

Curso práctico de actualización a

GeneXus™ 16

PARTE 3

Abril 2020

Copyright © GeneXus S.A. 1988-2020.

All rights reserved. This document may not be reproduced by any means without the express permission of GeneXus S.A. The information contained herein is intended for personal use only.

Registered Trademarks:

GeneXus is trademark or registered trademark of GeneXus S.A. All other trademarks mentioned herein are the property of their respective owners.

CONTENIDO

CONTENIDO.....	2
OBJETIVO.....	3
COMENCEMOS.....	3
ODATA.....	4
Importación de servicios	4
Posible solución	4
Utilizar los servicios importados: adiministración de vuelos, airlines, airports.....	8
Posible Solución.....	9
Utilizar los servicios importados: LISTADO DE VUELOS	12
Solución:.....	14
GENEXUS AI	17
Pasos previos.....	18
Agregar referencia al módulo GeneXusAI	18
Cambiar el idioma del simulador	19
Traducción automática de texto.....	19
Solución.....	21
Lectura de texto en voz alta	22
Solución.....	22
KB SOLUCIÓN	23

OBJETIVO

Para cumplir el objetivo de desarrollar aplicaciones modernas, mejorando la integración con servicios externos, en este práctico trataremos los temas OData e Inteligencia Artificial (AI).

La letra de este práctico se ha actualizado de acuerdo al upgrade 8 de GeneXus 16. Si lo realiza con una versión posterior, considere que puede haber cambios entre lo que aquí se le muestra y lo que usted ve.

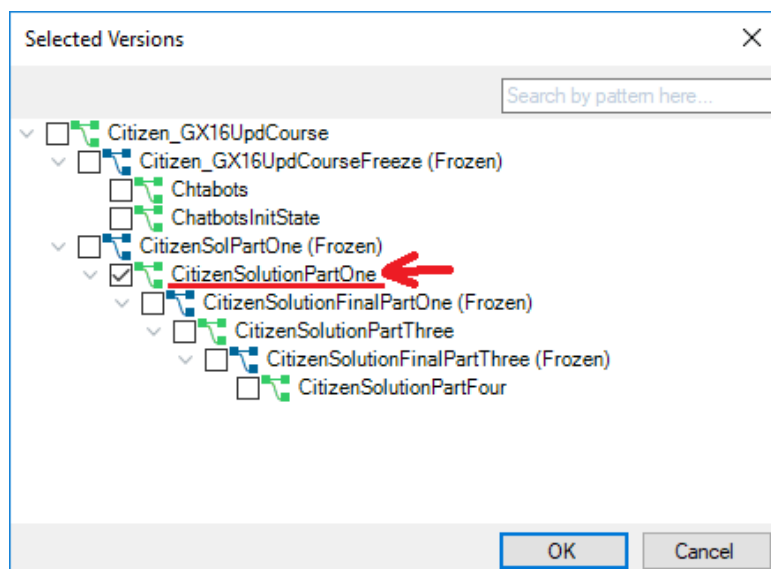
La aplicación sobre la que trabajaremos es una simplificación de una app para la municipalidad de una ciudad, que ofrece un frontend (Web/SD) para que los ciudadanos mediante su identificador de usuario puedan realizar reclamos (por ejemplo por árboles caídos, semáforos que no funcionan, coches mal estacionados, etc.), puedan realizar trámites (por ejemplo para obtener licencia de conducir, refinanciar una deuda con el municipio, instalar elementos de publicidad, etc), reservando un turno para ser atendidos por el personal del municipio. En el frontend se les muestran también las diversas actividades culturales que ofrece la ciudad. Y también se cuenta con un backoffice web para que ciertos funcionarios de la municipalidad manejen los datos y vean estadísticas.

Para mejorar el turismo desde la ciudad, se realizará una integración con servicios de TripPin que permitirán llevar un registro de vuelos realizados por los usuarios a distintos destinos.

Además se ofrecerá traducir la descripción de las actividades culturales al idioma del dispositivo, y para mejorar la accesibilidad, también se ofrecerá reproducir esa descripción como audio.

COMENCEMOS

Se partirá de la versión de la KB con la que usted trabajó en el práctico de Design Systems y UX, en el estado en el que le haya quedado al finalizar el práctico. Si por algún motivo la perdió o no la encuentra, créese otra desde <http://samples.genexusserver.com/v16>, (KB de nombre Citizen_GX16Course) eligiendo la versión de nombre **CitizenSolutionPartOne**.



Observe en el KB Explorer que hemos agregado el folder GeneXus/Web/Backend_ODATA para que allí coloque los objetos que vaya creando para esta parte del práctico. Y el folder AI para crear allí los procedimientos necesarios para conectarse al proveedor de AI que se le indicará.

ODATA

Se desea agregar dos nuevas funcionalidades para el backoffice de la aplicación, consumiendo OData desde TripPin. Lo que se quiere es:

1. Llevar un registro de vuelos salientes, administrando la información de los vuelos, aeropuertos y aerolíneas.
2. Listar toda la información relacionada a los vuelos con destino a cierto aeropuerto.

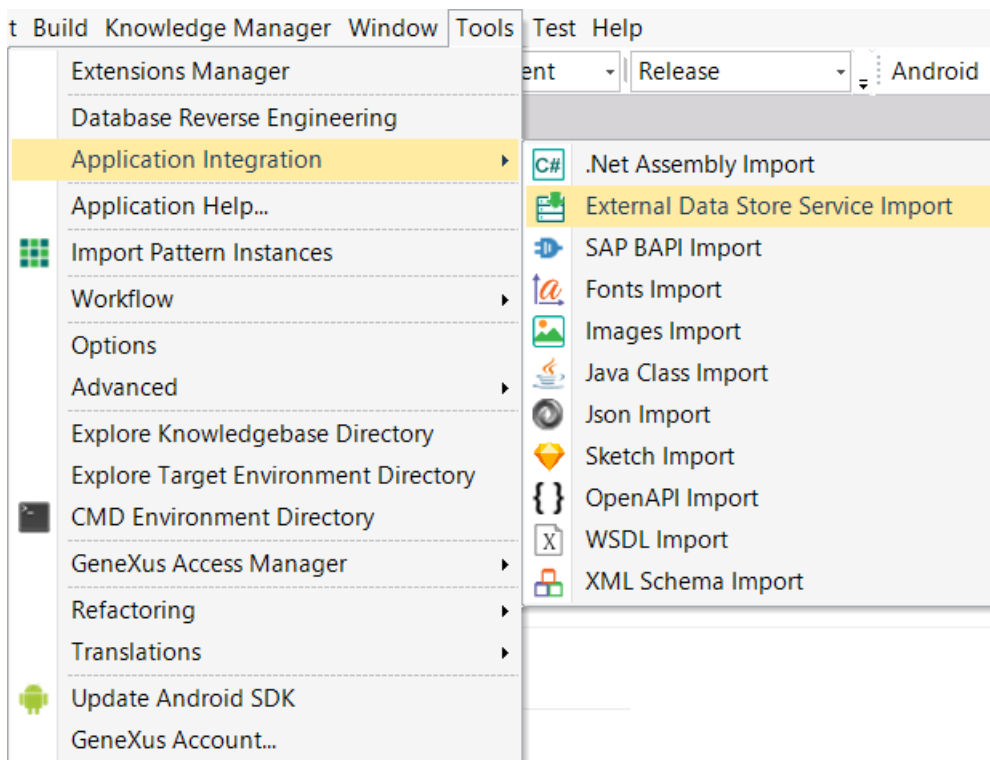
IMPORTACIÓN DE SERVICIOS

Para poder implementar las nuevas funcionalidades necesitaremos importar algunos servicios desde TripPin.

