Artículo
Yone Moreno · Dic 6, 2019 Lectura de 7 min

Haciendo una integración REST: Geolocalización

¡ Hola a tod@s!

comparto los pasos necesarios para crear y probar una integración REST.

Lo primero que necesitamos es acceder al contrato que nos aporta el equipo que se encarga de desarrollar el sistema externo. En concreto de esta forma sabremos varias cosas:

1 A qué URL apuntar, con qué método y qué parámetros incluir:

3.4 Geolocalización de centros de salud y hospitales

ORIGEN	• [redacted]
DESTINO	• [redacted]
RUTA	• /geolo[redacted]
SEGURIDAD	• [redacted]
LOG	• [redacted]

3.4.1 Listar centros

RECURSO	• /geolo[redacted]
METHOD	• GET
PARAMETROS	• String cod_cias • String tipo (Tipo de centro) • String isla

2 Los resultados esperados:

RESULTADO	• null → Acción Incorrecta • listado de centros o vacío


```
[
  {
  "cod_cias": "12025222 ",
  "nombre_centro": "C.P. LA RESTINGA",
  "direccion": "C/ AVENIDA MARITIMA S/N (LA RESTINGA) ",
  "localidad": "EL PINAR",
  "telefono": "[redacted]",
  "longitud": "-17.98384130001068",
  "latitud": "-17.98384130001068",
  "tipo_centro":"CP",
  "punto_urgencia":"N"
 },
 {
```


A continuación, lo que haremos será probar la URL de forma directa, para comprobar que en efecto, con los parámetros que nos dicen, accediendo a la URL dada, obtenemos lo que esperamos.

En nuestro caso accederemos a un sistema externo que nos devolverá la lista de centros, con su geolocalización, expresada como longitud y latitud. En caso de que pongamos sin parámetros, se nos devuelven todos los centros.

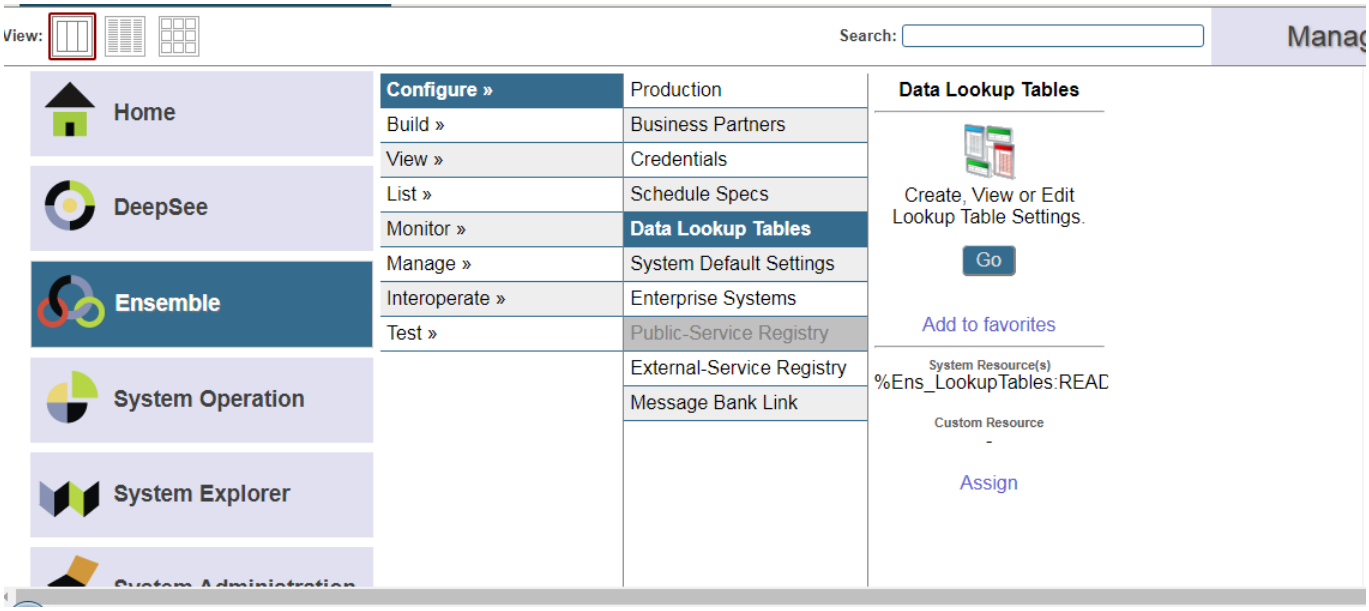
Si ponemos isla=HIE nos saldrán los centros del Hierro, con tipo=CS los centros de salud y con cod_cias=Numero nos saldría la infromación relativa a un centro con un código determinado.

