

Retrieving Data from a SOAP Web Service

Table of Contents

Introduction.....	2
Connect to an Environment	2
Get to know the scenario.....	5
Install the Contacts application	5
Business Case Overview	5
Consume the SOAP Web Service Method.....	6
List Contacts on the screen.....	12
End Lab.....	21

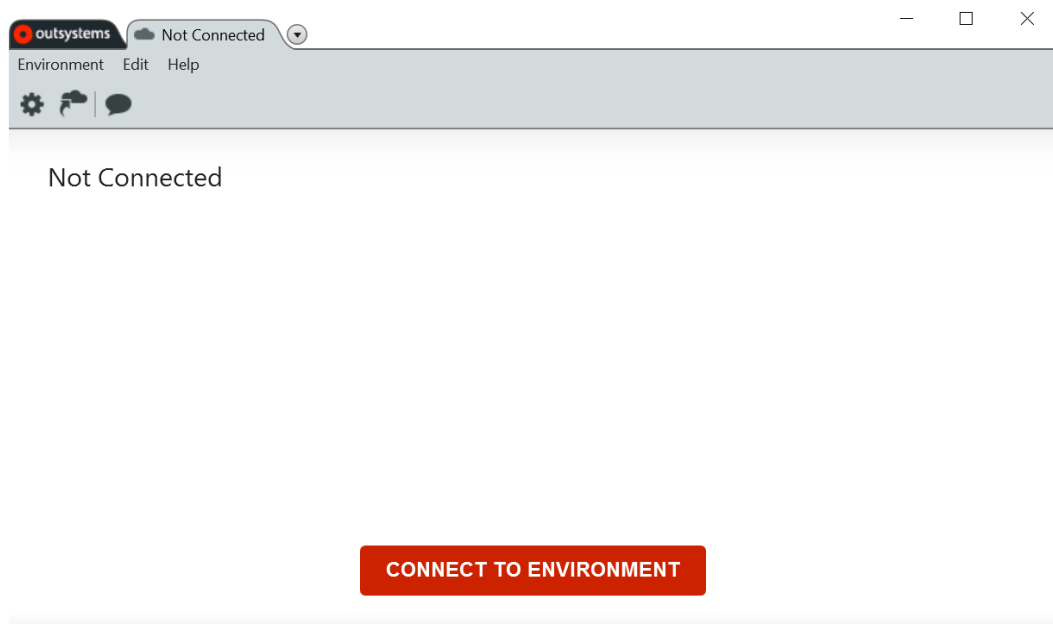
Introduction

In this lab, we are going to integrate with a SOAP Web Service to use data from an external system. We are going to use a SOAP method to read information of the Contacts in the external system and then display that data in our application.

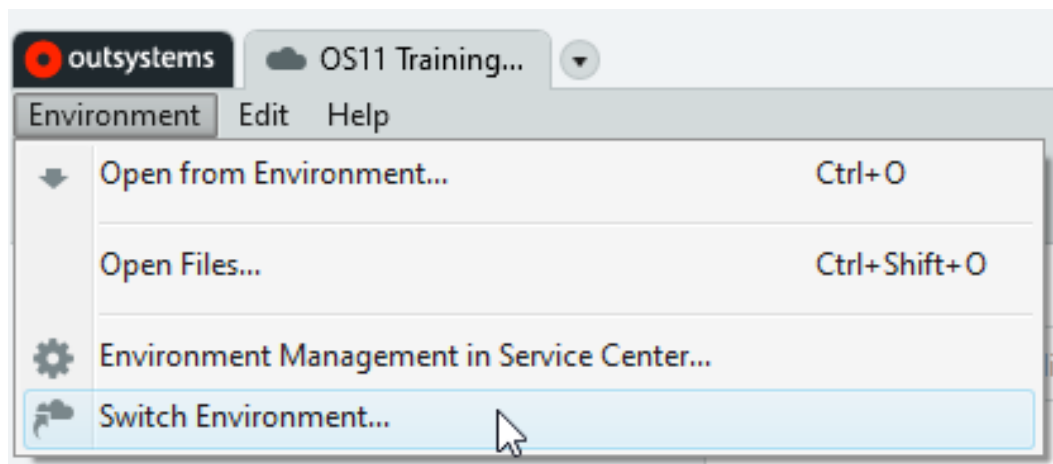
Connect to an Environment

When we open Service Studio for the first time, we will need to connect to an **environment** where the OutSystems platform server generates, optimizes, compiles, and deploys OutSystems applications.

- 1) Open Service Studio and access the **Connect to Environment** dialog. This can be done in two ways.
 - a) If you are not logged in to any environment, click on **Connect to Environment**.

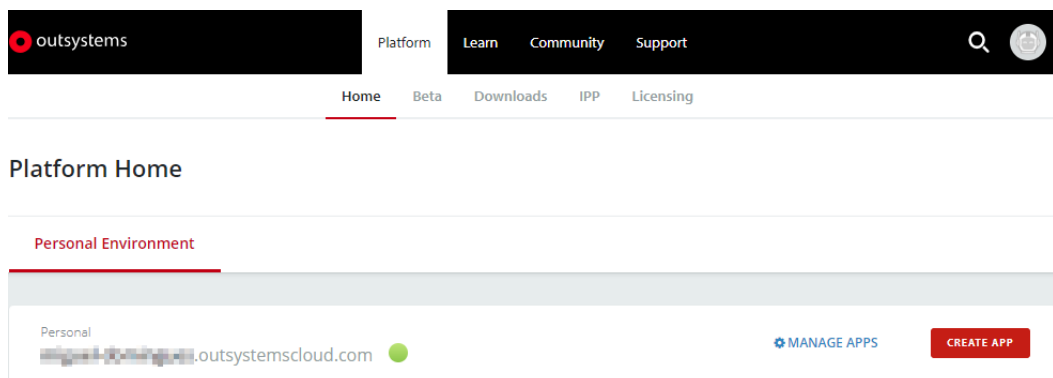


- b) If you are already logged in to an environment, select the **Switch Environment...** option from the Environment menu at the top.

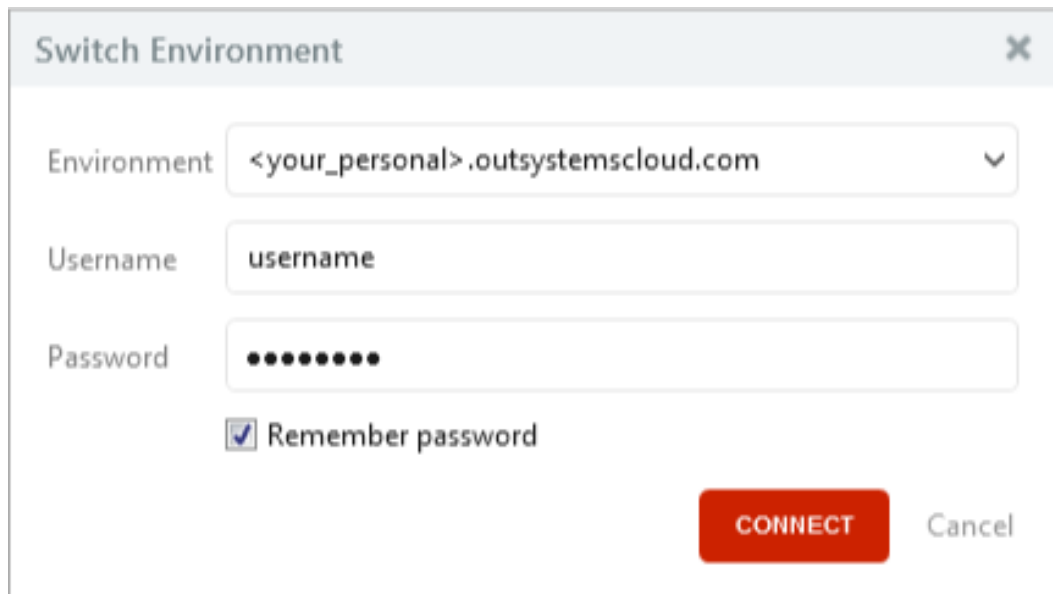


- 2) Connect to your OutSystems personal environment.

