

Practical update course

GeneXus™ 16

PART 3

September 2018

Copyright ©GeneXus S.A. 1988-2018.

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.

TABLE OF CONTENTS

TABLE OF CONTENTS..... 2

OBJECTIVE..... 3

GETTING STARTED..... 3

ODATA 4

Importing services.....4

Possible solution4

Using the imported services: flight, airline, airport management8

Possible solution9

Using the imported services: LIST OF FLIGHTS.....12

Solution: 14

GENEXUS AI 16

Previous steps.....17

Adding a reference to the GeneXusAI module 18

Changing the language of the simulator 18

Automatic Text translation.....18

Solution 20

Reading text out loud.....21

Solution 21

SOLUTION KB.....22

OBJECTIVE

This practical course will address the topics of OData and Artificial Intelligence (AI) to develop state-of-the-art applications while improving integration with external services.

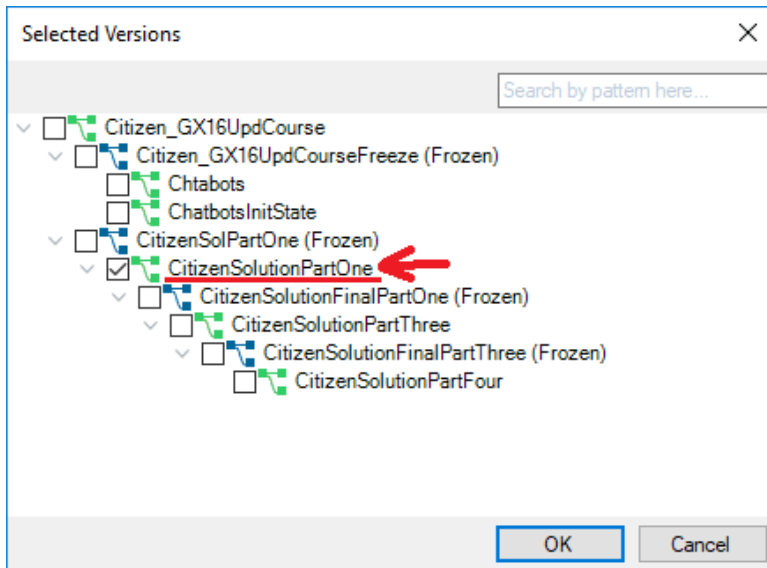
The application that you will work on is a simplification of an app for the municipality of a city, which offers a frontend (Web/SD) so that citizens, using their user ID, can make claims (for example, for fallen trees, traffic lights that do not work, improperly parked cars, etc.), or carry out formalities (for example, to obtain a driver's license, refinance a debt with the municipality, install advertising elements, etc.), booking a time to be assisted by municipal staff. The frontend also shows the various cultural activities offered by the city. In addition, a web backend is available for certain municipal officials to manage the data and view statistics.

To improve tourism from the city, integration will be made with TripPin services that will allow keeping a record of flights taken by users to different destinations.

In addition, the description of cultural activities can be translated into the language of the device and, to improve accessibility those descriptions can also be played back as audio.

GETTING STARTED

To get started, you will use the KB that you worked on in the Design Systems and UX practice exercises, as it was left at the end of the practice. If you've lost it or can't find it, create another from <http://samples.genexusserver.com/v16> (KB called Citizen_GX16Course). Select the version called CitizenSolutionPartOne.



Note that in the KB Explorer the folder GeneXus/Web/Backend_ODATA has been added for you to place the objects you create for this part of the practical course. Also, a folder called AI has been added to create the procedures required to connect with the AI provider that will be indicated.

ODATA

You have to add two new features for the application backend, consuming OData from TripPin. The objective is to:

1. Keep track of outbound flights, managing flight, airport and airline information.
2. List all information related to flights to a certain airport.

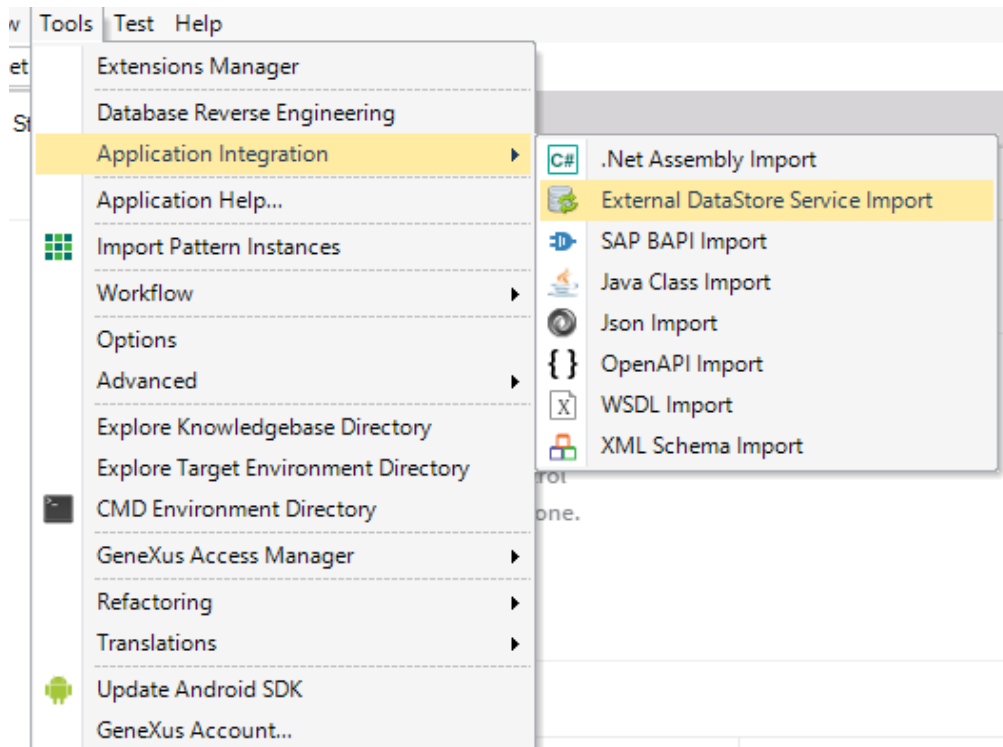
IMPORTING SERVICES

To implement the new features you will need to import some services from TripPin.