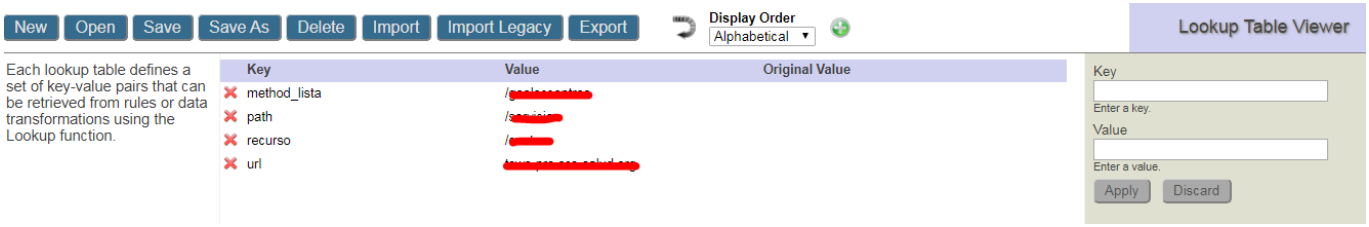
Después vamos a crearnos una Operación. Construiremos la URL así:

```
Method listarCentros(pRequest As Mensajes.Request.miSCS.GestionFavoritos.ListarCentrosRequest, Output pResponse As Mensajes.Response.miSCS.GestionFavoritos.TSI.ListarCentrosResponse) As %Status

{

set pResponse = ##class(Mensajes.Response.miSCS.GestionFavoritos.TSI.ListarCentrosResponse).%New()

try {

//preparamos la url para la petición
```

```
set url     = ##class(Util.TablasMaestras).getValorMaestra("GEOLOCALIZACION","url")

set path    = ##class(Util.TablasMaestras).getValorMaestra("GEOLOCALIZACION","path")

set servicio = ##class(Util.TablasMaestras).getValorMaestra("GEOLOCALIZACION","method_lista")

set recurso = ##class(Util.TablasMaestras).getValorMaestra("GEOLOCALIZACION","recurso")


set URL = "http://"_url_path_servicio_recurso
```

Donde estamos accediendo a la tabla GEOLOCALIZACION y obteniendo cada fragmento de la URL. Para que funcione, vamos a crearnos la tabla. Vamos a Ensemble > Configure > Data Lookup Tables.





Además queremos contemplar los casos, en los que queramos filtrar usando varios parámetros. Para eso en el código tenemos en cuenta cuál está definido y es el primero, entonces se precede con ?. Si no, se precede con &.

```
if pRequest.isla '= ""{

if $find(URL,"?") < 2{

set URL = URL_"?"_"isla="_pRequest.isla

}else{

set URL = URL_"&"_"isla="_pRequest.isla

}

}

if pRequest.tipo '= ""{

if $find(URL,"?") < 2{

set URL = URL_"?"_"tipo="_pRequest.tipo

}else{
```

```
set URL = URL_"&"_"tipo="_pRequest.tipo

}

}

if pRequest.codCias '= ""{

$$$LOGINFO("codCias: "_pRequest.codCias)

if $find(URL,"?") < 2{

set URL = URL_"?"_"cod_cias="_pRequest.codCias

}else{

set URL = URL_"&"_"cod_cias="_pRequest.codCias

}

}
```