- a) If you are using your Personal Environment, you can find its address in the OutSystems [website](#) and log in.
- b) Under the **Platform** tab and then under the **Personal Environment** tab the environment address (or **Server Address**) can be found.



- c) Back in Service Studio, use that Environment and login with your OutSystems community email (username) and password.



The image shows a 'Switch Environment' dialog box with a close button (X) in the top right corner. It contains three input fields: 'Environment' with a dropdown menu showing '<your_personal>.outsystemscloud.com', 'Username' with the text 'username', and 'Password' with masked characters '••••••••'. Below the password field is a checkbox labeled 'Remember password' which is checked. At the bottom right, there is a red 'CONNECT' button and a 'Cancel' button.

Switch Environment

Environment <your_personal>.outsystemscloud.com

Username username

Password ••••••••

☒ Remember password

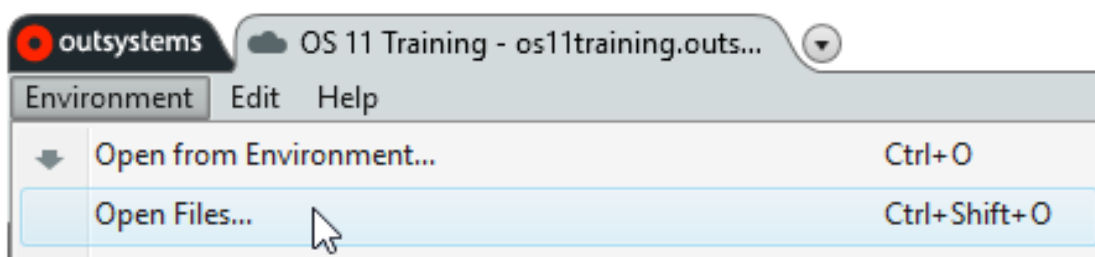
CONNECT Cancel

Get to know the scenario

Install the Contacts application

Open and publish the **SOAP Contacts - Using Data.oap** in your personal environment. The oap file can be found in the Resources folder.

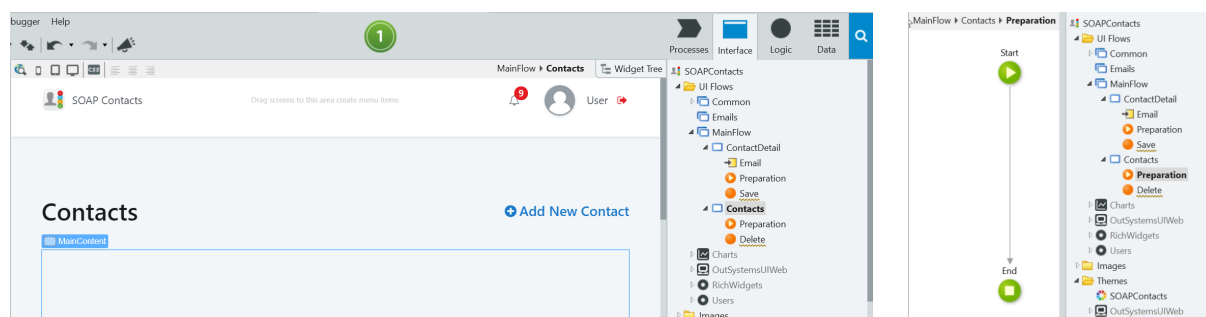
- 1) In the Applications, open the Environment menu and select Open Files...



- 2) In the Open dialog, change the File Type dropdown option to **OutSystems Application Pack (*.oap)** and then open the SOAP Contacts - Using Data.oap.
- 3) Click Proceed when asked.
- 4) Wait for the installation to complete and then proceed.

Business Case Overview

The SOAP Contacts application is a simple application, only containing a couple of empty Screens, Preparation and some Screen Actions, also empty. These elements are where we'll create the logic to get the Contacts and display the list on the screen. This quick start application is simply to speed up the setup part and start right away working with the external SOAP web service.



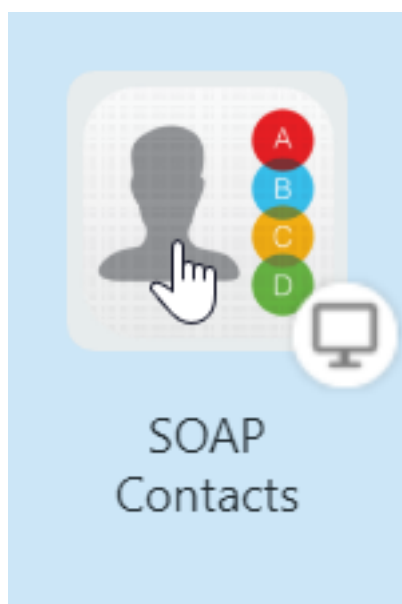
Consume the SOAP Web Service Method

In OutSystems, integration with SOAP Services can be straightforward. OutSystems helps us to generate all the methods and data structures needed to integrate with an external system.

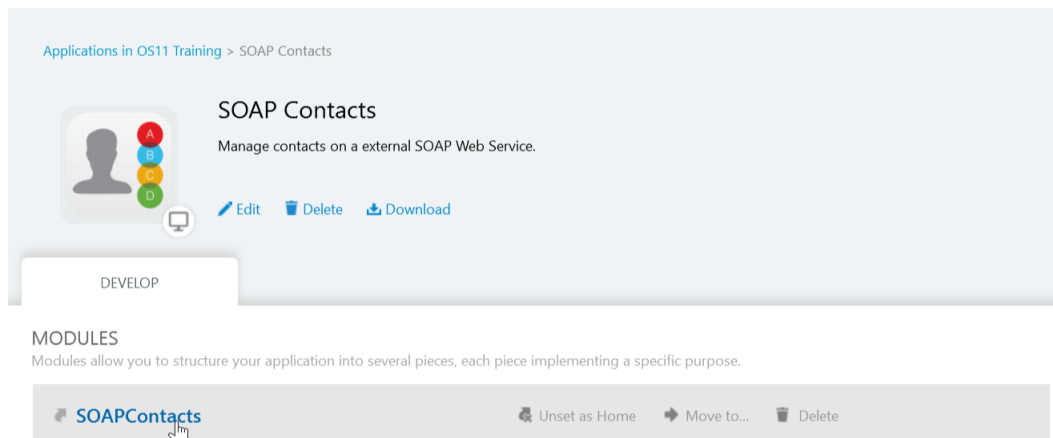
Before you consume any SOAP Web Service it's important to gather all the information you need from the SOAP Web Service documentation. Information such as the expected structures and some examples may be really useful when consuming an external service. In this lab, the documentation for the external SOAP Web Service is available [here](#).

In this section we will create the integration with the external SOAP Web Service. The methods from this Web Service will be used to display data on a Web Screen.

- 1) Open the Contacts module
 - a) In the Applications list, locate the **SOAP Contacts** Web application and open it.



