Developing User Interfaces

Exercises

Contents

1. Module

1.1 Objectives

After completing this course module, you will be able to:

- Explain the usage of user interfaces in Cordys solutions.
- Explain how user interfaces are handled in Cordys
- Create user interfaces using web service operations.
- Deploy user interfaces for users

1.2 Overview

User interfaces enable the application users to work with data (read/insert/update/delete) from back ends. In this module you will be introduced to the concept of user interfaces in Cordys, you will create user interfaces to display data using the available web service operations.

2. About Developing User Interfaces

2.1 Introduction

In Cordys, there are a number of ways of adding a user interface to the application. In most cases user interfaces based on the XForms standard are used to create front-end web applications. XForms nicely integrates with the Cordys Application Server called WS-AppServer and BPM integration. WS-AppServer is used as the central spot to execute server side business logic and possibly service composition.

However, you can also make use of HTML, JSP or even non browser-oriented client applications.

With this module, you will build user interfaces using the Cordys User Interface based on XForm.

You can apply user interfaces to add human interaction into a business processes. For example, you can start a BPM from a user interface, from within a BPM and you can use a user interface to send a task (and relevant details) to a user. Typically, that task must be completed before the process continues.

You can use a user interface to search for, view, add or modify data. User interfaces can be applied to integrate services from several systems together in one user interface to have a single view of the relevant information. For example, you can combine customer data from your CRM, ERP and document systems together into one user interface, thus creating one single view on your customer information.

2.2 References

More information about this subject is available:

- Online Cordys Documentation
  Developing Applications→ Working with Web Applications
  Reference → Application Development
  Reference → API → Web APIs
- http://community.cordys.com
- http://www.w3schools.com

3. User Interfaces and Web Services

3.1 Prerequisites

Before you can start with this module, take note of the following prerequisites. The exercises are written based on successfully completion of those prerequisites.

You must have completed the following modules

- Application Management
- Developing Processes

You must have ONLY the following roles assigned to yourself

- Administrator
- Developer
- Cordys Fundamentals Trainee

3.2 Configure Design and Deployment structure

In this exercise you will setup the folder structure for the design time and deployment time components for user interfaces. For detailed information see module *Application Management*.

3.2.1 Creating the Design time Folder Structure

1. Open the *Workspace Documents* (Workspace Documents).
2. Open the *Fundamentals training* workspace.
3. Right click the *My Application Project* and select *New → Folder*.
4. Provide the name: **Cordys Web dir**.

3.2.2 Creating Deployment Structure

<u>Web Application files</u>

Web Application files are files like style sheets, JavaScript files, icons, images etc. These files need to be accessible by the application users in order to correctly display and work with the user interfaces. These files need to be created in the Web folder of Cordys, using a directory path structure to make them unique.

Creating a Web Library Definition

A Web Library Definition document serves as starting point and indicates that all files regardless of their type need to be published to the *Cordys Web dir* location.

1. Right click *Cordys Web dir* and select *set as Start Point of Qualified name*

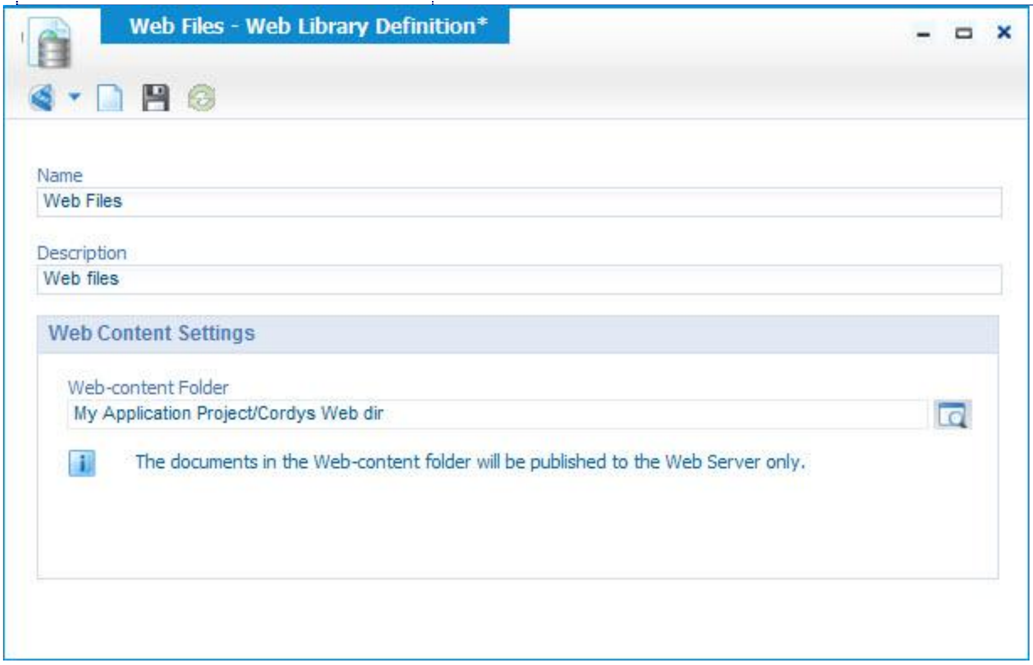
2. Right click the *Cordys Web dir* folder and add a *Web Library Definition* (

Web Library Definition
Define and publish web libraries

) document.

3. Provide the following values:

Field	Value
Name	Web Files
Description	Web files
Location	Prefilled with: My Application Project/Cordys Web dir



Web Files - Web Library Definition*

Name
Web Files

Description
Web files

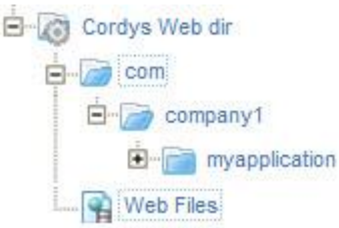
Web Content Settings

Web-content Folder
My Application Project/Cordys Web dir

The documents in the Web-content folder will be published to the Web Server only.

4. Click **Save** and close the screen.

5. In the *Cordys Web dir* folder add a subfolder **com**.
6. Add a subfolder **companyX** (X is your student number).
7. Add a subfolder **myapplication**.



Cordys Web dir
com
company1
myapplication
Web Files

3.3 Single Occurrence Form

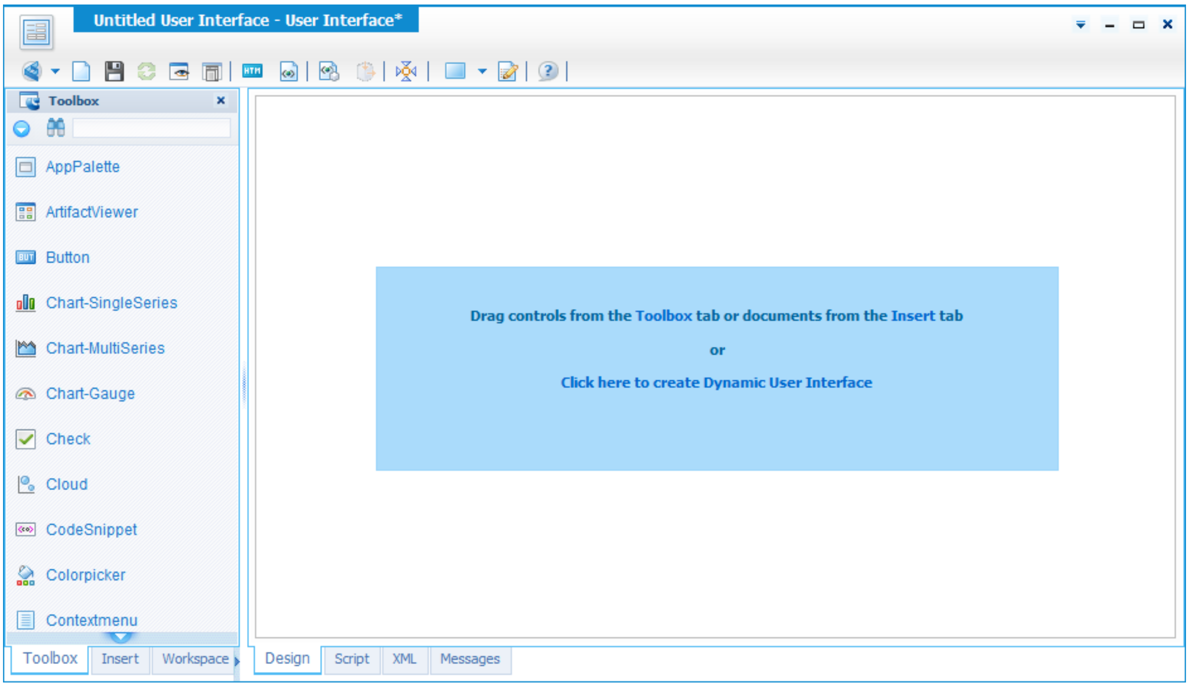
A single occurrence form is a detail screen; the form displays one instance of an underlying data model at the time. This type of form will be generated when the web service that is used returns one instance.

3.3.1 Creating the Form

1. In *My Application Project,* navigate to *User Interfaces → com → companyX → myapplication*

2. Right click the *myapplication* folder and create a new document of type *User Interface*.

The User Interface editor is started:



Drag controls from the Toolbox tab or documents from the Insert tab

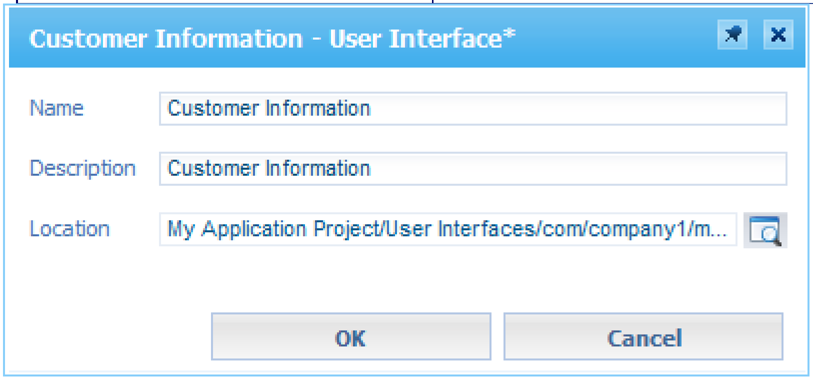
or

Click here to create Dynamic User Interface

3. Click **Save**.

4. Provide the following values:

Field	Value
Name	Customer Information
Description	Customer Information
Location	Prefilled with: My Application Project/User Interfaces/com/companyX/myapplication



Customer Information - User Interface*

Name	Customer Information
Description	Customer Information
Location	My Application Project/User Interfaces/com/company1/m...

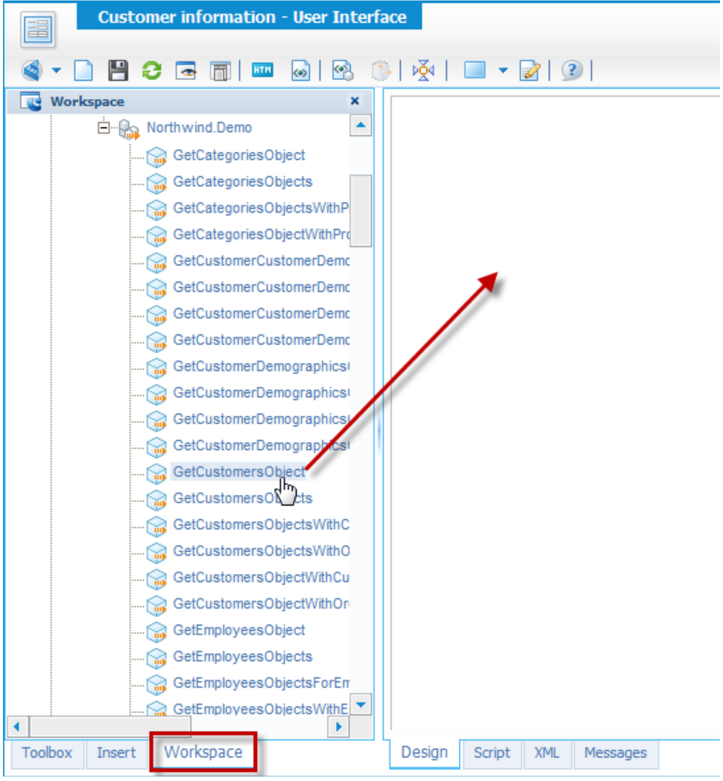
5. Click **OK**.

> **NOTE**
>
> The name of the user interface is by default also used as the name that will be displayed in the *My Application App palette*. Via the form Task properties you can change the display name.