Por tanto si nos viniera isla=HIE y tipo=CS, la URL se formaría como:

http://url?isla=HIE&tipo=CS

Luego:

```
        set tSC = ..Adapter.GetURL(URL,.res)

if $$$ISERR(tSC)
```

```
{

set textoError = $System.Status.GetErrorText(tSC)

    set error = ##class(EsquemasDatos.Seguridad.Error).%New()

set error.codigo = "-1"

set error.descripcion = textoError

set pResponse.error = error

} else {

$$$LOGINFO("RECIBIMOS RESPUESTA: "_res)

set linea = ""

While (res.Data.AtEnd = 0) {

    set linea = linea_res.Data.Read()

    $$$LOGINFO("LINEA: "_linea)


    }
```

Enviamos el GET, la petición, a la URL formada, en caso de error lo mostramos y si va bien, leemos la respuesta. Lo que ocurre es que en este momento el mensaje que recibimos, lo que imprimimos en línea, sería un JSON donde todo está pegado:

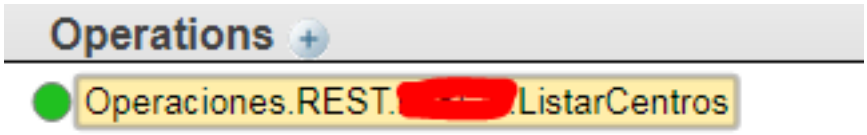
Lo que nos interesa es crear un mensaje legible, en XML. Para ello usamos:

```
// se transforma el objeto JSON a un objeto local ★

set claseAux = ##class(%ZEN.Auxiliary.jsonProvider).%New()

set tSC= claseAux.%ConvertJSONToObject(.linea,
"EsquemasDatos.miSCS.GestionFavoritos.tns.infoCentro",.objeto,1)



set i = 0, pResponse.numCentro = objeto.Size

While (objeto.Size > i) {

set i = i + 1

set centro = ##class(EsquemasDatos.miSCS.GestionFavoritos.tns.infoCentro).%New()

set centro.codCias = objeto.GetAt(i)."cod_cias"

set centro.nombreCentro = objeto.GetAt(i)."nombre_centro"

set centro.direccion = objeto.GetAt(i).direccion

set centro.localidad = objeto.GetAt(i).localidad

set centro.telefono = objeto.GetAt(i).telefono

set centro.longitud = objeto.GetAt(i).longitud

set centro.latitud = objeto.GetAt(i).latitud

set centro.tipoCentro = objeto.GetAt(i)."tipo_centro"

set centro.puntoUrgencia = objeto.GetAt(i)."punto_urgencia"
```