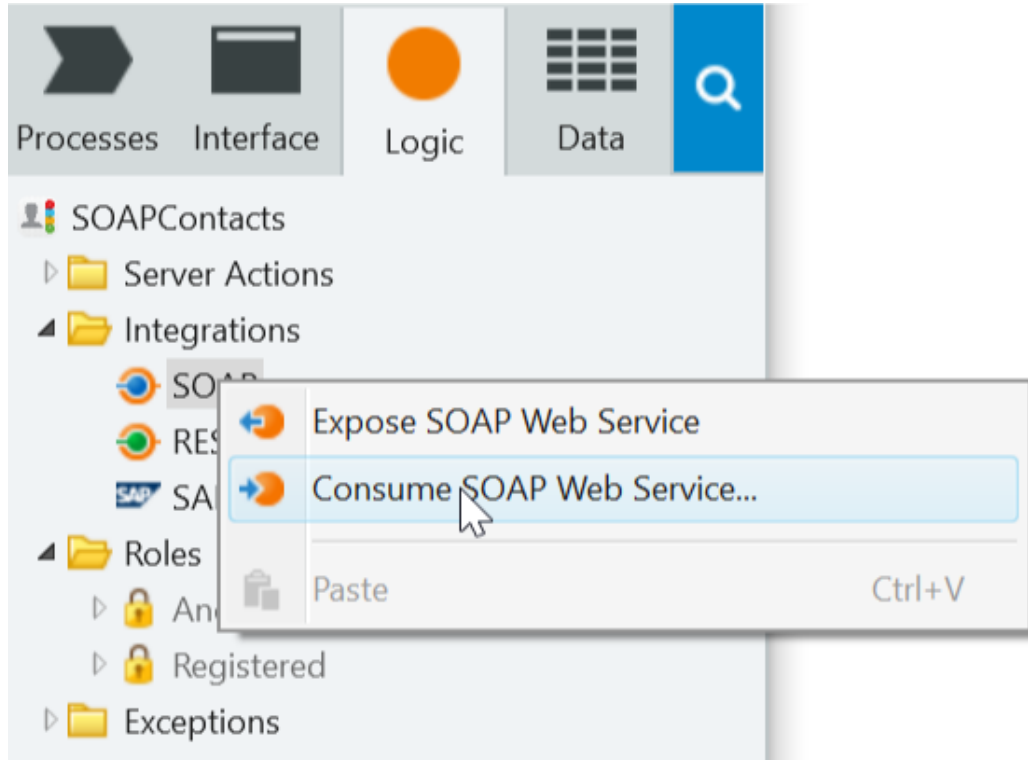
- b) Open the **SOAPContacts** module.



- c) If a dialog appears asking to update references, click **Yes** to open the Manage Dependencies window. There, click on **Refresh All** and then OK.

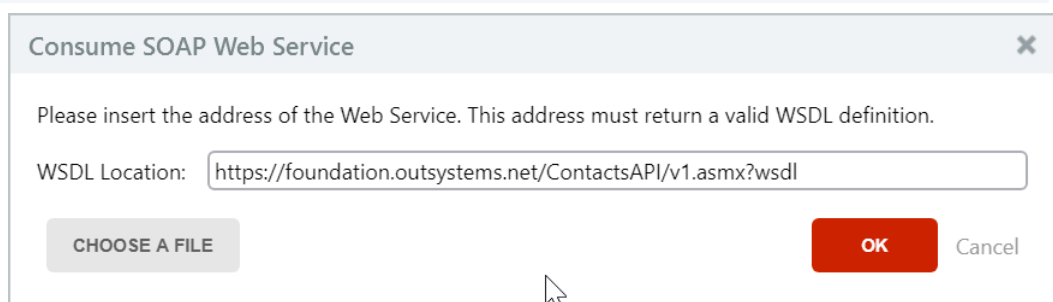


- 3) Consume the SOAP methods of the external SOAP Web Service.
- a) Switch to the Logic tab and, in the Integrations folder, right-click the SOAP integration element and select *Consume SOAP Web Service...*

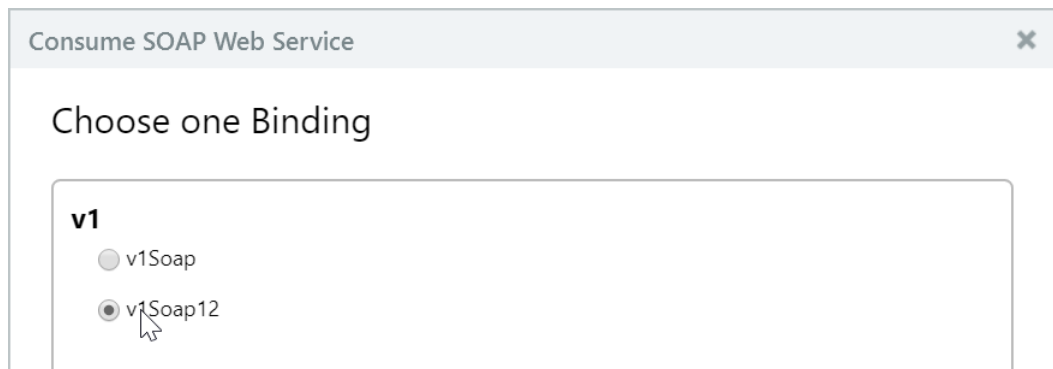


- b) On the new window, set the **WSDL Location** to

```
https://foundation.outsystems.net/ContactsAPI/v1.asmx?wsdl
```

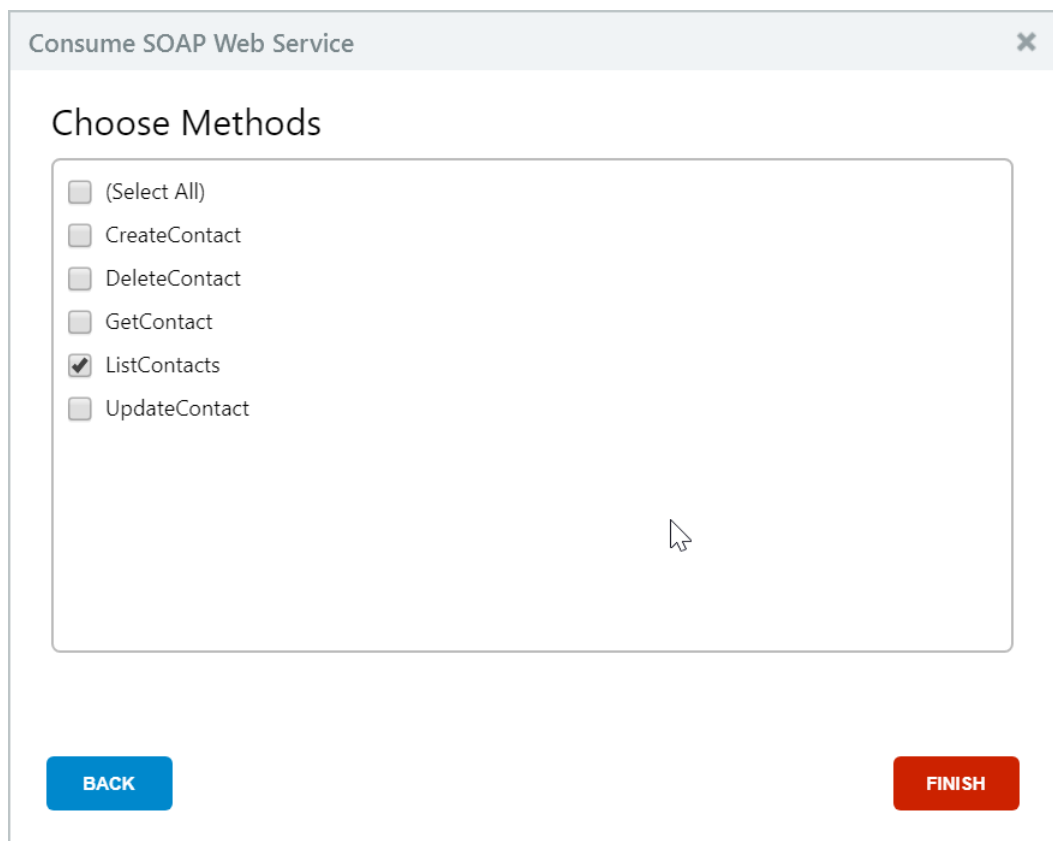


- c) Select the Binding, in this case **v1Soap12**, and click Next.