The entities whose implementation you need to use are Airlines and Airports. Flights will be an entity of your own KB, which you will have to create later.

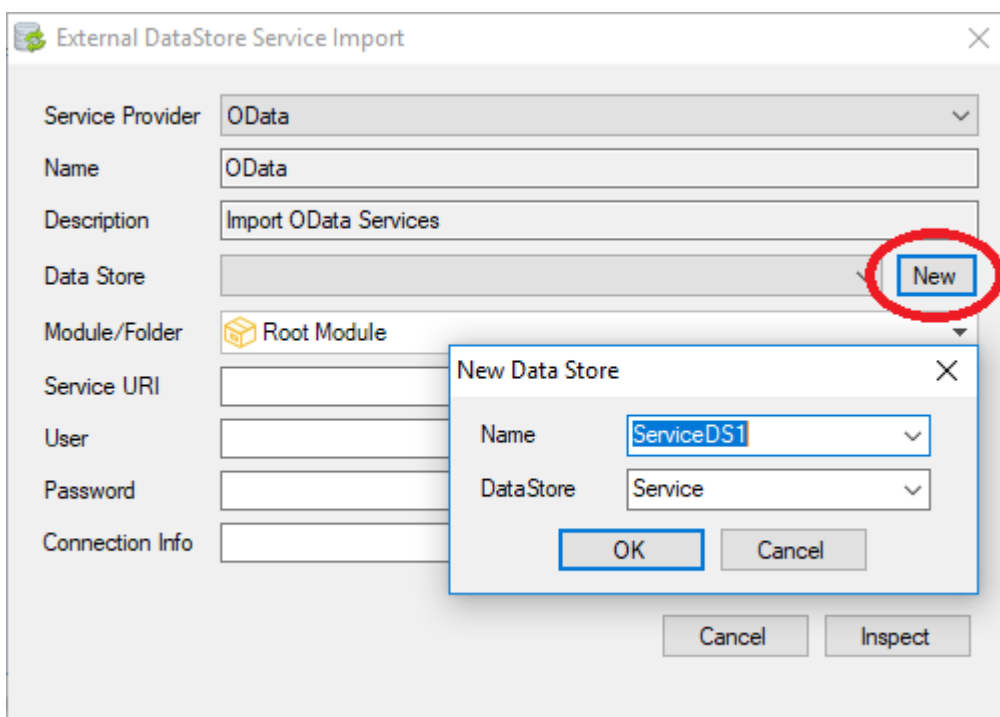
POSSIBLE SOLUTION

Import from TripPin what you need to implement the new features. To do so:



A new window will be opened to enter the data required to consume the service.

First, create a new Data Store of Service type from the New button located to the right of the DataStore option:



Next, complete the following fields:

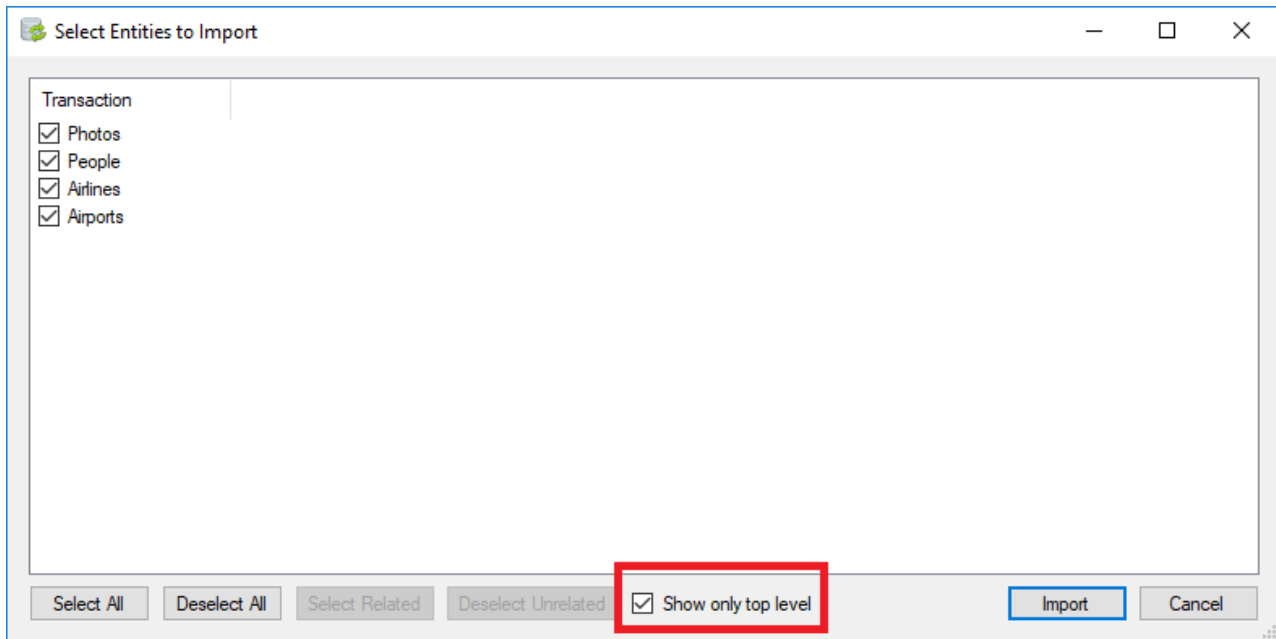
- Service URI: Enter the address of the service to consume. In this case, use the TripPin service: [https://services.odata.org/V4/\(S\(40gwcqlhjmuyfayfnplov0o\)\)/TripPinServiceRW/](https://services.odata.org/V4/(S(40gwcqlhjmuyfayfnplov0o))/TripPinServiceRW/)
- User: Enter the username, if necessary. In this case, it is not necessary so leave it empty.
- Password: User's password. In this case, it is not necessary so leave it empty.
- Connection Info: Additional information for the connection. In this case it is necessary, so enter "filter_strings=n" to indicate that conditions such as <, >, <=, >= are not accepted with String type data.