Las entidades de las que deseamos utilizar la implementación son la de Airlines y Airports. Los vuelos serán una entidad de nuestra propia KB, que deberemos crear después.

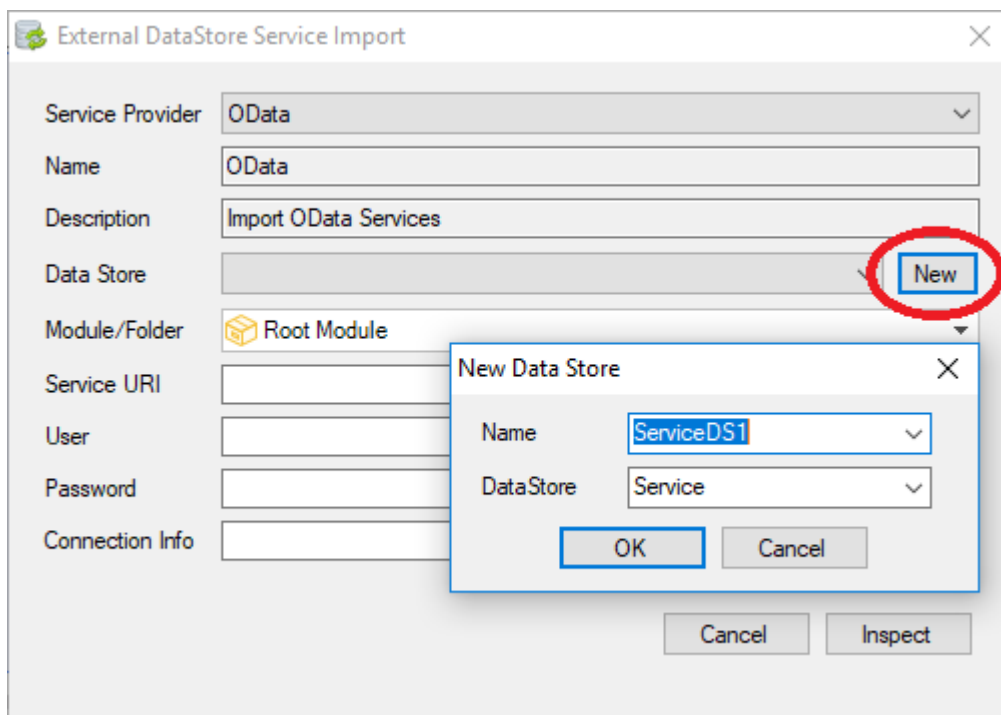
POSIBLE SOLUCIÓN

Importaremos desde TripPin lo necesario para poder implementar las nuevas funcionalidades. Para hacer esto:



Se le abrirá una nueva ventana donde ingresaremos los datos para poder consumir el servicio.

Lo primero que debemos hacer es crear un nuevo Data Store de tipo Service, desde el botón New ubicado a la derecha de la opción de DataStore:



Luego debemos rellenar los campos:

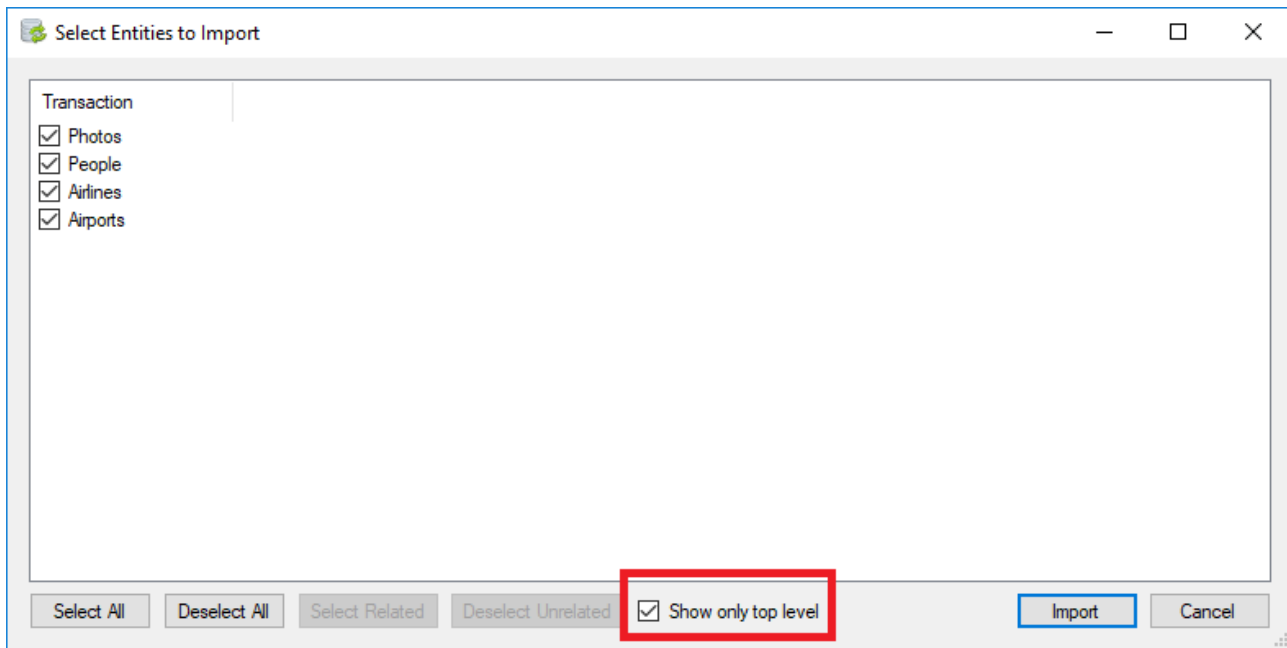
- Service URI: aquí debemos de ingresar la dirección del servicio a consumir. En este caso utilizaremos el servicio de TripPin:
[https://services.odata.org/V4/\(S\(40gwcqlhjmuyfayfnplov0o\)\)/TripPinServiceRW/](https://services.odata.org/V4/(S(40gwcqlhjmuyfayfnplov0o))/TripPinServiceRW/)
- User: aquí debemos ingresar nombre de usuario, en caso de ser necesario. En este caso no se necesita por lo cual lo dejaremos vacío.
- Password: aquí debemos ingresar password del usuario. En este caso no se necesita por lo cual lo dejaremos vacío.
- Connection Info: aquí se ingresa información extra para la conexión, en este caso es necesario que ingresemos “filter_strings=n” para indicar que no acepta condiciones como <,>,<=,>= con datos de tipo String.

