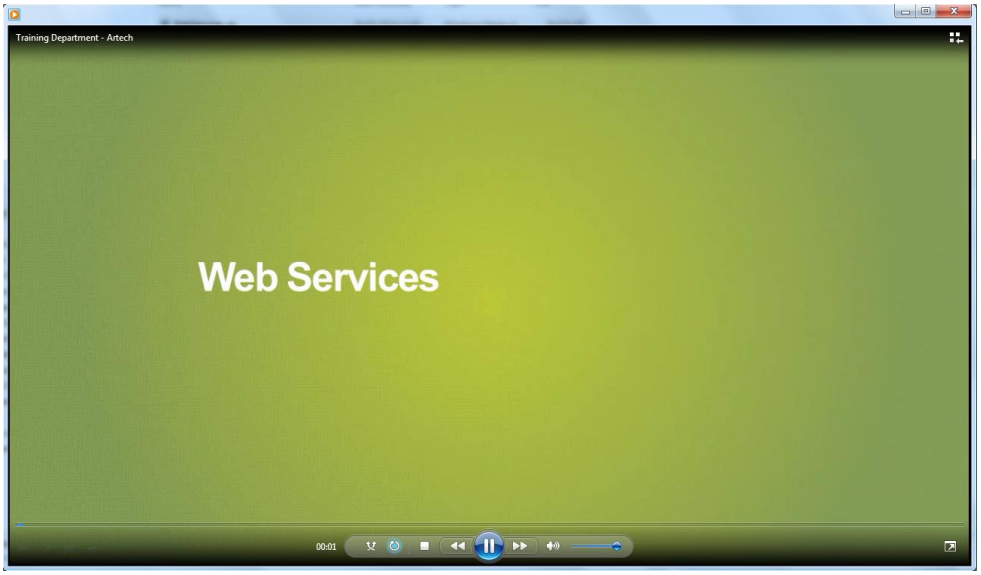
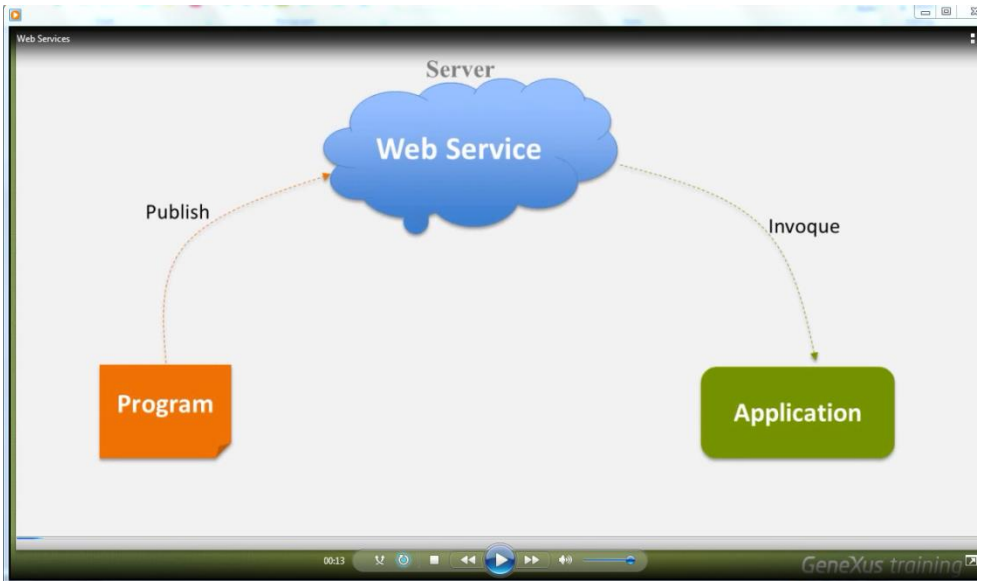
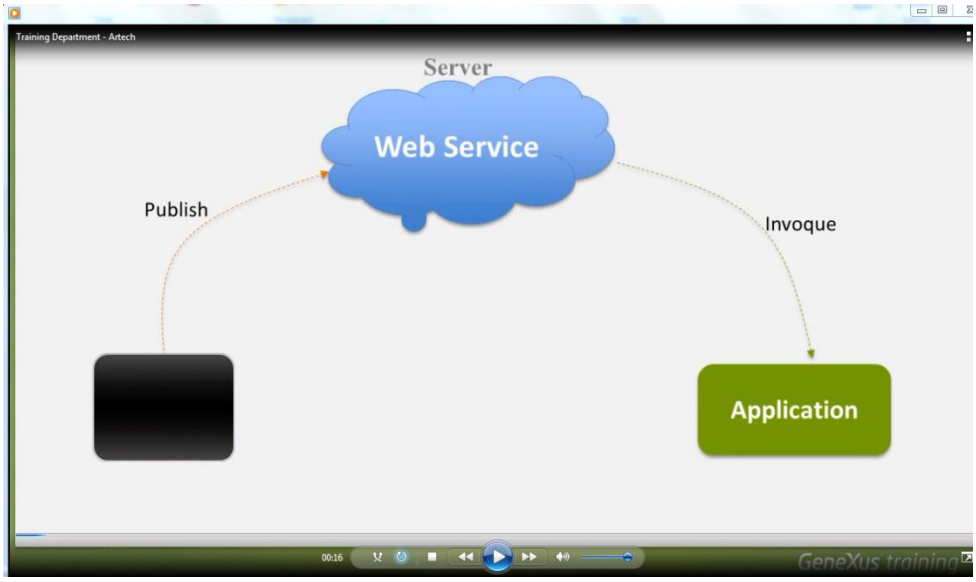
Introducción a Web Services



Los Web Services son programas que brindan funcionalidades útiles y son provistos, es decir publicados en servidores para que puedan ser localizados e invocados a través de una red, generalmente Internet.

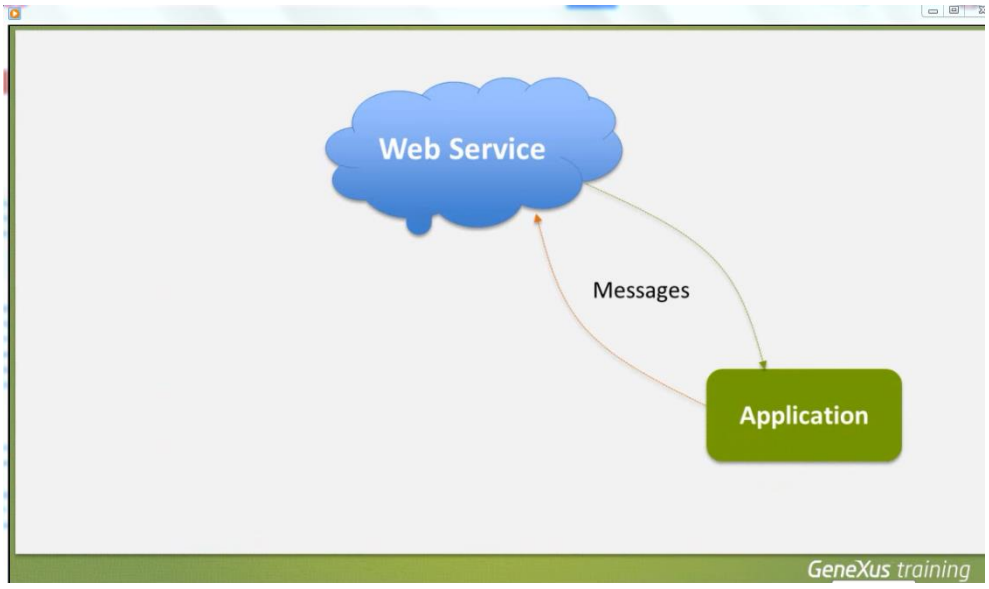


Son cajas negras

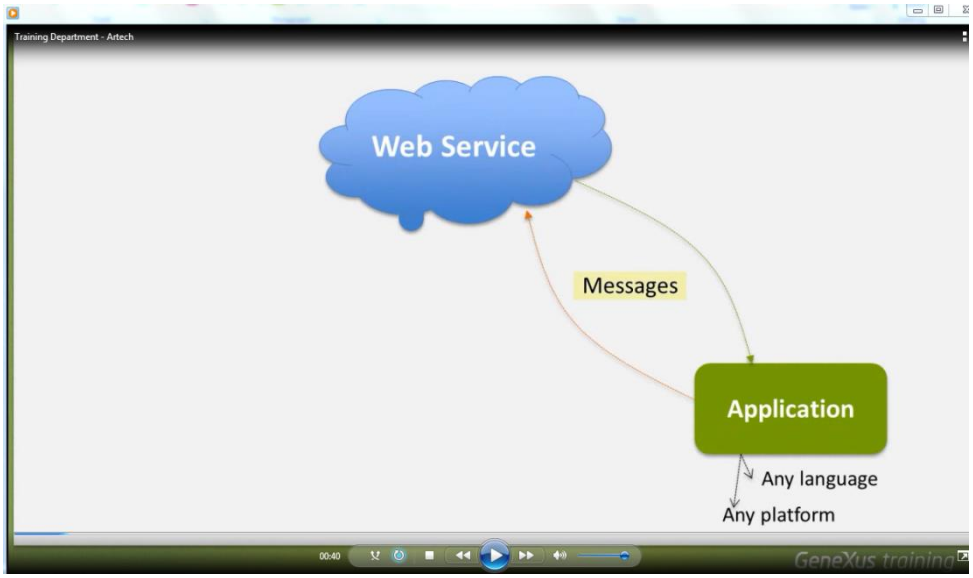


en el sentido que no conocemos su implementación interna ni hay que preocuparse por ello, ya que lo importante es conocer qué funcionalidad brindan, qué parámetros necesitan recibir y qué devuelven.

Los parámetros que participan en la comunicación con Web Services, son denominados “mensajes”

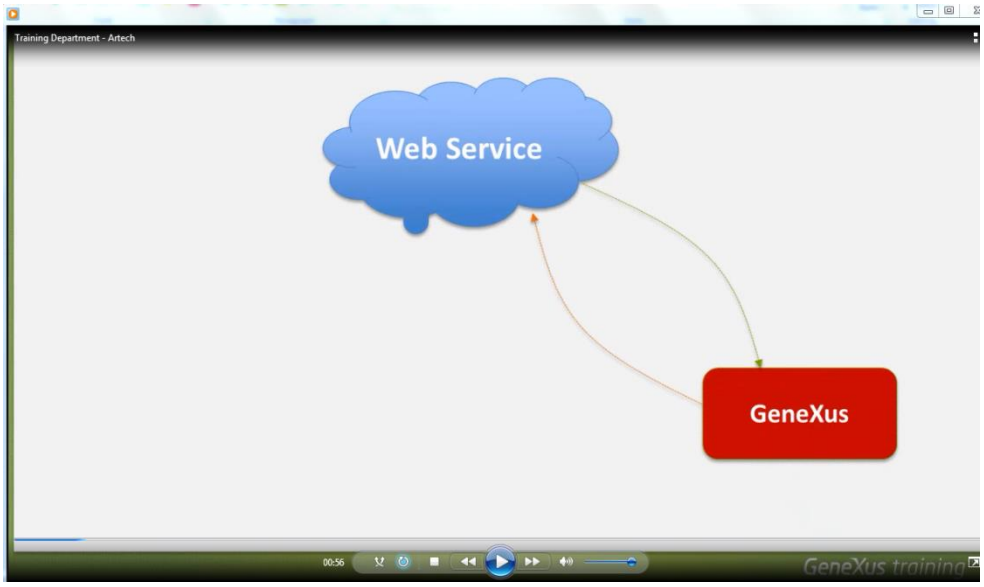


y las aplicaciones que invocan a un Web Service pueden ser desarrolladas en cualquier lenguaje o plataforma.

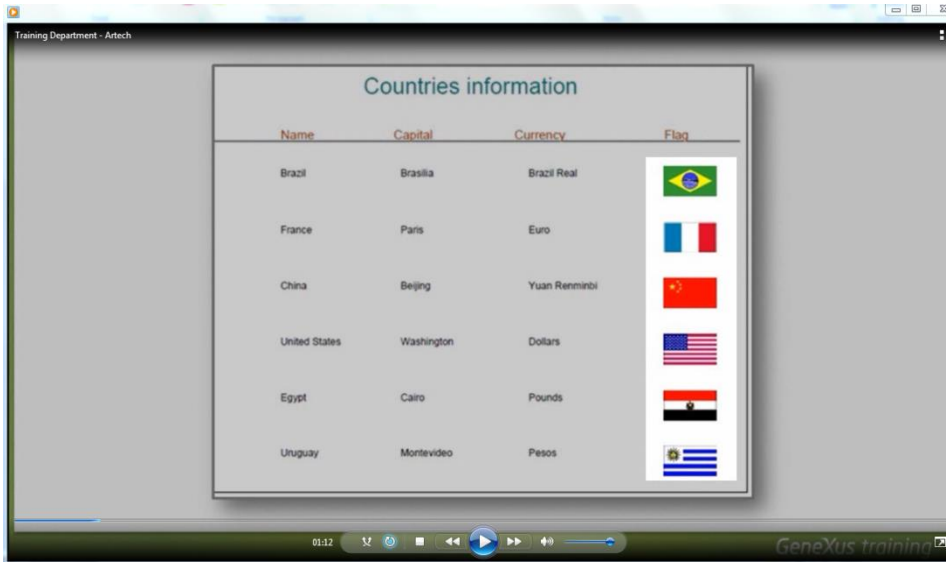


Solamente deben ser capaces de crear y consumir los “mensajes” especificados por el Web Service.