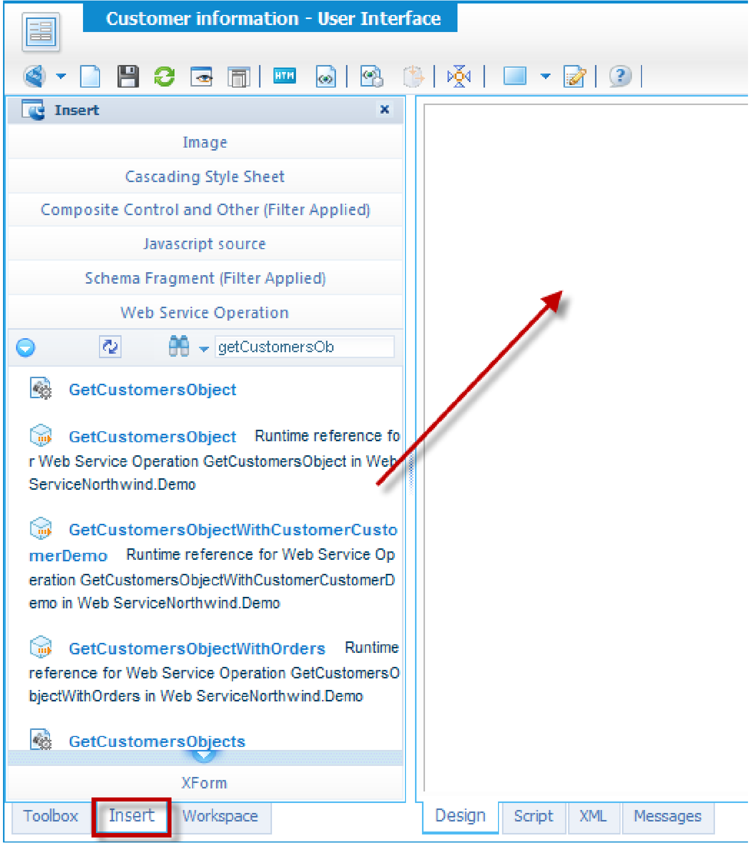
3.3.2 Adding Web Service Operations to the Form

Adding Documents to your editor can be done in the following ways:

- Select the W*orkspace* tab, find the required object in the tree, and drag and drop it in the modeling area.



- Select the *Insert* tab, expand the object tab box, and drag and drop the required object in the modeling area.

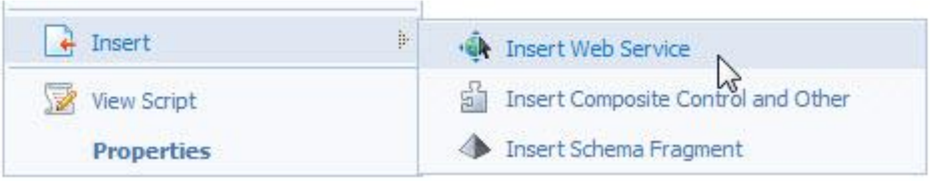


GetCustomersObject Runtime reference for Web Service Operation GetCustomersObject in Web ServiceNorthwind.Demo

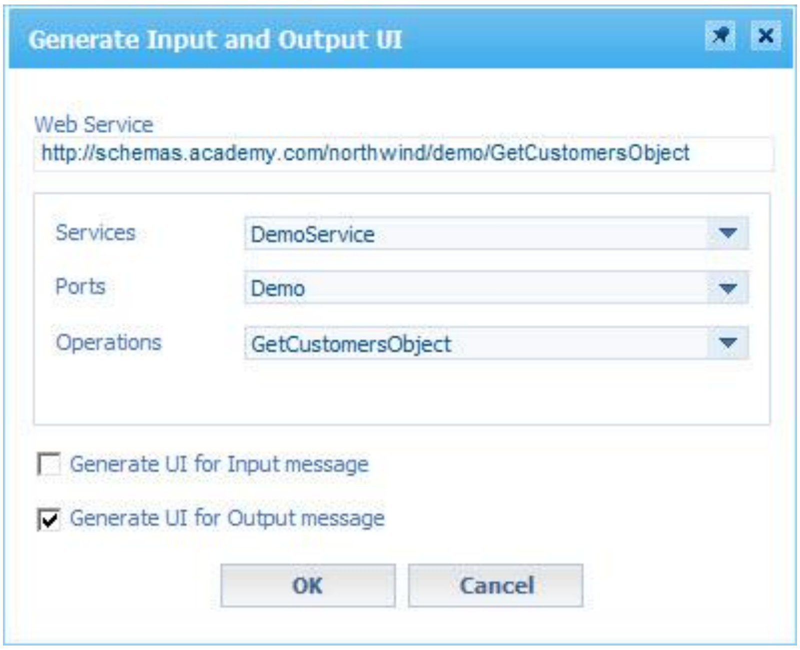
GetCustomersObjectWithCustomerCustomerDemo Runtime reference for Web Service Operation GetCustomersObjectWithCustomerCustomerDemo in Web ServiceNorthwind.Demo

GetCustomersObjectWithOrders Runtime reference for Web Service Operation GetCustomersObjectWithOrders in Web ServiceNorthwind.Demo

- Right click on the form, via the *Insert* option select the required object type and from the opened box, and select the required object.



1. In the *Workspace* tab navigate to *Runtime References → Web Services → Northwind.Demo*.
2. Drag and Drop *GetCustomersObject* in to the form editor.



Web Service
http://schemas.academy.com/northwind/demo/GetCustomersObject

Services	DemoService
Ports	Demo
Operations	GetCustomersObject

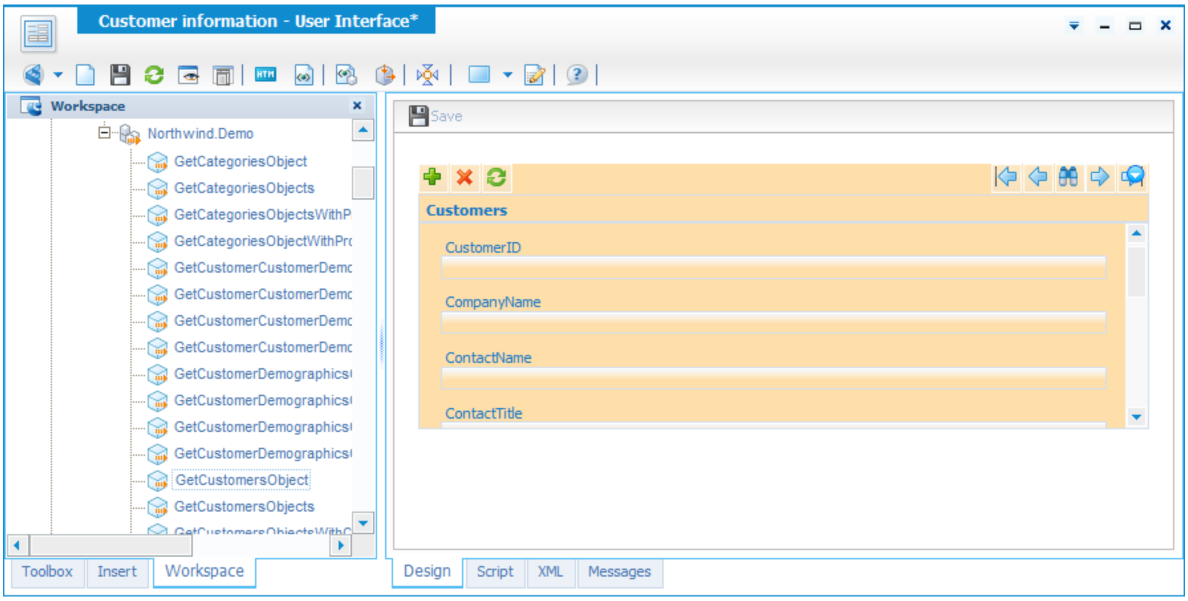
☐ Generate UI for Input message

☑ Generate UI for Output message

3. Make sure only *Generate UI for Output messages* is checked.

> **NOTE**
>
> A service operation has an input message and an output message. You can choose to generate form elements for those messages.
>
> **Input**: will generate a field to enter a search value and a search button
>
> **Output**: will generate fields based on the response message

4. Click **OK**.

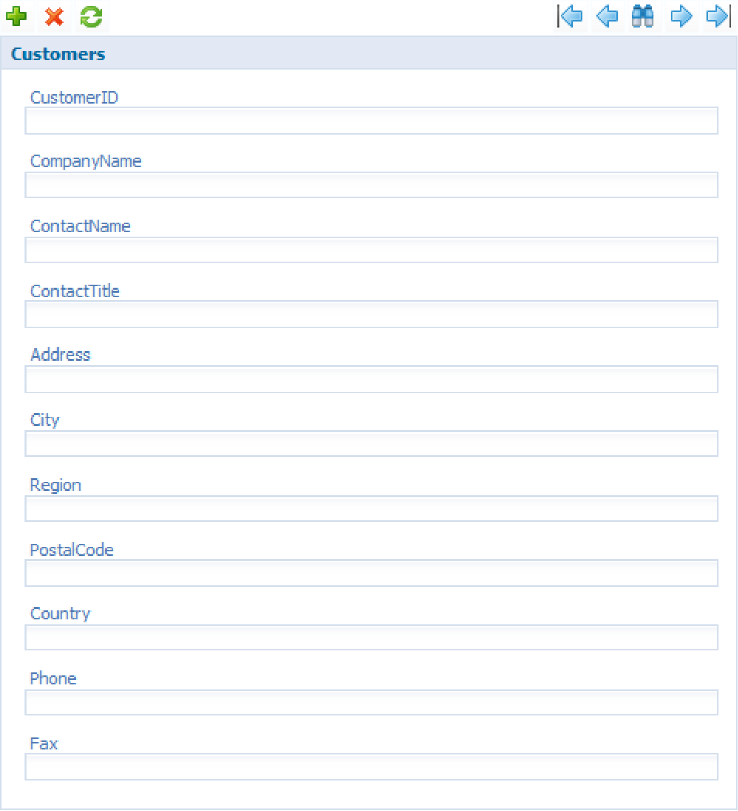


How is it possible that the UI is created just by adding a Web Service Operation on your form?

____

____

5. Resize the *Customers* group box so it shows all the fields:



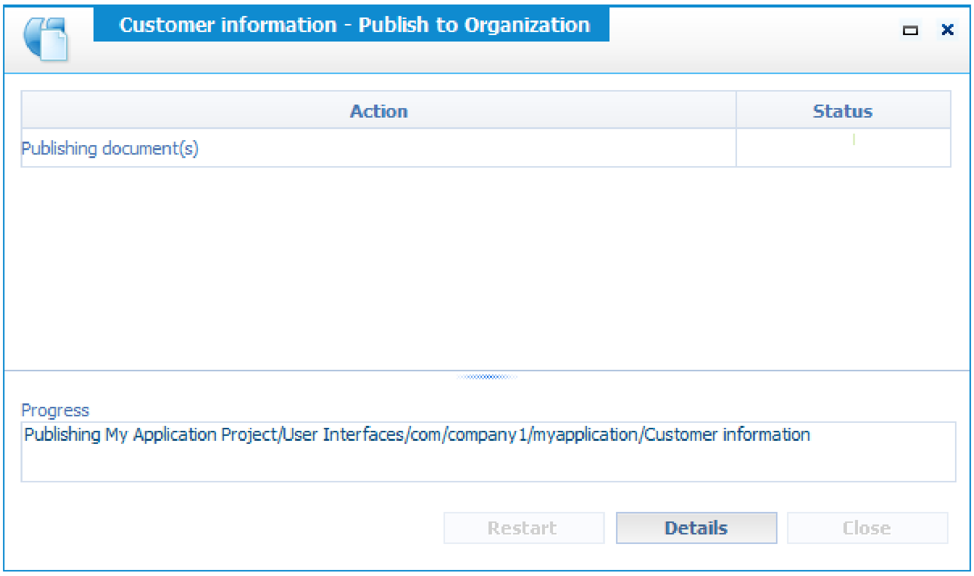
6. **Save** your form.

3.3.3 Previewing the Form

You can preview your form from the editor. When doing so the form will be automatically published to the organization to create a runtime version of the form.

In preview mode you can test all functionality.