The screenshot shows a dialog box titled "External DataStore Service Import". It has a close button (X) in the top right corner. The fields are as follows:

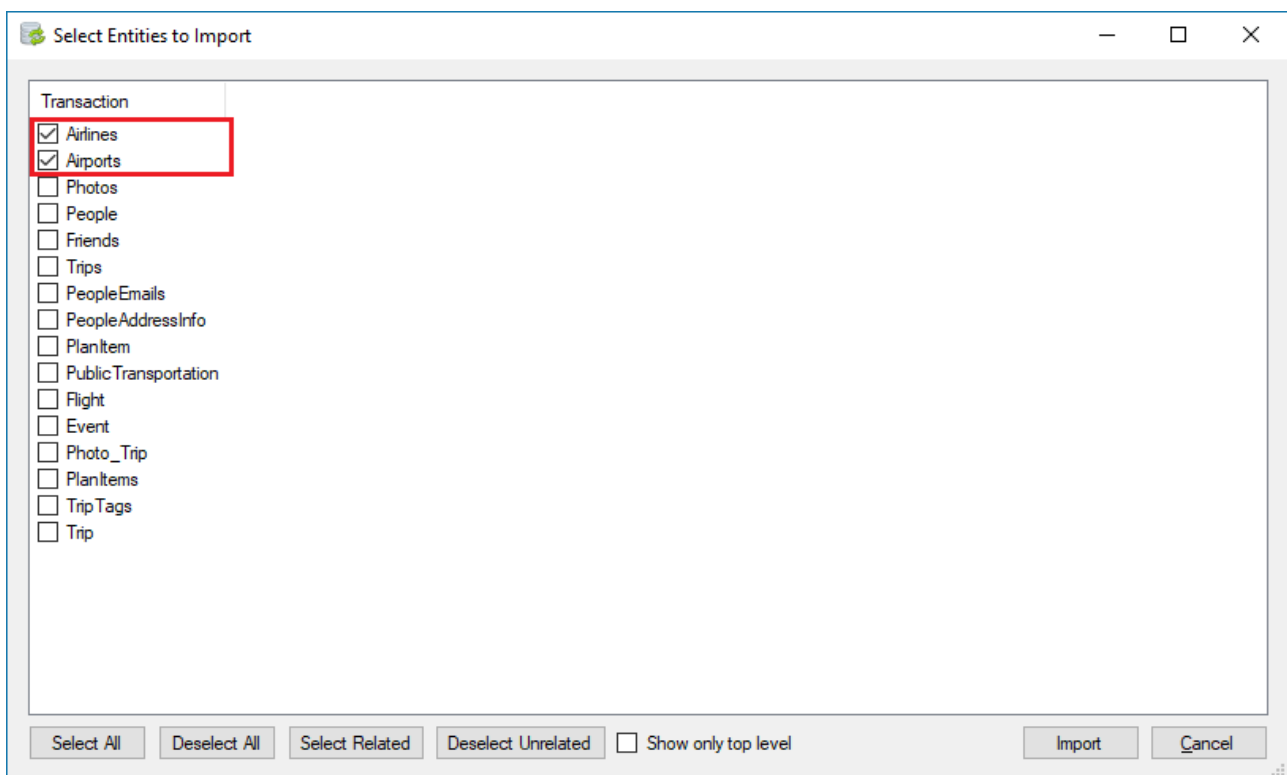
- Service Provider: OData (dropdown menu)
- Name: OData (text input)
- Description: Import OData Services (text input)
- Data Store: ServiceDS1 (dropdown menu) with a "New" button to its right
- Module/Folder: Root Module (dropdown menu)
- Service URI: https://services.odata.org/V4/(S(40gwcqlhjmuyfayfnplov0o))/TripPinServiceRW/ (text input)
- User: (empty text input)
- Password: (empty text input)
- Connection Info: filter_strings=n (text input)

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

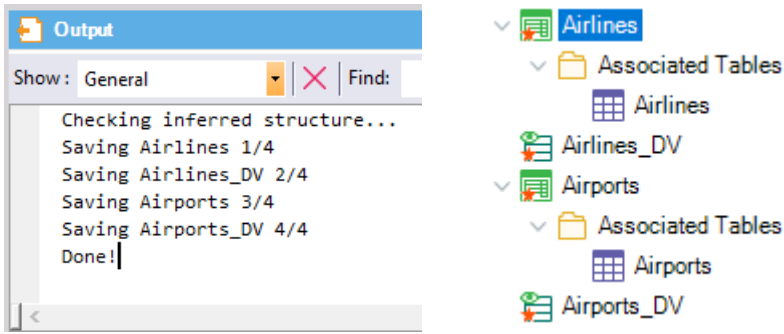
Once you complete the fields, click on Inspect. This will open a new window where the entities to be imported from the service will be displayed. You will only need Airports and Airlines. To this end, clear the checkbox "Show only top level":



And leave only the options Airlines and Airports selected.