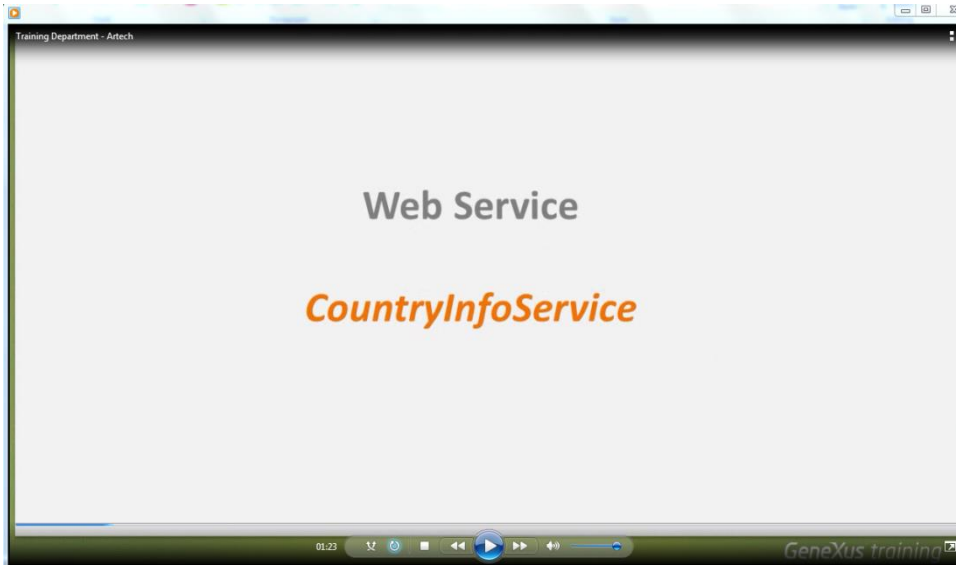
Luego de esta introducción al concepto, vamos a ver cómo invocar, es decir, consumir un web service desde una aplicación GeneXus… y por supuesto que también es posible definir web services con GeneXus y publicarlos.



Bien. Empecemos por suponer que la agencia de viajes nos ha solicitado una nueva funcionalidad: desea poder ofrecer a los clientes, un listado de todos los países, mostrando para cada uno su cuidad capital, su moneda de uso y bandera.

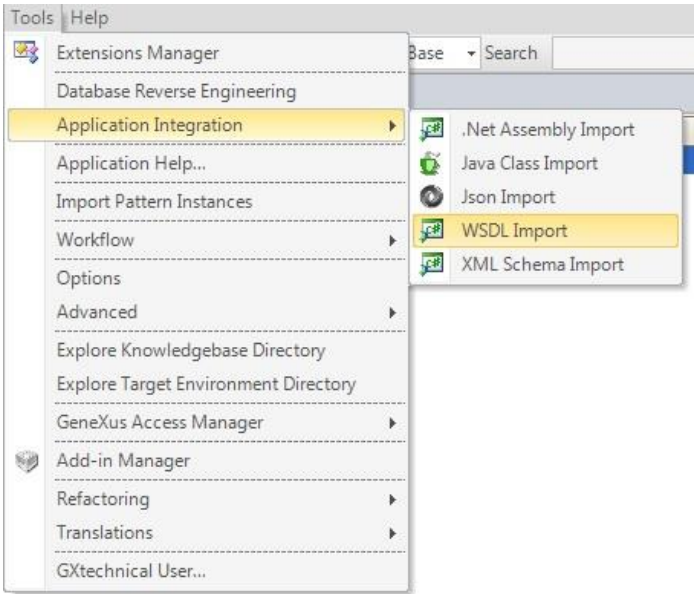


Para resolver esto vamos a invocar a un Web Service específico, que hemos buscado y encontrado publicado. Su nombre es: CountryInfoService

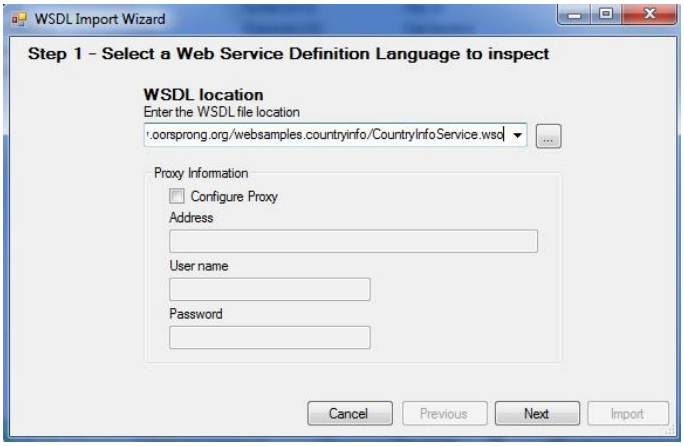


y devuelve dicha información y más.

En GeneXus entonces, en el menú **Tools**, seleccionamos la opción: **Application Integration** y luego **WSDL Import:**

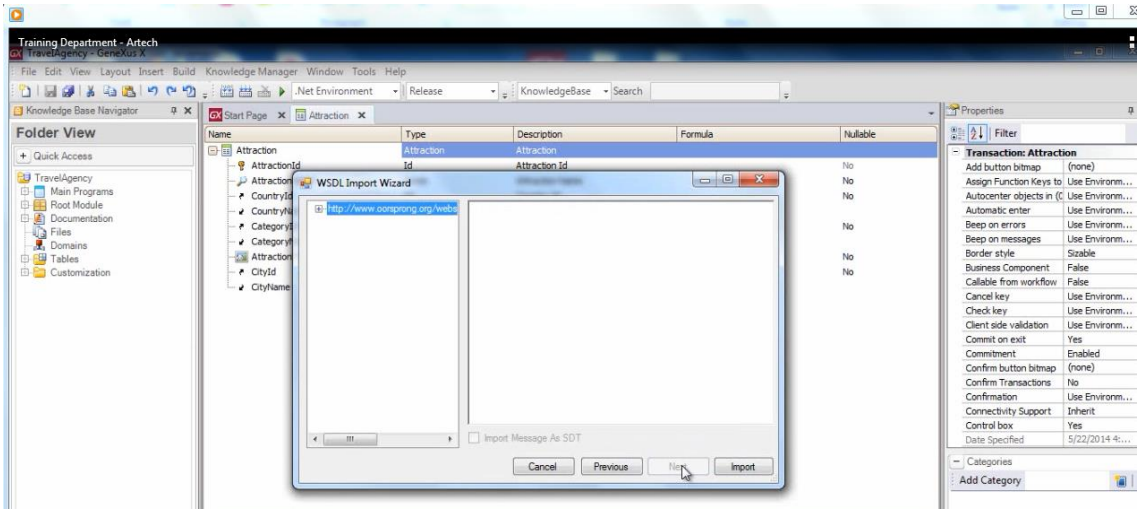


Observemos que se abre un wizard, donde lo primero que debemos indicar es la ubicación del Web Service a consumir.

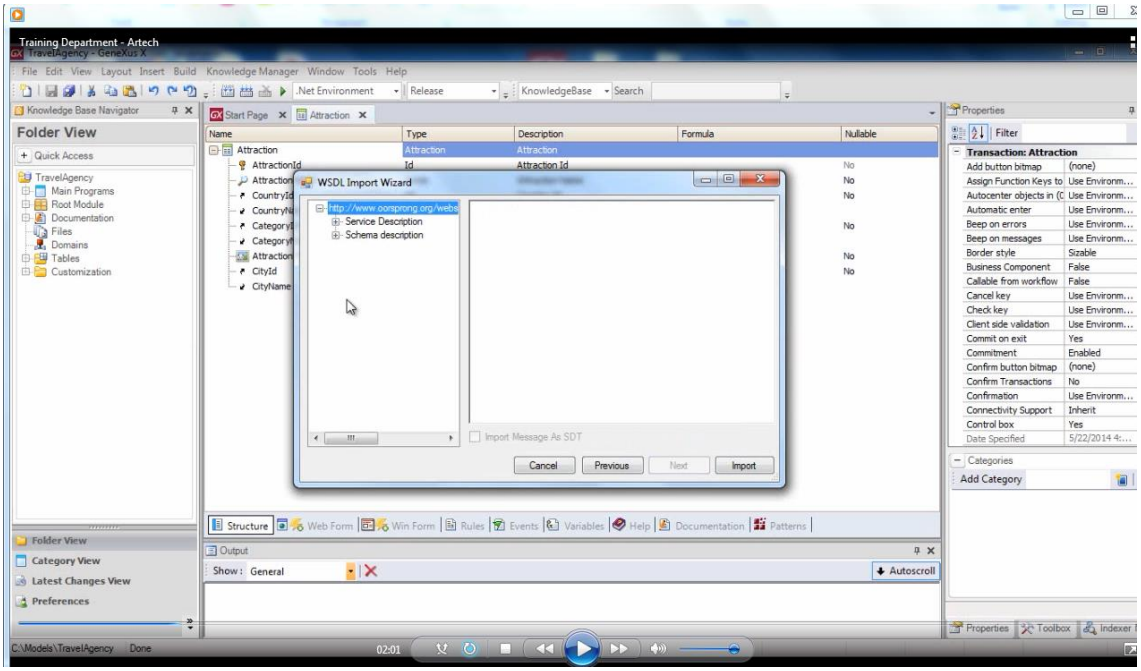


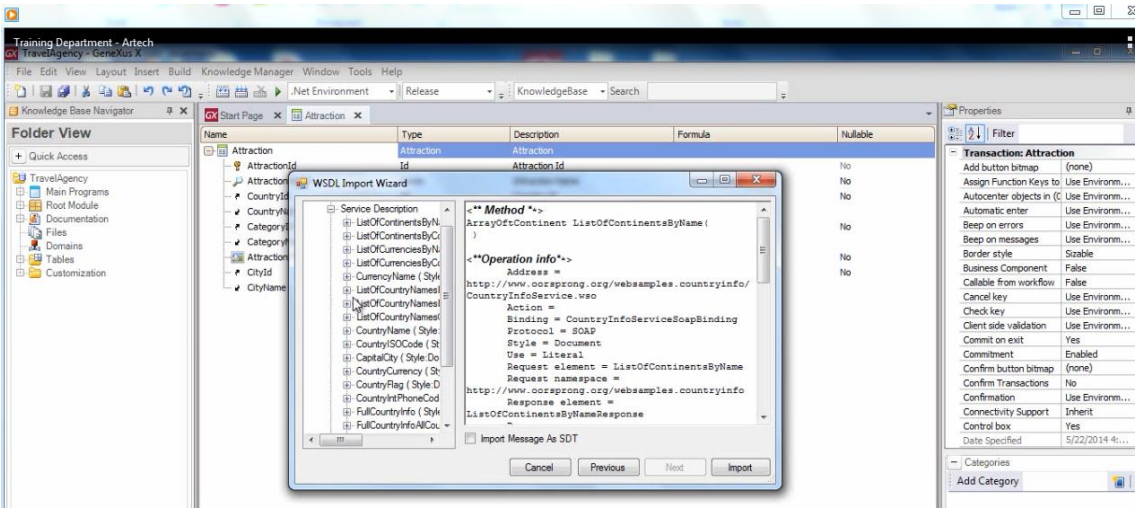
Luego presionamos Next:

Dejamos esta información por defecto… y presionamos nuevamente Next:

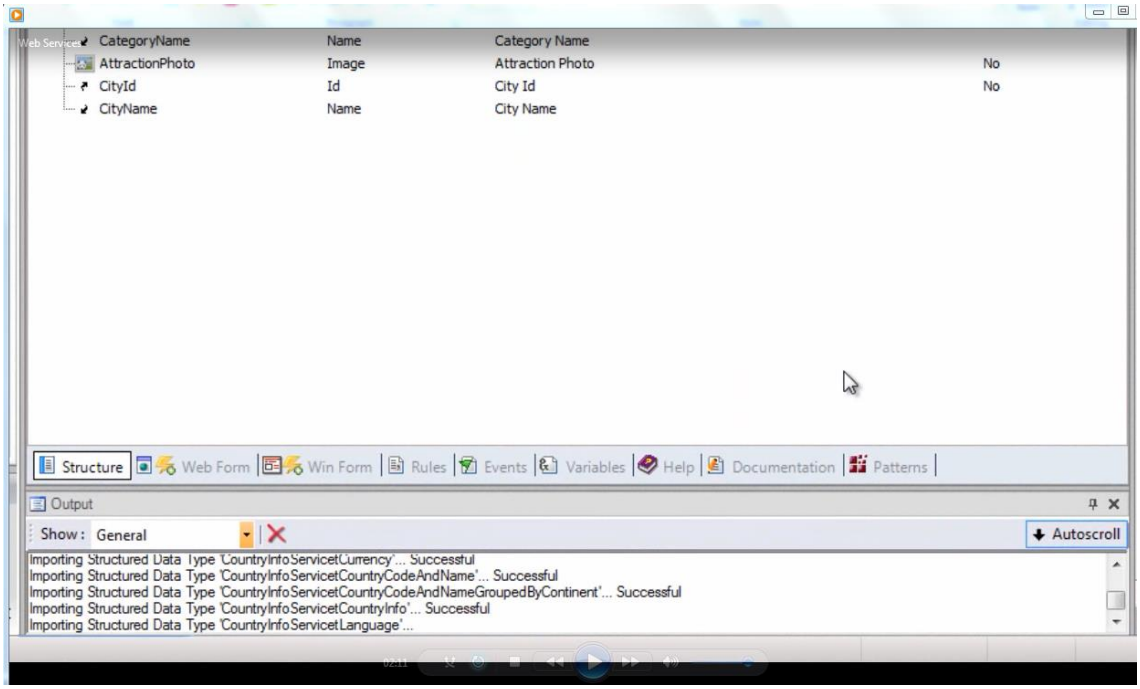


Desde aquí podemos ver la lista de servicios que provee

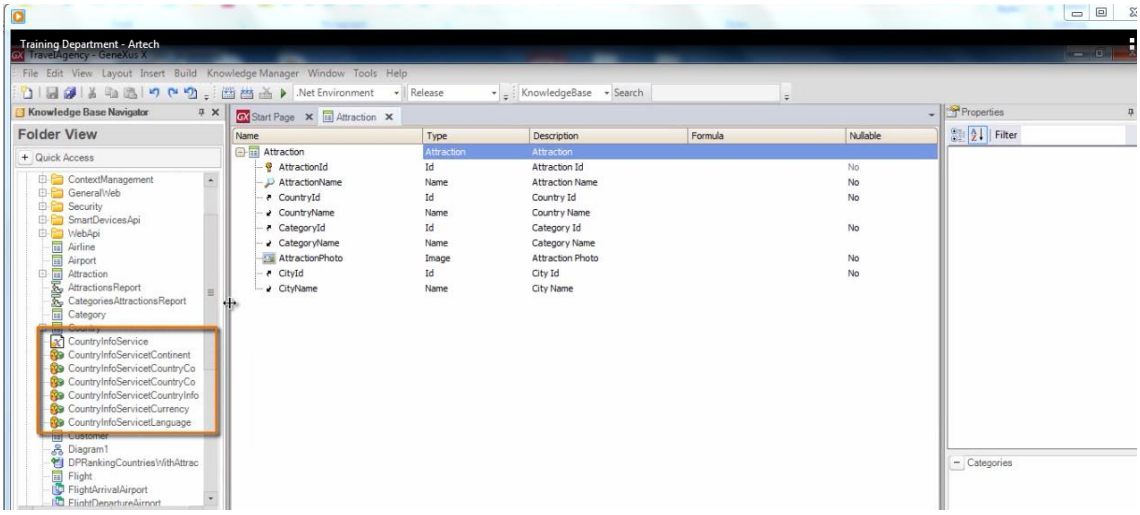




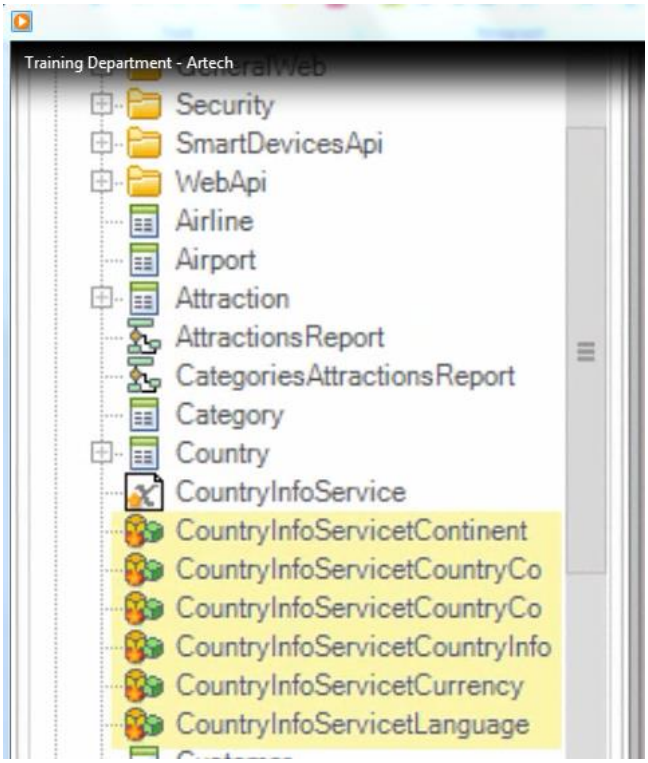
y finalmente presionamos: **Import**



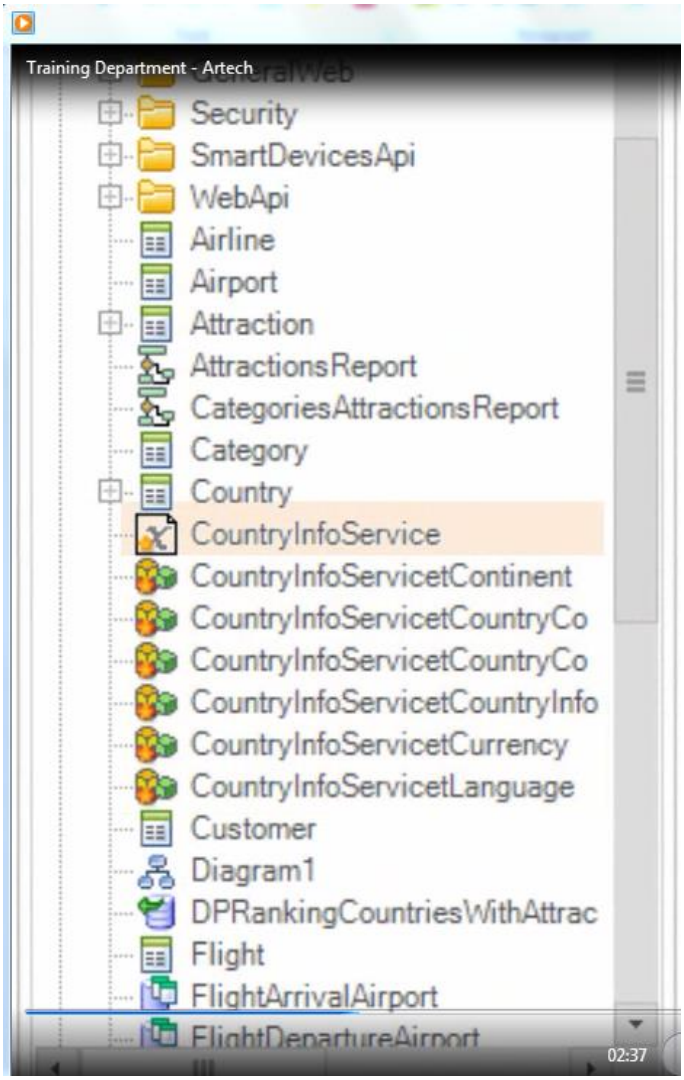
Observemos que ahora en la ventana Folder View, aparecen un conjunto de objetos que GeneXus creó como resultado de la importación:



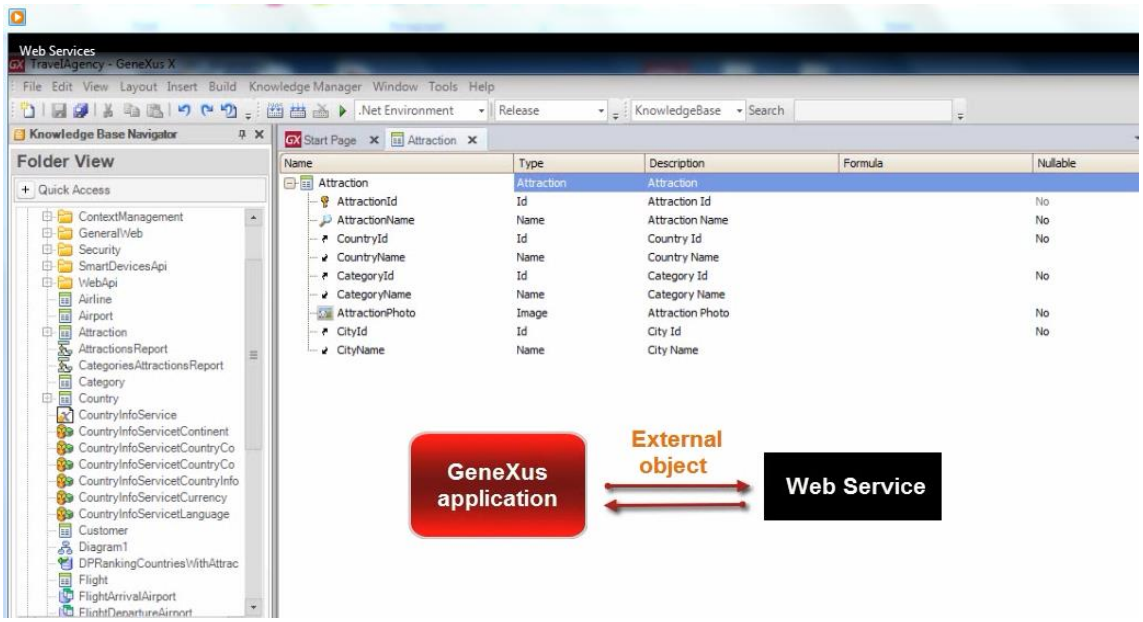
Vemos un conjunto de tipos de datos estructurados



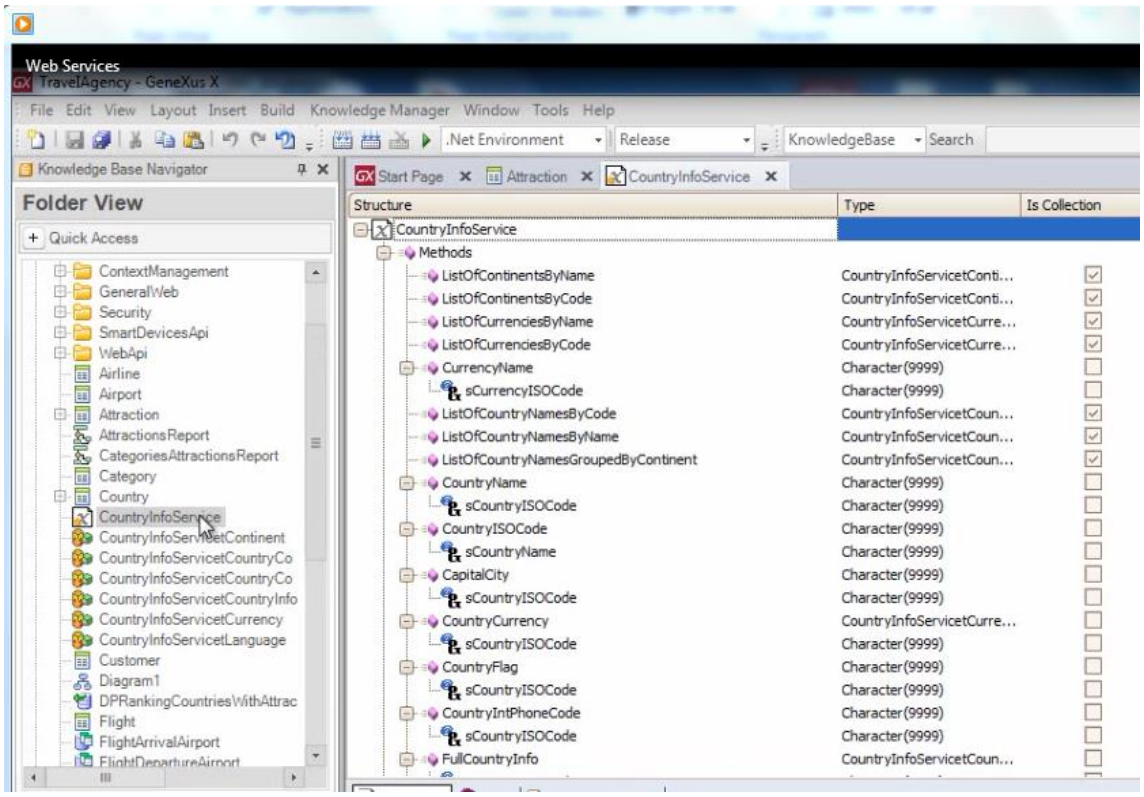
y un External Object con el nombre del web service importado:



Un external object (u Objeto externo) es un tipo de objeto GeneXus que permite la comunicación entre nuestra aplicación y un recurso externo, en este caso con el web service a consumir.



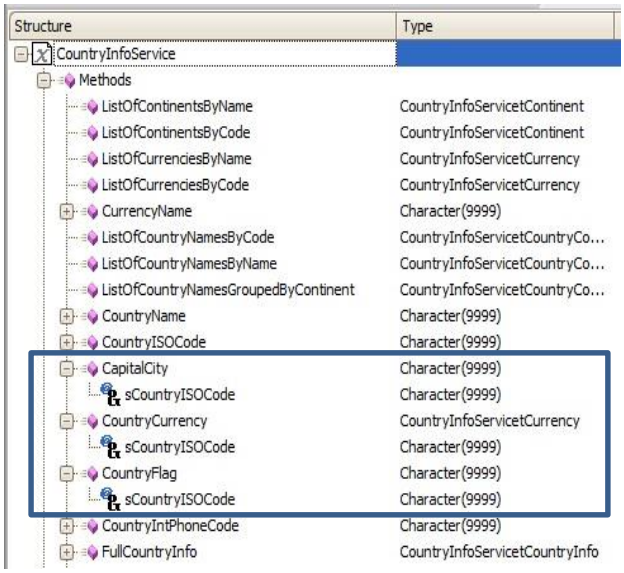
Si hacemos doble click sobre este external object



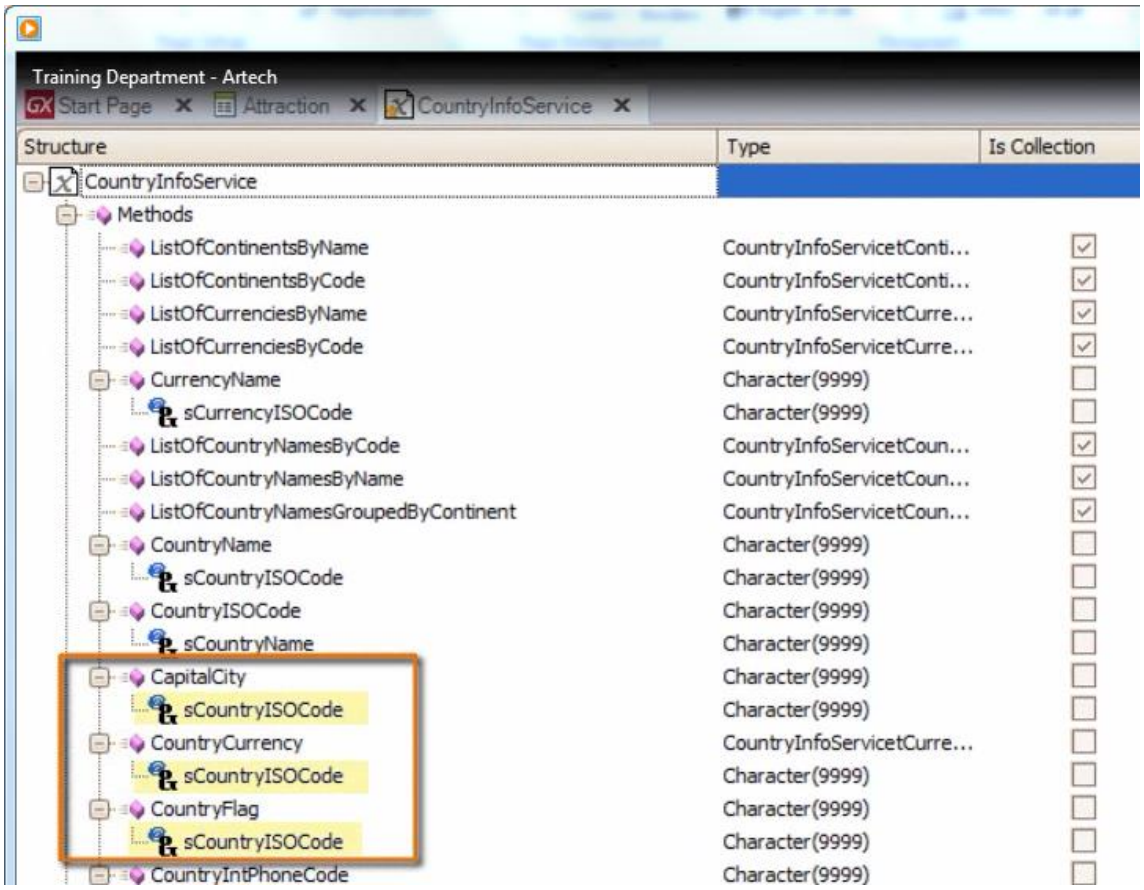
vemos todos los métodos que ofrece el web service, o sea, todas las funcionalidades que ofrece.