The screenshot shows a dialog box titled "External DataStore Service Import". It contains the following fields and values:

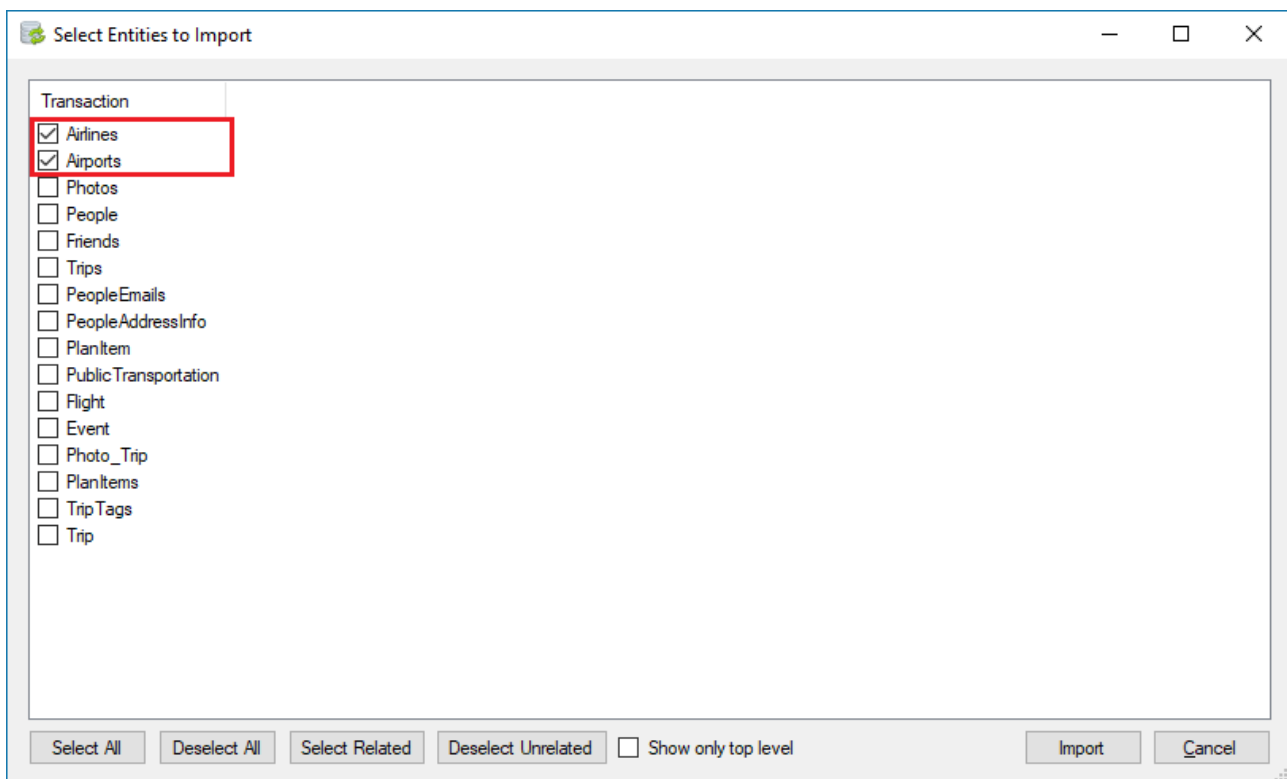
- Service Provider: OData
- Name: OData
- Description: Import OData Services
- Data Store: ServiceDS1
- Module/Folder: Root Module
- Service URI: https://services.odata.org/V4/(S(40gwcqlhjmuyfayfnplov0o))/TripPinServiceRW/
- User: (empty)
- Password: (empty)
- Connection Info: filter_strings=n

At the bottom right, there are two buttons: "Cancel" and "Inspect".

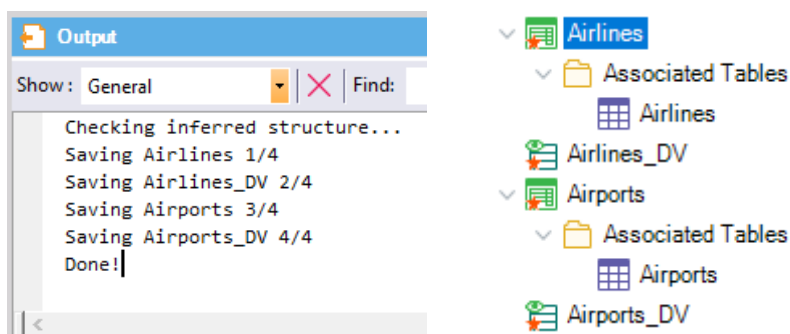
Una vez que completamos los campos hacemos clic en Inspect. Esto abrirá una nueva ventana en donde se mostrarán las entidades a ser importadas desde el servicio. Para este caso necesitaremos solo Airports y Airlines. Para esto debemos desmarcar el checkbox “Show only top level”:



Y dejar marcado solamente las opciones de Airlines y Airports



Una vez hecho esto presionamos Import. Esto le importará las entidades Airports y Airlines como transacciones, con los data views asociados:



La estructura de las transacciones es la siguiente:

Name	Type
Airlines	Airlines
AirlinesAirlineCode	Character(40)
AirlinesName	Character(40)

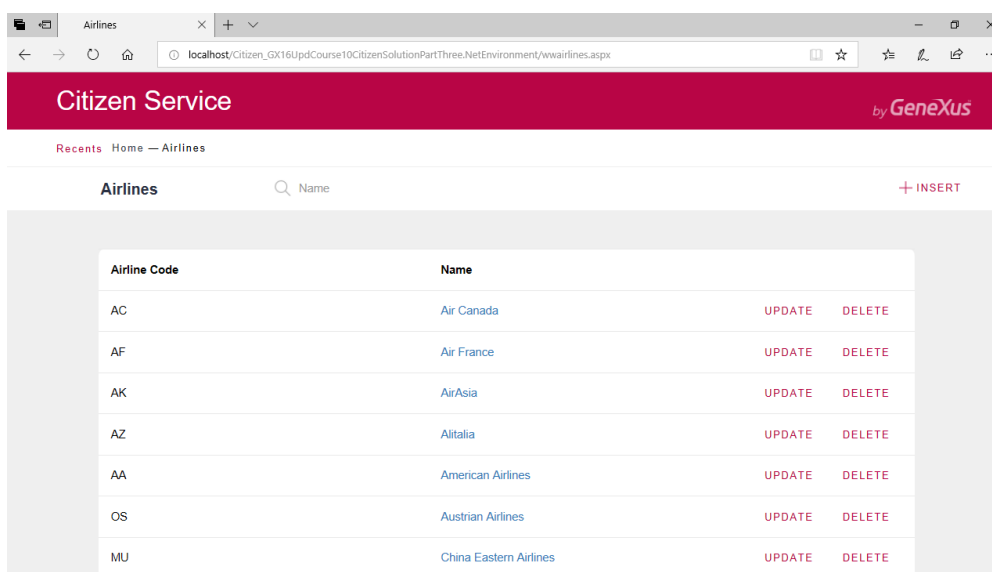
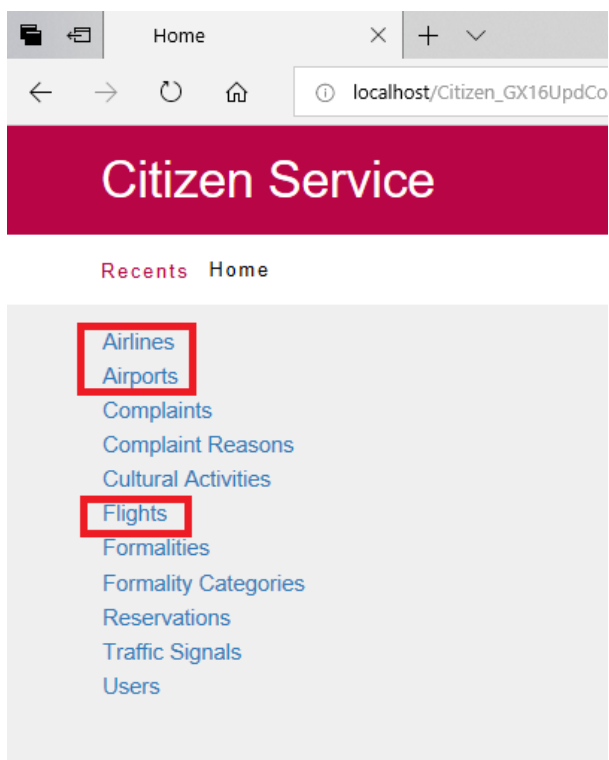
Name	Type
Airports	Airports
AirportsIcaoCode	Character(40)
AirportsName	Character(40)
AirportsIataCode	Character(40)
AirportsLocation_Address	Character(40)
AirportsLocation_City_CountryRegion	Character(40)
AirportsLocation_City_Name	Character(40)
AirportsLocation_City_Region	Character(40)
AirportsLocation_Loc	GeoPoint