Next, click on Import. This will import the entities Airports and Airlines as transactions, with the associated data views:



The transactions' structure is as follows:

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

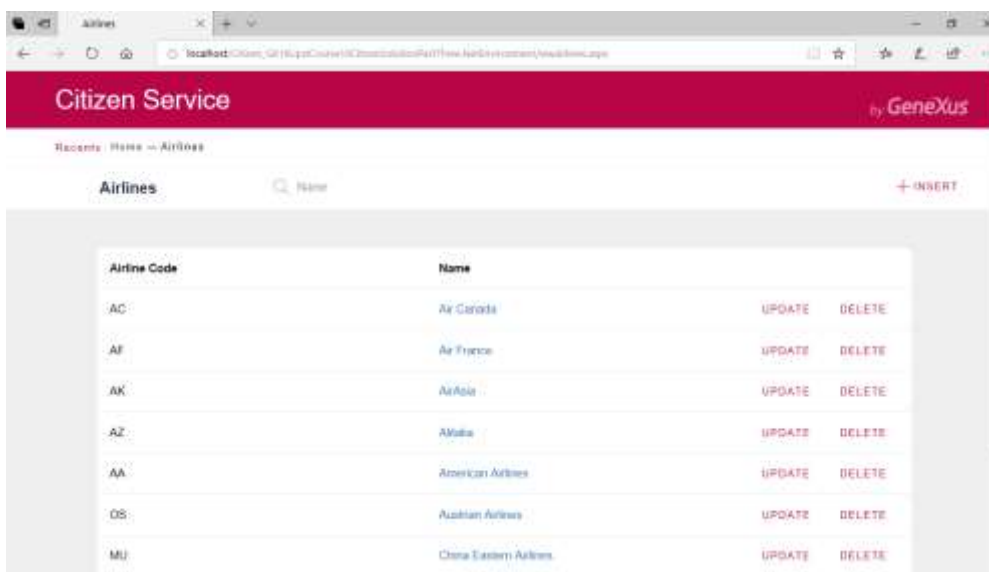
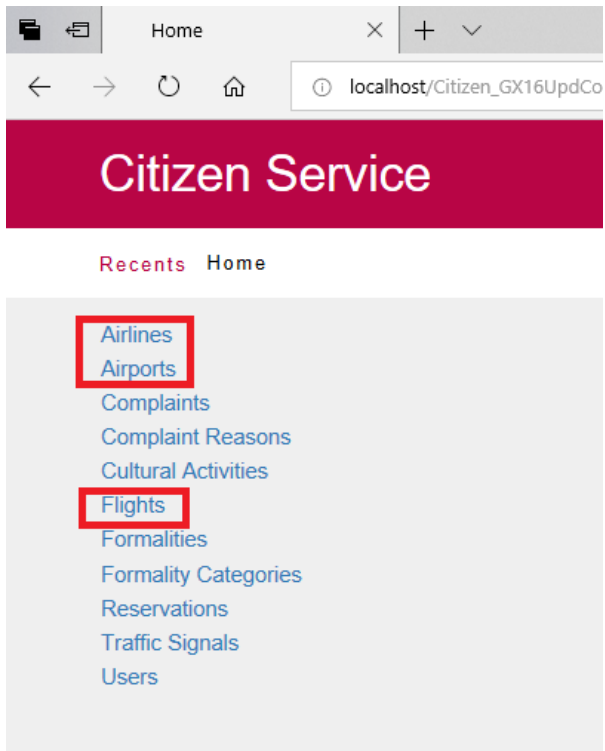
Name	Type
Airports	Airports
AirportsIcaoCode	Character(40)
AirportsName	Character(40)
AirportsIataCode	Character(40)
AirportsLocation_Address	Character(40)
AirportsLocation_City_CountryRegion	Character(40)
AirportsLocation_City_Name	Character(40)
AirportsLocation_City_Region	Character(40)
AirportsLocation_Loc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

Moving the entities to the folder GeneXus/Web/Backend_ODATA is recommended.

USING THE IMPORTED SERVICES: FLIGHT, AIRLINE, AIRPORT MANAGEMENT

Once the data has been imported, use it to implement the functionalities requested.

First of all, you need to be able to manage (CRUD) airline and flight data, adding them to the backend.