1. Click **Preview** (👁) in the toolbar.
2. A *Publish* screen is opened, it will close automatically when successful:



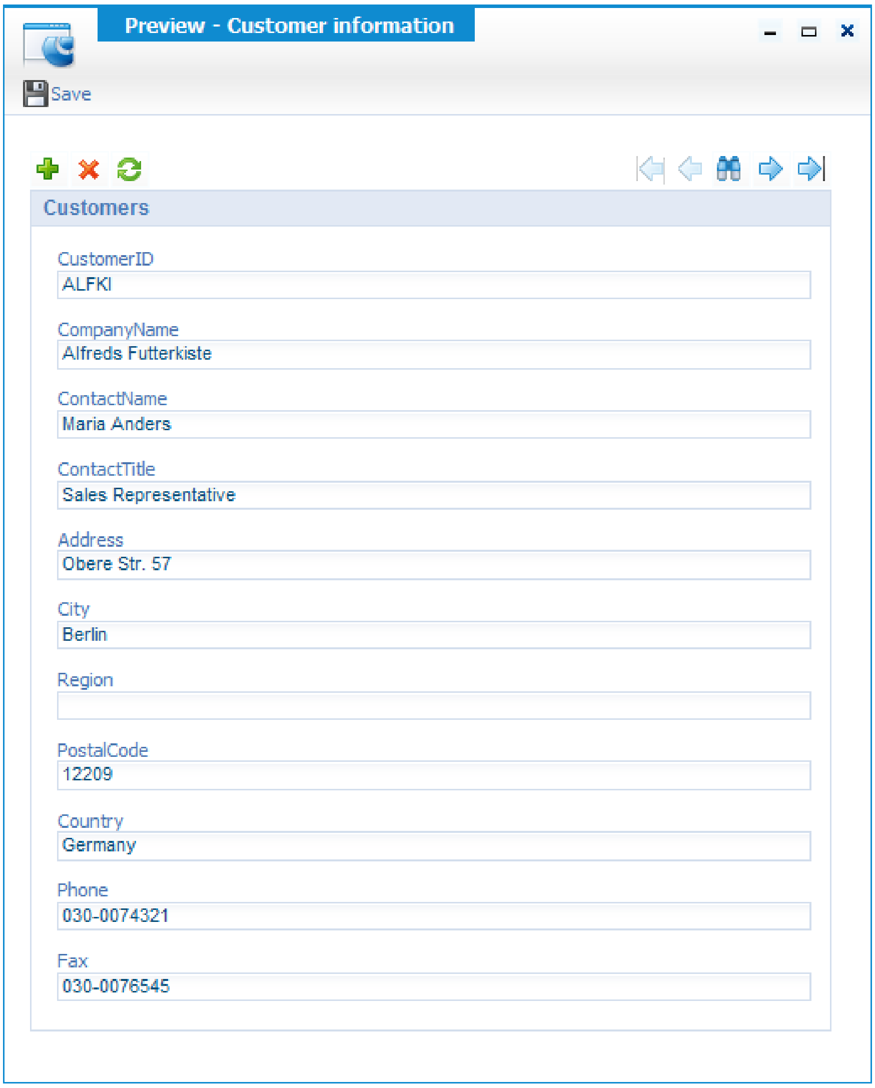
Customer information - Publish to Organization

Action	Status
Publishing document(s)	

Progress

Publishing My Application Project/User Interfaces/com/company1/myapplication/Customer information

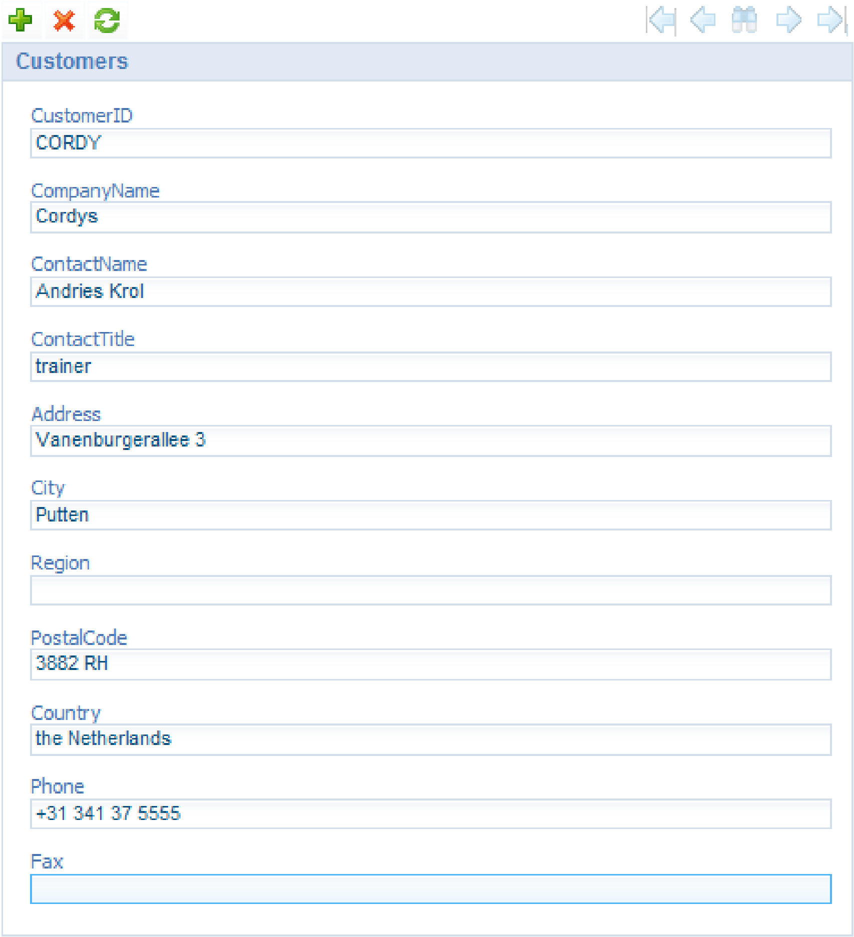
3. The preview of your form is opened:



Preview - Customer information

CustomerID	ALFKI
CompanyName	Alfreds Futterkiste
ContactName	Maria Anders
ContactTitle	Sales Representative
Address	Obere Str. 57
City	Berlin
Region	
PostalCode	12209
Country	Germany
Phone	030-0074321
Fax	030-0076545

4. Use the pagination bar (⇤ ← 🔍 → ⇥) to move to the first, last, next and previous record.
5. Use the Search button to find Customer **HILAA**.
6. Click **Insert** (✚).
7. Provide any values to insert a new customer,
   CustomerID: is a string of 5 characters which must be unique
   CompanyName: Use your own company name
   ContactName: use your own name
    Other fields are optional:



Customers

CustomerID
CORDY

CompanyName
Cordys

ContactName
Andries Krol

ContactTitle
trainer

Address
Vanenburgerallee 3

City
Putten

Region

PostalCode
3882 RH

Country
the Netherlands

Phone
+31 341 37 5555

Fax

8. Click **Save** (💾Save) to save your customer details.

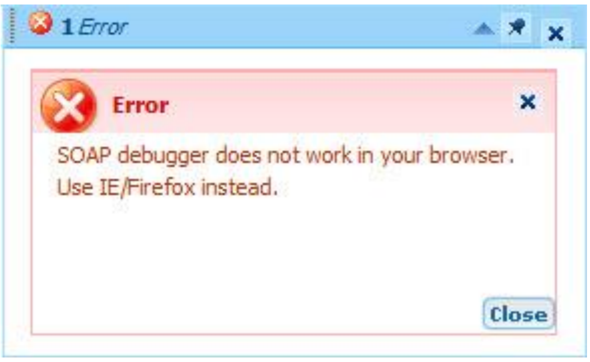
How is it possible that you can insert a new customer, although you only used a Get operation (GetCustomersObject)?

Tip open the SOAP Debugger (SOAP Debugger*) before inserting a new customer. Prior to starting the SOAP Debugger make sure to close the Inbox and System Resource Manager and to open the preview of your form.*

____

____

Note that the functionality of the *SOAP Debugger* is only supported by MS Internet Explorer and Firefox. You can also use tools like Fiddler to monitor browser communication.



1 Error

Error

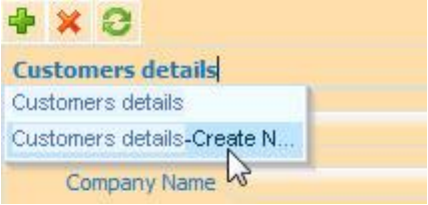
SOAP debugger does not work in your browser. Use IE/Firefox instead.

9. Close the preview screen.

3.3.4 Changing the Look and Feel

In this part you will make changes to the generated form to meet your needs regarding the look and feel.

1. If closed, open the *Customer Information* design form.
2. Change the label of the group box to **Customers details**:



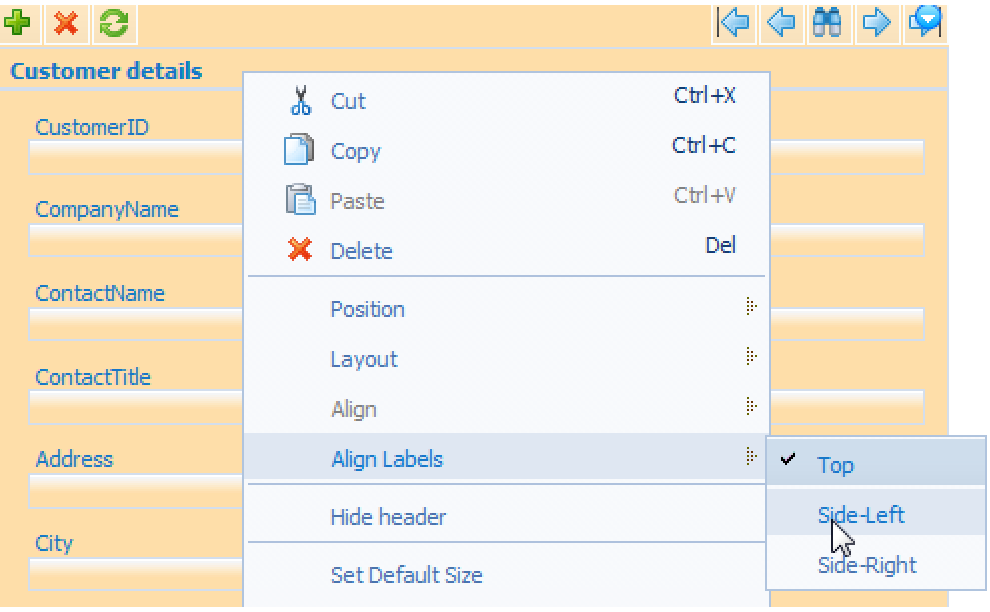
Customers details

NOTE

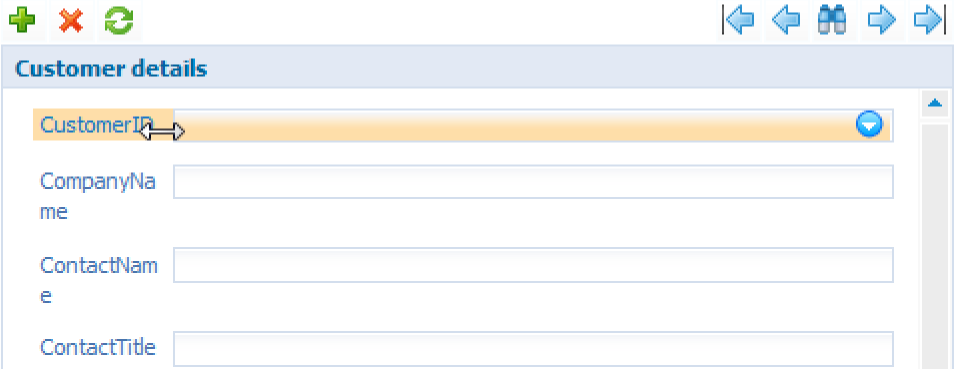
Text in forms, bpms, etc can be translated to different languages. Objects like labels will be connected to an identifier which can be used in other fields, forms, etc. This way the same label only needs to be translated once. From the context menu you can select an existing language label, or create a new one when the text is used in a different context and requires a different translation.

You can learn more about translation/internationalization in the Developing User Interface course.