Recomendamos mover las entidades al folder GeneXus/Web/Backend_ODATA

UTILIZAR LOS SERVICIOS IMPORTADOS: ADMINISTRACIÓN DE VUELOS, AIRLINES, AIRPORTS

Luego de importados los datos, debemos utilizarlos para implementar las funcionalidades pedidas.

Primero que nada, queremos poder administrar (CRUD) los datos de aerolíneas y vuelos, agregándolos al backoffice.

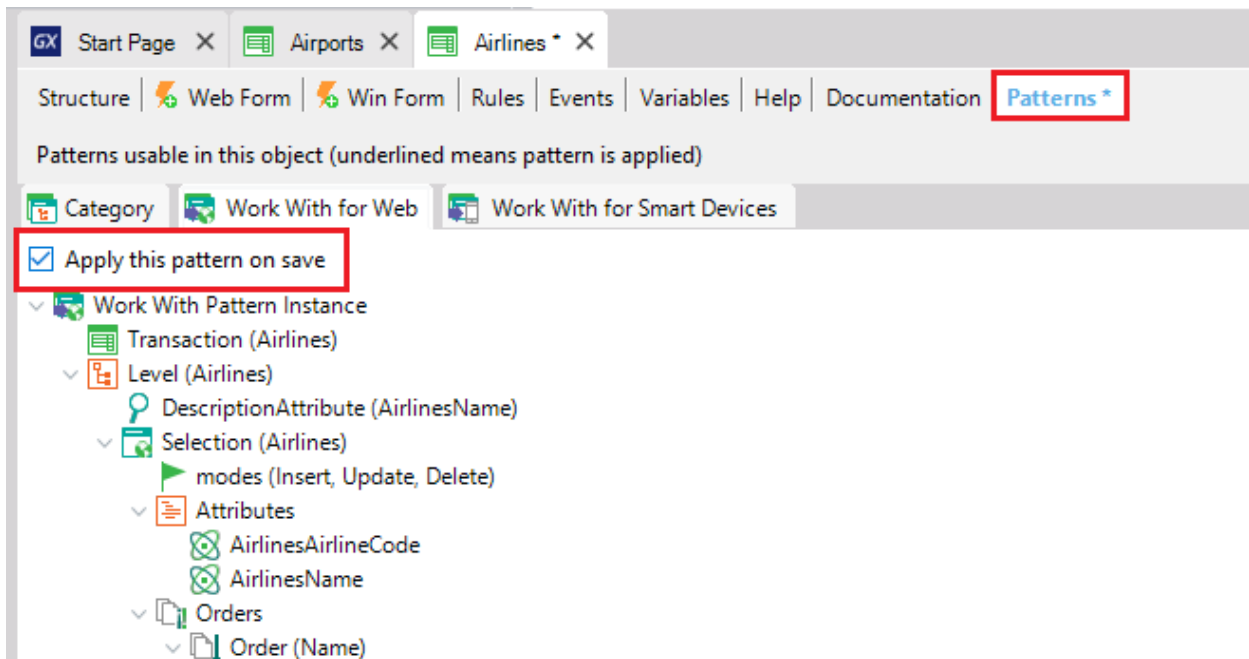


Nota: no se pueden insertar ni eliminar aeropuertos, pues el servicio provisto no lo permite.

Una vez implementado, pruebe cambiar el nombre de un aeropuerto, por ejemplo, e ingrese vuelos.

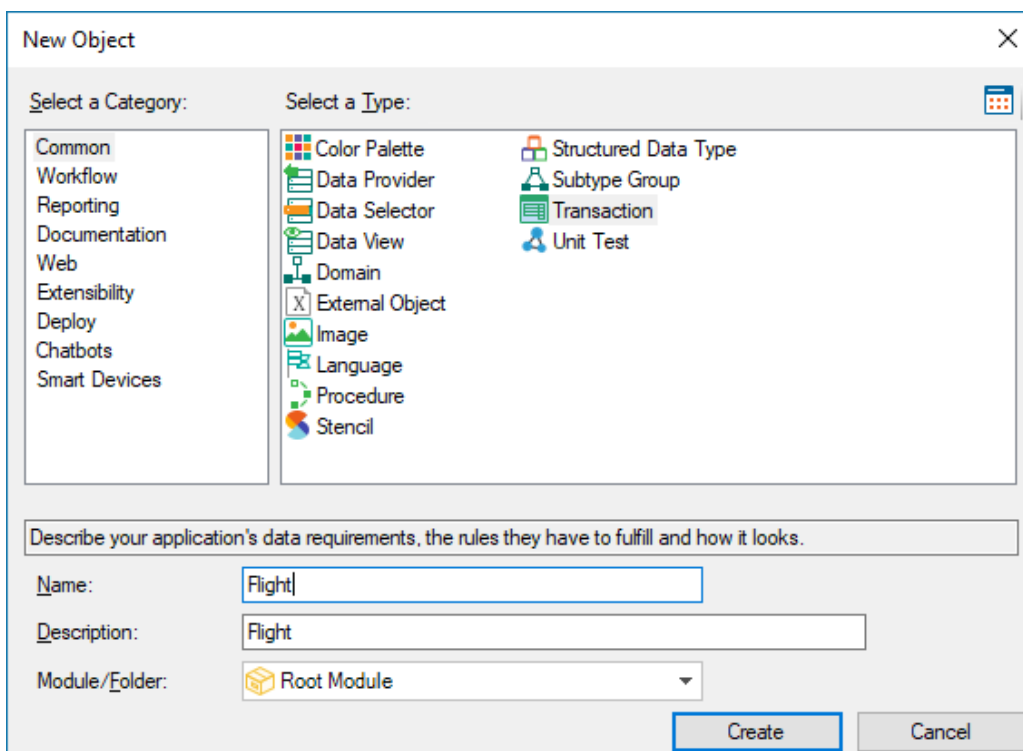
POSIBLE SOLUCIÓN

Aplicará el pattern Work With for Web a las transacciones Airports y Airlines.