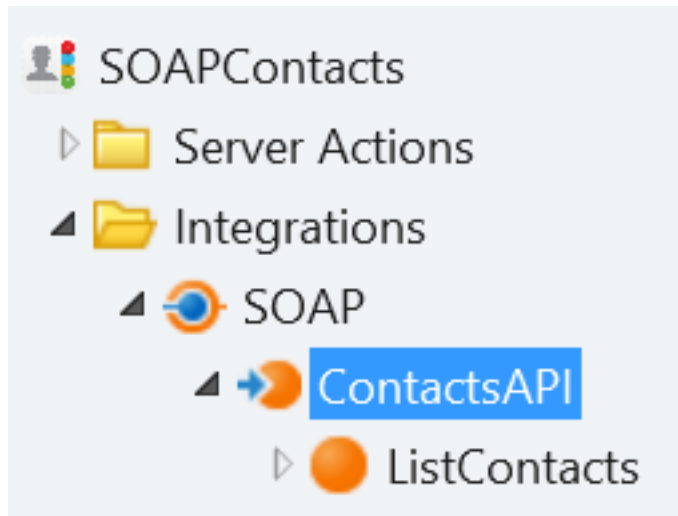
The screenshot shows a dialog box titled "Consume SOAP Web Service" with a close button (X) in the top right corner. The main heading is "Choose one Binding". Below this, there is a section labeled "v1" containing two radio button options: "v1Soap" and "v1Soap12". The "v1Soap12" option is selected, and a mouse cursor is pointing at it.

- d) Select the **ListContacts** method and click Finish.



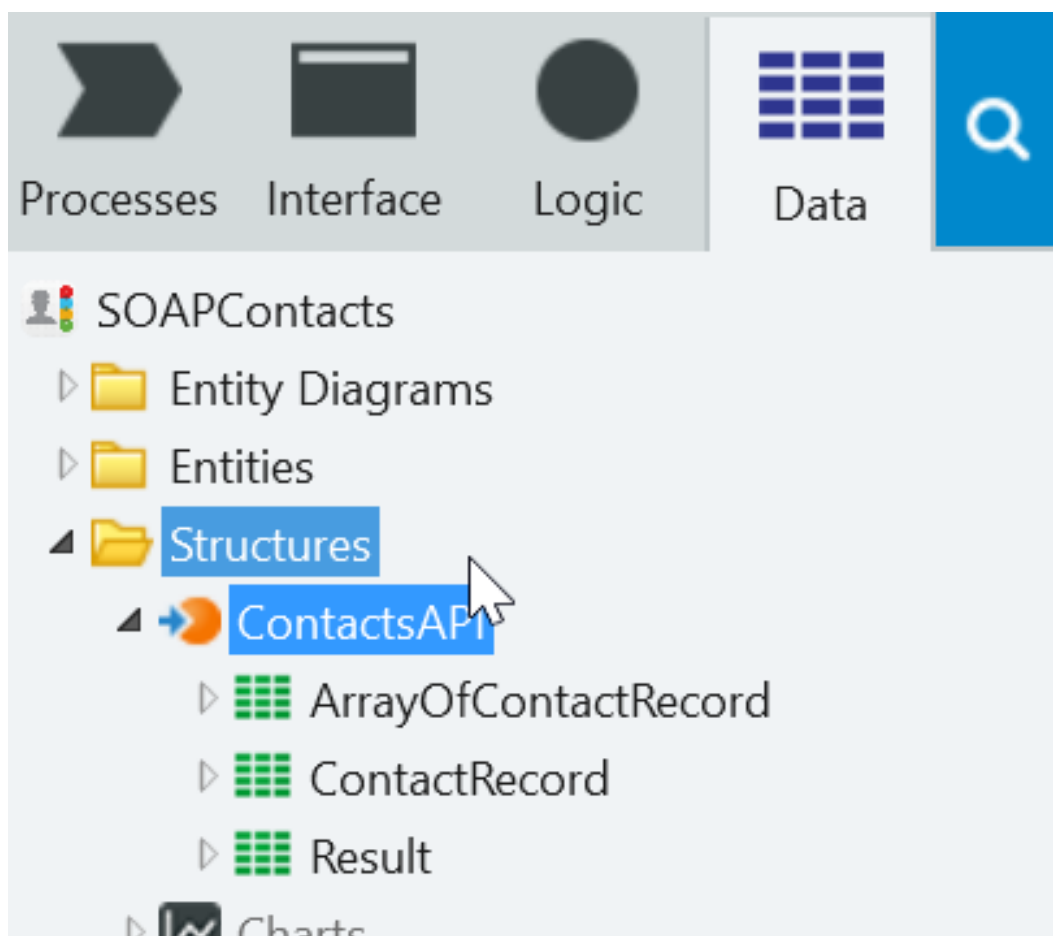
The screenshot shows the same dialog box titled "Consume SOAP Web Service" with a close button (X) in the top right corner. The main heading is "Choose Methods". Below this, there is a list of methods with checkboxes: "(Select All)", "CreateContact", "DeleteContact", "GetContact", "ListContacts", and "UpdateContact". The "ListContacts" method is selected with a checked checkbox. At the bottom of the dialog, there are two buttons: a blue "BACK" button on the left and a red "FINISH" button on the right. A mouse cursor is visible near the bottom right of the method list.

- e) Change the name of the SOAP Integration from *v1* to **ContactsAPI**.



NOTE: This step is not mandatory, but it allows developers to have custom or more familiar names in their integrations.

- f) Switch to the Data tab, and locate the structures under the *ContactsAPI*.



The *ArrayOfContactRecord*, *ContactRecord* and *Result* structures were automatically created by Service Studio when the SOAP Web Service was added in previous steps.