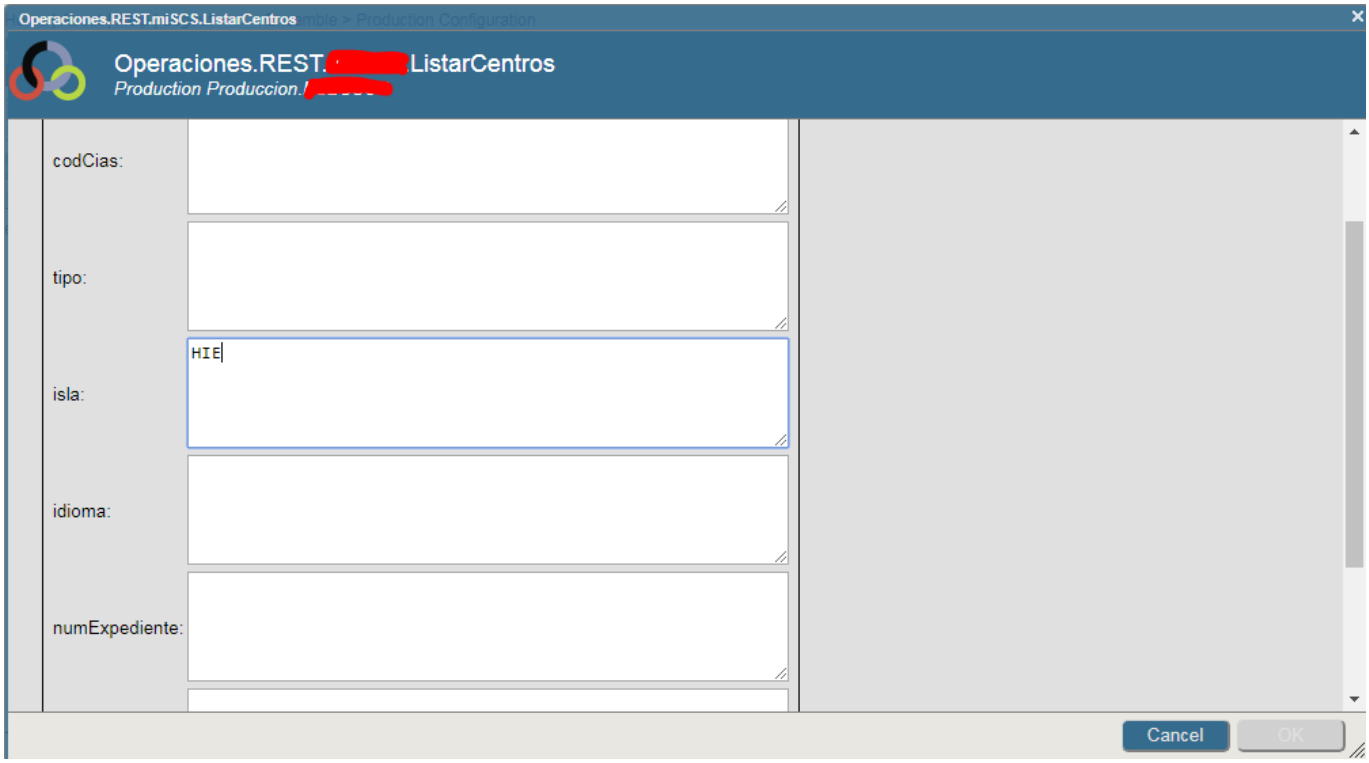
```
do pResponse.datos.Insert(centro)

}

}
```

De forma que el mensaje quedaría:

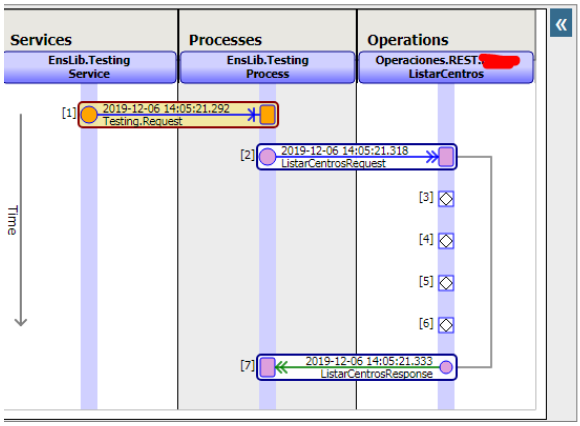
Ahora, si queremos probar lo escrito, necesitamos desde la Producción, subir la Operación que hemos terminado:

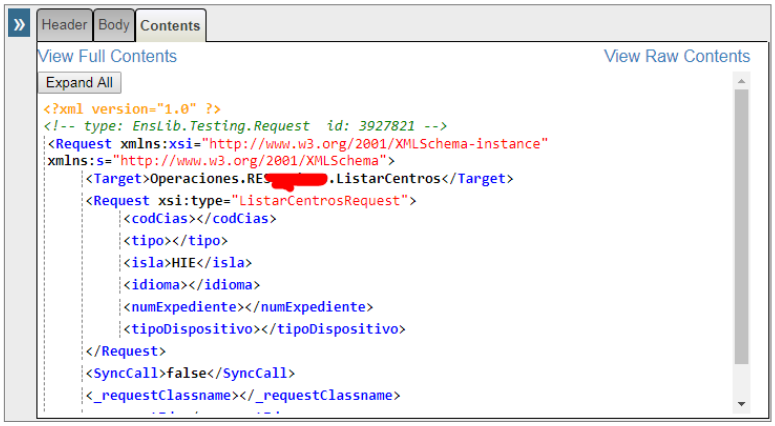


Después podemos pulsar en la Operación > Actions > Test y rellenamos los parámetros del mensaje:



De forma que veremos el circuito y el mensaje, en el Message Viewer:





A continuación, crearemos el Servicio que aceptará peticiones REST y las enviará al sistema externo que acabamos de probar.