En nuestro requerimiento necesitamos obtener la capital, la moneda y la bandera de cada país, así que vamos a utilizar los métodos CapitalCity, CountryCurrency y CountryFlag, respectivamente.

Observemos que los tres métodos que vamos a invocar

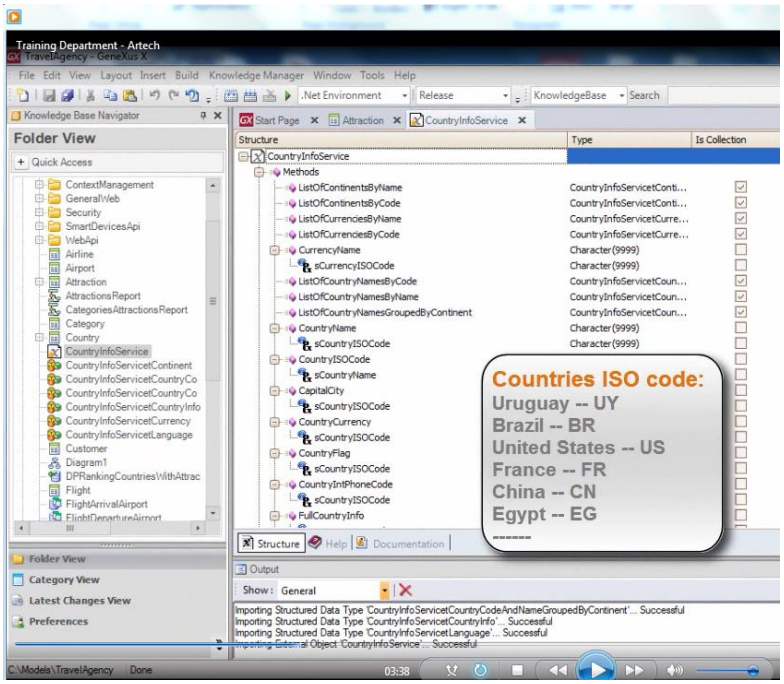


reciben como parámetro el “código ISO del país”

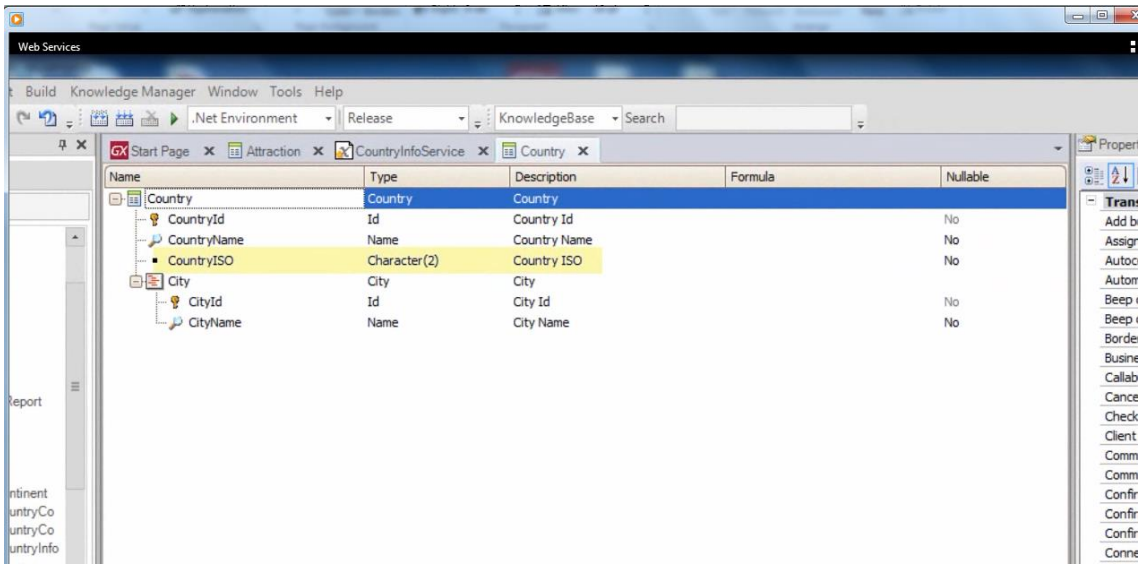


Se trata de un estándar internacional que proporciona códigos para los países, los cuales son utilizados por muchas organizaciones, empresas y gobiernos de todo el mundo.

Por ejemplo, en el caso de Uruguay es “UY”, Brasil “BR”, Estados Unidos “US”, etc.



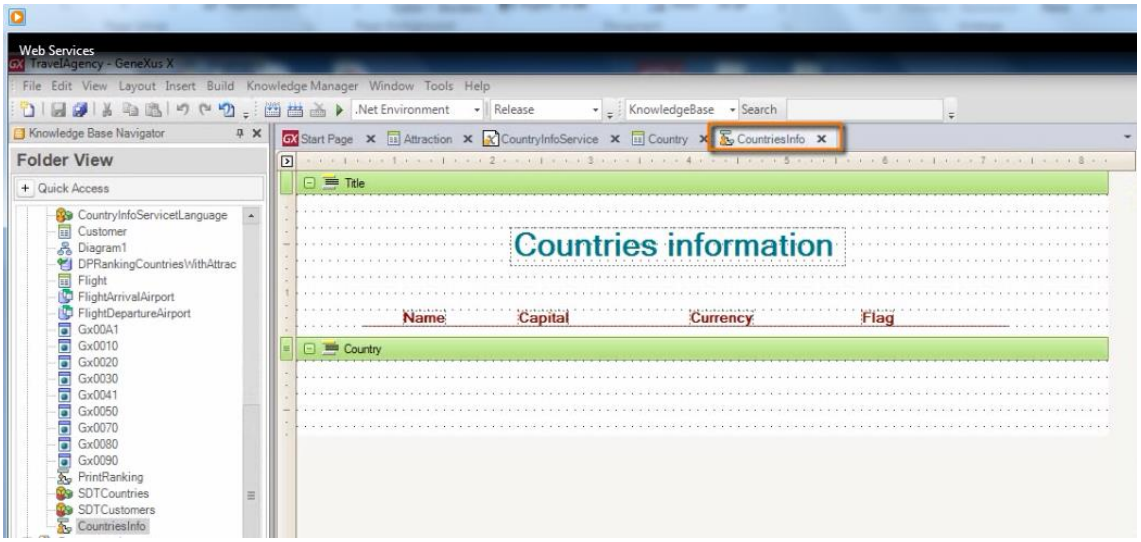
Bien. Vayamos ahora a la transacción Country y observemos que hemos definido un nuevo atributo CountryISO



para almacenar el valor del código ISO correspondiente a cada país.

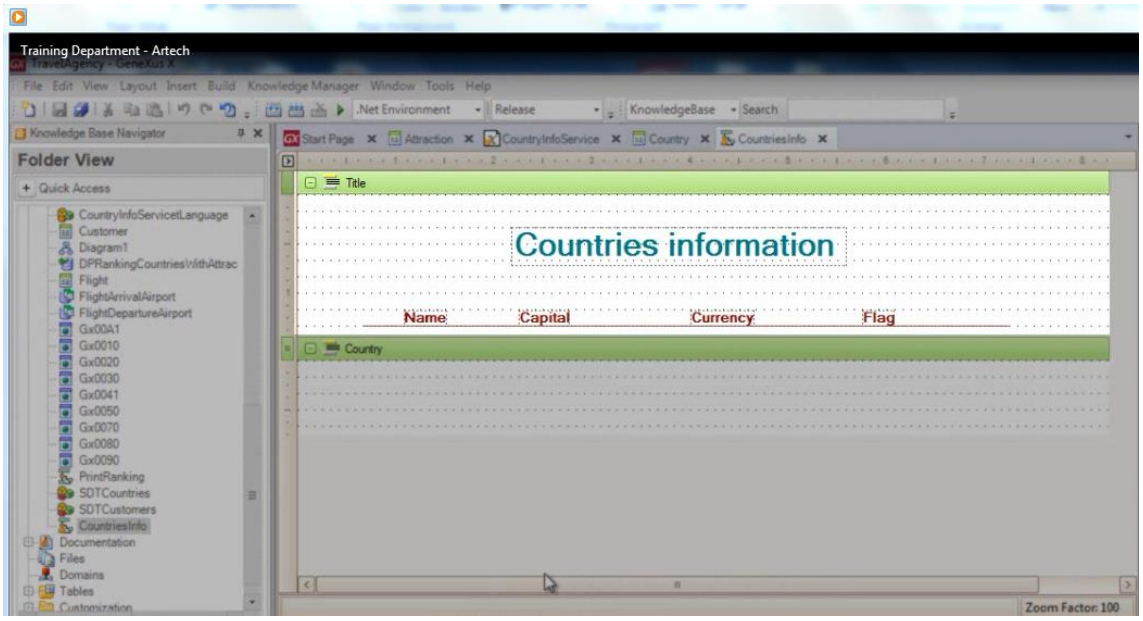
Ya lo hemos resuelto con el fin de ganar tiempo en lo que respecta a la reorganización física de la tabla y la carga de los códigos ISO para cada país.

También ya hemos comenzado a crear un procedimiento de nombre CountriesInfo



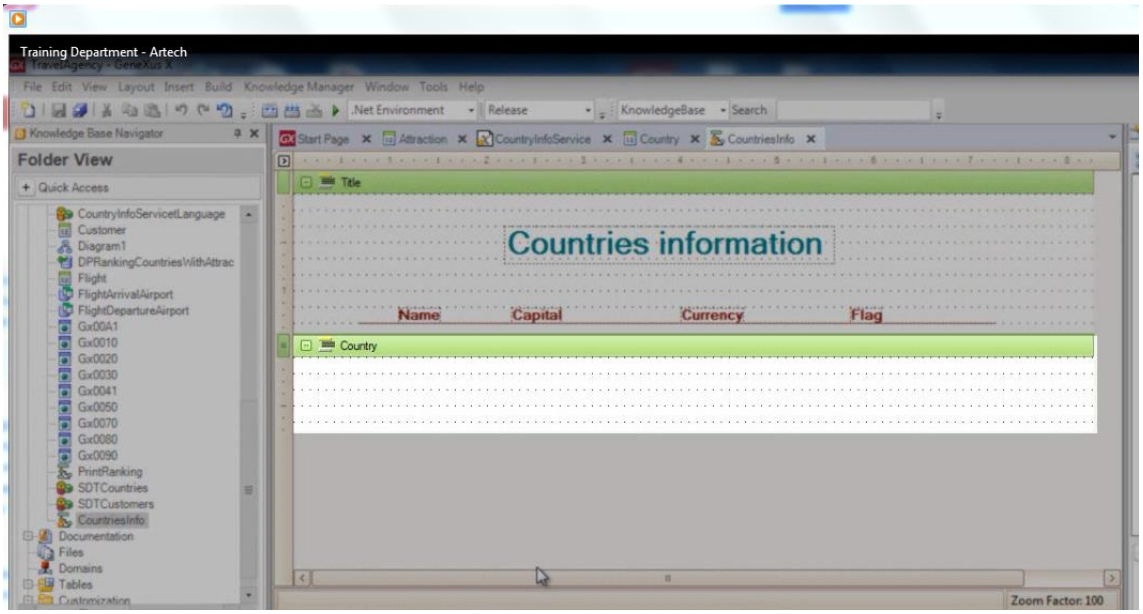
para implementar el listado pdf solicitado por la agencia de viajes.

Observamos en la sección Layout, el printblock de nombre “Title”



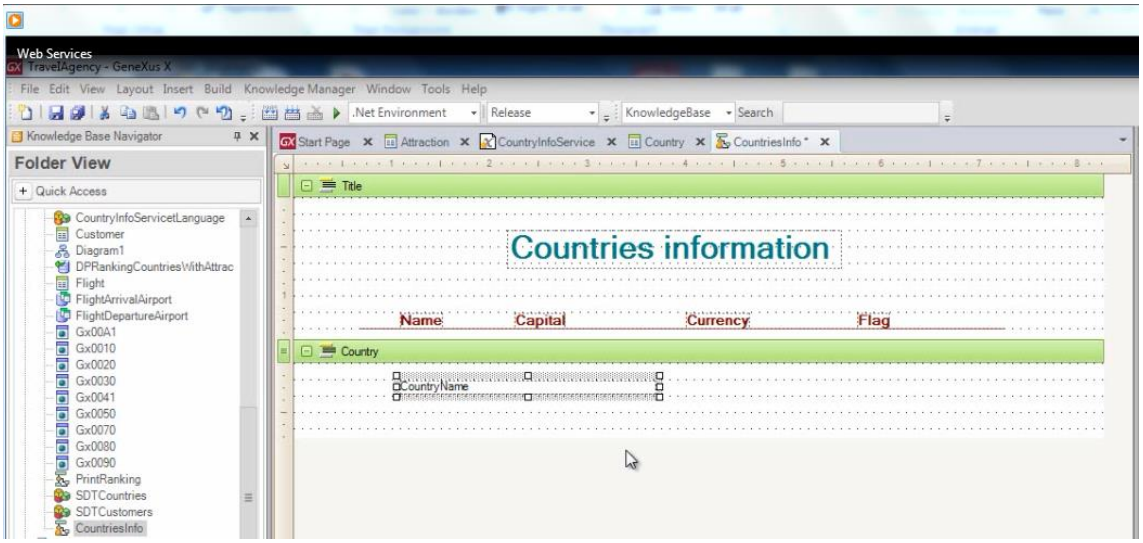
con los títulos..

y el printblock de nombre Country

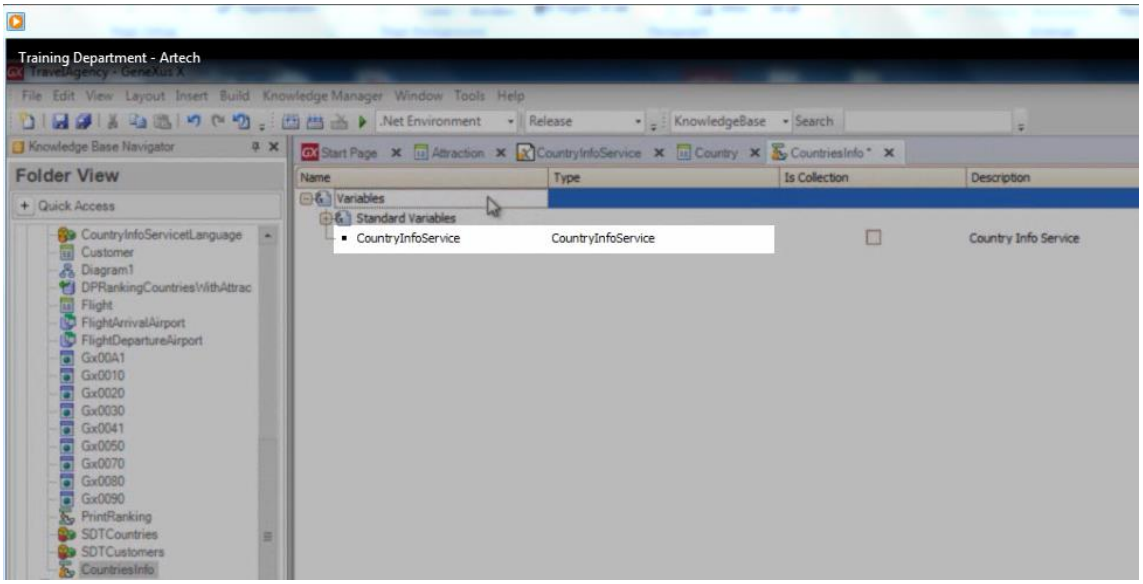


donde insertaremos los atributos y variables que necesitemos.