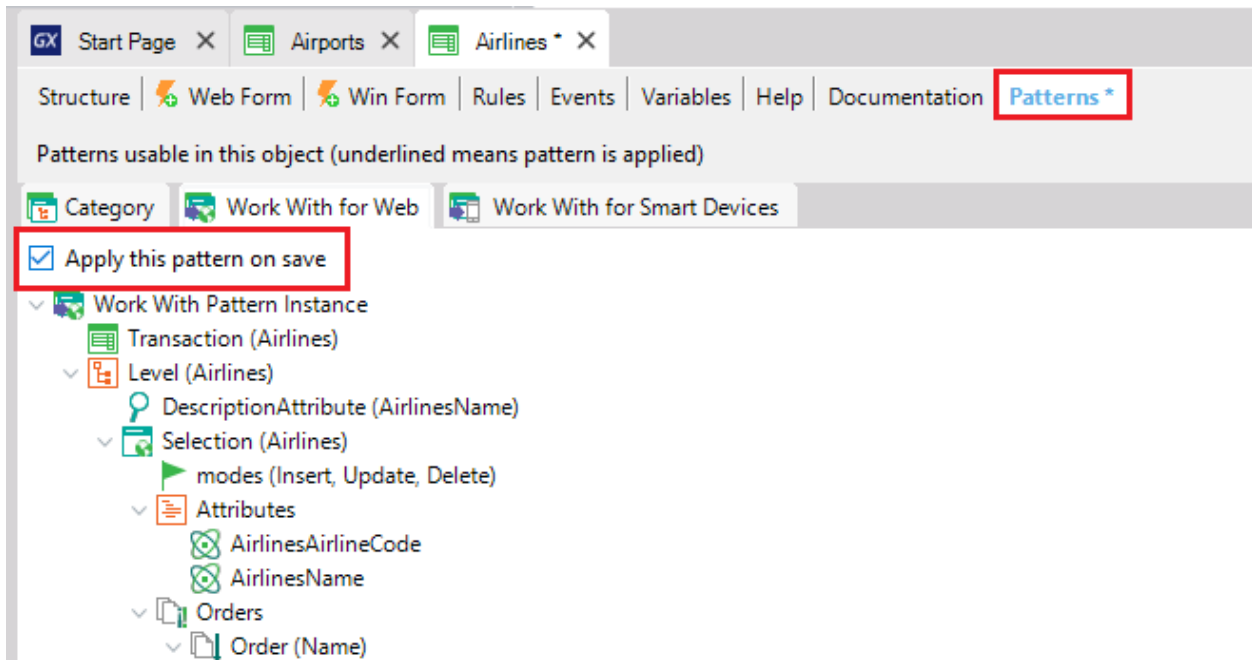


Note: It is not possible to insert or delete airports, because the service provided doesn't allow it.

Once this has been implemented, try changing the name of an airport, for example, and enter flights.

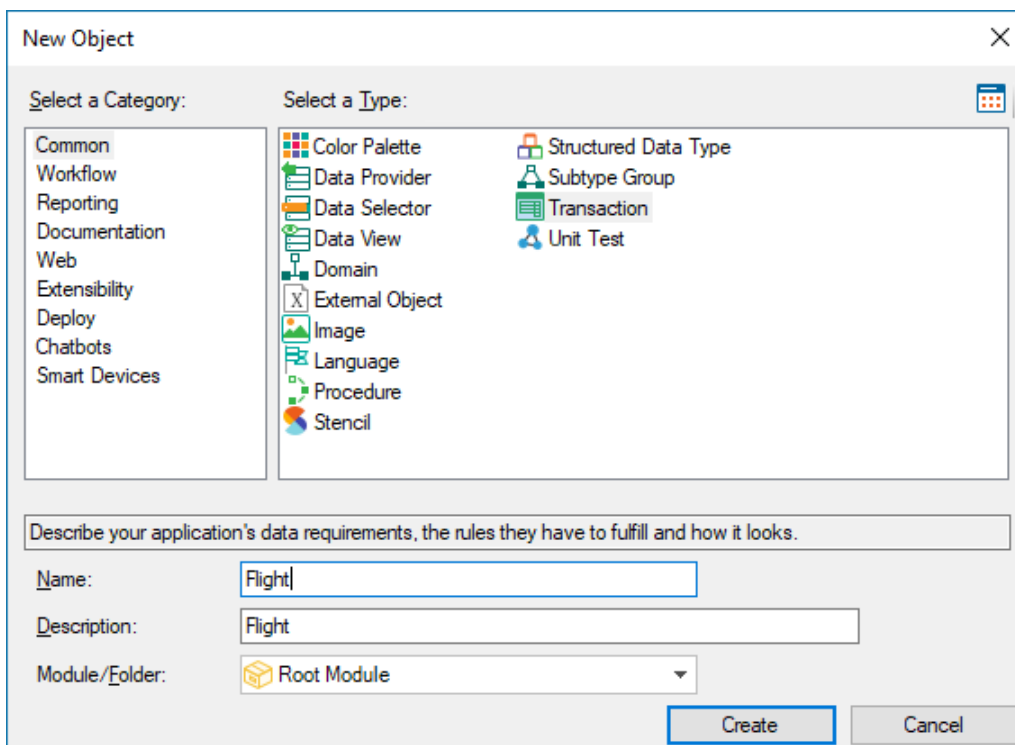
POSSIBLE SOLUTION

Apply the Work With for Web pattern to the Airports and Airlines transaction.



In this way, they will be automatically added to the backend.

Next, create a Flight transaction that will keep a record of flights and communicate with Airports and Airlines:



The structure of this transaction will be as follows:

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

Also, apply the Work With Web pattern to this transaction.

You have already implemented the flight management feature.

Run and view the data to enter more information. If you wish, you can use the following sample data.

Flight 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

Flight 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

Flight 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

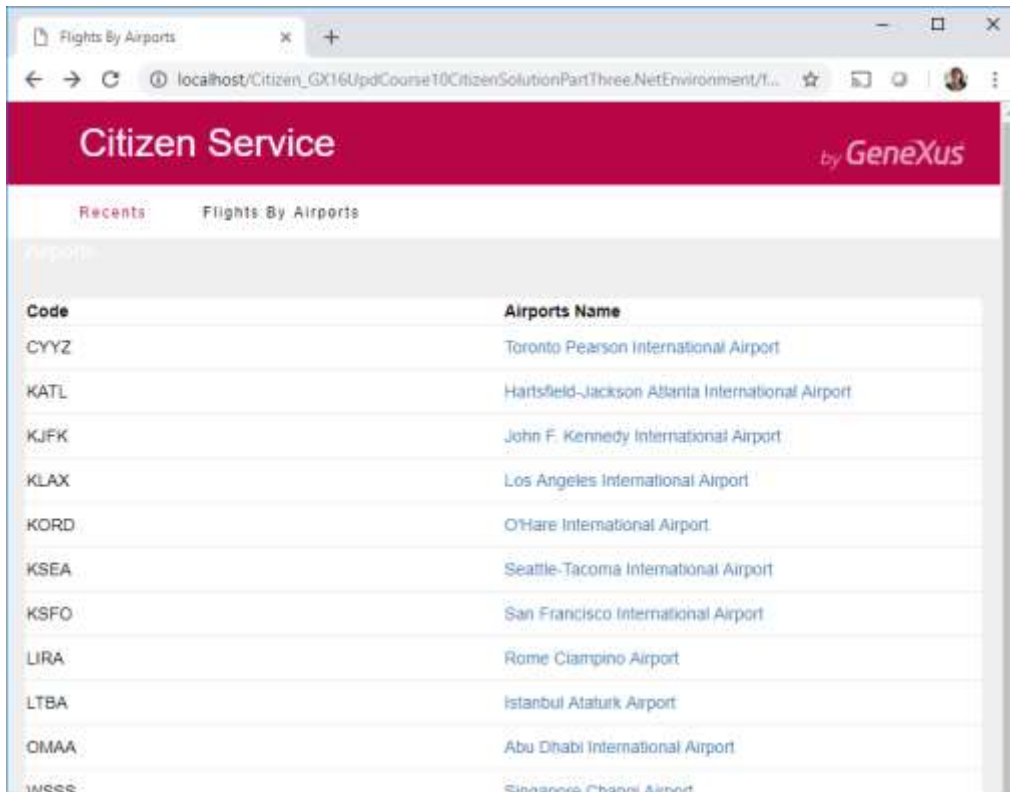
Recents | Airlines — Airports — Home — Flight — Flights