Un Servicio REST se define como una clase que hereda de: Ens.BusinessService, tiene la propiedad: Adapter: "EnsLib.HTTP.InboundAdapter" e implementa el Método: OnProcessInput

Con la primera parte del código le diremos que las peticiones deben incluir el prefijo /geolocalizacion para que sean atendidas por este servicio.

```
Class Servicios.Seguridad.REST.miSCS.Geolocalizacion
Extends (Ens.BusinessService, Ens.Util.FunctionSet)

{

Parameter ADAPTER = "EnsLib.HTTP.InboundAdapter";

Parameter EnsServicePrefix = "/geolocalizacion";
```

Luego recibimos un objeto, del cual obtenemos su Header request, la pasamos de JSON a objeto y es ahí donde nos enviarán isla, tipo, codigo_cias; es decir, los parámetros mediante los cuales se filtra. Además enviaremos a un proceso llamado Geolocalizacion que crearemos posteriormente.

```
Method listarCentros(pInput As %Stream.Object, Output pOutput As %Stream.Object) As %Status

{

Set pOutput=##class(%GlobalBinaryStream).%New()

set claseAux = ##class(%ZEN.Auxiliary.jsonProvider).%New()

try{

set req = $tr(pInput.GetAttribute("request"),"")

set tSC= claseAux.%ConvertJSONToObject(.req,
"Mensajes.Request.miSCS.GestionFavoritos.ListarCentrosRequest",.objetoEntrada,1)

if $$$ERROR(tSC) {

set hayErrorInterno = -1
```

```
}

} catch{

set objetoSalida = ##class(
Mensajes.Response.miSCS.GestionFavoritos.ListarCentrosResponse).%New()

set error = ##class(EsquemasDatos.Seguridad.Error).%New()

set error.codigo = "50"

set error.descripcion = ##class(Util.TablasMaestras).getValorMaestra(
"ACCESOS_CIUDADANOS",error.codigo)

set objetoSalida.error = error

set hayError = 1

}

set tSC = ..SendRequestSync("Geolocalizacion",objetoEntrada,.objetoSalida)

set tSC = claseAux.%WriteJSONStreamFromObject(.pOutput,.objetoSalida,,,,"aeloqtuw")

Do:$$$ISOK(tSC) pOutput.SetAttribute("Content-Type","application/json")

do pOutput.SetAttribute("Access-Control-Allow-Origin","*")

do pOutput.SetAttribute("Access-Control-Allow-Credentials","true")

do pOutput.SetAttribute("Access-Control-Allow-Methods","GET")
```

