulo de Dibujo es una aplicación embebible que permite graficar objetos geográficos sobre un mapa de infraestructura base, realizar la selección de los mismos, edición y su posterior exportación en diferentes formatos basados en estándares, siempre dentro de los límites provinciales, utilizando los servicios de la Infraestructura de Datos Espaciales de la provincia de Santa Fe (IDESF).

2- Funcionalidades

- El Módulo de ubicación permite superponer una capa vectorial, basada en filtros de selección, ya sea desde Geoserver como desde un objeto GeoJSON, sobre una capa base WMST de infraestructura Provincial.
- Permite realizar la selección directa por elemento geográfico cargado.
- Permite dibujar Puntos Líneas y Polígonos
- Brinda la opción de exportar los elementos seleccionados en formatos estándar de intercambio de datos geográficos.
- Es embebible e interactivo.

3 - Cómo embeber el widget "Módulo de Dibujo"

Para hacer uso del Módulo se deben incorporar algunas librerías Javascript que permitirán la inicialización correcta del mismo. A continuación, se explican los pasos a seguir para lograr su correcto funcionamiento:

3.1.- Librerías básicas

```
<script src="https://www.santafe.gob.ar/idesf/mod-
dibujo/web/js/easyXDM/easyXDM.min.js"></script>
```

3.2.- Página principal del cliente

Incorporar, en la posición donde se desea embeber el módulo, el siguiente bloque de código HTML:

```
<!-- Tag BASICO PARA  MODULO-->
<div id='embebido' class="fieldset" height="auto" scrolling="no" frameborder="0"></div>
```

3.3.- Funcionalidad principal <script>

A continuación del elemento *div* anterior, insertar una sección **<script>.** En dicha sección debe ir el código JavaScript que se encarga de cargar el módulo. Éste utiliza la librería incorporada en la sección 3.1 y permite inyectar un elemento HTML -iframe- dentro del contenedor definido, realizando el intercambio de mensajes entre el cliente y el módulo:

```
<script>

        var resultado;
        var datos;
        var imagen;
        var comunicacion;
        var socket;

        comunicacion = {  //instancia local de XDM
                        easyXDM: window.easyXDM.noConflict("comunicacion")
                    };
        if(coordenadas !== '')
        {
            string_coordenadas = '&COORDENADAS='+coordenadas;
        }
        else
        {
            string_coordenadas = '';
        }

        var opciones = '&CONTROLES='+JSON.stringify({
                                                    "poligonos": true ,
                                                    "puntos":true,
                                                    "lineas":true,
                                                    "buscador":true
                                                }
                                                 );
      function getRandomIntInclusive() {
        return Math.floor(Math.random() * (10000 - 0 + 1) + 0);
      }
    $(document).ready(function()
    {

        socket = new comunicacion.easyXDM.Socket({
            "https://www.santafe.gob.ar/idesf/mod-dibujo/index_iframe.php?NAMESPACE=comunicacion"+string_coordenadas+opciones+"&TIPO_SERVICIO=PARCELAS&cache="+getRandomIntInclusive(), //NAMESPACE : como debe llamarse la instancia llamada
```

```
            container:embebido,
            onReady: function(){
                this.container.getElementsByTagName("iframe")[0].style.height =
(window.innerHeight-100)+"px";
                this.container.getElementsByTagName("iframe")[0].style.width =
window.innerWidth+"px";
                this.container.getElementsByTagName("iframe")[0].scrolling = "no";
                this.container.getElementsByTagName("iframe")[0].frameborder = "0";
            },
            onMessage:function(messageFuente, origin)
            {
        //TODOS LOS MENSAJES SON ASINCRONOS Y DEBEN MANEJARSE DENTRO DE ESTE CONTEXTO
        // ENVIAR MENSAJES Y REALIZAR TAREAS SOLO LUEGO DE LA RECEPCION DE LOS MISMOS
        // PARA EVITAR PERDIDA DE INFORMACION
                var message = null;
                if(typeof  messageFuente === "string")
                {
                    try{
                        message = JSON.parse(messageFuente);
                    }
                    catch( e)
                    {
                        console.log("Error al parsear json, error"+ e );
                    }
                    console.log(message);
                }

                if(typeof message === "number")
                    console.log(message);

                if(message)
                {
                    if(message.error !== undefined)
                    {
                        console.log(message.error);
                        return;
                    }

                    if(typeof message === 'object')
                    {
                        if(message.tipo !== undefined)
                        {
                            if(message.tipo === 'array')
                            {
                                resultado = message.valores;
                                if(message.origen !== undefined) //puede venir de
localidad o de calles
                                {
                                    if(message.origen === "getLocalidades")
                                    {
                                        jQuery('#localidades').children().remove();
                                        jQuery('#callesLocalidad').children().remove();
                                        resultado.forEach(function(val,idx){
                                    var unaOpcion = document.createElement("option");
                                     unaOpcion.value = val.codigo_distrito;
                                     unaOpcion.textContent= val.nombre_localidad;

                                                $('#localidades').append(unaOpcion);
                                            });
```

```
                                    }

                                    if(message.origen === "getCallesLocalidad")
                                    {
                                        jQuery('#callesLocalidad').children().remove();
                                        resultado.forEach(function(val,idx){
                                      var unaOpcion = document.createElement("option");
                                            unaOpcion.value = val.codigo_via;
                                            unaOpcion.textContent= val.nombre_via;

                                            $('#callesLocalidad').append(unaOpcion);
                                        });
                                    }
                                }

                            }

                            if(message.tipo === 'imagen')
                            {
                                imagen = message.imagen;
                                datos = message.info;
                            }

                        }
                        else
                        {
                            if(message.GeoJSON !== undefined)
                                resultado = message.GeoJSON;
                            else
                                resultado = JSON.parse(message);
                        }

                    }
                    else
                        resultado = message;
                }
            }
        });
    });
</script>
```