Flights
🔍
📄
+ INSERT

Id	Airports Icao Code	Airports Name	Airlines Airline Code	Airlines Name	UPDATE	DELETE
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

USING THE IMPORTED SERVICES: LIST OF FLIGHTS

You need to show in a panel all the possible destination airports:









The screenshot shows a web browser window with the following content:

- Browser tab: Flights By Airports
- Address bar: localhost/Citizen_GX16UpdCourse10CitizenSolutionPartThree.NetEnvironment/f...
- Page Header: Citizen Service by GeneXus
- Navigation: Recents, Flights By Airports
- Table:

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

And, after selecting one airport, issue a PDF file containing all the flights to that airport. For each flight you want to list passenger information (Citizen users):

Toronto Pearson International Airport			
Flight id: 1		Airline: Air Canada	
Seat	Passenger		
1 A	Luciano Silveira	08/05/76	
1 B	Mary Smith	07/30/69	
2 A	Martin Torral	02/01/89	
2 B	Ann Bert	04/11/75	
Flight id: 3		Airline: Air France	
Seat	Passenger		
1 A	Brian Goals	12/01/60	
1 B	John Palera	07/29/73	
2 A	Eleanor Johnson	03/12/90	

The FlightsByAirports web panel and ListAirportFlight procedure have already been created for you to change what you need and save time. You can find them in the folder GeneXus/Web/Backend_ODATA.

SOLUTION:

Use a procedure (it is already created and called ListAirportFlight) to implement the list, which will receive as parameter the airport identifier, in order to list the flights to it.

In the Rules section you will find:

```
parm(in: &AirportCode);
```

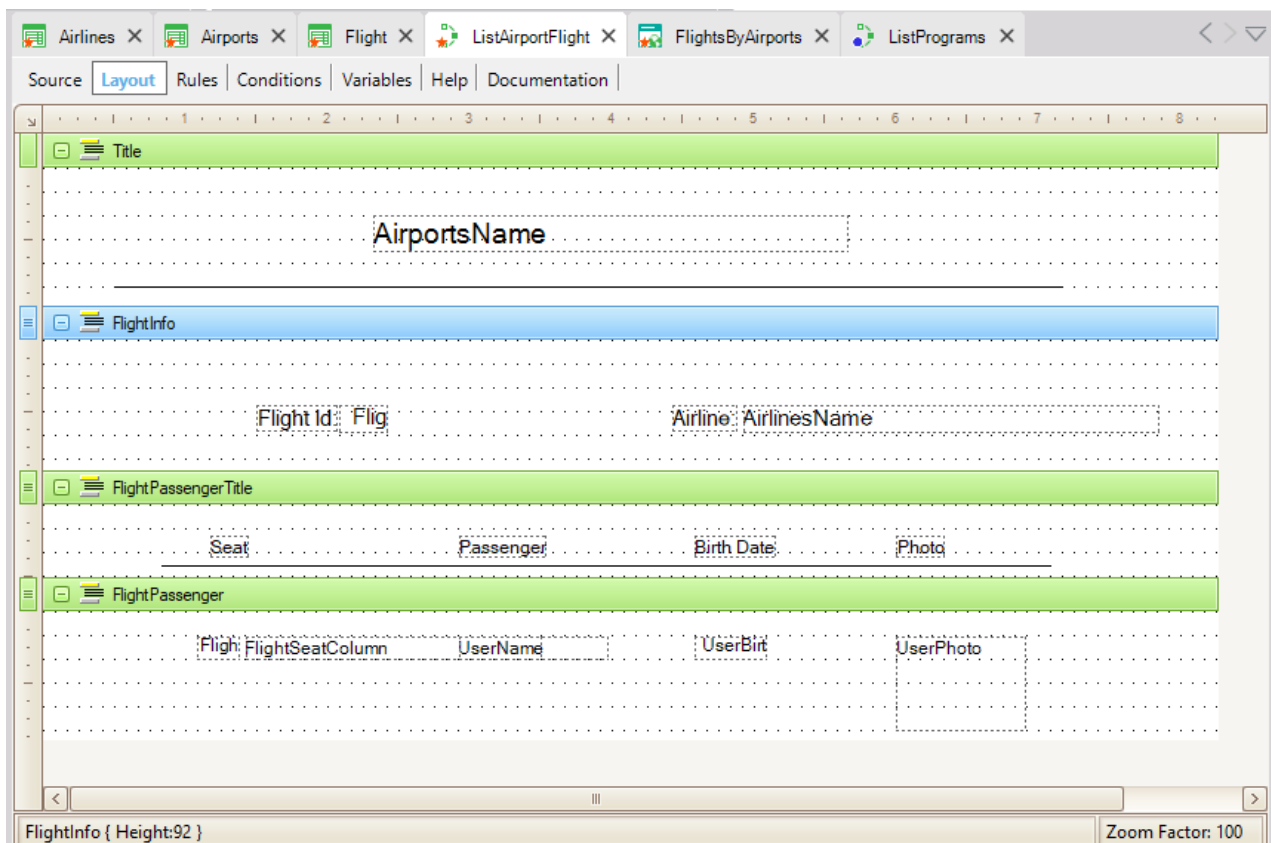
```
Output_file("AirportFlightsList.pdf","pdf");
```