Queremos ver el nombre de los países, y este dato lo tenemos en nuestra base de datos así que insertamos el atributo CountryName:



Ahora bien. Para cada cada objeto externo, GeneXus crea un tipo de datos asociado, así que para poder utilizar al web service vamos a definir una varible del tipo: CountryInfoService y así podremos acceder a todos sus métodos



Ahora vayamos al source y comencemos a escribir el código necesario. La primer sentencia ya está escrita y como ya hemos aprendido, es para imprimir los títulos.

```
1  Print Title
```

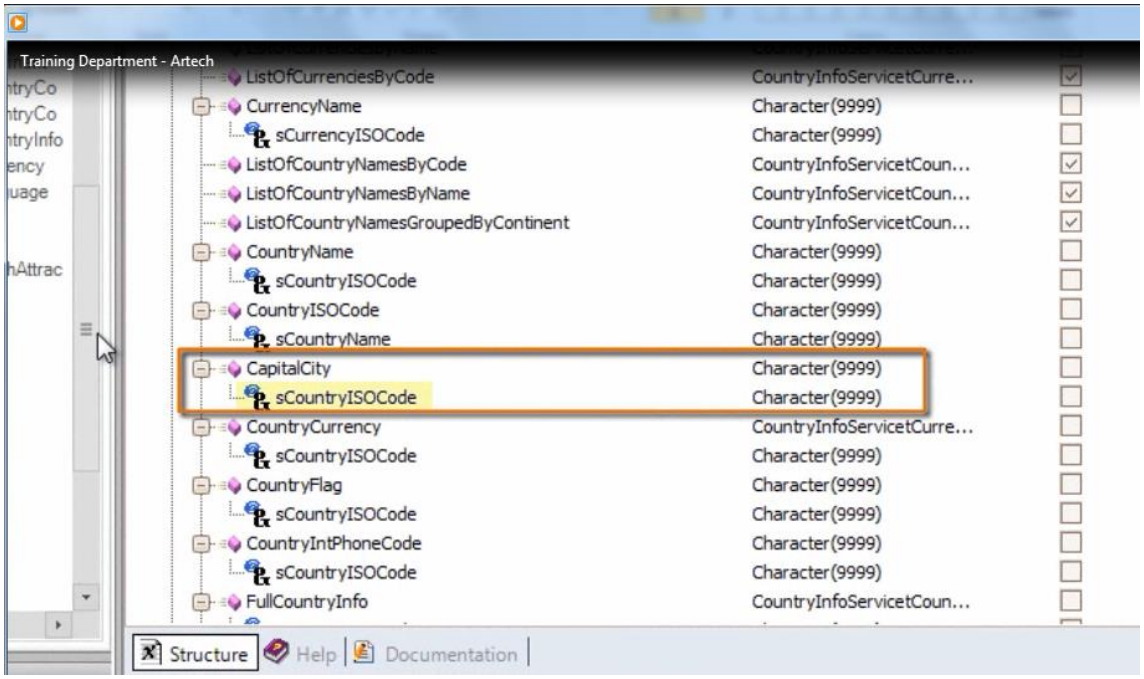
Ahora queremos recorrer la tabla COUNTRY, así que escribimos For each y al lado Country, por ser el nombre de la transacción cuya tabla física asociada es COUNTRY.

```
1  Print Title
2  For each Country
3
4
5
```

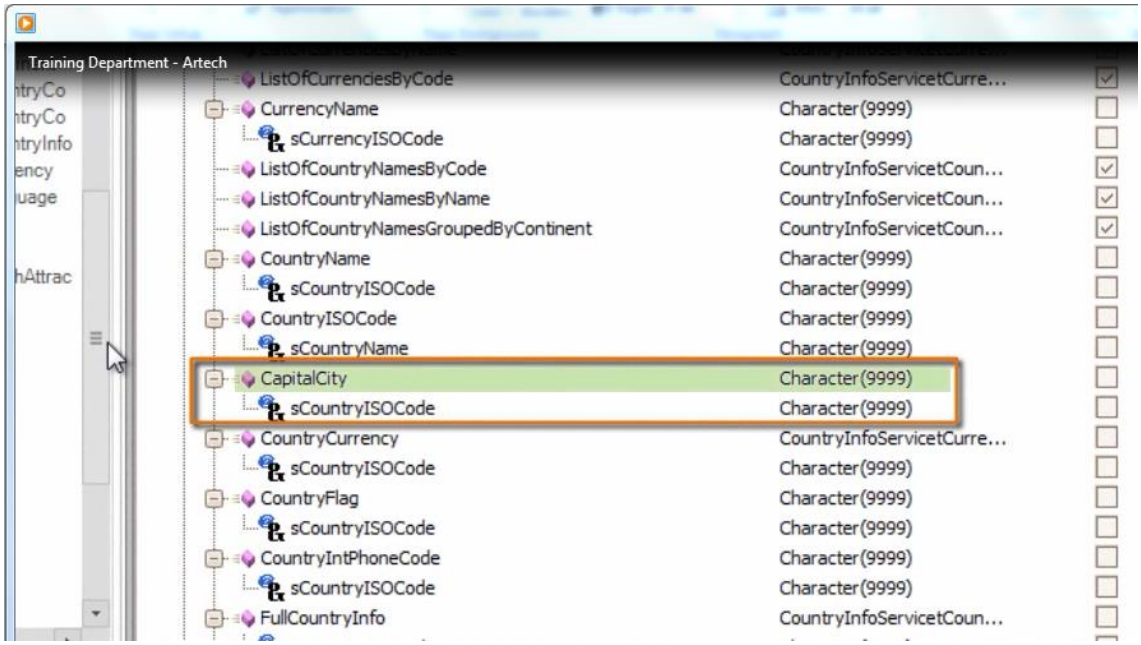
Vamos a ver ahora qué más debemos declarar y definir para que el web service nos devuelva la información que necesitamos.

Comencemos por ver cómo obtener la capital del país.

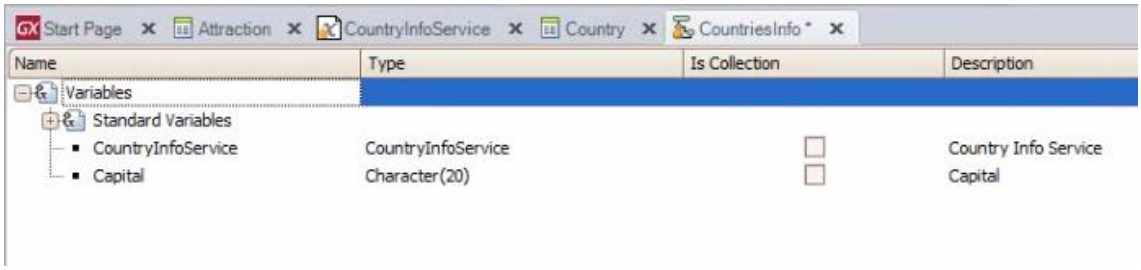
Si observamos nuevamente la lista de métodos, vemos que el método CapitalCity recibe el código ISO del país



y devuelve un tipo de dato Character

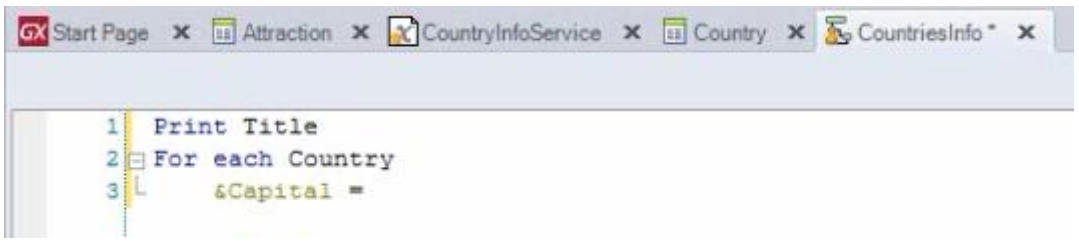


Así que en nuestro procedimiento definimos una variable de nombre “Capital” y de tipo de dato Character:



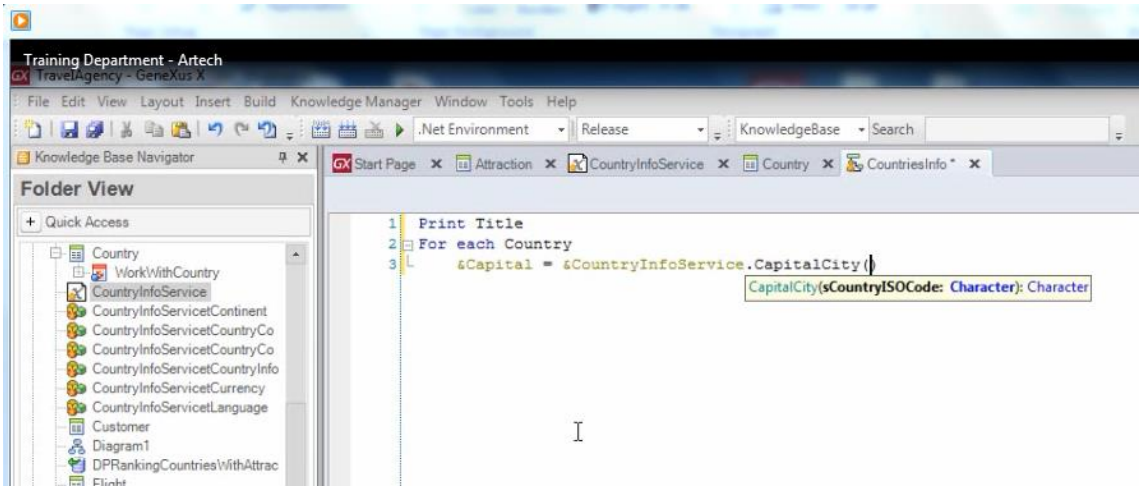
En el source, dentro del For each (porque queremos hacer todo lo siguiente, para cada país navegado) incluimos a la variable &Capital

a continuación el signo de igual, es decir, el signo de asignación

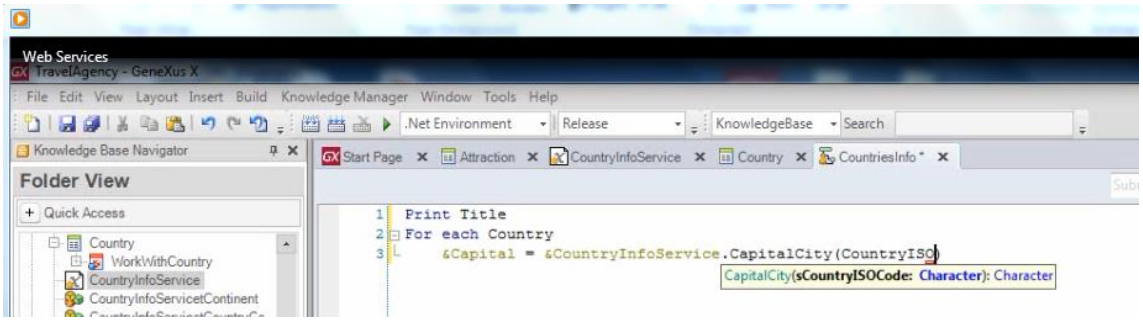


y para recibir la capital del país en esta variable &Capital, ponemos a la derecha del signo esta sintaxis:

la variable &CountryInfoService, punto, y el método que queremos ejecutar del web service, en este caso: CapitalCity

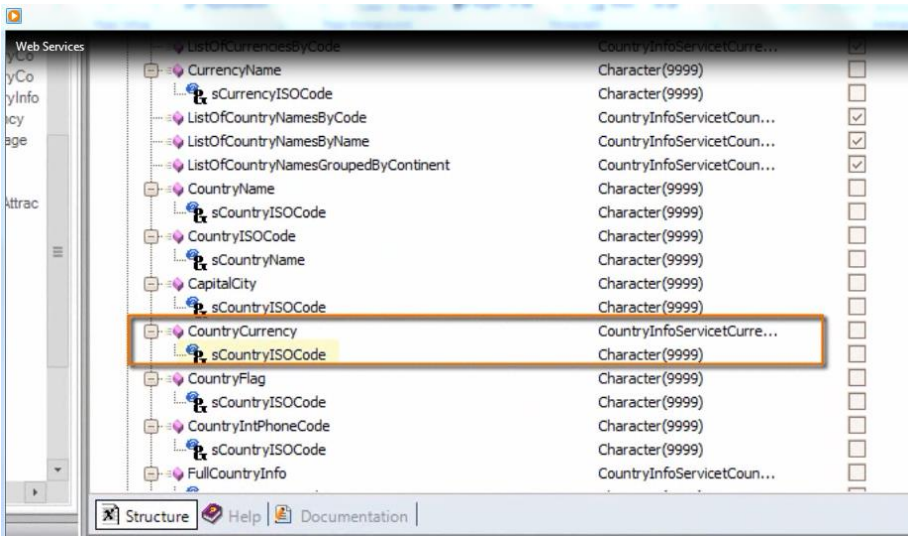


Y por último enviamos por parámetro al método invocado, al atributo CountryISO:

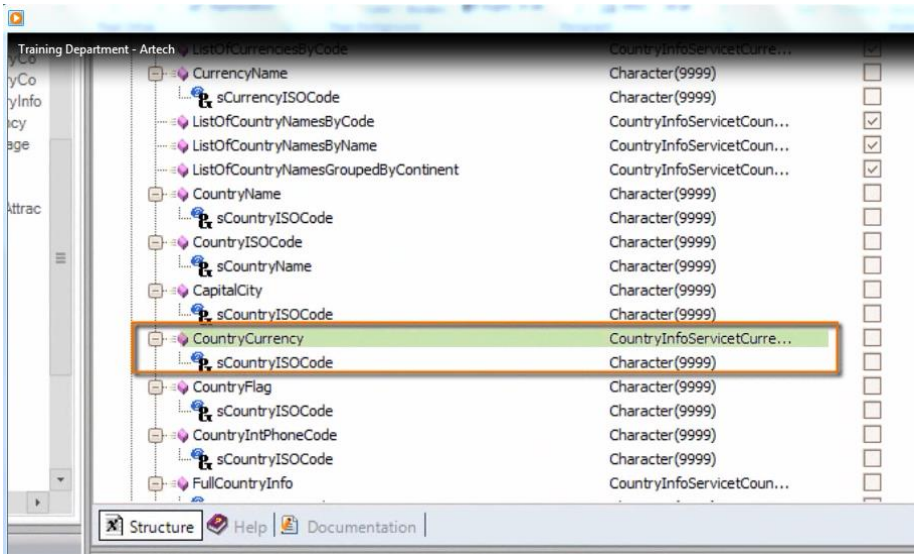


Ahora continuemos con la obtención de la moneda usada en el país.