3.4.- Ejemplo práctico

Solicitar ejemplo PHP.

NOTA: Ejecutar siempre el código del script dentro de un bloque $(document).ready pero teniendo la precaución de dejar como globales a las variables:

```
var resultado;
var datos;
var imagen;
var comunicacion;
var socket;
```

3.4.1 – Definir controles del modulo

Para gestionar como se presenta el modulo podemos definir parámetros

```
var opciones = '&CONTROLES='+JSON.stringify({
                                                "poligonos": true ,
                                                "puntos":false,
                                                "lineas":false,
                                                "buscador":true
                                            }
                                             );
```

"poligonos": true	Habilita la herramienta para dibujar poligonos
"puntos":false	Habilita la herramienta para dibujar puntos
"lineas":false	Habilita la herramienta para dibujar lineas
"buscador":true	Habilita el buscador de domicilios.

3.4.2 – Posicionar Mapa en coordenadas iniciales.

Para gestionar la coordenada inicial del mapa debemos pasar las mismas en la URL de parametrización con el siguiente formato.

```
var coordenadas= "-31.64653122425081_-60.705519318580656"   //latitud_longitud
string_coordenadas = '&COORDENADAS='+coordenadas;
```

El formato es: primero latitud y luego longitud , separadas con _ en grados decimales utilizando sistema de coordenadas WGS84.

3.4.3 – Tipos de servicios o mapas base

Se configuran mediante el uso del parámetro de URL `TIPO_SERVICIO`

Si lo omitimos el mapa se configura con capas base oficiales de infraestructura del servicio de cache de mapas de la provincia.

Si la definimos tenemos varias alternativas:

`TIPO_SERVICIO=PARCELAS`
Mapa de infraestructura oficial con la superposición de la capa de parcelario catastral

`TIPO_SERVICIO=OS`
Mapa de base OpenStreet, con la superposición de la capa de infraestructura oficial.