3. Right click the title bar and select *Align Labels → Side-Left*:



4. Hold your mouse a little left of the *CustomerID* input field; you will get a horizontal resize mouse icon. Click and hold to change the length of all fields at once:



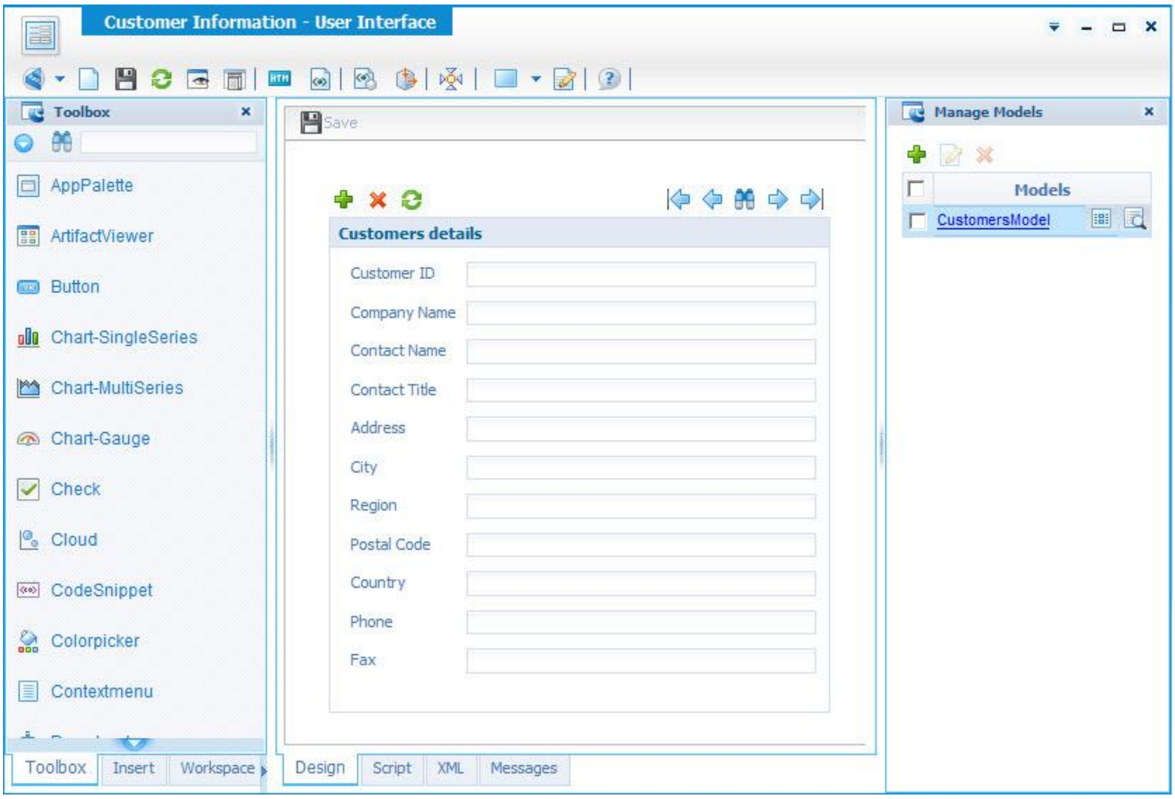
5. Rename the labels to include spaces etc, like *Company Name*.
6. Try to move the *Customers* groupbox (location is fixed).
7. Right click a free spot on the form (not in the group box) and select *Layout → Free*. You can now move the box to any location in the screen, but the layout of the *Customers* group box has still a vertical layout.
8. Resize the group box so it looks nice.
9. **Save** the form.
10. Preview the form to see changes at runtime.

3.3.5 Manage Model

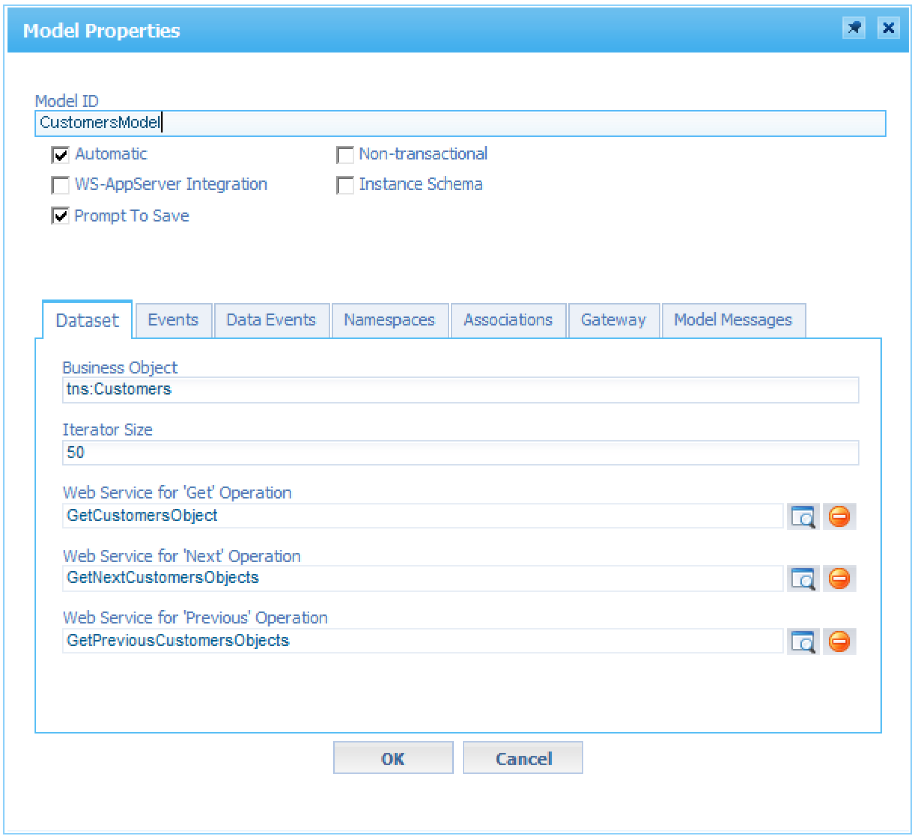
When you add a web service operation to a form, apart from visible fields (View), there is also a Model added to your form. The Model acts as intermediate between the view and the web service operation (Model - View - Controller (MVC)).

1. If closed, open the *Customer Information* design form.
2. Click **Manage Models** ( ) in the toolbar.

The *Manage Model* frame is opened at the right:



3. Click CustomersModel to open the model properties:



Model Properties

Model ID: CustomersModel

☑ Automatic ☐ Non-transactional
☐ WS-AppServer Integration ☐ Instance Schema
☑ Prompt To Save

Business Object: tns:Customers

Iterator Size: 50

Web Service for 'Get' Operation: GetCustomersObject

Web Service for 'Next' Operation: GetNextCustomersObjects

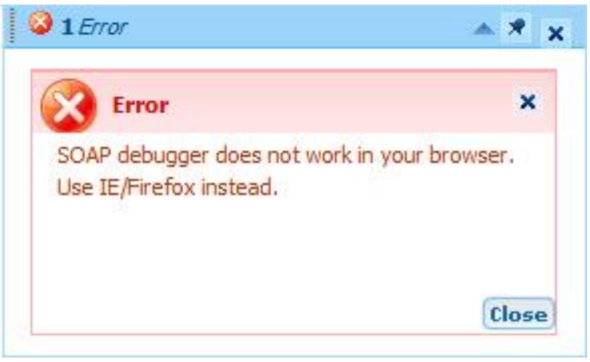
Web Service for 'Previous' Operation: GetPreviousCustomersObjects

4. Change *Iterator* size to **2** (cursors data in sets of 2).
5. Notice the operations used for the actions *Get*, *Next* and *Previous*.
6. Click **OK**.
7. **Save** and **Preview** your form.
8. If the *My Inbox* is open, close it before continuing.

9. If the *System Resource Manager* is open, close it before continuing.

10. Open the *SOAP Debugger* (SOAP Debugger) artifact, via *My Application*.

Note that the functionality of the *SOAP Debugger* is only supported by MS Internet Explorer and Firefox. You can also use tools like Fiddler to monitor browser communication.



SOAP debugger does not work in your browser. Use IE/Firefox instead.

You will now study the relation between the pagination bar buttons and the web service operation available on the model.

NOTE

When the SOAP Debugger becomes active, only use the Single Step (▶) button in the toolbar and continue until the form becomes active again.

11. Perform the actions mentioned in the table and fill in the result:

Use pagination button	Web service operation called
Last	
First	
Next 1st time	
Next second time	
Next third time	
Search **FRANS**	

12. Close the *SOAP Debugger*.

13. Close the preview and the design form.

3.4 Multi Occurrence Form

A Multi occurrence form displays multiple instances of an underlying data model at once in a table view (which can be changed). A multi occurrence form will be generated when the related web service operation returns multiple instances.

3.4.1 Creating the Form

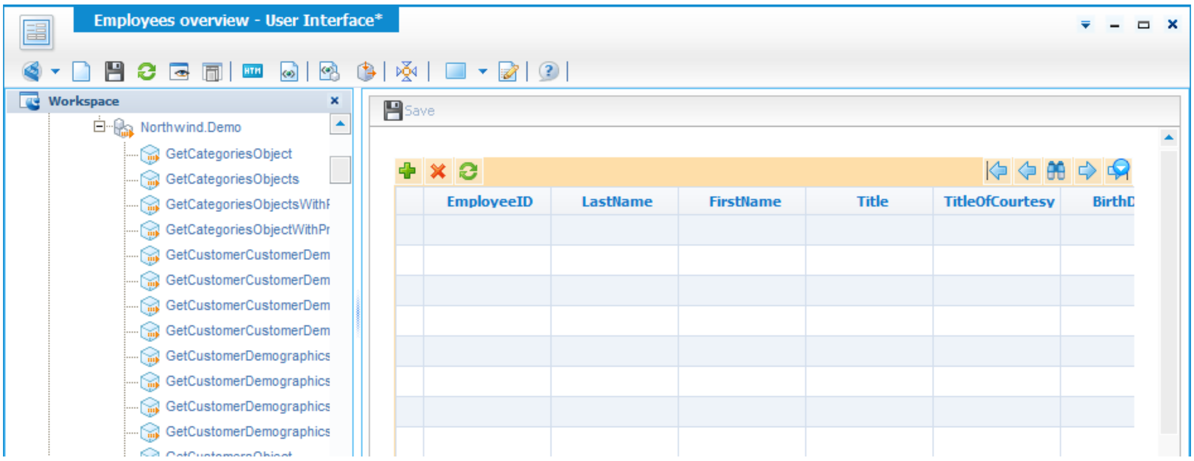
1. If closed, open the *Workspace Documents*.
2. In the *Workspace Documents* toolbar click **New**.
3. Select a *User Interface* document.
4. In the form editor click **Save**.
5. Provide the following values:

Field	Value
Name	Employees Overview
Description	Employees Overview
Location	My Application Project/User Interfaces/com/companyX/myapplication

6. Click **OK**.

3.4.2 Adding the Web Service Operation

1. Select the *Workspace tab* and navigate to *Runtime References → Web Services → Northwind.Demo*.
2. Drag and Drop *GetEmployeesObjects* in to the form editor.
3. Check only *Generate UI for Output messages*.
4. Click **OK**.



5. **Save** the form.

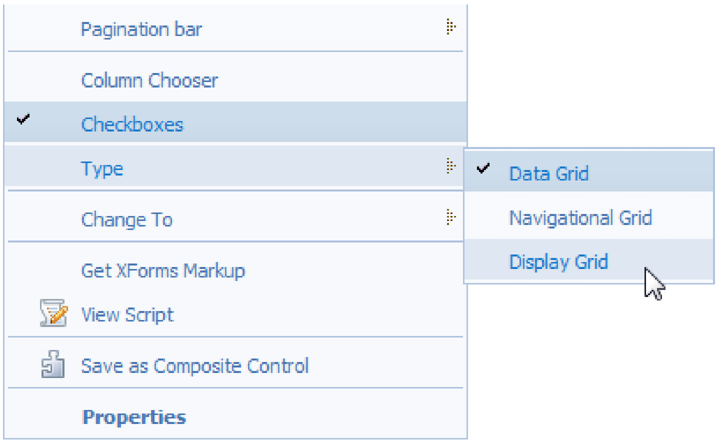
3.4.3 Changing the Look and Feel

1. Resize the height of the table, so you can see *10* employees in one overview.
2. Right click the table and click *Column Chooser*:

3. Check only the following fields, to display them in the table:
   - *EmployeeID*
   - *LastName*
   - *FirstName*
   - *Address*
   - *City*
   - *Country*
   - *HomePhone*
4. Change the labels in the column header of the table.

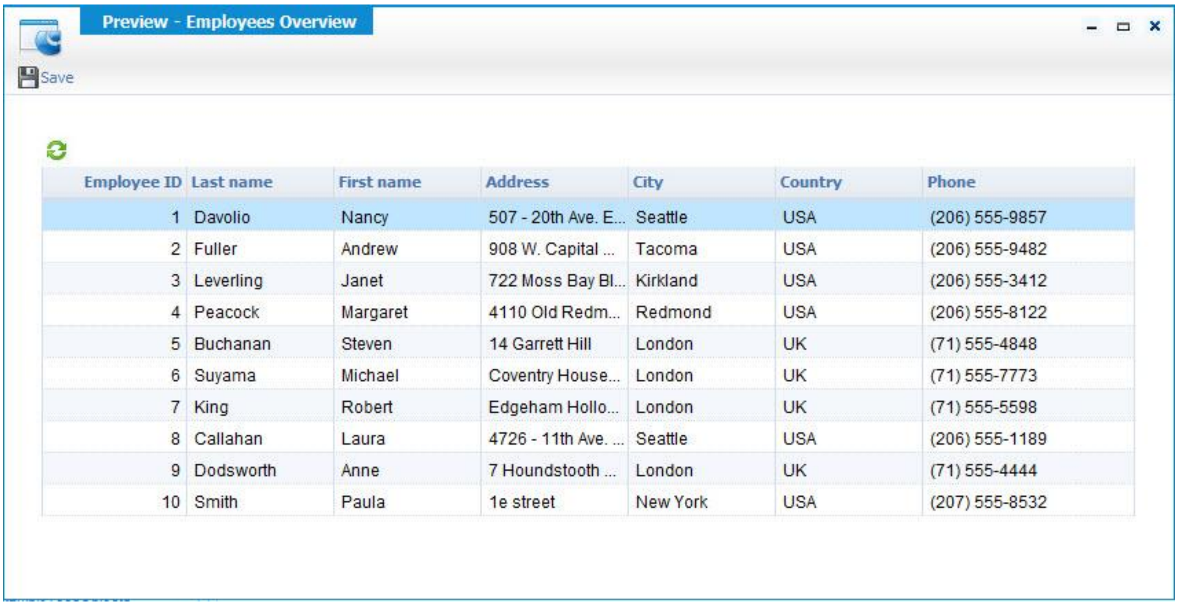
> **NOTE**
> You can use the tab key to go to the next column.