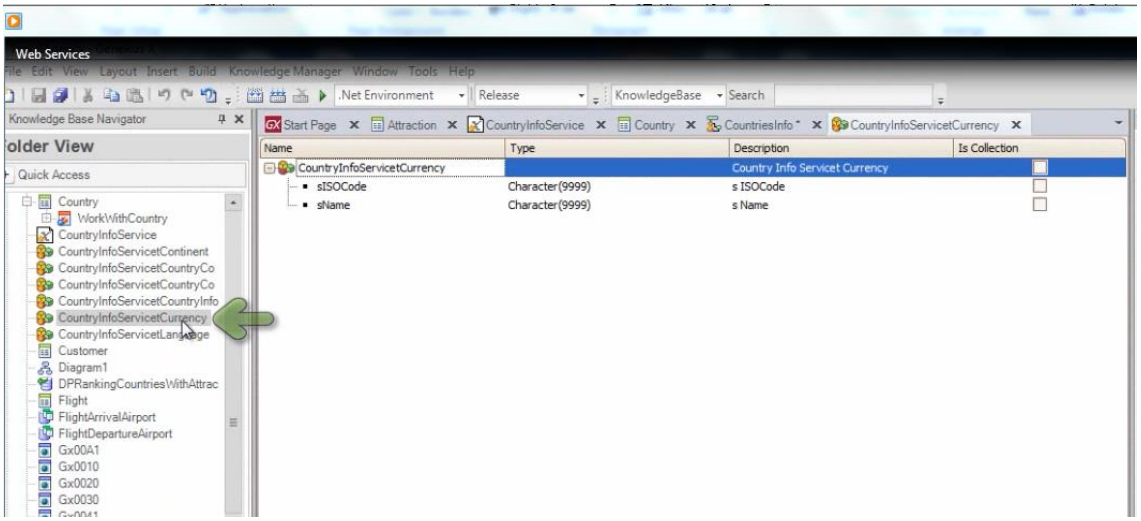
El método CountryCurrency recibe también como parámetro el código ISO del país



pero devuelve el tipo de dato: CountryInfoServiceCurrency

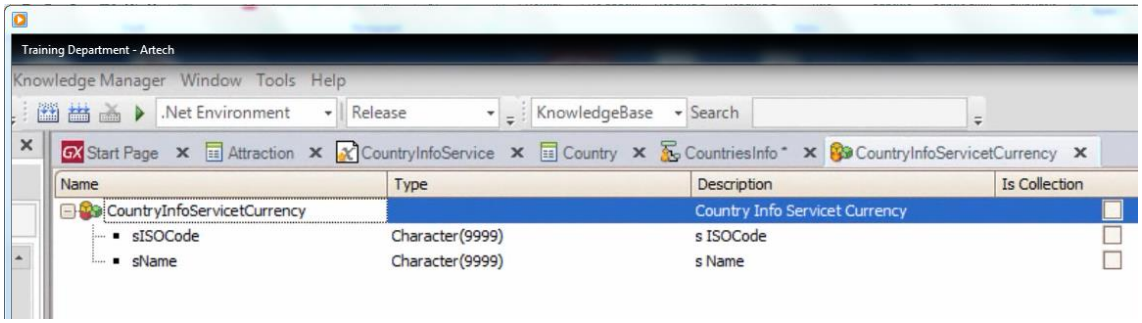


La definición del tipo de dato

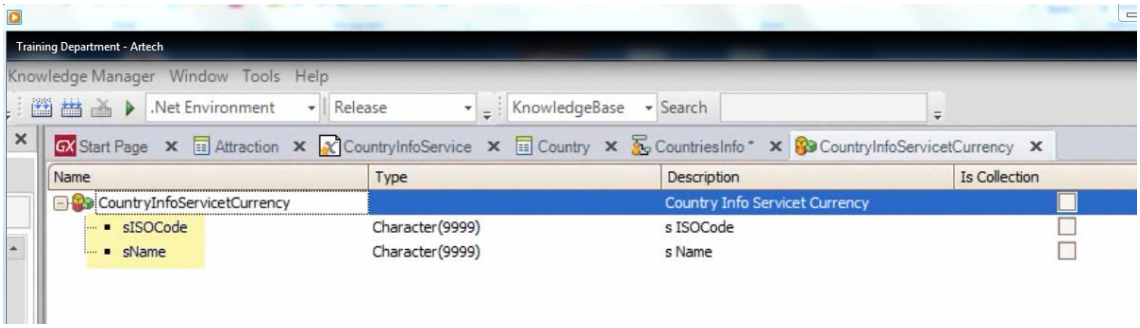


es este SDT que se creó en la base de conocimiento .

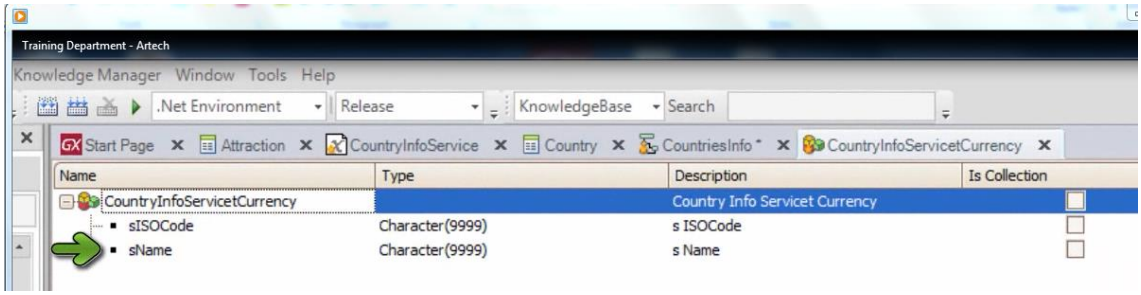
Observemos su estructura:



Vemos que se compone de dos items simples: el código y el nombre de la moneda

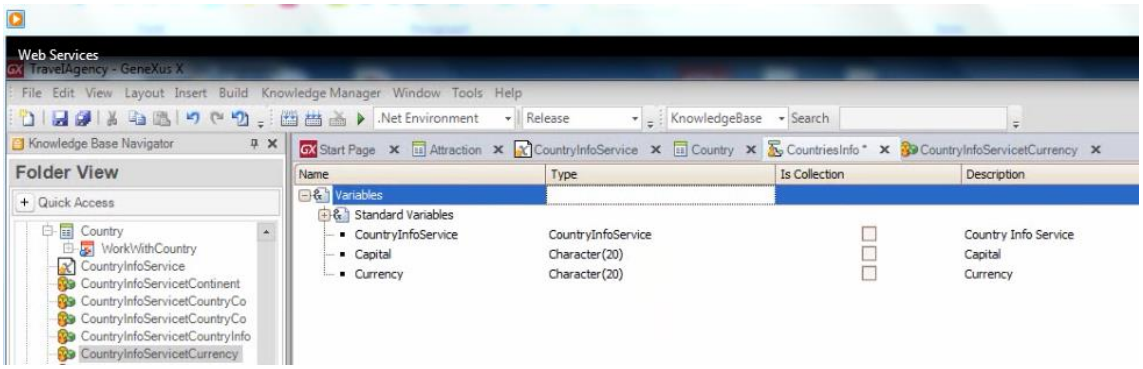


Nosotros queremos obtener el nombre de la moneda



o sea, el valor cargado en el campo sName.

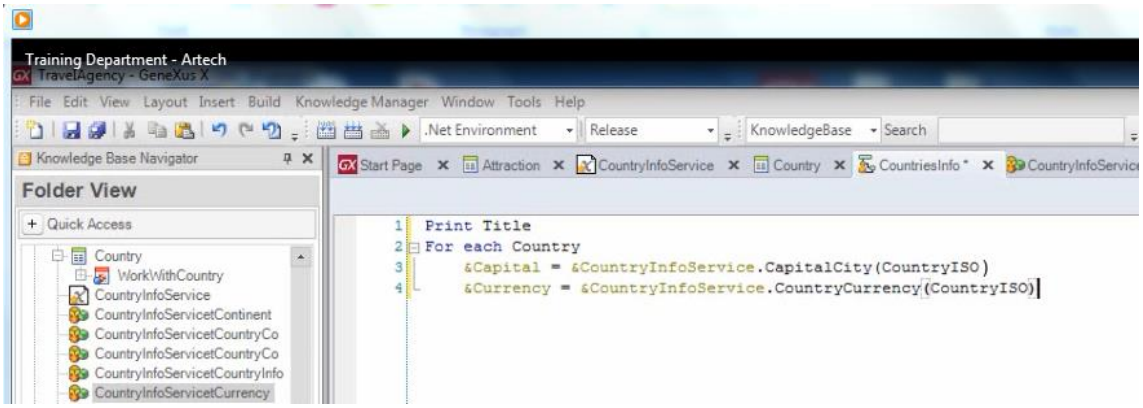
Vamos a definir entonces una nueva variable en nuestro procedimiento, de nombre Currency y tipo de dato Character:



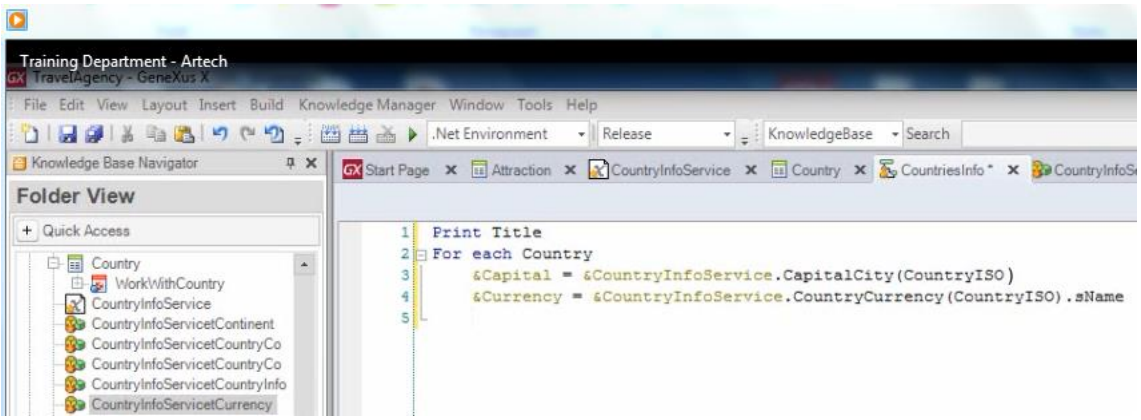
Volvamos al source a definir la siguiente sentencia:

A la variable &Currency, le asignamos la variable &CountryInfoService, seguida del punto y le aplicamos el método CountryCurrency.

Le enviamos por parámetro el código ISO

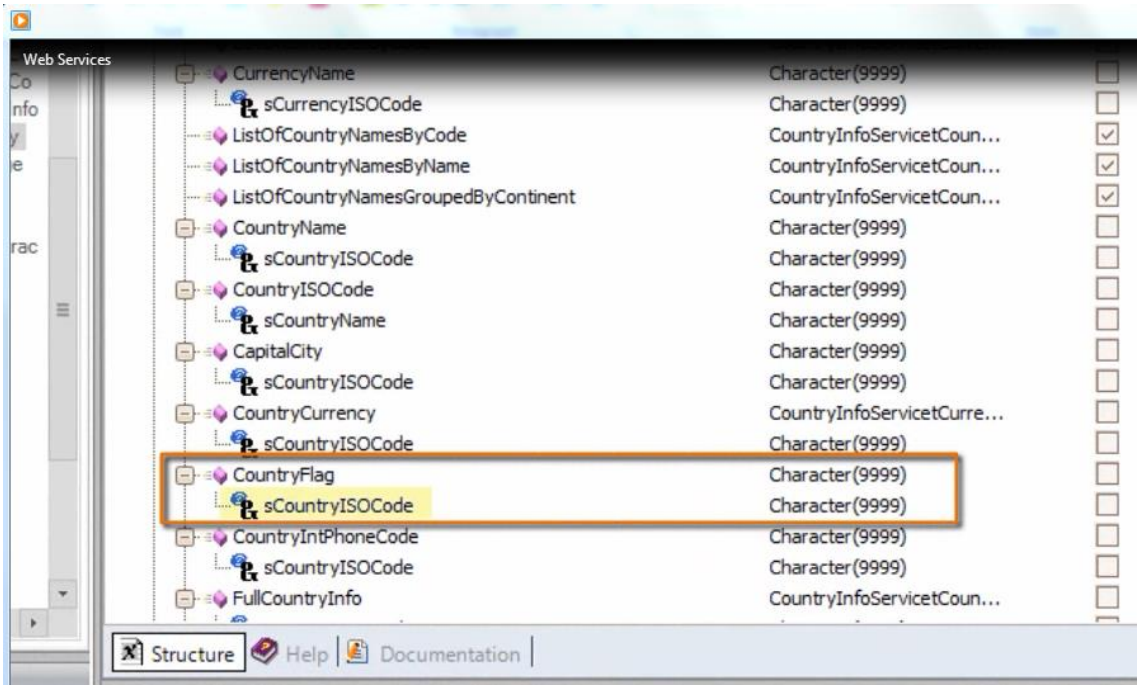


y si lo dejamos así, la devolución es de un tipo de datos compuesto por 2 datos, pero como queremos solamente el nombre de la moneda, ponemos nuevamente: punto y elegimos: sName