`TIPO_SERVICIO=WMTS  // modo por default`
Mapa de base Provincias de Argentina con la capa de infraestructura oficial con la superposición del parcelario catastral

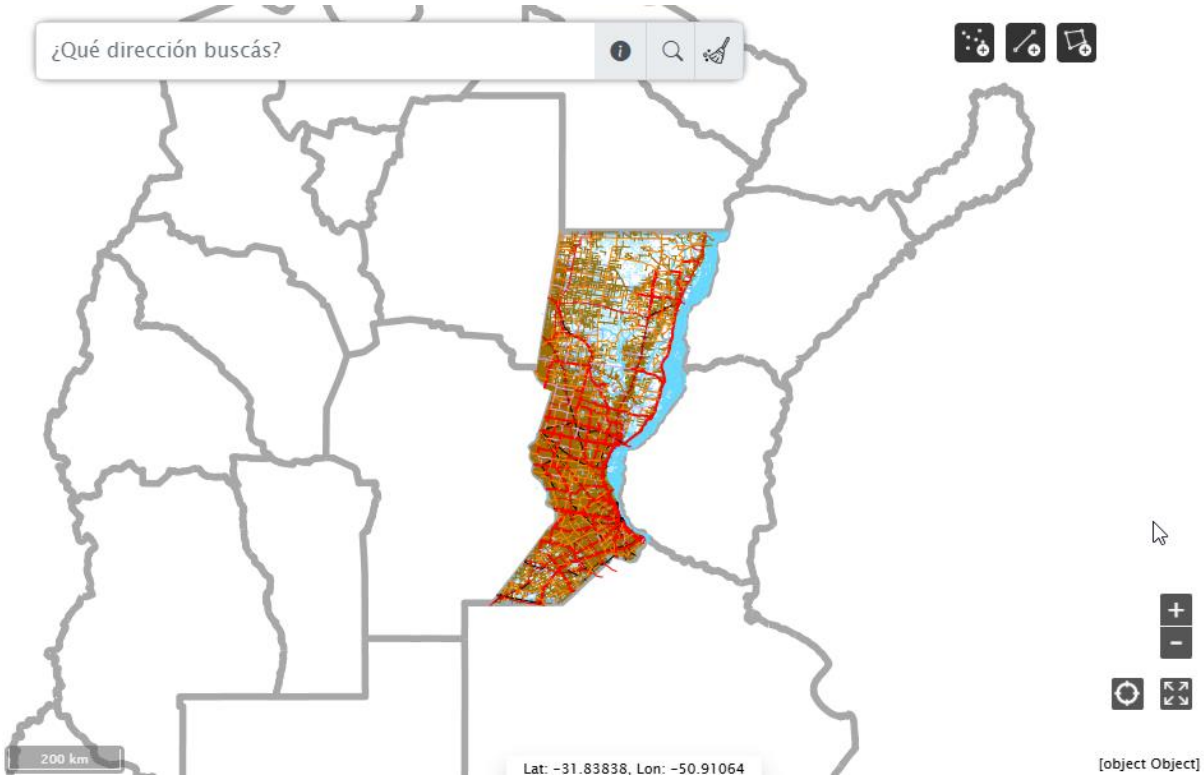
3.5.- Cambiar las dimensiones del módulo

Para modificar el tamaño del módulo, se debe modificar el valor de las siguientes líneas presentes en el script:

```
onReady: function(){
            this.container.getElementsByTagName("iframe")[0].style.height =
(window.innerHeight-100)+"px";
            this.container.getElementsByTagName("iframe")[0].style.width =
window.innerWidth+"px";
            this.container.getElementsByTagName("iframe")[0].scrolling = "no";
            this.container.getElementsByTagName("iframe")[0].frameborder = "0";
            },x";
```

3.6.- Uso del módulo

Una vez inicializado el módulo, se debería ver algo similar a lo siguiente:



A cada mensaje enviado utilizando el socket, el cliente que tiene embebido el módulo, recibirá un mensaje en formato String representando un objeto *JSON,* con la información solicitada o el resultado de la operación. Este objeto podrá ser procesado en la porción del

cuerpo del script bajo la etiqueta **onMessage**:

NOTA: Tener en cuenta que los mensajes son siempre asíncronos.

```
onMessage:function(messageFuente, origin)
            {
          //TODOS LOS MENSAJES SON ASINCRONOS Y DEBEN MANEJARSE DENTRO DE ESTE CONTEXTO
          // ENVIAR MENSAJES Y REALIZAR TAREAS SOLO LUEGO DE LA RECEPCION DE LOS MISMOS
          // PARA EVITAR PERDIDA DE INFORMACION
```

3.7.- Invocación de métodos del IFRAME

Para invocar cualquier método disponible en el widget embebido se debe utilizar el socket generado al momento de la instanciación del módulo.

```
socket.postMessage({"message":"<unMensaje>"
[,"capa":"idesf:scit_parcelas_query","parametros":" < params >"]});
```

El campo **message** es obligatorio. Los demás campos son opcionales y varían según el método utilizado.

3.8.- Dibujar

3.8.1 – iniciar Dibujo:

```
var mensaje = {"message":"iniciarDibujo", "tipo":[ punto | linea | poligono | circulo |]};
socket.postMessage(JSON.stringify(mensaje));
```

3.8.2 – Detener Dibujo:

```
var mensaje ={"message":"detenerDibujo"};
socket.postMessage(JSON.stringify(mensaje));
```

3.8.3 – Borrar Dibujo:

```
var mensaje = { "message":"borrarDibujo"};
socket.postMessage(JSON.stringify(mensaje));
```

3.8.4 – Ajustar Vista:

```
var mensaje = { "message":"ajustarVista"};
socket.postMessage(JSON.stringify(mensaje));
```

3.8.5 – Iniciar Selección:

```
var mensaje = { "message":"iniciarSeleccion"};
socket.postMessage(JSON.stringify(mensaje));
```

3.8.6 – Detener Selección:

```
var mensaje = { "message":"detenerSeleccion"};
socket.postMessage(JSON.stringify(mensaje));
```

3.8.6 – Detener Selección:

```
var mensaje = { "message":"detenerSeleccion"};
socket.postMessage(JSON.stringify(mensaje));
```

3.9.- Exportar en diferentes Formatos

3.9.1 – Exportar selección en GEOJSON:

```
var mensaje = { "message":"getSeleccionGEOJSON"};
socket.postMessage(JSON.stringify(mensaje));
```

3.9.2 – Exportar selección en WKT:

```
var mensaje = { "message":"getSeleccionWKT"};
socket.postMessage(JSON.stringify(mensaje));
```

3.9.3 – Exportar selección en WKT EPSG:5347:

```
var mensaje = { "message":"getSeleccionWKT", "epsg":"5347"};
socket.postMessage(JSON.stringify(mensaje));
```

3.9.4 – Exportar todo en GEOJSON:

```
var mensaje = { "message":"getGEOJSON"};
socket.postMessage(JSON.stringify(mensaje));
```

3.9.5 – Exportar todo en WKT:

```
var mensaje = { "message":"getWKT"};
socket.postMessage(JSON.stringify(mensaje));
```

3.9.6 – Exportar todo en WKT EPSG:5347:

```
var mensaje = { "message":"getWKT","epsg":"5347"};
socket.postMessage(JSON.stringify(mensaje));
```

3.10.– Métodos Adicionales

3.10.1 – Recuperar Área de la selección

```
var mensaje = { "message":"getAreaSeleccion"};
socket.postMessage(JSON.stringify(mensaje));
```