Con esto se agregarán automáticamente al backoffice.

Luego creará una nueva transacción Flight, que será la encargada de llevar el registro de vuelos y se relacionará con Airports y Airlines:



La estructura de esta transacción va a ser:

Name	Type
Flight	Flight
FlightId	Id
AirportsIcaoCode	Character(40)
AirportsName	Character(40)
AirlinesAirlineCode	Character(40)
AirlinesName	Character(40)
Seat	Seat
FlightSeatRow	Numeric(4,0)
FlightSeatColumn	VarChar(40)
UserIdentification	Identification
UserName	Name
UserBirthDate	Date
UserPhoto	Image

A esta transacción también le aplicaremos el pattern Work With Web.

Con esto ya logramos implementar la administración de vuelos.

Ejecute y visualice los datos e ingrese alguno. Si lo desea puede utilizar los siguientes datos de prueba.

Vuelo 1:

FlightId	1
Airport	Toronto Pearson International Airport
Airline	Air Canada
Seat Row 1 Column A	User Luciano Silveira
Seat Row 1 Column B	User Mary Smith
Seat Row 2 Column A	User Martin Torad
Seat Row 2 Column B	User Ann Bert

Vuelo 2:

FlightId	2
Airport	John F. Kennedy International Airport
Airline	American Airlines
Seat Row 1 Column A	User Rudolph Roball
Seat Row 1 Column B	User Cecilia Lemos
Seat Row 2 Column A	User Jessica Loyarte

Vuelo 3

FlightId	3
Airport	Toronto Pearson International Airport
Airline	Air France
Seat Row 1 Column A	User Brian Goals
Seat Row 1 Column B	User John Peters
Seat Row 2 Column A	User Eleonor Johnson
Seat Row 2 Column B	User Valerie Molins

Citizen Service

by GeneXus

Recents Airlines — Airports — Home — Flight — Flights

Flights



+ INSERT







Id	Airports Icao Code	Airports Name	Airlines Airline Code	Airlines Name		
1	CYYZ	Toronto Pearson International Airport	AC	Air Canada	UPDATE	DELETE
2	KJFK	John F. Kennedy International Airport	AA	American Airlines	UPDATE	DELETE
3	CYYZ	Toronto Pearson International Airport	AF	Air France	UPDATE	DELETE

UTILIZAR LOS SERVICIOS IMPORTADOS: LISTADO DE VUELOS

Queremos mostrar en un panel todos los aeropuertos de destino posibles:

Code	Airports Name
CYYZ	Toronto Pearson International Airport
KATL	Hartsfield-Jackson Atlanta International Airport
KJFK	John F. Kennedy International Airport
KLAX	Los Angeles International Airport
KORD	O'Hare International Airport
KSEA	Seattle-Tacoma International Airport
KSFO	San Francisco International Airport
LIRA	Rome Ciampino Airport
LTBA	Istanbul Ataturk Airport
OMAA	Abu Dhabi International Airport
WSSS	Singapore Changi Airport

Y eligiendo uno emitir un pdf con todos los vuelos con destino a ese aeropuerto. De cada vuelo se quieren listar información de los pasajeros (users de Citizen):

Toronto Pearson International Airport			
Flight Id: 1		Airline: Air Canada	
Seat	Passenger		
1 A	Luciano Silveira	05/05/76	
1 B	Mary Smith	07/30/68	
2 A	Martin Torad	02/01/89	
2 B	Ann Bert	04/11/75	
Flight Id: 3		Airline: Air France	
Seat	Passenger		
1 A	Brian Goals	12/01/40	
1 B	John Peters	07/29/73	
2 A	Eleanor Johnson	03/12/90	

Ya le damos creado el web panel FlightsByAirports y el procedimiento ListAirportFlight, para que usted modifique allí lo que necesite y no pierda tiempo. Se encuentran en el folder GeneXus/Web/Backend_ODATA.

SOLUCIÓN:

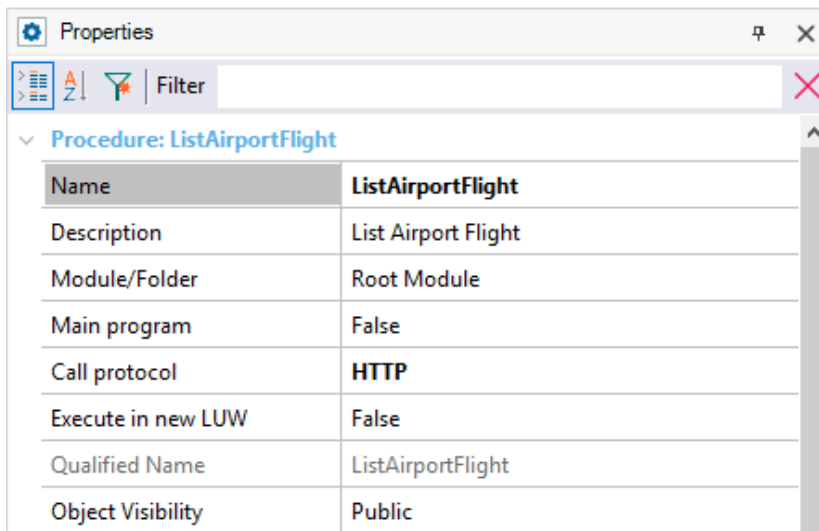
Utilizaremos un procedure (ya se lo damos creado, es el de nombre ListAirportFlight) para implementar el listado, que recibirá por parámetro el identificador de aeropuerto, para poder listar los vuelos que llegan a él.

En la sección de Rules tiene:

```
parm(in: &AirportCode);  
Output_file("AirportFlightsList.pdf","pdf");
```