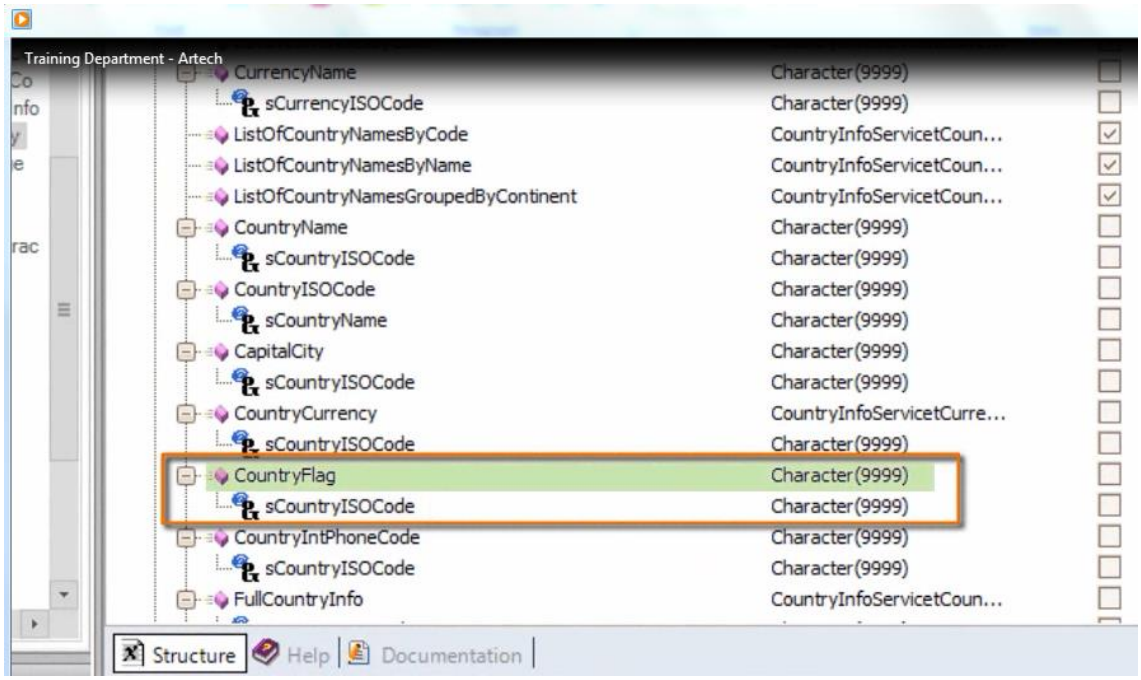


Ahora vamos a obtener la bandera del país.

Observemos que el método CountryFlag recibe también el código ISO del país

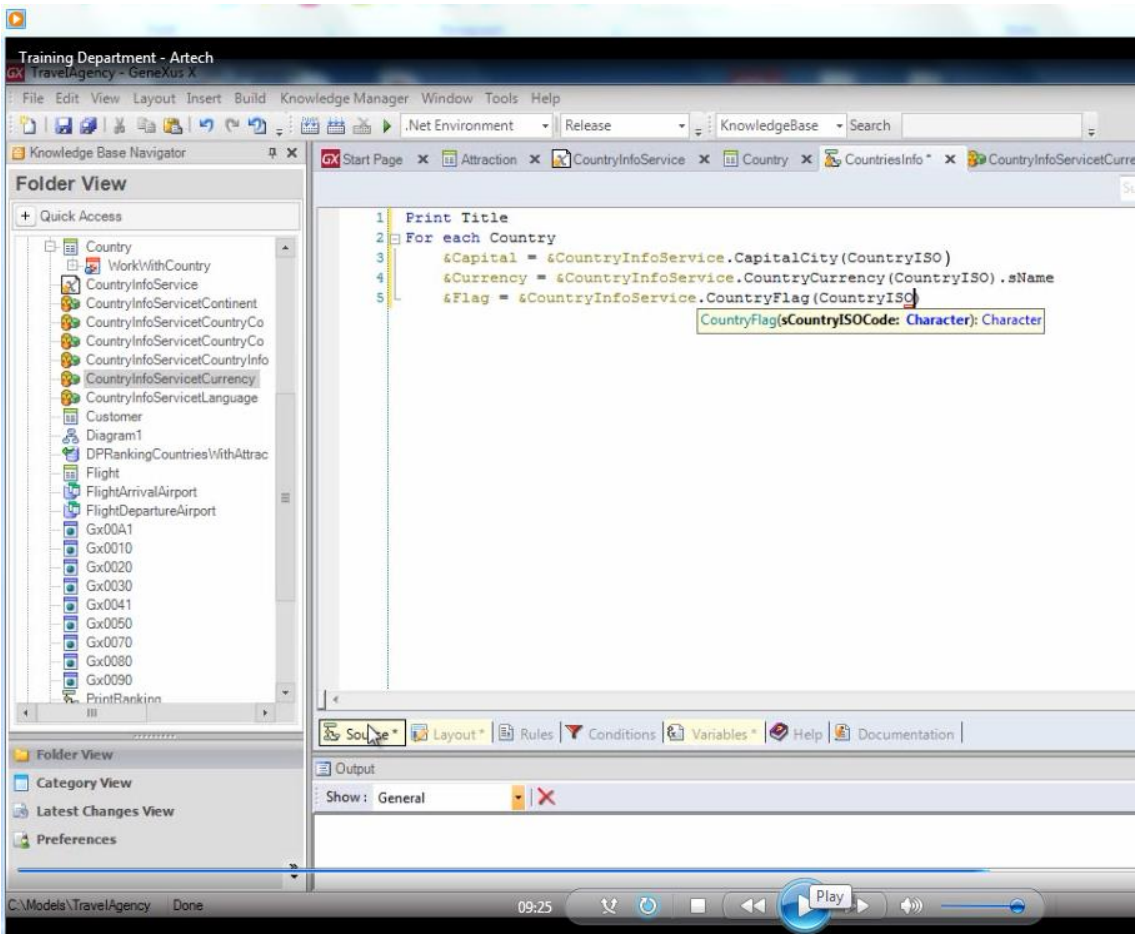


y devuelve el tipo de dato character



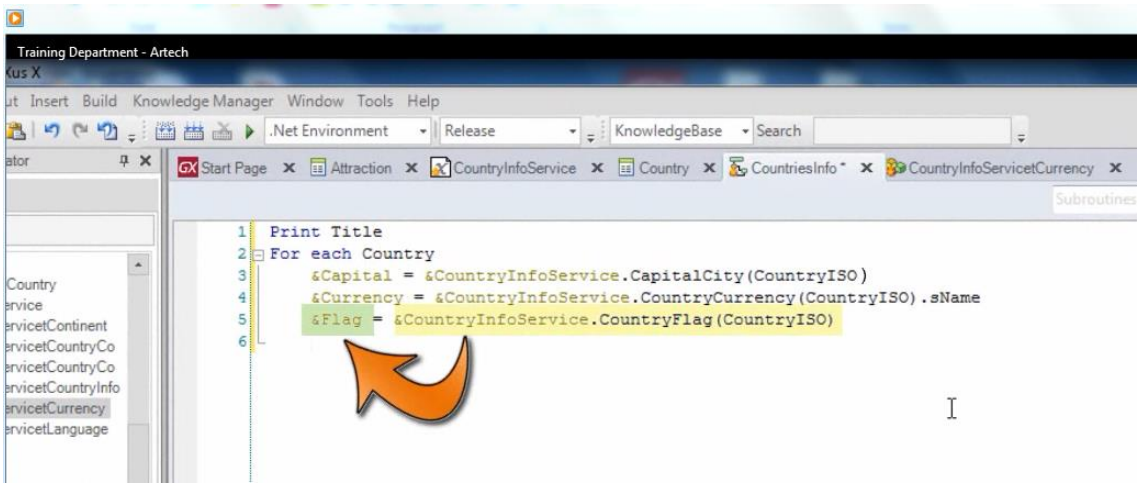
Así que volvemos a nuestro procedimiento y vamos a definir una variable de nombre: Flag y tipo de dato: Character.

Vamos al source y a la variable &flag le asignamos la variable &CountryInfoService, punto, y el método: CountryFlag, pasándole por parámetro el ISO del país, o sea el atribo CountryISO



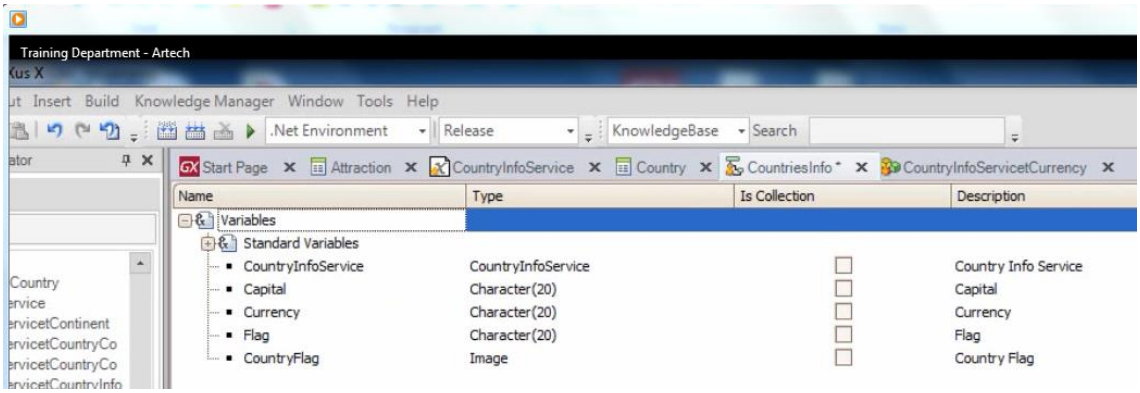
Analicemos ahora lo siguiente:

En la variable &Flag tenemos almacenado un valor character que devolvió el método CountryFlag del webservice CountryInfoService.



Pero nosotros queremos ver la imagen de la bandera, así que de alguna forma necesitamos convertir ese dato character en un dato de tipo Image.

Vayamos nuevamente al sector de las variables del procedimiento y vamos a definir esta vez una variable de nombre CountryFlag de tipo Image:



y en el source, definimos la siguiente sentencia:

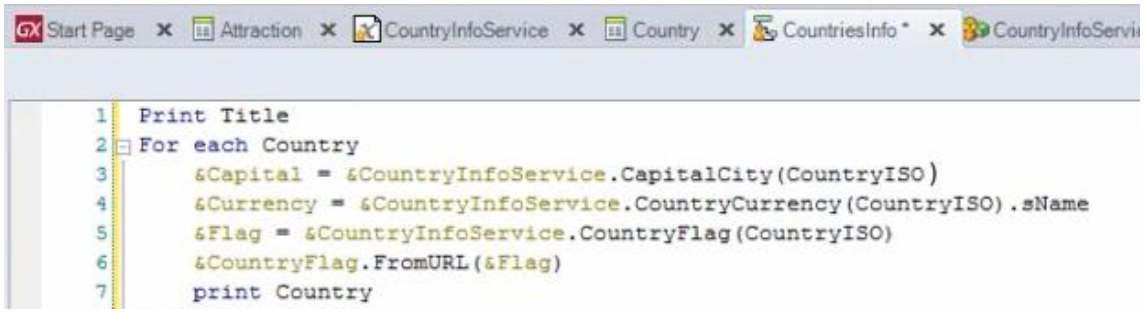
la variable &CountryFlag, punto, FromURL, pasándole la variable &Flag:

Y de esta forma convertimos el character de la variable &Flag en una imagen.

El método FromURL puede ser aplicado a toda variable o atributo de tipo image, pasándole por parámetro un character con el path.

Ahora que ya tenemos todas las variables que necesitamos mostrar, cargadas con los valores correspondiente, vamos a la sección del Layout, e insertamos estas variables en el printblock Country:

Y nuevamente en el source, completamos nuestro código indicando que se imprima el printblock Country:

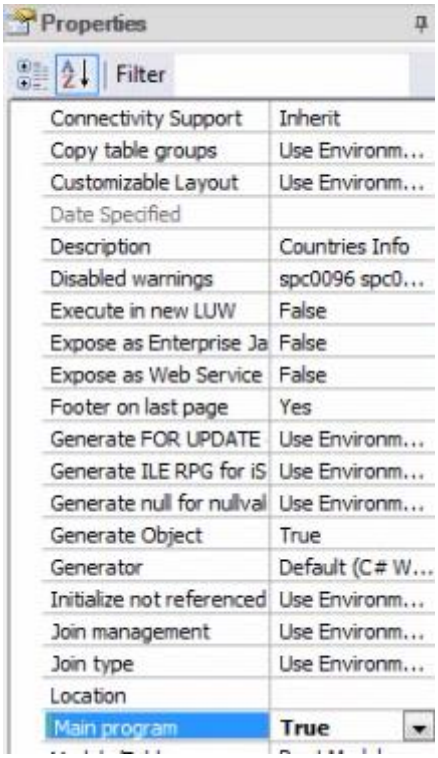


Terminamos con Endfor

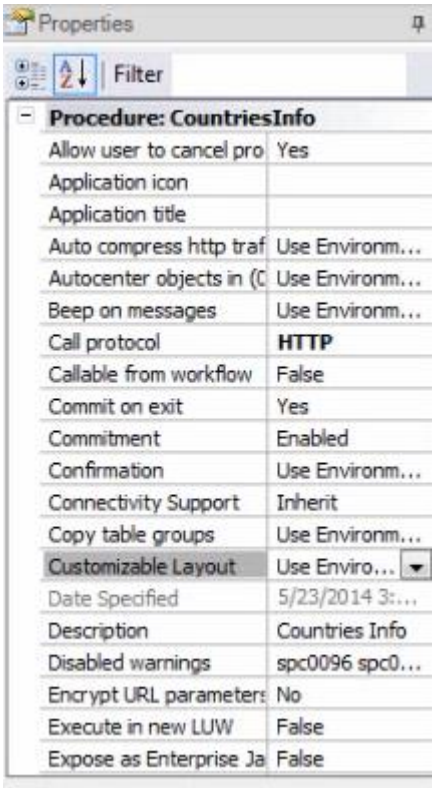
Y ya estamos listos para ver nuestro listado en ejecución!