5. Drag and drop the field *First name* before *Last name* (a blue line will indicate the target location before dropping).
6. Right click the *Employee ID* field and select *Change To → Output*.
7. Preview the form (it will automatically ask you to save).
8. Look at the layout/structure of the table in the preview (type = Data Grid).
9. Add yourself as a new employee.
10. Close the preview.
11. Right click somewhere in the middle of the table and select *Type → Display grid*.



12. Preview the form.

13. Notice the difference in layout and functions available in/for the table.



Employee ID	Last name	First name	Address	City	Country	Phone
1	Davolio	Nancy	507 - 20th Ave. E...	Seattle	USA	(206) 555-9857
2	Fuller	Andrew	908 W. Capital ...	Tacoma	USA	(206) 555-9482
3	Leverling	Janet	722 Moss Bay Bl...	Kirkland	USA	(206) 555-3412
4	Peacock	Margaret	4110 Old Redm...	Redmond	USA	(206) 555-8122
5	Buchanan	Steven	14 Garrett Hill	London	UK	(71) 555-4848
6	Suyama	Michael	Coventry House...	London	UK	(71) 555-7773
7	King	Robert	Edgeham Hollo...	London	UK	(71) 555-5598
8	Callahan	Laura	4726 - 11th Ave. ...	Seattle	USA	(206) 555-1189
9	Dodsworth	Anne	7 Houndstooth ...	London	UK	(71) 555-4444
10	Smith	Paula	1e street	New York	USA	(207) 555-8532

14. Close the preview and design form.

3.5 Multiple Models

In this exercise you will use multiple models on the same form with and without using association between the models.

3.5.1 Applying a Model for a Select Box

Select boxes can be used to let people select data from a “small” list that is either stored in a backend application or hardcoded in the form. When there is a lot of data and the data is not changed or selected frequently, using a zoom page is a better solution, performance wise.

Creating the Form

1. In the My Application Project, navigate to *User Interfaces → com → companyX → myapplication*
2. In the *myapplication* folder create a new *User Interface*.
3. Provide the following values:

Field	Value
Name	Product Details
Description	Product Details
Location	Prefilled with: My Application Project/User Interfaces/com/companyX/myapplication

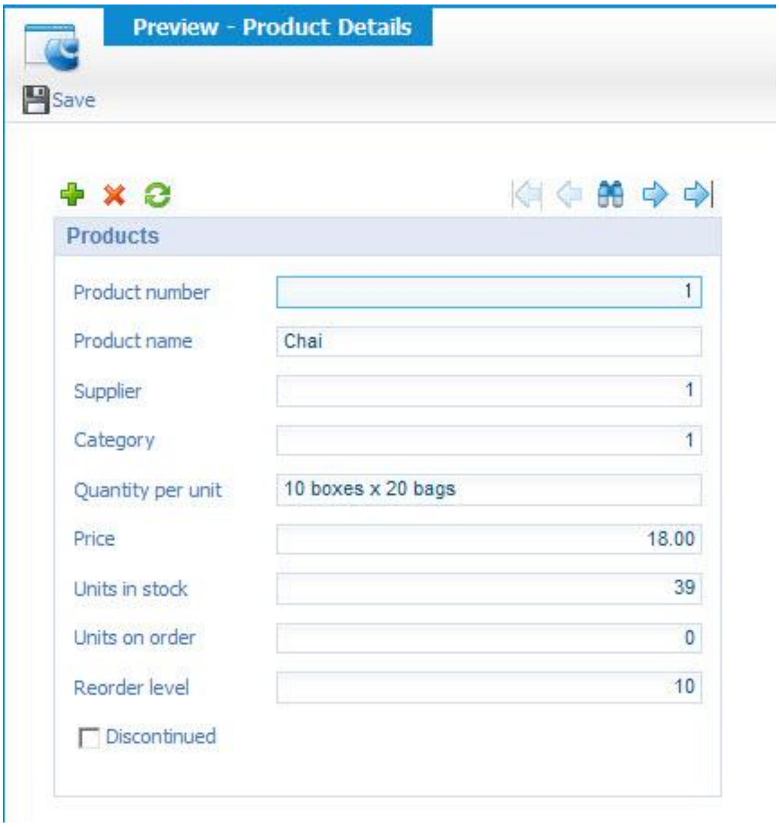
Adding Web Services Operation to the Form

1. In the *Workspace* tab, navigate to *Runtime References → Web Services → Northwind.Demo*.

2. Drag and Drop *GetProductsObject* in to the form editor.
3. Check only *Generate UI for Output messages*.
4. Click **OK**.
5. **Save** the form.

Changing the Look and Feel

1. Resize the Group Box to your needs.
2. Change the labels of the fields.
3. Preview the form.



Products

Product number	1
Product name	Chai
Supplier	1
Category	1
Quantity per unit	10 boxes x 20 bags
Price	18.00
Units in stock	39
Units on order	0
Reorder level	10

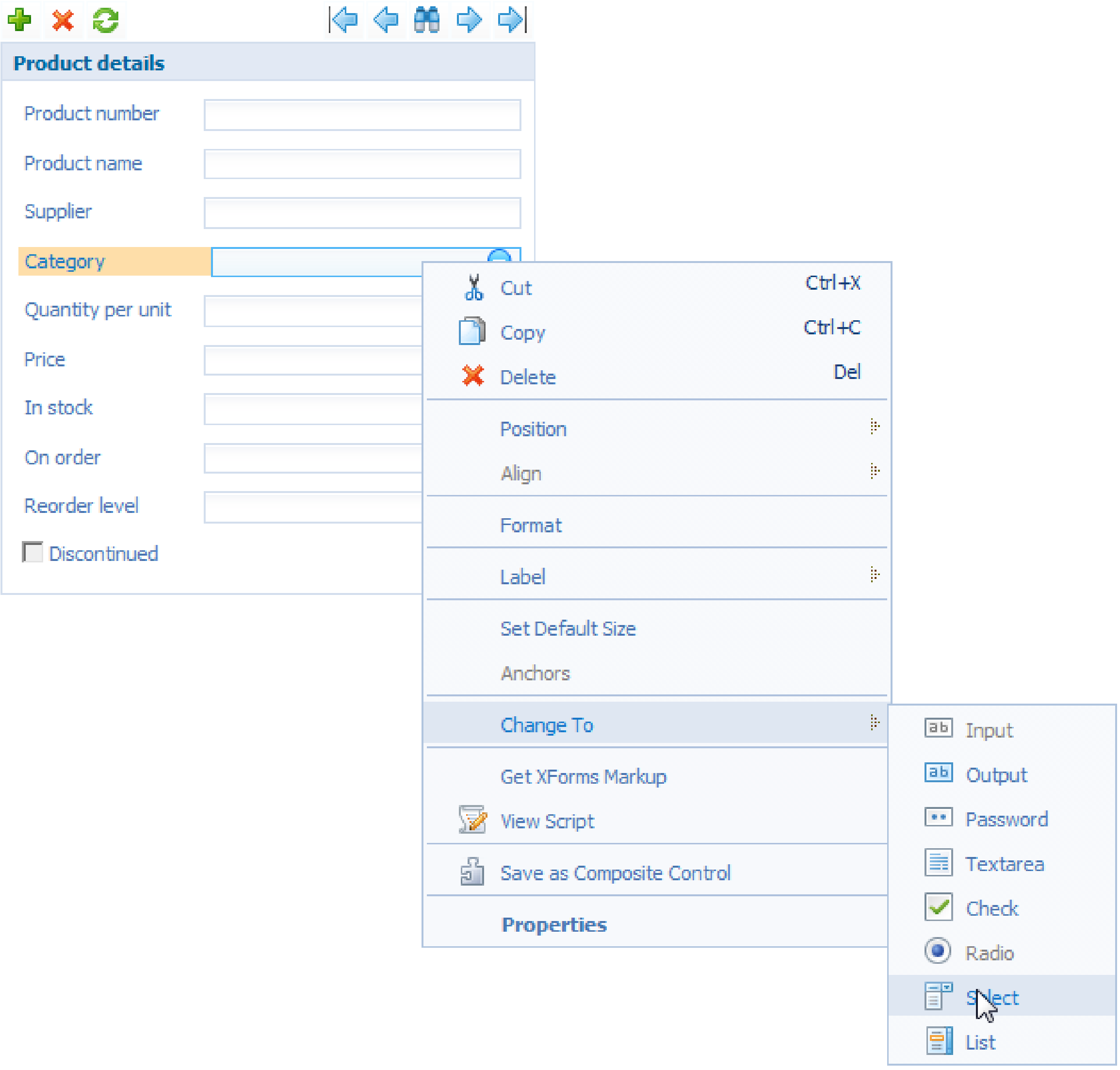
□ Discontinued

Configuring the Select field

As the *Category* is currently a number you will change this field to a select box where you can select the category by name.

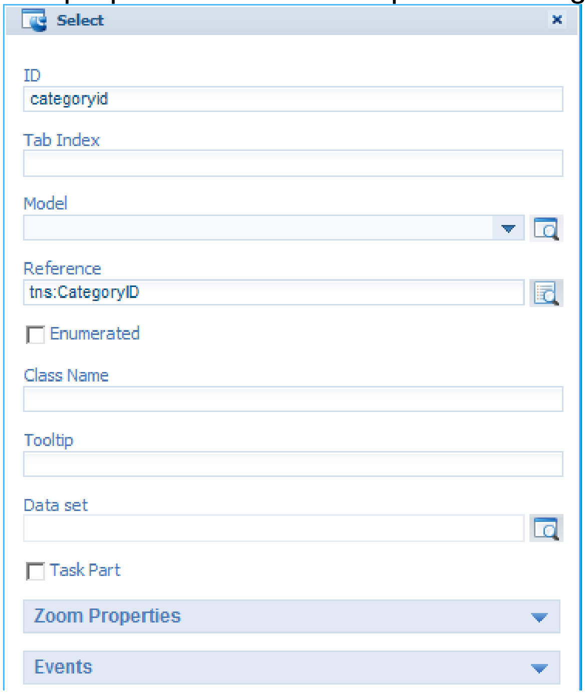
1. Return to the design form.

2. Right click the *Category(ID)* field and select *Change To → Select*.



3. Right click the *Category* field and select *Properties* (or double click).

4. The properties screen will open on the right.



ID: categoryid

Tab Index:

Model:

Reference: tns:CategoryID

□ Enumerated

Class Name:

Tooltip:

Data set:

□ Task Part

NOTE

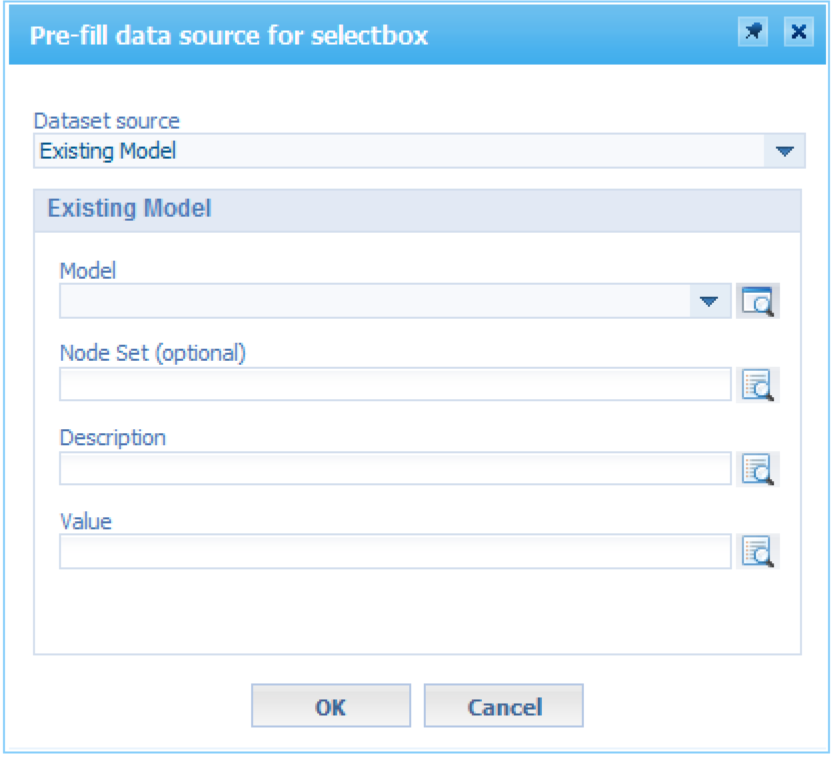
The fields on your form are in fact controls. Each control type has his own set of properties like ID, reference etc.