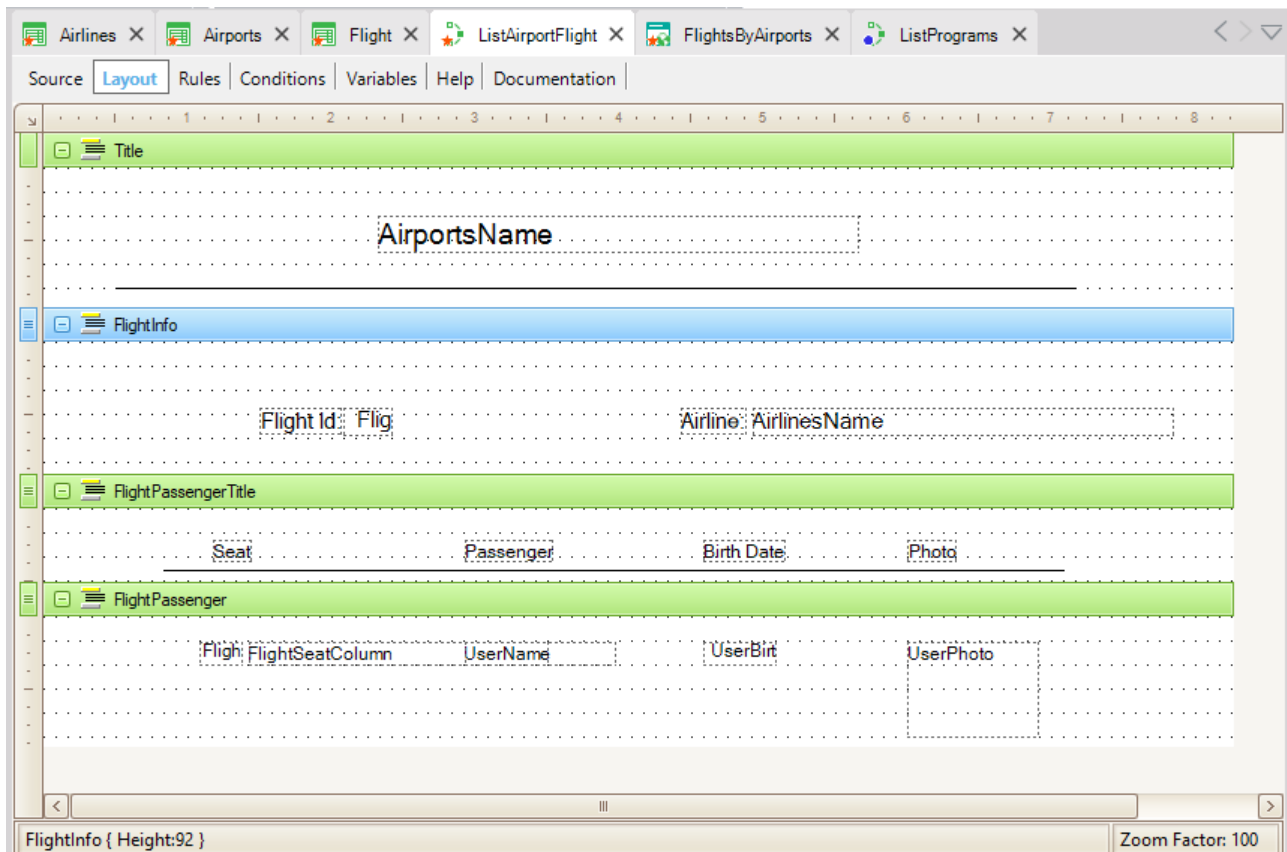
Donde **&AirportCode** es una variable basada en el tipo de datos del atributo AirlinesAirlineCode (que es Character(40)). Recuerde que **Output_file** se utiliza para definir la salida del listado (en nuestro caso un documento PDF).

Recuerde también que hubo que cambiar la propiedad Call protocol a HTTP:



Procedure: ListAirportFlight	
Name	ListAirportFlight
Description	List Airport Flight
Module/Folder	Root Module
Main program	False
Call protocol	HTTP
Execute in new LUW	False
Qualified Name	ListAirportFlight
Object Visibility	Public

El layout debería quedarle más o menos así:



Y la sección Source:

```

For each Airports
  where AirportsIcaoCode = &AirportCode
  print Title
  For each Flight
    print FlightInfo
    print FlightPassengerTitle
    For each Flight.Seat
      print FlightPassenger
    endfor
  endfor
endfor

```

Invocamos este procedure desde el Web Panel FlightsByAirports, donde insertamos un Grid con los atributos AirportsIcaoCode y AirportsName.

En la sección de Events:

```

Event AirportsName.Click
  ListAirportFlight( AirportsIcaoCode )
endevent

```


Para enganchar este panel al backoffice (al web panel Home), editamos el procedure ListPrograms y agregamos el siguiente código:

```
&name = !"FlightsByAirports"  
&description = "Flights By Airports"  
&link = FlightsByAirports.Link()  
  
Do 'AddProgram'
```

Ejecutamos la aplicación.

GENEXUS AI

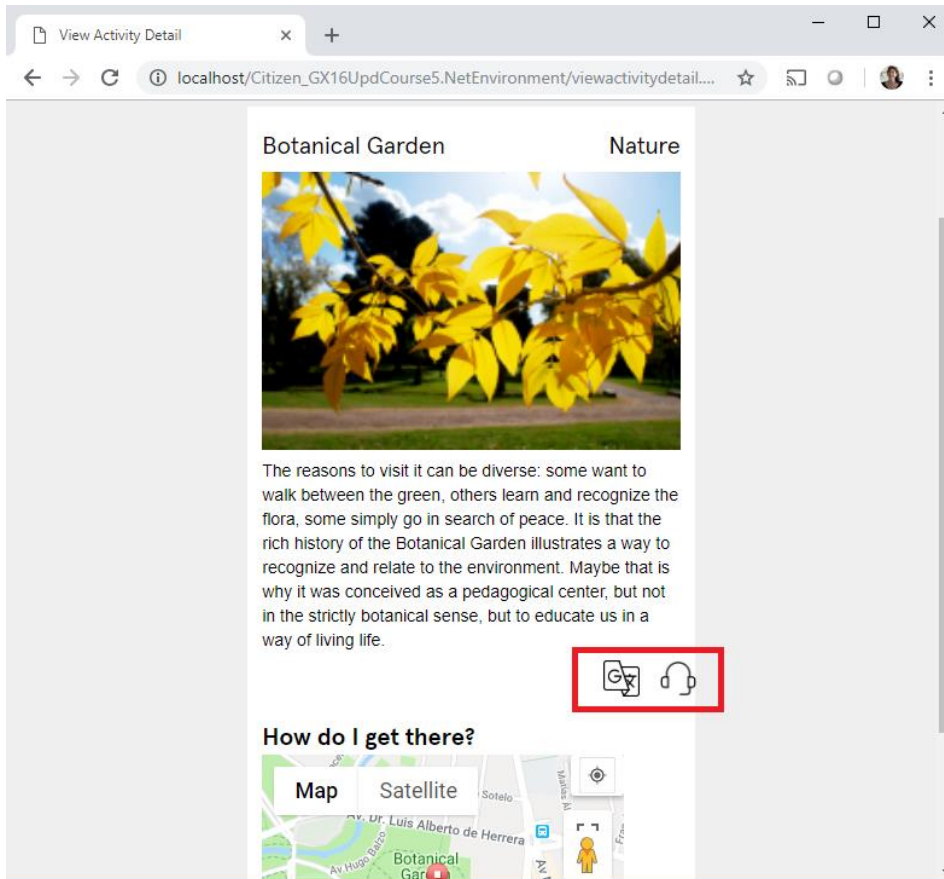
Se desea agregar dos nuevas funciones a la aplicación para poder:

1. traducir la descripción de una actividad cultural dada al idioma del dispositivo, y
2. permitir que la aplicación lea en voz alta dicha descripción.