Solamente recordemos que es necesario declarar las porpiedades para que el formato de la salida sea PDF..

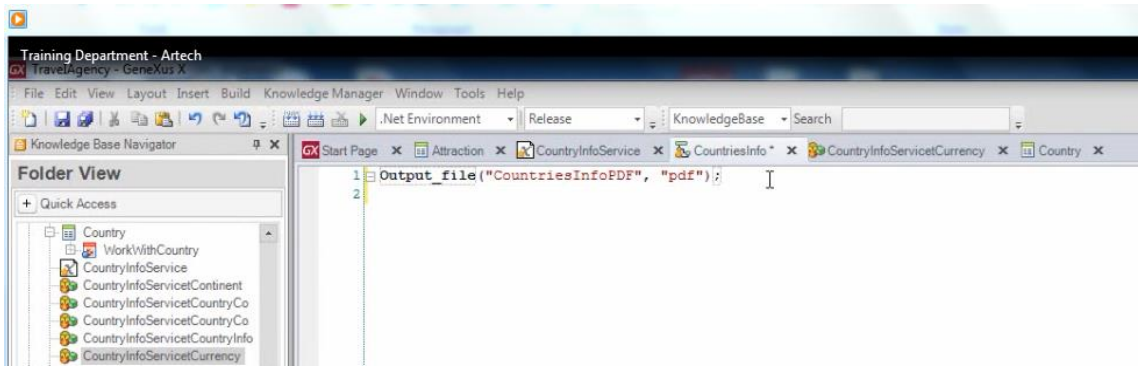
Así que a la propiedad “Main program” le asignamos el valor True:



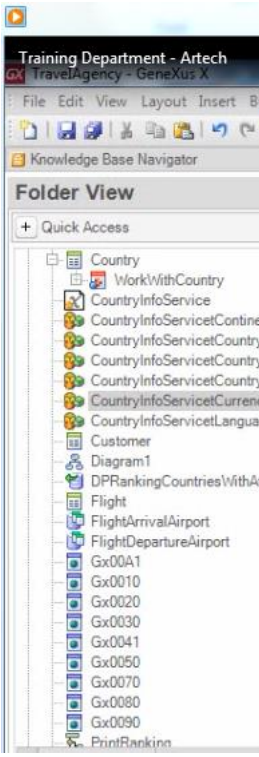
a la propiedad Call protocol, HTTP:



e insertamos en las rules la regla output_file para generar el listado pdf:



Ahora sí, presionamos click con el botón derecho del mouse sobre la solapa del procedimiento y elegimos Run



Countries information

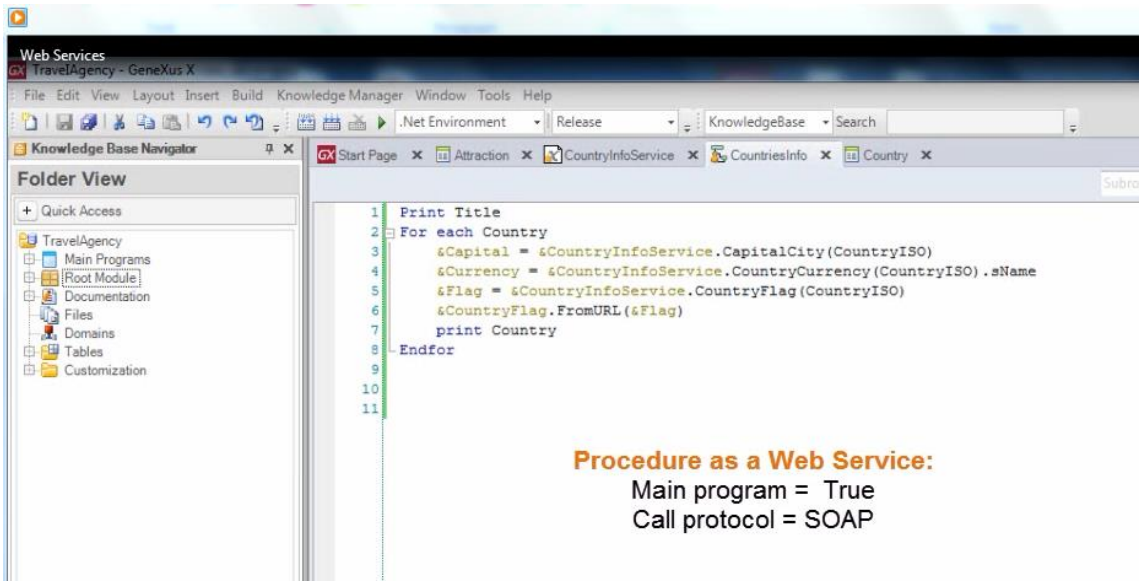
Name	Capital	Currency	Flag
Uruguay	Montevideo	Pesos	
Brazil	Brasilia	Brazil Real	
France	Paris	Euro	
China	Beijing	Yuan Renminbi	
United States	Washington	Dollars	
Egypt	Cairo	Pounds	

Y vemos el listado con la información completa.

De esta forma hemos visto cómo importar y consumir un web service disponible en inernet…. y por supuesto que también podemos crear con GeneXus web services y publicarlos para que puedan ser invocados por otras aplicaciones, o por nuestras propias aplicaciones.

A grandes razgos, para definir un web service en GeneXus, hay que crear un objeto procedimiento y configurar sus propiedades:

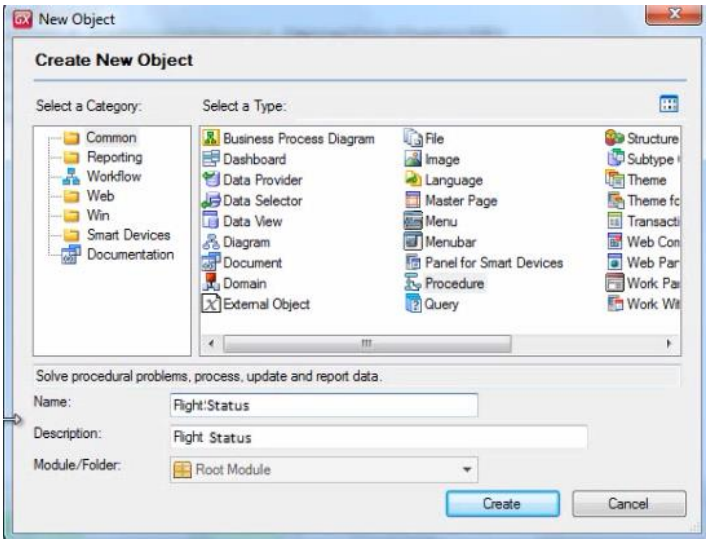
- Main program, con el valor True
- Y Call Protocol, con el valor SOAP



e implementar por supuesto la funcionalidad deseada.

Por ejemplo, supongamos que queremos implementar un procedimiento para publicarlo como web service. Este procedimiento recibirá el identificador de un vuelo, realizará ciertas validaciones y controles y devuelvorá un status.

Así que creamos un objeto de tipo procedimiento, le ponemos como nombre: FlightStatus



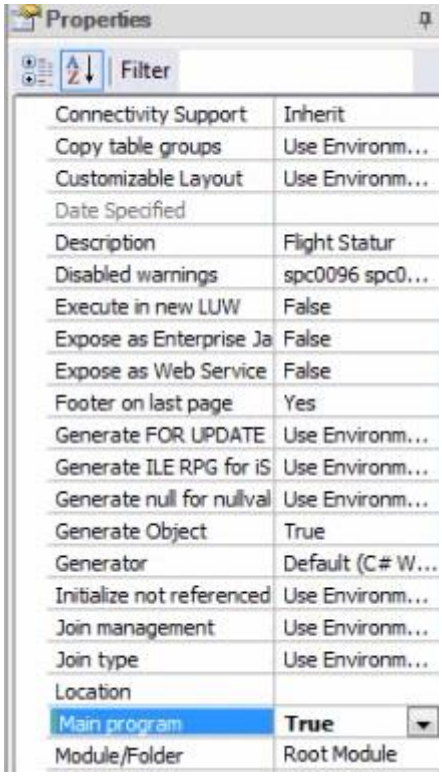
Y tendrá declarada la siguiente regla Parm:

Parm(&FlightId, &Status);

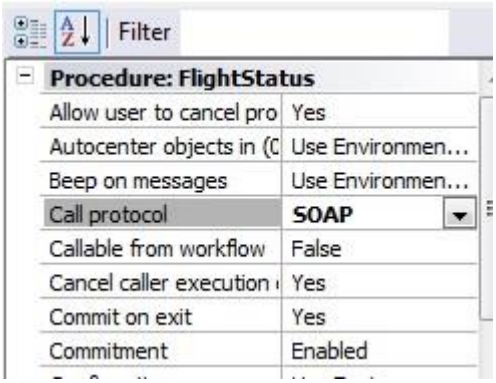
En el source, se ubicaría el flight, se validaría lo necesario y cargaría el status……

Y para que dicho procedimiento sea publicado como un web service y quede realmente disponible para ser utilizado por otras aplicacones, restaría configurar las propiedades :

Main program con el valor True:



y Call protocol con el valor SOAP:



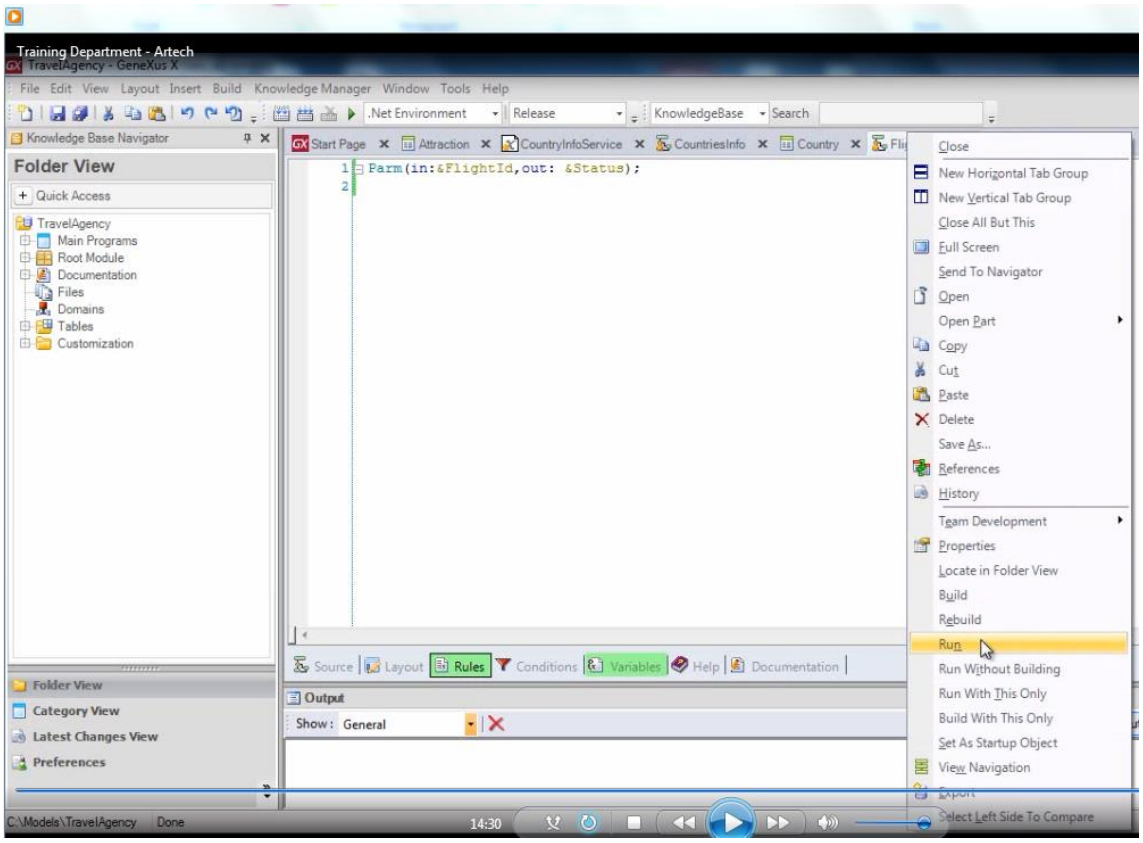
Los datos de entrada y salida, son los que se especifican en la regla parm() del procedimiento.

```
Parm(in:&FlightId, out:&Status);
```

El procedimiento recibe el identificador del vuelo y devuelve un status.

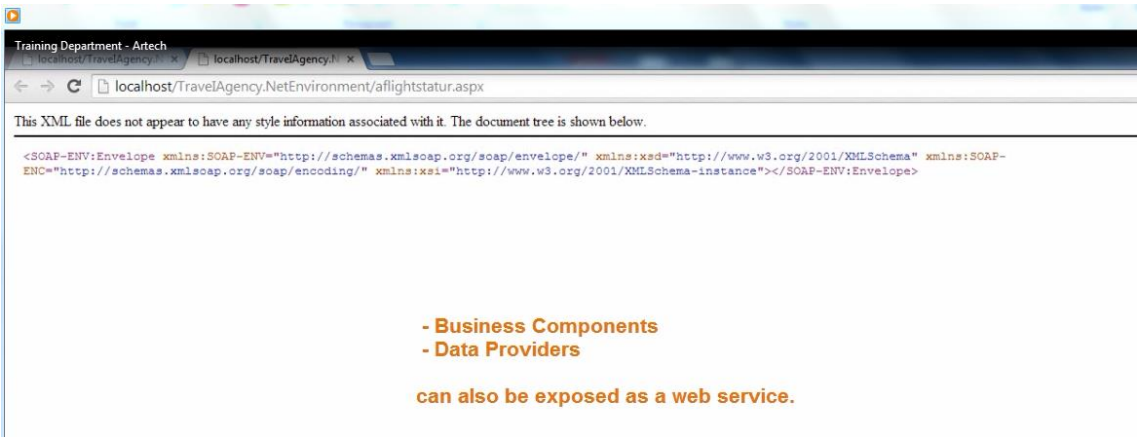
Una vez compilado, el procedimiento queda disponible para ser consumido por cualquier aplicación (sea de GeneXus o no, de la misma KB o no.).

Así que ejecutemos la opción Run sobre nuestro procedimiento



y vemos que esta ejecución permite ver el XML generado y se establece el camino que posteriormente se utilizará para consumir este WebService:

Es importante mencionar que también posible crear un web service publicando un Business component como servicio o un Data Provider como servicio.



Y para finalizar, actualizamos los cambios en GXserver