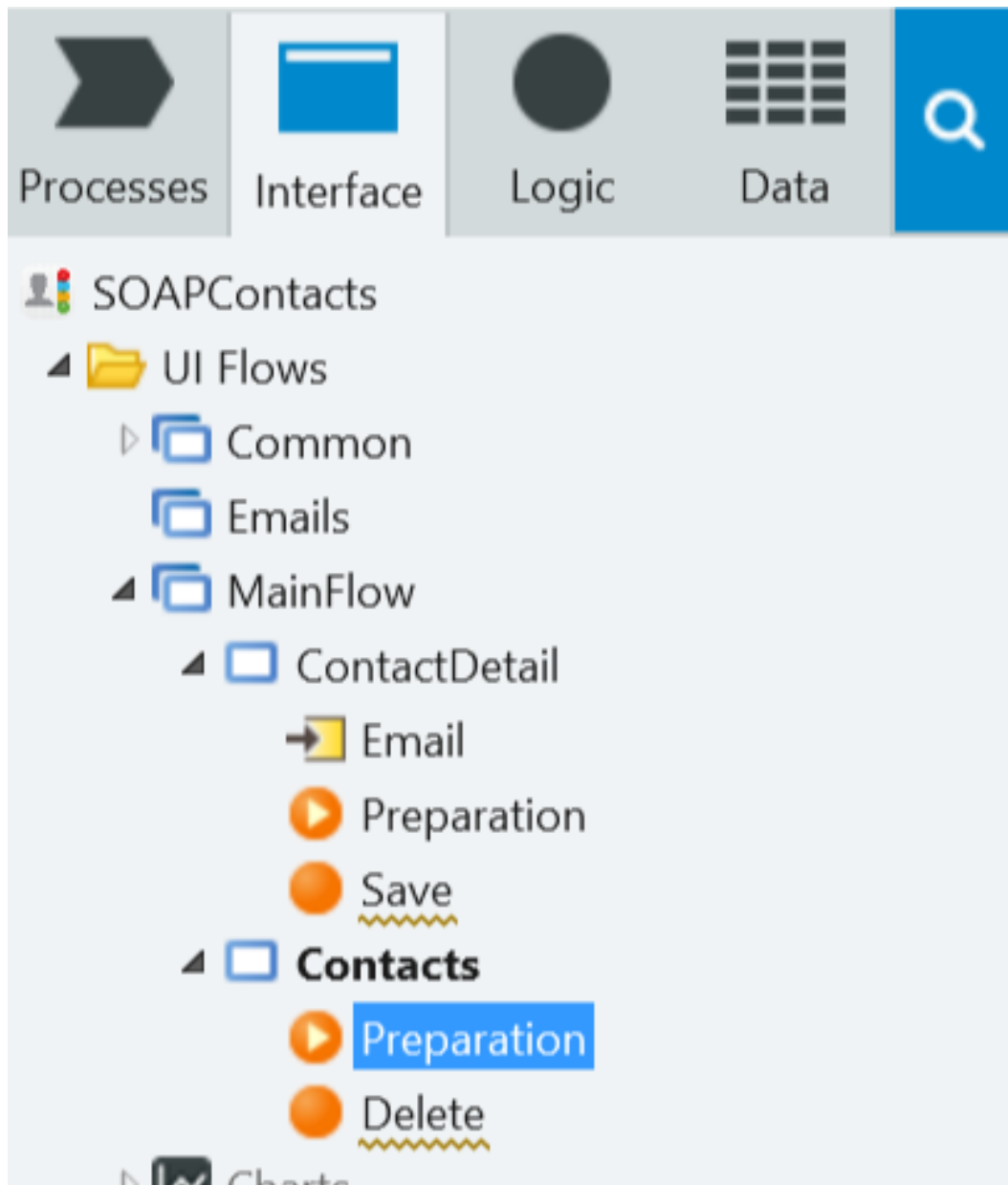
Expanding the *ContactRecord* structure allows you to see all attributes of Contact (Title, Name, Gender, ..). These attributes were also automatically inferred by Service Studio from the provided WSDL.

List Contacts on the screen

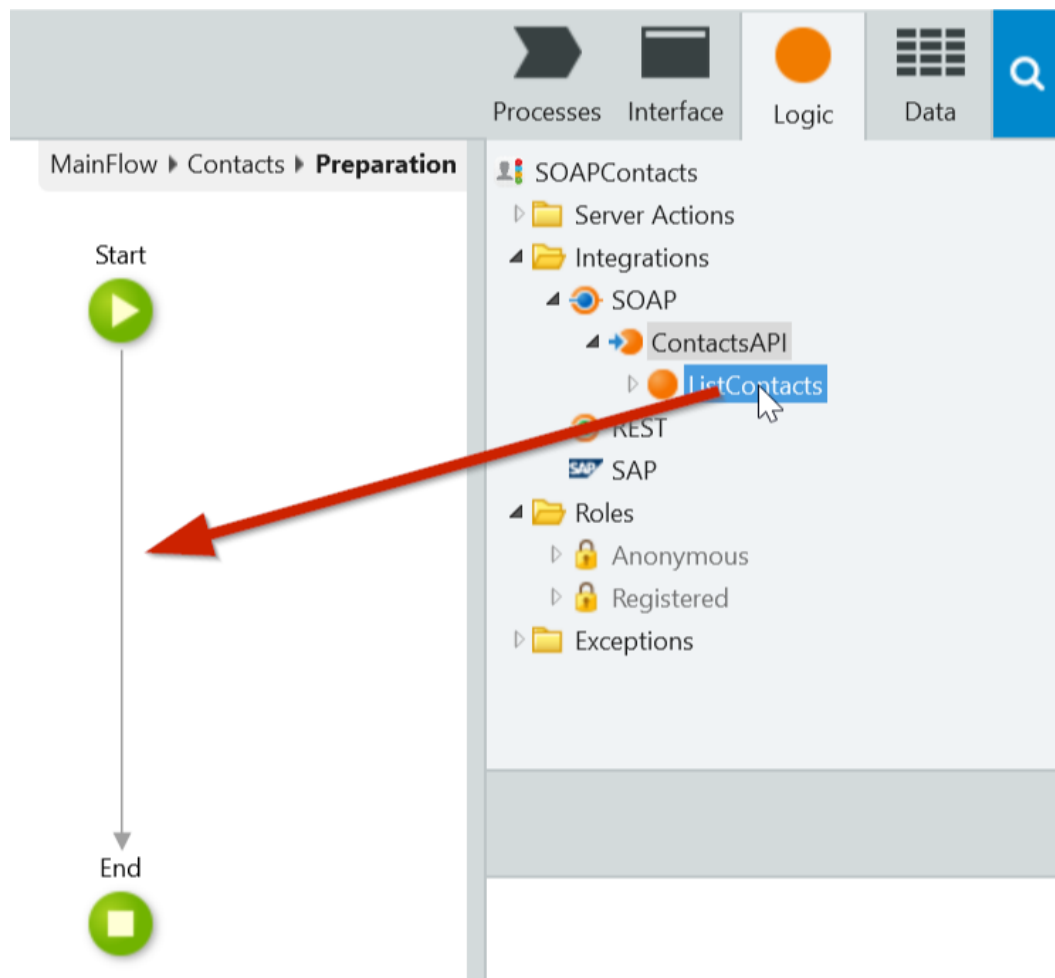
According to the Screen lifecycle of OutSystems web applications, when the user is redirected to the Contacts Screen, firstly, the Preparation will be executed. Therefore, the Preparation of the Contacts Screen is where we will request the list of contacts from the external SOAP Service.

In our sample app, the Contacts screen has already been created. Let's now implement the Preparation that will retrieve the list of Contacts.

- 1) In the Preparation of the Contacts screen, invoke the Contacts API to retrieve the list of contacts from the external SOAP Web Service.
 - a) In the Interface tab, double-click the Preparation of the **Contacts** Web Screen to open it in the canvas.



- b) Switch to the Logic tab and locate the **ListContacts** SOAP method that we have just consumed. Drag and drop the method to the Preparation flow.