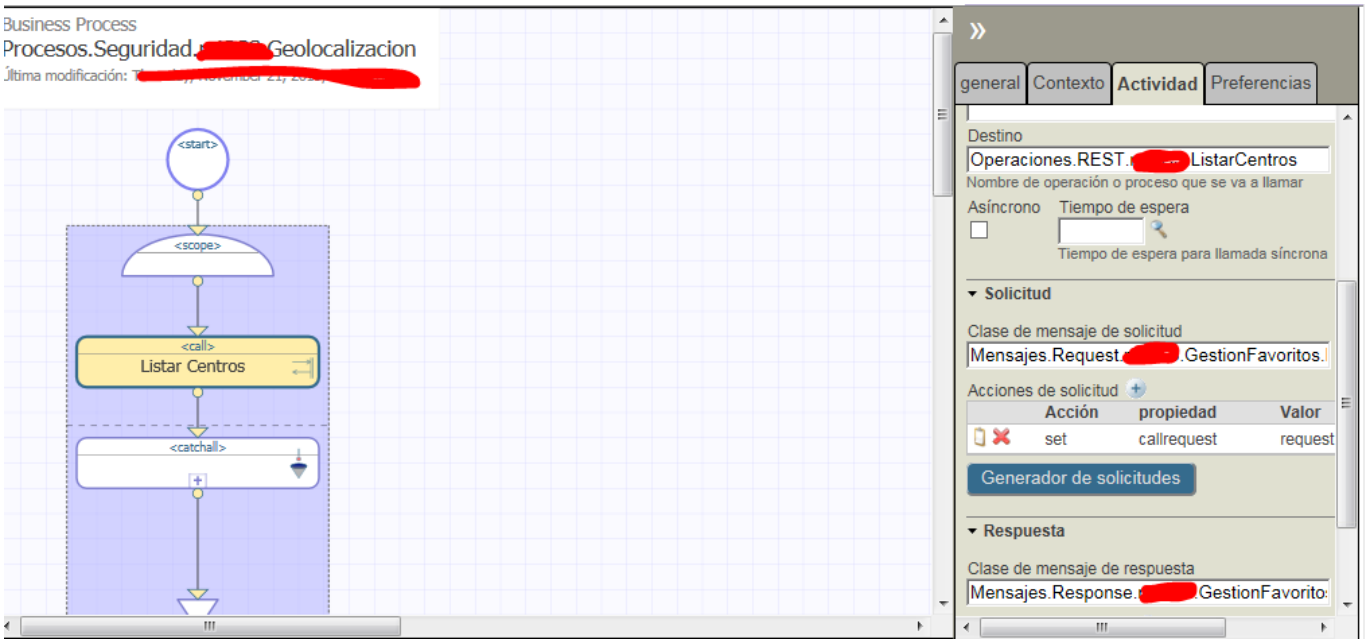
```
    do pOutput.SetAttribute("Access-Control-Allow-Headers","request,Access-Control-Allow-Headers, Origin,Accept, X-Requested-With, Content-Type, Access-Control-Request-Method, Access-Control-Request-Headers")

Quit tSC

}
```

Y el método que obtiene la URL y llama a listarCentros en caso de que se indique en esa URL, es el OnProcessInput.

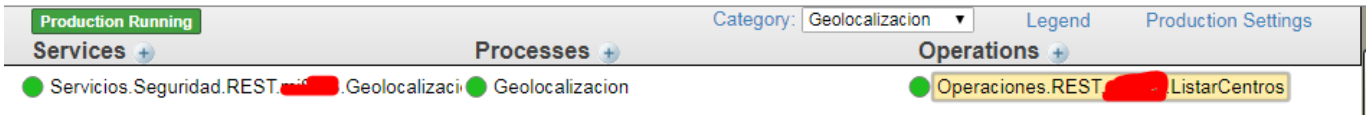
```
Method OnProcessInput(pInput As %Stream.Object, Output pOutput As %Stream.Object) As %Status

{

Set tCmd=$ZConvert(pInput.Attributes("HttpRequest"),"U")

Set tURL =$ZConvert(pInput.Attributes("URL"),"I","URL")

Set tService="/"_$Piece(tURL,"/",5)

Set tVersion="/"_$Piece(tURL,"/",6)

$$$LOGINFO("Servicio: "_tService)

$$$LOGINFO("Version: "_tVersion)

Quit:..#EnsServicePrefix'=tService $$$ERROR($$$EnsErrGeneral,"Service "_tService_"/ not supported.")
```

```
if (%Version = "/listarCentros") {

do ..listarCentros(pInput,.pOutput)

}

Quit $$$OK

}

}
```

Ahora nos queda conectar el servicio y la operación. Para ello vamos a crearnos un Proceso nuevo llamado Geolocalizacion. En él aceptamos el Mensaje Request listarCentros, lo enviamos a la Operación listarCentros, y recibimos su Mensaje Response listarCentros.



Una vez subimos el servicio, el proceso y la operación a la producción:



Vamos a probar el circuito entero. Para ello enviamos desde el POSTMAN peticiones, a la URL definida en el servicio, por ejemplo:

http://IP:PUERTO/x/x/test/geolocalizacion/listarCentros

Y enviando en las Headers el JSON apropiado, obtendremos la respuesta que queremos:

Un saludo

#Studio #Caché

URL de fuente: https://es.community.intersystems.com/post/haciendo-una-integraci%C3%B3n-rest-geolocalizaci%C3%B3n