Para eso en el panel ActivityDetail del frontend SD tenemos dos botones donde se agregará la funcionalidad:



Lo mismo queremos en el panel ViewActivityDetail análogo, del frontend web:

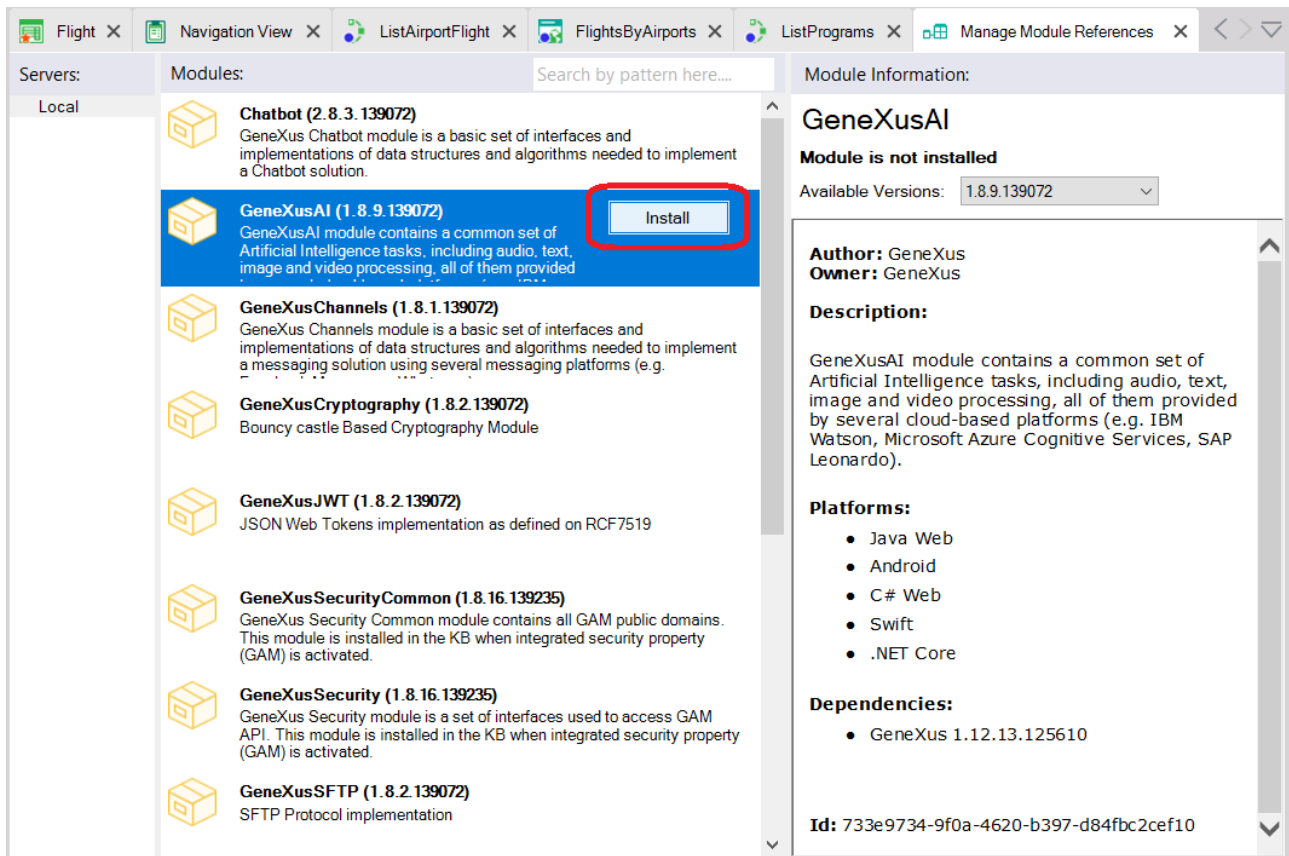


PASOS PREVIOS

AGREGAR REFERENCIA AL MÓDULO GENEXUSAI

Antes de comenzar a agregar la funcionalidad, debemos agregar la referencia al módulo GeneXusAI.

Eso lo hacemos en la opción Knowledge Manager > Manage Module References



CAMBIAR EL IDIOMA DEL SIMULADOR

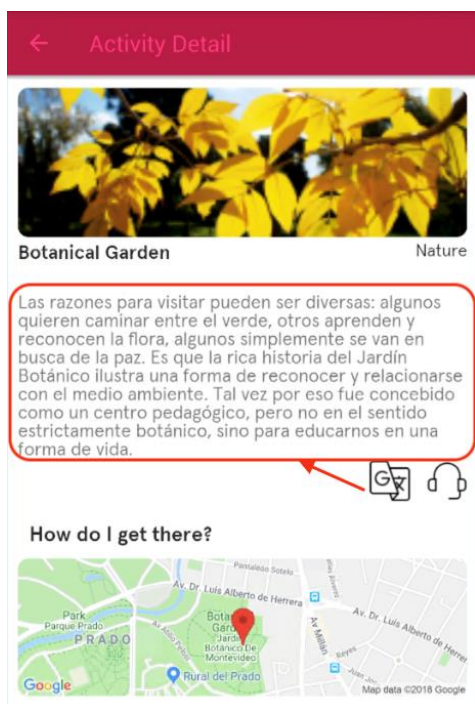
Si el simulador está en inglés, cambiarlo a otro idioma para poder ejecutar las pruebas.

Tip: para cambiar el idioma del emulador de Android ir a Settings > System > Languages & input > tap en la opción Languages > agregar el nuevo idioma, por ejemplo Español (Estados Unidos) y posicionarlo en el primer lugar de la lista.

TRADUCCIÓN AUTOMÁTICA DE TEXTO

Implemente la traducción de la descripción en el panel para SD ActivityDetail (luego puede repetirlo en el web panel análogo ViewActivityDetail).

Una vez programado el evento, esperamos ver el texto traducido como se ve a continuación:



El servicio que vamos a utilizar para traducción es el de GeneXusAI.Text.Translate. Para esto debemos configurar el Provider con Type = ProviderType.IBM y la siguiente pareja de clave-valor:

- Key: PropertyKey.Key; Value: euY0twBoSB4qs8jDj4gvuxVlWt9soLHOKu5O8lYv5o3p

Tip: Recuerde que esto se hace en un procedimiento. Se creó un folder AI para que lo cree allí. Y se hace en base al SDT predefinido, que viene dentro del módulo GeneXusAI/Configuration:

Name	Type	Description	Is Collection
Provider		Provider runtime configuration	<input type="checkbox"/>
Name	VarChar(40)	Provider name which identifies the ...	<input type="checkbox"/>
Type	ProviderType, GeneXusAI.Configur...	Provider type	<input type="checkbox"/>
Properties		Provider configuration specific pro...	<input checked="" type="checkbox"/>
Property			
Key	VarChar(64)	Property key based on PropertyKe...	<input type="checkbox"/>
Value	VarChar(128)	Property value	<input type="checkbox"/>

Tip: el idioma de origen siempre será inglés, pero si se desea, se puede incorporar también la función de GeneXusAI.Text.DetectLanguage.

Tip: el idioma de destino es el del dispositivo, el cual se puede obtener usando ClientInformation.Language en SD (que luego se debe convertir a los códigos de lenguaje usados por GeneXusAI).

Nota: En Web es más difícil obtener el idioma del cliente, por lo que se puede dejar fijo el idioma de destino.