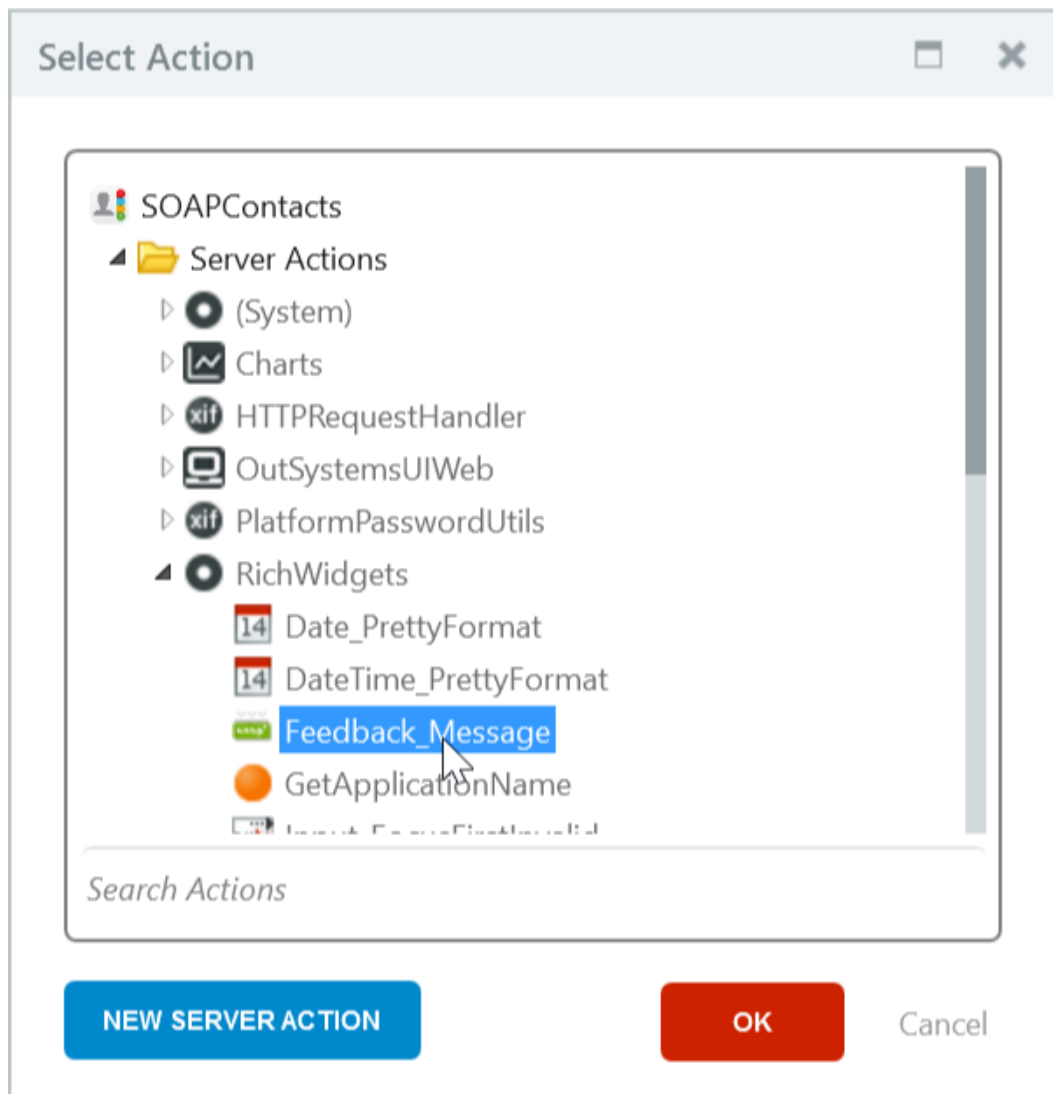
- 2) Validate the result of the SOAP method and show the error message on fail.

- a) Drag an **If** and drop it after the ListContacts, then set its **Condition** to

```
ListContacts.Result.Success
```

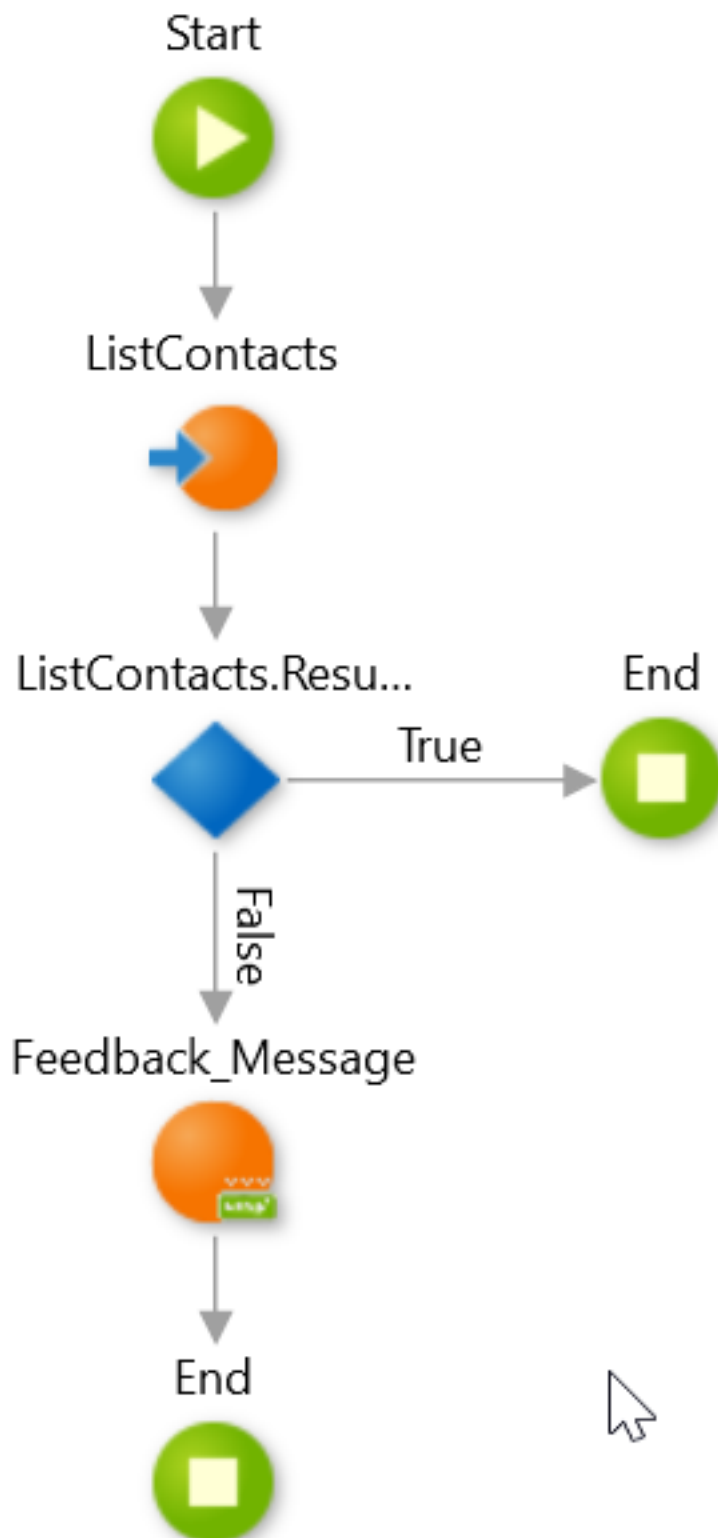
- b) Drag an **End** and drop it to the right of the If, then create the *True* branch connector from the If to the new End.
- c) Drag a **Run Server Action** and drop it on the *False* branch connector of the If.

- d) In the **Select Action** dialog, choose the *Feedback_Message* action



- e) Set the **MessageText** parameter to `ListContacts.Result.ErrorMessage` and the **MessageType** to `Entities.MessageType.Error`.