The reference indicates that the field CategoryID is used for the value. However the model field is not filled here, how does the control know in which model to look for the CategoryID?

____

____

5. Click **Edit Dataset Options** ( ) for the *Data set*.



Pre-fill data source for selectbox

Dataset source: Existing Model

Existing Model

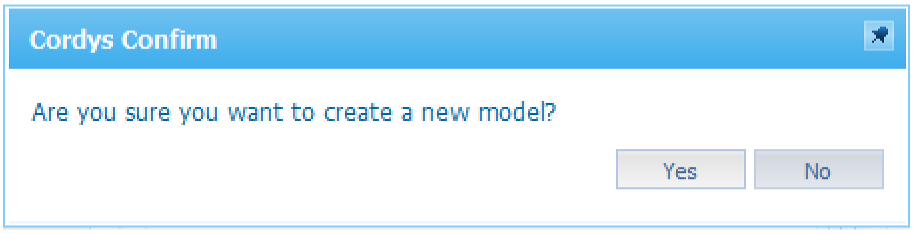
Model:

Node Set (optional):

Description:

Value:

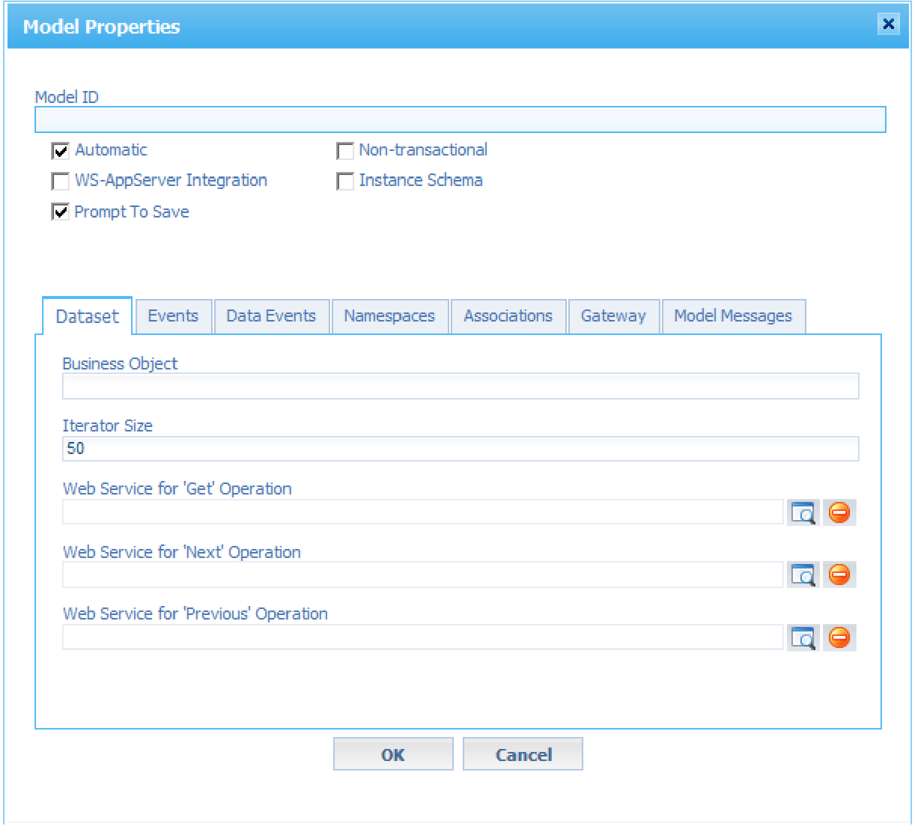
6. Select *Existing Model* as the Dataset source.
7. Click **Create New Model** ( ).
8. Click **Yes** to confirm creation of the model.



Cordys Confirm

Are you sure you want to create a new model?

9. The *Model Properties* are opened.



Model Properties

Model ID

☑ Automatic
☐ WS-AppServer Integration
☑ Prompt To Save
☐ Non-transactional
☐ Instance Schema

Business Object

Iterator Size: 50

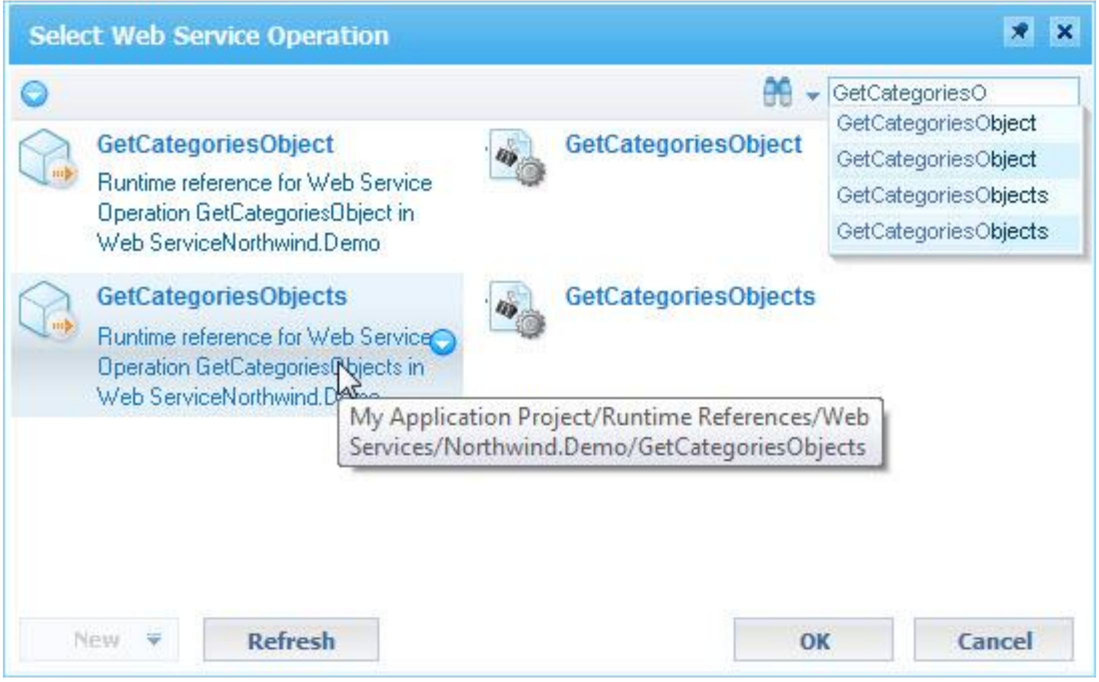
Web Service for 'Get' Operation

Web Service for 'Next' Operation

Web Service for 'Previous' Operation

10. Click the lookup button for the *Web Services for 'Get' Operation*.

11. Select the *GetCategoriesObjects* of the *Northwind.Demo* Web Service.



Select Web Service Operation

GetCategoriesO

GetCategoriesObject
Runtime reference for Web Service Operation GetCategoriesObject in Web ServiceNorthwind.Demo

GetCategoriesObject

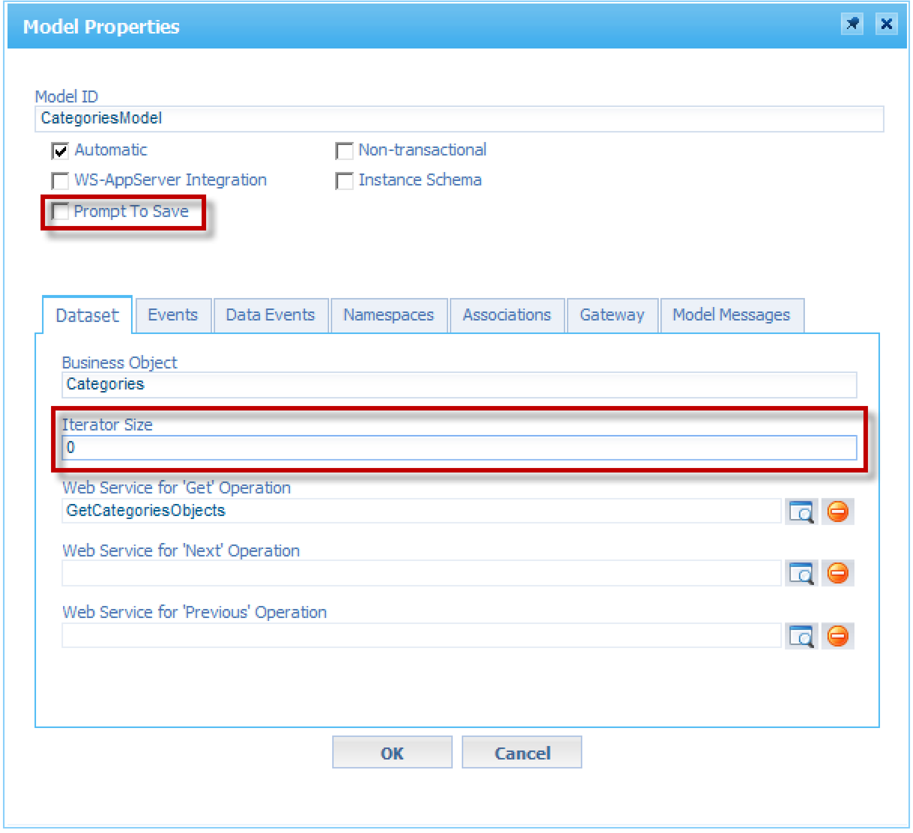
GetCategoriesObjects
Runtime reference for Web Service Operation GetCategoriesObjects in Web ServiceNorthwind.Demo

GetCategoriesObjects

My Application Project/Runtime References/Web Services/Northwind.Demo/GetCategoriesObjects

12. In the model properties, Uncheck *Prompt To Save*.

13. Change the *Iterator Size* to **0** (all categories need to be fetched).



Model Properties

Model ID
CategoriesModel

☑ Automatic
□ Non-transactional
□ WS-AppServer Integration
□ Instance Schema
□ Prompt To Save

Business Object
Categories

Iterator Size
0

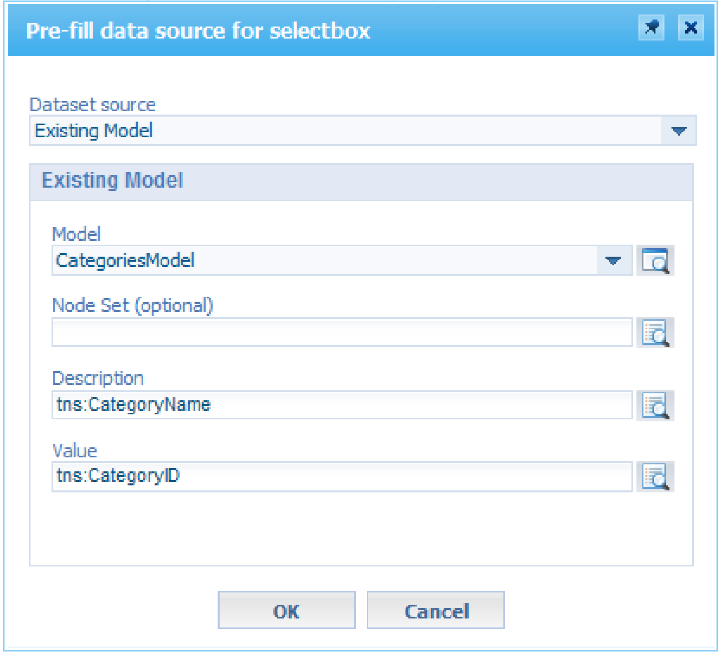
Web Service for 'Get' Operation
GetCategoriesObjects

Web Service for 'Next' Operation

Web Service for 'Previous' Operation

14. Click **OK**.

15. For *Description* select *CategoryName*, for *Value*: *CategoryID*.



Pre-fill data source for selectbox

Dataset source
Existing Model

Existing Model

Model
CategoriesModel

Node Set (optional)

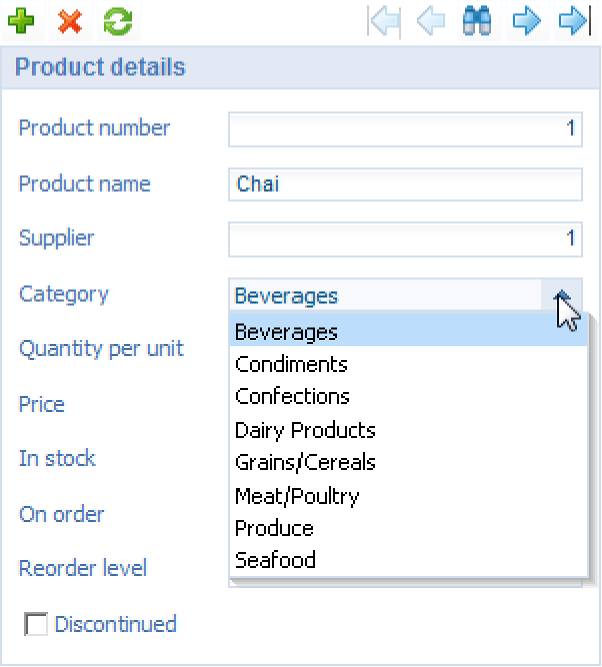
Description
tns:CategoryName

Value
tns:CategoryID

16. Click **OK**.

17. **Save** your form.