Where `&AirportCode` is a variable based on the data type of the attribute AirlinesAirlineCode (Character(40)). Remember that `Output_file` is used to define the list output (in this case, a PDF document).

Also, remember that the Call protocol property had to be changed to HTTP:

Properties	
Filter	
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

The layout should look as shown below:



And the Source section should look as follows:

```

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

```

Invoke this procedure from the Web Panel FlightsByAirports, where you insert a Grid with the attributes AirportsIcaoCode and AirportsName.

In the Events section:

```

Event AirportsName.Click
  ListAirportFlight( AirportsIcaoCode )
endevent

```

To connect this panel to the backend (to the Home web panel), edit the ListPrograms procedure and add the following code:

```

&name = !" FlightsByAirports"
&description = "Flights By Airports"
&link = FlightsByAirports.Link()

Do 'AddProgram'

```

Run the application.

GENEXUS AI

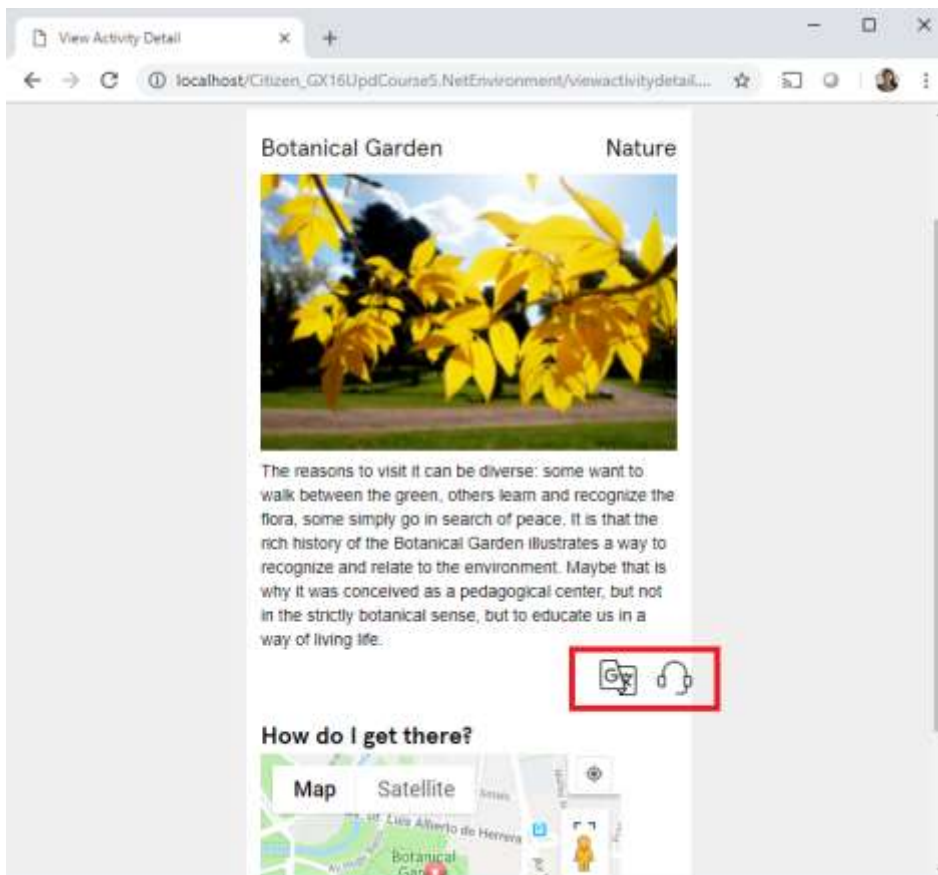
You need to add two new functions to the application in order to:

1. Translate the description of a cultural activity to the language of the device, and
2. Have the application read this description out loud.

To do so, in the ActivityDetail panel of the SD frontend you have two buttons where the functionality will be added:



You want the same for the similar panel ViewActivityDetail, in the web frontend:



PREVIOUS STEPS

ADDING A REFERENCE TO THE GENEXUSAI MODULE

Before you begin to add the functionality, you must add the reference to the GeneXusAI module.

Do this from the option Knowledge Manager > Manage Module References:



CHANGING THE LANGUAGE OF THE SIMULATOR

If the simulator is in English, change it to another language to run the tests.

Tip: To change the language of the Android simulator, go to Settings > System > Languages & input > tap on the Languages option. Add the new language, for example, Spanish (United States) and place it at the top of the list.

AUTOMATIC TEXT TRANSLATION

Implement the translation of the description in the SD panel ActivityDetail (you can later repeat it in the similar web panel ViewActivityDetail).

Once the event has been programmed, the translated text should be displayed as shown below:



The service that you will use for the translation is GeneXusAI.Text.Translate. To this end, set the Provider as Type = ProviderType.IBM and with the following key-value pair:

- Key: PropertyKey.Key; Value: euY0twBoSB4qs8jDj4gvuxVlWt9soLHOKu5O8lYv5o3p

Tip: Remember that this is done in a procedure. An AI folder has been created to this end. This is done based on the predefined SDT that is included inside the module 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: The source language will always be English. However, you may also incorporate the function GeneXusAI.Text.DetectLanguage.