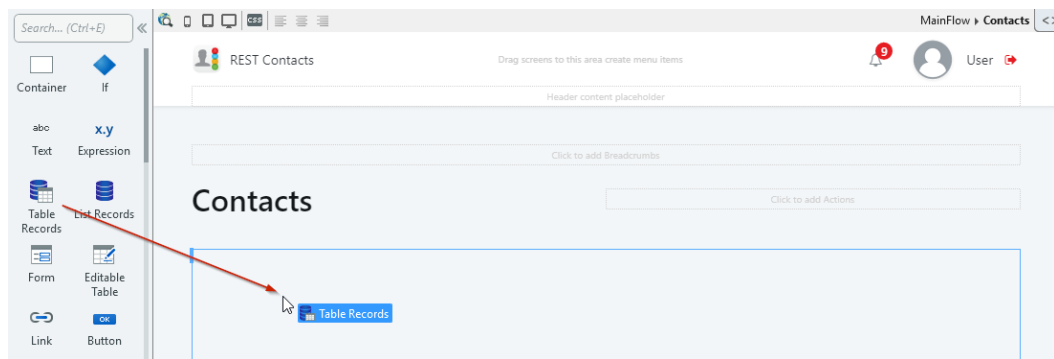
f) Your Preparation flow should like like this



3) Now that we have a way to retrieve a list of records, we need to create the

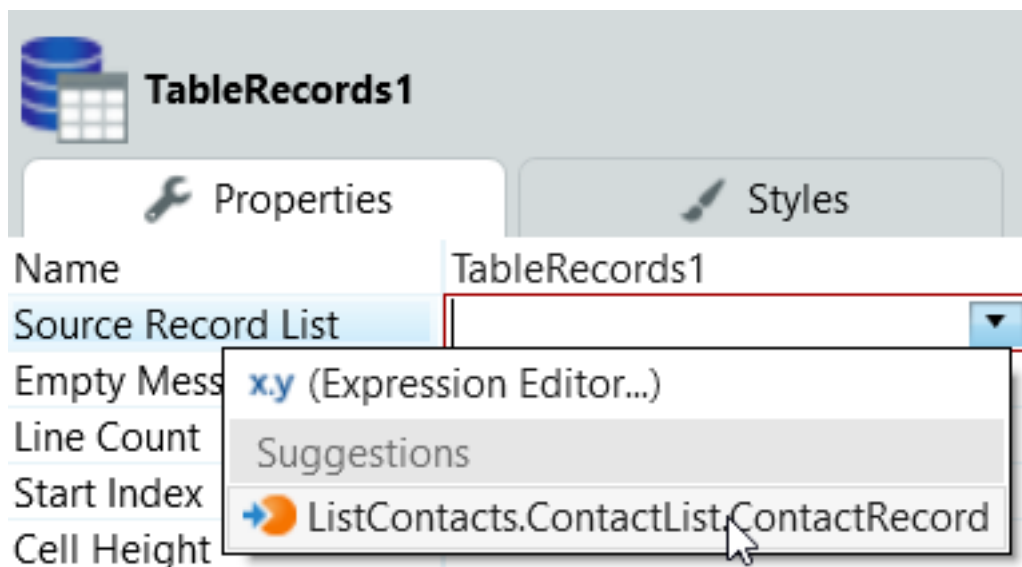
interface to display them to the user. For that we are going to show them in a Table Records widget.

- Switching back to the Interface tab, double-click the **Contacts** screen to open it on the canvas.
- Drag and drop a Table Records to the Main Content.