SOLUCIÓN

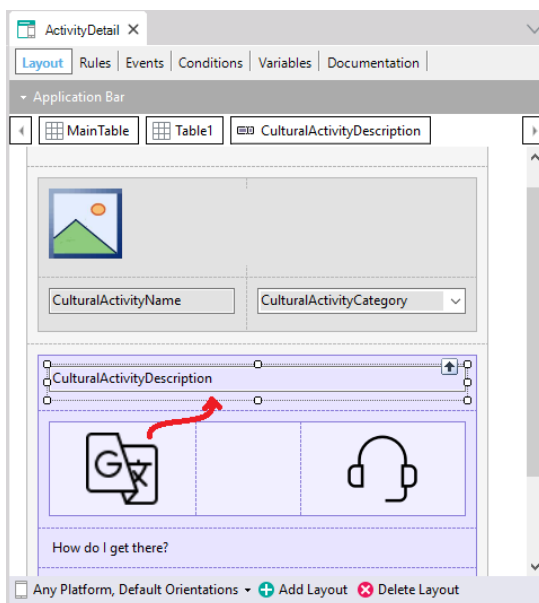
Necesitaremos un procedimiento al que llamaremos **TranslationProvider** para configurar el proveedor.

```
// Rules
parm(out:&provider);

// Source
&provider.Name = !'translate'
&provider.Type = ProviderType.IBM
&prop = new()
&prop.Key = PropertyKey.Key
&prop.Value = !'euY0twBoSB4qs8jDj4gvuxVlWt9soLH0Ku5081Yv5o3p'
&provider.Properties.Add(&prop)

// Variables
&provider : Provider data type
&prop: Provider.Property data type
```

Luego vamos al panel ActivityDetail:



Debemos primero cambiar en pantalla el atributo **CulturalActivityDescription** por una variable (para poder cambiarla por el texto traducido) que cargamos en el evento **Refresh**.

```
Event Refresh
    &CulturalActivityDescription = CulturalActivityDescription
EndEvent
```

Para la traducción del texto, debemos programar el evento **ImageTranslate.Tap**:

```
Event ImageTranslate.Tap
```

```

composite
    &provider = TranslationProvider()
    &languageStr = ClientInformation.Language
    &languageStr = substr(&languageStr, 1, 2)
    &language.FromString(&languageStr)
    &CulturalActivityDescription = Translate(&CulturalActivityDescription,
Language.English, &language, &provider, &messages)
endcomposite

```

Endevent

Donde ClientInformation es el EO que viene por defecto con el módulo GeneXUs, submódulo Client. Si observa el tipo de datos de ClientInformation.Language verá que es C(20), por lo que el tipo de datos de la variable &languageStr tendrá que ser de ese tipo.

Y el tipo de datos devuelto por el proc Translate es Text, y el de &language es Language y &messages es del tipo Messages.

LECTURA DE TEXTO EN VOZ ALTA

Ahora queremos implementar la traducción de la descripción de la actividad cultural a audio, y reproducirlo en el dispositivo en el momento. Recuerde que para reproducir un audio contamos con el EO Audio.

El servicio que vamos a utilizar en esta parte es el de GeneXusAI.Audio.TextToSpeech. Para esto debemos configurar el Provider con Type = ProviderType.IBM y el par clave-valor:

- Key: Property.Key
- Value: pedir este valor al instructor

SOLUCIÓN

Para usar la función de text to speech, debemos implementar el evento **ImageSpeak.Tap**:

```

Event ImageSpeak.Tap
    composite
        &provider = TextToSpeechProvider()
        &localeStr = ClientInformation.Language
        &localeStr = substr(&localeStr, 1, 5)
        &locale.FromString(&localeStr)
        &audio = TextToSpeech(&CulturalActivityDescription, VoiceType.Female,
&locale, &provider, &messages)
        Audio.PlayBackground(&audio)
    endcomposite

```

Endevent

Donde TextToSpeechProvider es un Procedure con el siguiente código:

```
// Rules
```

```

parm(out:&provider);

// Source
&provider.Name = !'textToSpeech'
&provider.Type = ProviderType.IBM

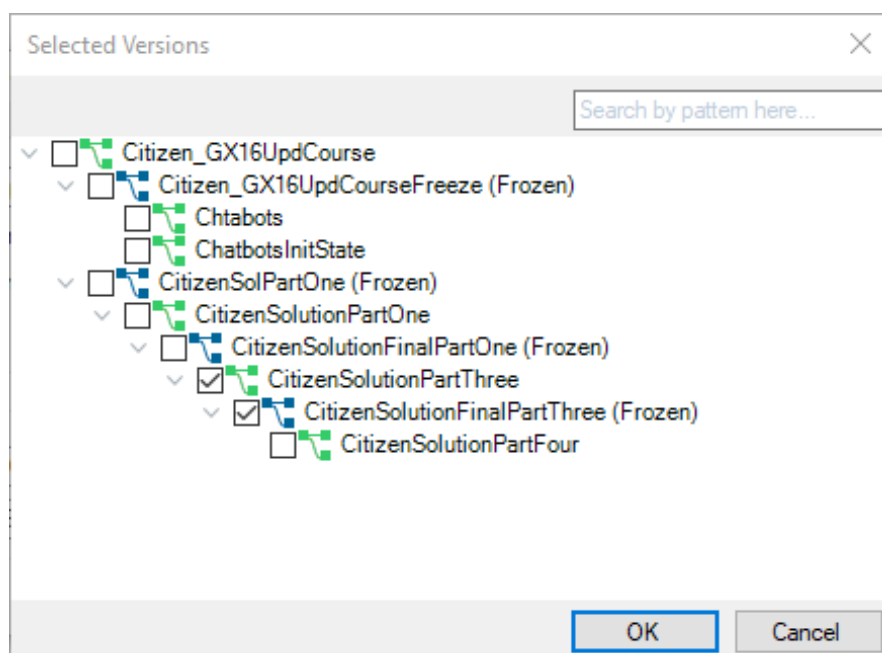
&prop = new()
&prop.Key = PropertyKey.Key
&prop.Value = !'ApiKey'
&provider.Properties.Add(&prop)

```

Donde ApiKey tiene que ser la clave que le pasó el instructor.

KB SOLUCIÓN

Podrá descargar de GeneXus Server la KB solución de este práctico para comparar resultados. Es la versión de la KB de nombre CitizenSolutionPartThree.



Corrobore que en el Data Store “ServiceDS1 (Service)” la propiedad de conexión Server name haya quedado así: [https://services.odata.org/V4/\(S\(40gwjqclhjmuyfayfnplov0o\)\)/TripPinServiceRW/](https://services.odata.org/V4/(S(40gwjqclhjmuyfayfnplov0o))/TripPinServiceRW/)

Y no vacía. Si está vacía, copie el valor anterior.

Se prendió la propiedad “Data provider” de la transacción “Flight” para poblar la tabla con datos iniciales, así no tiene que hacer nada más que ejecutar para probar las funcionalidades.

Le conviene actualizar el módulo GeneXusAI antes de hacer el build. Recuerde que esto se hace a través de Knowledge Manager/Manage Module References.