3.10.2 – Get listado Localidades NUC

```
var mensaje = {"message":"getLocalidades"};
socket.postMessage(JSON.stringify(mensaje));
```

3.10.2.1 – Get calles de una Localidades NUC

```
var mensaje =  {"message":"getCallesLocalidad","codigoLocalidad":<CODIGO
                  DE LA LOCALIDAD DEL METODO ANTERIOR>};

socket.postMessage(JSON.stringify(mensaje));
```

3.10.2 – Marcar calle en el mapa

```
var mensaje =
{"message":"marcarCalle","codigoLocalidad":CODIGO_LOCALIDAD,"codigoVia":CODIGO_V
IA,"limpiar":[TRUE | FALSE]};

socket.postMessage(JSON.stringify(mensaje));
```

3.10.3 – Mostrar Selector de capas

```
var mostrarWKT = { "message":"activarSelectorCapas", "estado": true };
socket.postMessage(JSON.stringify(mostrarWKT));
```

3.10.4 – Ocultar Selector de capas

```
var mostrarWKT = { "message":"activarSelectorCapas", "estado": false };
socket.postMessage(JSON.stringify(mostrarWKT));
```

3.10.5 – Ocultar Selector de capas

```
var mostrarWKT = { "message":"activarSelectorCapas", "estado": false };
socket.postMessage(JSON.stringify(mostrarWKT));
```

3.10.6 – Cargar parcelas catastrales vectoriales.

```
var nomenclatura = "NOMENCLATURA:1011PA000000022~1011PA000000023";
```

```
socket.postMessage('{ "message":"loadPoligonos",
"capa":"idesf:scit_parcelas_query","parametros": "'+
nomenclatura+'","nombreCapaFantasia":"dibujo"}');
```

La variable `NOMENCLATURA` soporta el ingreso de uno o múltiples identificadores de parcelas separados por el carácter **~** (tilde o virgulilla)

Una parcela: `var nomenclatura = "NOMENCLATURA:1011PA000000022";`

Varias parcelas: `var nomenclatura = "NOMENCLATURA:1011PA000000022~1011PA000000023~1011PA000000024";`

3.11 – Múltiples instancias del modulo de dibujo

Inicializar el buscador de la siguiente manera:

```
comunicacion = {
      easyXDM:  window.easyXDM.noConflict("comunicacion")
                };
 //Definir variable de no-conflict para referirse a la
 //instancia del módulo

var socket = new comunicacion.easyXDM.Socket({
      .
      .
      .
      . });
```

3.12 – Exportar imagen PNG

Retorna el estado actual de los popups:

```
socket.postMessage({message:'exporta2Png'});
```

Genera una imagen en Base64, y la retorna en un mensaje, en el ejemplo queda almacenada en una variable global llamada **imagen**.

3.13 – Agregar capas WMS

```
var cargarCapas = { "message":"agregarCapasGeograficas", "capas":{
                  "listado":[
                  {
```

```
                    "nombreCapaFantasia":"Distritos",
                    "nombreCapa":"distritos",
                    "servidor":"https://aswe.santafe.gov.ar/idesf/geoserver/wms",
                    "visible": false,
                     "opacidad":1
                            },
                            {
                     "nombreCapaFantasia":"Departamentos",
                      "nombreCapa":"departamentos",
                      "servidor":"https://aswe.santafe.gov.ar/idesf/geoserver/wms",
                      "visible": false,
                       "opacidad":1
                              }
                     ]}
          };
socket.postMessage(JSON.stringify(cargarCapas));
```

3.14 – Cargar un Poligono WKT

```
var mostrarWKT = { "message":"muestraPoligonosWKT", "WKT": < CADENA WKT> };

socket.postMessage(JSON.stringify(mostrarWKT));
```

3.15 – MUY IMPORTANTE - *Recomendaciones de desarrollo*

Inicializar el modulo siempre dentro de una función:

```
$( document ).ready(function() {
      <llamada al script>
});
```