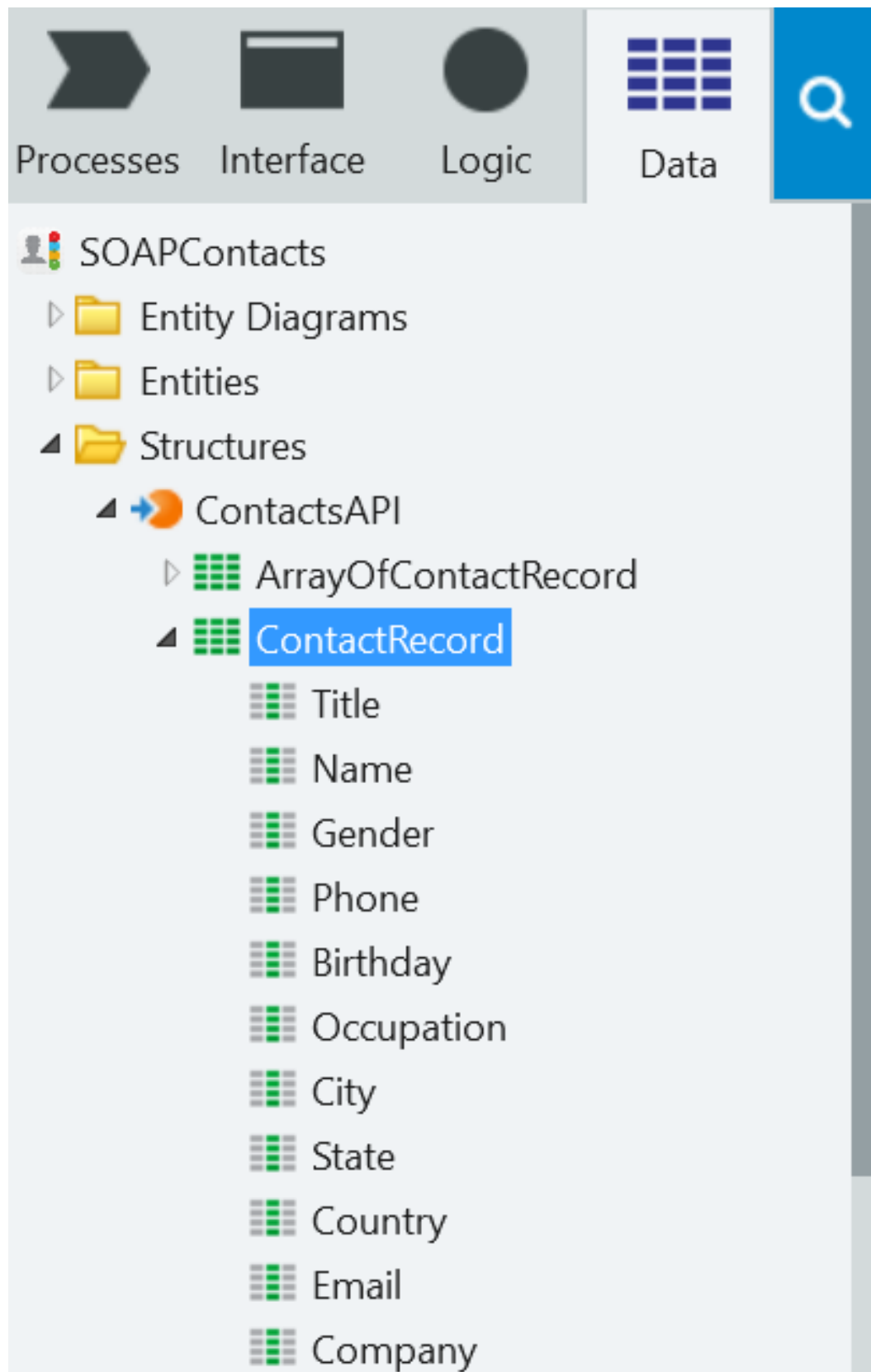


- Set the **Source Record List** property to

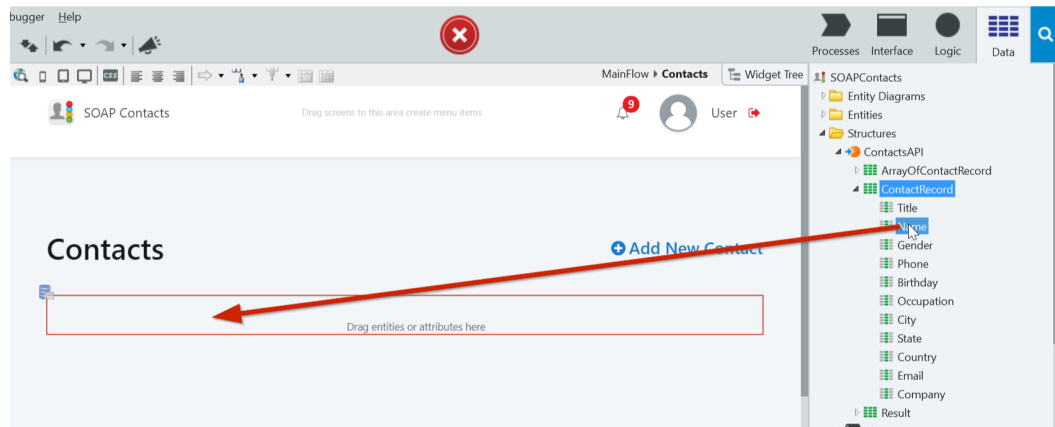
```
ListContacts.ContactList.ContactRecord
```



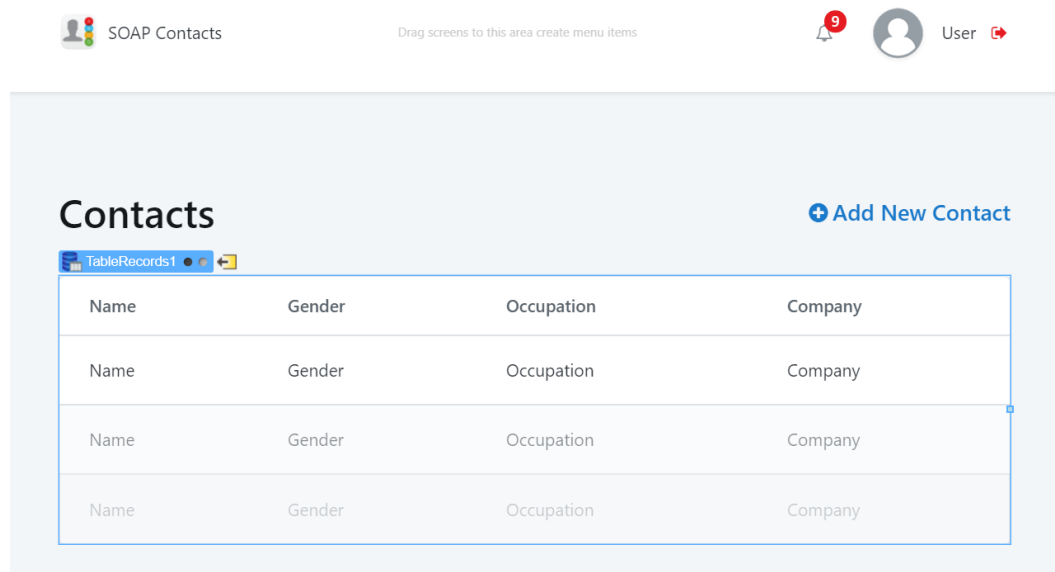
- d) Let's switch to the Data tab and under the Structures folder locate the *ContactRecord* structure and expand to see its attributes.



- e) Drag the *Name* attribute and drop it in the Table Records.



- f) Repeat the previous step for the *Gender*, *Occupation* and *Company* fields.
- g) You should end up having the Contacts screen with a Table Records with 4 columns like this:



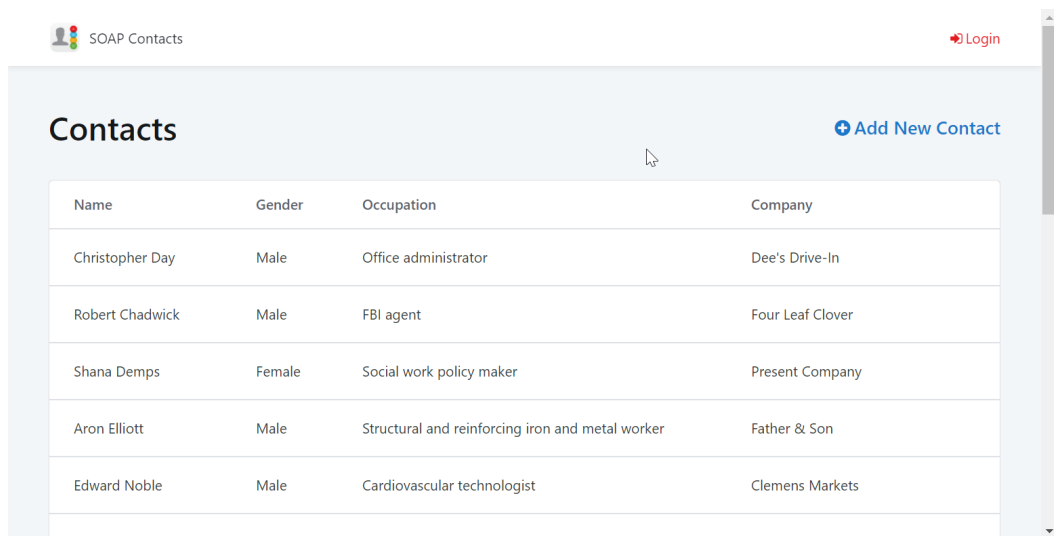
- 4) Publish the application using 1-Click Publish button and verify that the publish completed successfully in the 1-Click Publish tab.
- a) Click on the **1-Click Publish** button to publish the module to the server.

b) Verify in the 1-Click Publish tab that the publishing process was successful.

1 Warning		Debugger	1-Click Publish
1	Uploading	Storing a new version into 'https://os11training.outsystems.net/ServiceCenter'.	
2	Compiling	Generating and compiling optimized ASP.NET C# code and creating SQL scripts.	
3	Deploying	Updating SQL Server database model and deploying the web application to IIS.	
✓	Done	'SOAPContacts' is now available at 'https://os11training.outsystems.net/SOAPContacts'.	

c) Preview the app in browser by clicking on the **Open in Browser** button.

d) The *Contacts* screen should be presented and display existing contacts.



The screenshot shows the 'SOAP Contacts' application interface. At the top, there is a header bar with the application name 'SOAP Contacts' on the left and a 'Login' button on the right. Below the header, the main content area is titled 'Contacts' and includes an 'Add New Contact' button. A table displays a list of contacts with the following data:

Name	Gender	Occupation	Company
Christopher Day	Male	Office administrator	Dee's Drive-In
Robert Chadwick	Male	FBI agent	Four Leaf Clover
Shana Demps	Female	Social work policy maker	Present Company
Aron Elliott	Male	Structural and reinforcing iron and metal worker	Father & Son
Edward Noble	Male	Cardiovascular technologist	Clemens Markets

NOTE: The contacts shown may vary depending on the data provided by the external system.

End Lab

In this lab, we integrated with a SOAP Web Service, namely a method to retrieve the list of Contacts provided by the external system.

To accomplish that, we have used the OutSystems visual interface to consume the SOAP method, that automatically infers the output structure of the method.

Once we consumed the SOAP Web Service, we implemented the logic to display the information on the Contacts screen, firstly by adding the method to the Preparation and then by creating the interface to show the Contacts data on the screen.

At the end of this exercise you should be able to integrate with a simple SOAP Web Service to retrieve data and display it in your application.