Tip: The target language is that of the device, which can be obtained using ClientInformation.Language in SD (later it must be converted to the language codes used by GeneXusAI).

Note: On the Web it is more difficult to obtain the client language, so the target language can be left unchanged.

SOLUTION

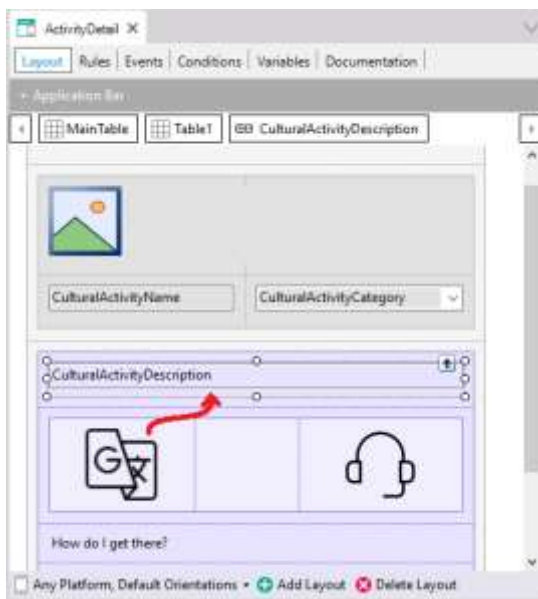
You will need a procedure called `TranslationProvider` to configure the provider.

```
// 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
```

Next, go to the `ActivityDetail` panel:



First, change the `CulturalActivityDescription` attribute on screen for a variable (to be able to change it for the translated text) loaded in the `Refresh` event.

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

For translating the text, program the event `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

Where ClientInformation is the EO included by default in the GeneXus module, Client submodule. Note that the data type ClientInformation.Language is C(20); therefore, the data type of the &languageStr variable will have to be of that type.

The data type returned by the Translate procedure is Text; that of &language is Language and &messages is Messages.

READING TEXT OUT LOUD

Now you need to implement the translation of the cultural activity's description to audio, and play it on the device. Remember that to play audio you have the Audio EO.

The service that you will use in this part is GeneXusAI.Audio.TextToSpeech. To this end, set the Provider as Type = ProviderType.IBM and the following key-value pairs:

- Key: PropertyKey.Username; Value: 9e579369-9b47-4c8e-84ec-122affee5ae1
- Key: PropertyKey.Password; Value: 5BiwVDLGd0H4

SOLUTION

To use the text to speech function, you need to implement the event **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

Where TextToSpeechProvider is a Procedure with the following code:

```

// Rules
parm(out:&provider);

```

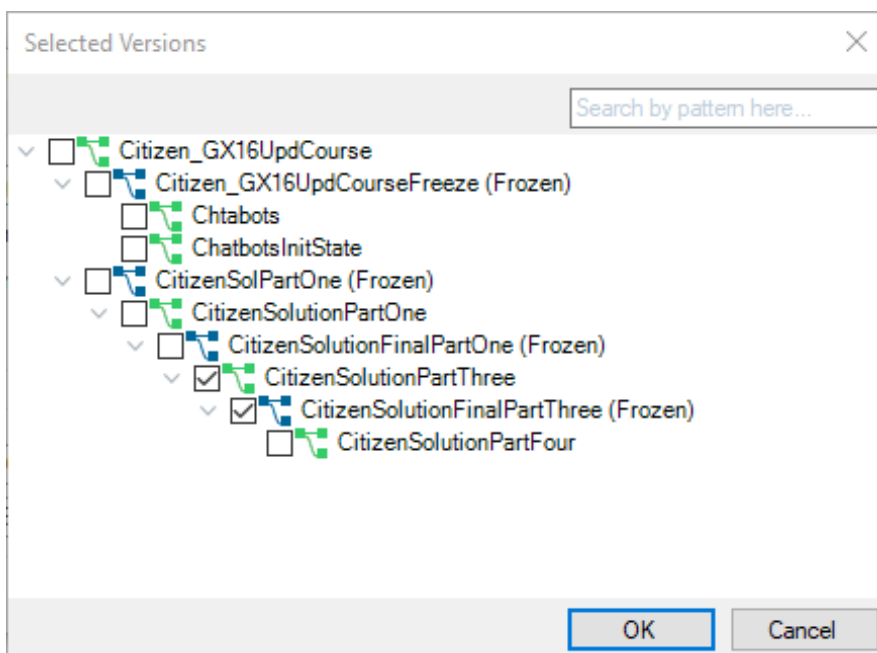
```
// Source
&provider.Name = !'textToSpeech'
&provider.Type = ProviderType.IBM

&prop = new()
&prop.Key = PropertyKey.Username
&prop.Value = !'9e579369-9b47-4c8e-84ec-122affee5ae1'
&provider.Properties.Add(&prop)

&prop = new()
&prop.Key = PropertyKey.Password
&prop.Value = !'5BiwVDLGd0H4'
&provider.Properties.Add(&prop)
```

SOLUTION KB

You can download the KB containing the solution to these practice exercises from GeneXus Server to compare results. This KB is called CitizenSolutionPartThree.



Confirm that in the Data Store “ServiceDS1 (Service)” the Server Name connection property is as follows: [https://services.odata.org/V4/\(S\(40gwjqclhjmuyfayfnplo0o\)\)/TripPinServiceRW/](https://services.odata.org/V4/(S(40gwjqclhjmuyfayfnplo0o))/TripPinServiceRW/)

And that it is not empty. Otherwise, copy the previous value.

The “Data provider” property of the “Flight” transaction has been enabled to populate the table with initial data. In this way, you only have to run it to test the features.