18. Preview the form and check proper functioning.



Product details

Product number 1

Product name Chai

Supplier 1

Category Beverages

Quantity per unit

Price

In stock

On order

Reorder level

□ Discontinued

3.5.2 Associating Models

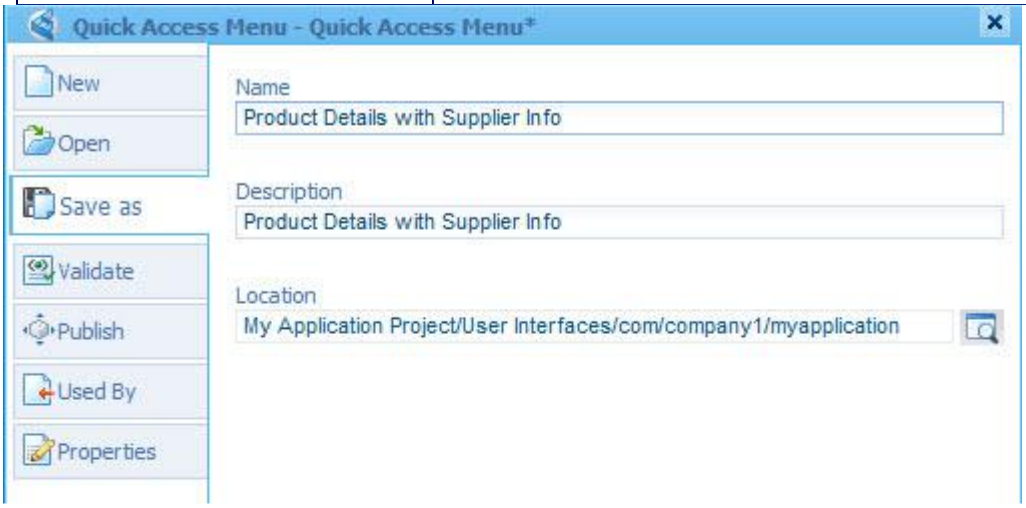
In previous exercises you have used models as standalone data instances in one form. In this part you will use multiple models and associate them, to retrieve related data. You will create a master-detail form, where employees of a department are shown on a separate tab page. You will use services from the same backend but note that this is not required; you can link any type of web service operations together.

Creating the Form

1. Open the User Interface *Product Details*.
2. Click **Menu** ( ) in the toolbar.
3. Click **Save as** (Save as).

4. Provide the following values:

Field	Value
Name	Product Details with Supplier Info
Description	Product details with supplier info
Location	Prefilled with: My Application Project/User Interfaces/com/companyX/myapplication



Name: Product Details with Supplier Info

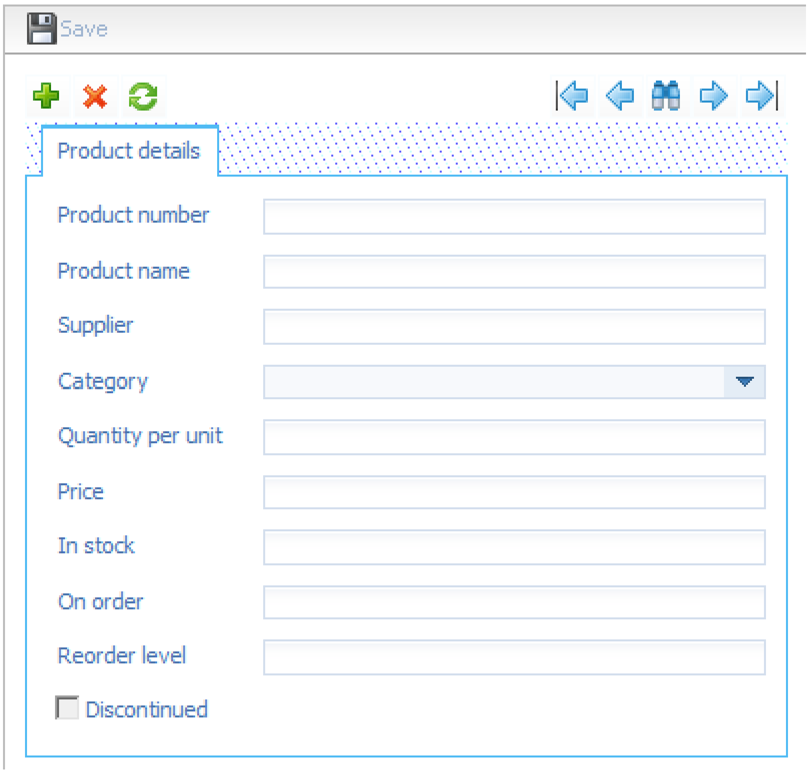
Description: Product Details with Supplier Info

Location: My Application Project/User Interfaces/com/company1/myapplication

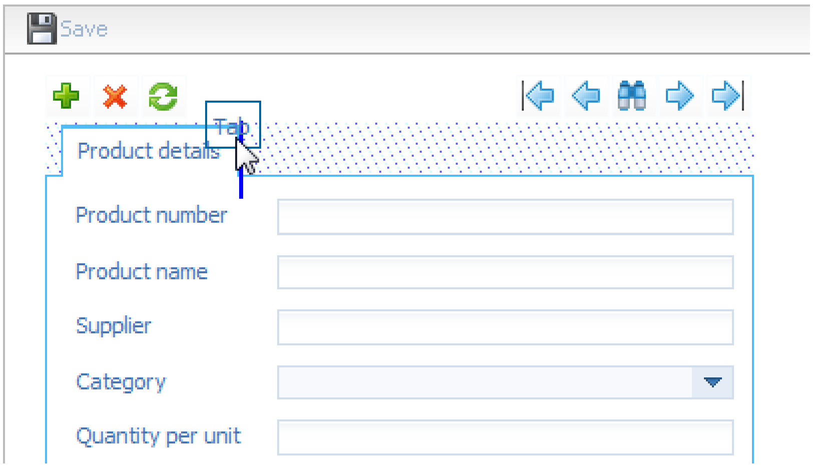
6. Click **Save**.

Creating Tab Pages

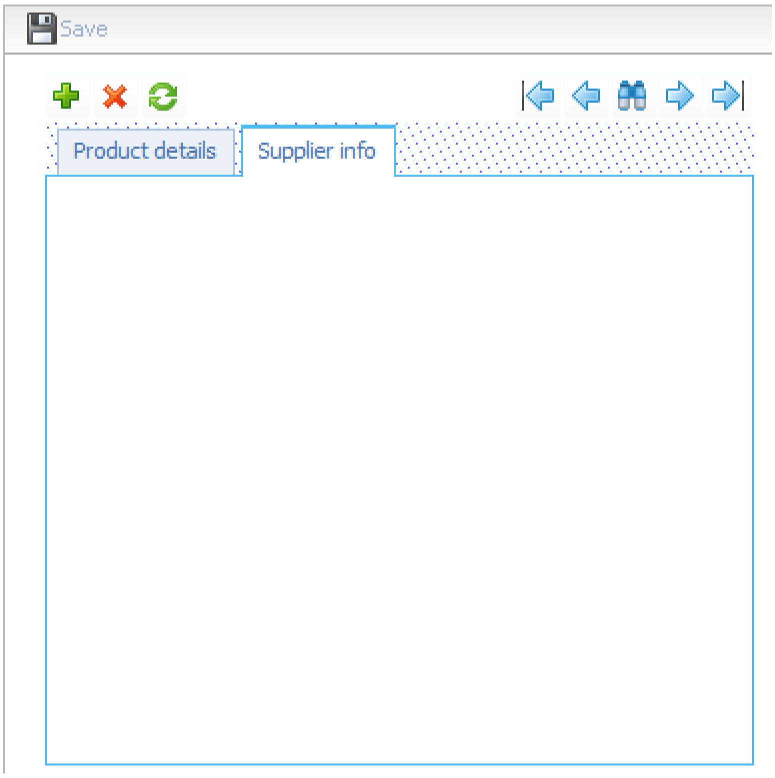
1. Right click the *Product details* Group Box and select *Change To → Tabs*.



2. Drag and drop a new *Tabpage* construct to the right of the existing one (a blue line will indicate target location before dropping).



3. Change the label of the tab to **Supplier info**.
4. Your form should like this:



5. **Save** your form.

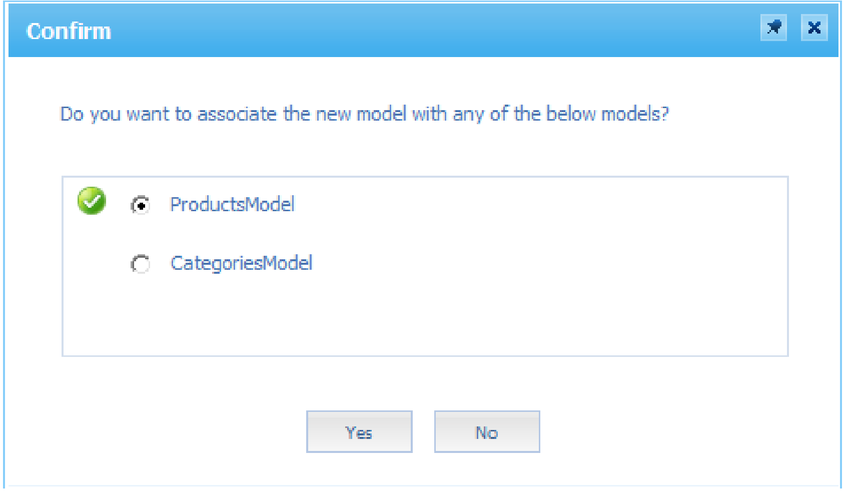
Associating Models

You want to see the supplier for the currently displayed product. Therefore you need a web service operation that fetches the supplier. In this case you use the *GetSuppliersObject* web service operation.

1. Select the *Workspace* tab and navigate to *Runtime References → Web Services → Northwind.Demo*.
2. Drag and drop the operation *GetSuppliersObject* in the created *Supplier info* tab page.
3. Click **OK** to generate UI for output only.

An association screens opens. This box shows all available models you can use to associate this new model to.

4. Check the *ProductsModel* (view the tooltip of the icon that appears next to the model).



Do you want to associate the new model with any of the below models?

✓ ● ProductsModel

○ CategoriesModel

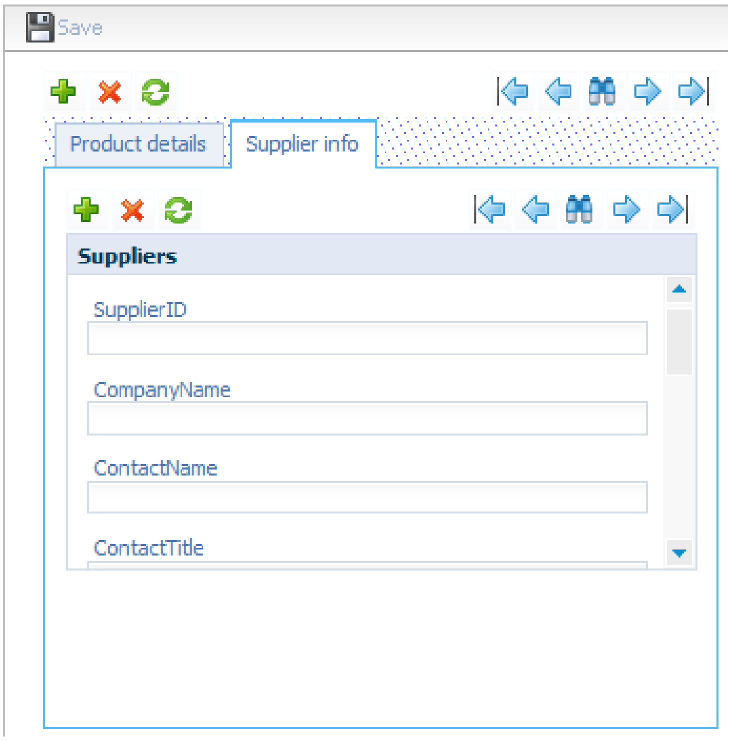
NOTE

An association will be successfully made when the input parameter(s) of the web service operation you are adding can be found with the exact same name in the model that you are associating to. In the case of this exercise: the *GetSuppliersObject* web service operation has *SupplierID* as the input parameter which can be found as well in the *ProductsModel* with the exact same name.

In case these names do not exactly match, the association between the models is made, but needs to be manually edited. For this, go to the *Associations* tab of the model properties.

5. Click **Yes**.

6. Your form should look like this:



7. Right click the Suppliers groupbox and delete the *control bar* and the *pagination bar*.
8. Right click the groupbox and change it to a group.
9. Set the labels to the left.
10. Change the labels and size of the fields, tab, etc to your liking.
11. **Save** your form.
12. Preview the form.

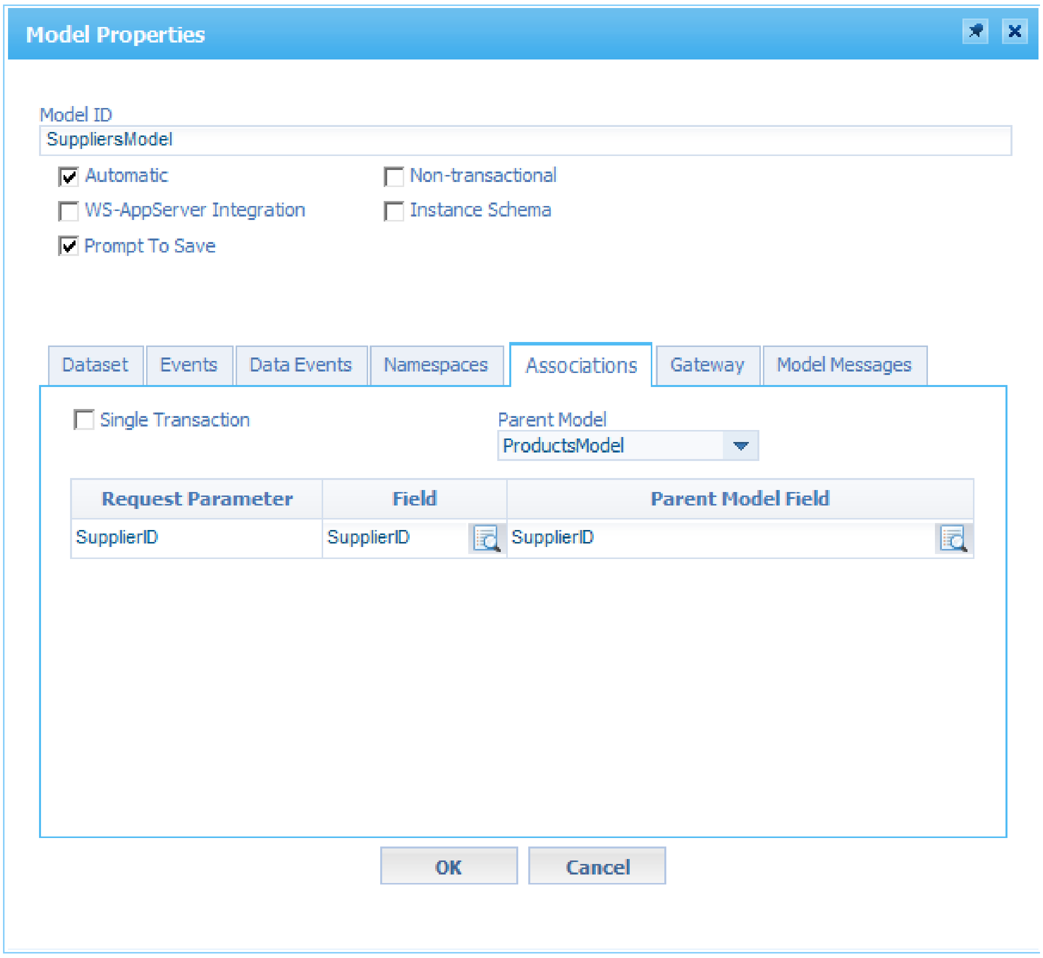
Is the correct Supplier shown for the selected product?

____

____

13. Return to the design form.
14. Click **Manage Models** in the toolbar.
15. Click the *SuppliersModel* model to open it.
16. Remove the web service for the *Next* and *Previous* operations.
17. Go to the *Associations* tab page.

18. Check the association that is automatically made, but do not make any changes.



Model ID
SuppliersModel

☑ Automatic ☐ Non-transactional
☐ WS-AppServer Integration ☐ Instance Schema
☑ Prompt To Save

☐ Single Transaction

Parent Model
ProductsModel

Request Parameter	Field	Parent Model Field
SupplierID	SupplierID	SupplierID

19. Click **OK**.

20. Save the form and preview.

21. Navigate in the preview through some of the products and check the suppliers tab.

Is the correct Supplier for a product shown now?

____

____

How can the performance of this form be improved (use SOAP Debugger)?

____

____

3.6 Make Changes Available to SCM

In this exercise you will make your developed content/changes available to your team members by sending the changes to the SCM application.

You should only make changes available after you have tested these to work correctly. This is to ensure that your team members' work is not affected when they incorporate your changes.

> **NOTE**
> This only applies when your workspace is created using an SCM application.

1. If closed open the Workspace Documents.
2. Click **Make Changes Available to Others** ( ) in the toolbar.
3. Review the modified content.
4. Provide as comment **User Interfaces and Web Services**.
5. Click **Make Available**.

4. User Interfaces at Runtime

At runtime forms can be accessed via the CUSP, but also stand alone via a URL.

In this chapter you will first configure access for the user interface via CUSP. Afterwards you will access it directly without going to Cordys.

4.1 Accessing a User Interface via the CUSP

In this exercise you will configure access to a user interface via CUSP.

4.1.1 Creating the Role

> **NOTE**
> The following steps are only required if you do not have the *Purchase Manager* role in the Roles folder.

1. If closed, open the *My Application Project*.
2. In *My Application Project* navigate to *Roles*.
3. Right click *Roles* and select *New → Other*.
4. Select *Role*.
5. Click **Save**.
6. Provide the following values:

Field	Value
Name	Purchase Manager
Description	Purchase managers
Location	Prefilled with: My Application Project/Roles

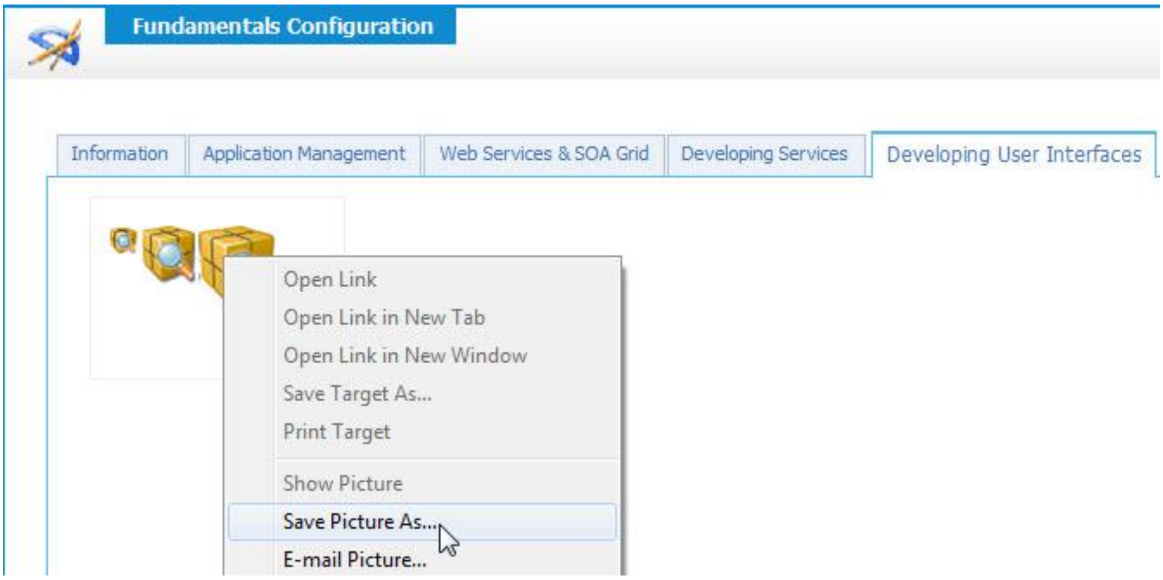
7. Click **Save**.

4.1.2 Creating Task Icon

The task will be added to the CUSP so you want to add an icon to go along with this task. In this part you will add the task icon to your project so it will be deployed with your package.

1. Open the *Fundamentals Configuration* (Fundamentals Configuration) via *My Application*.
2. Go to the tab *Developing User Interfaces*.

3. Right click in the image and Select *Save Picture As...*
→ Menu option name differs per browser



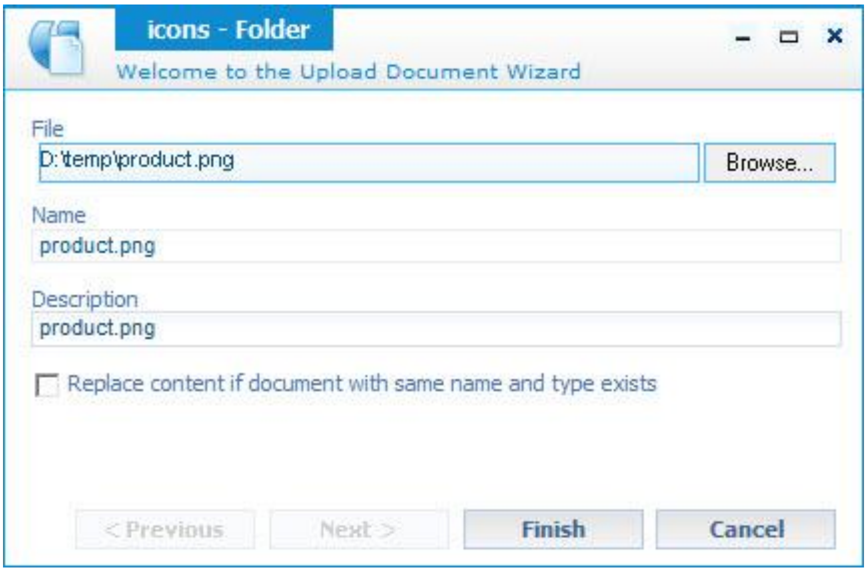
4. Save the file on your local machine as **product.png**.

7. In the *Workspace Document* tree navigate to *Cordys Web dir → com → companyX → myapplication*

5. Add a subfolder named **icons**.

6. Right click the *icons* folder and select *Upload Document*.

7. **Browse** and select the downloaded file.



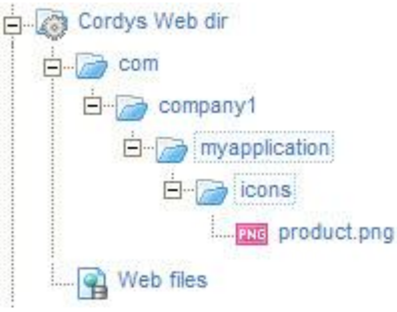
File
D:\temp\product.png

Name
product.png

Description
product.png

□ Replace content if document with same name and type exists

8. Accept the default values and click **Finish**.



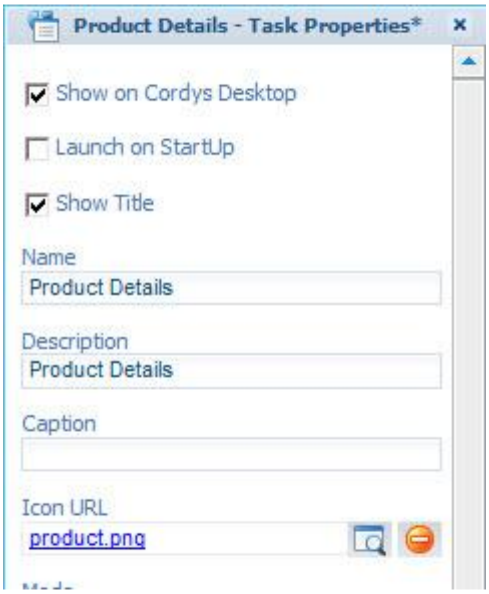
NOTE

When you have multiple files to upload to your project (e.g. styles, JavaScript) it is easier to use the import function to your SCM application or use the synchronize function of the Workspace Document.

4.1.3 Task Properties

In this part you will set the task properties of a user interface.

1. Open the *Product Details* form you created earlier.
2. Click **Task Properties** () in the toolbar.
3. Check *Show on Cordys Desktop*, otherwise the task will not be displayed in the CUSP.
4. Click the **Lookup** button to select *Icon URL*.
5. Select the *product.png*, you uploaded before.



Product Details - Task Properties*

☑ Show on Cordys Desktop

□ Launch on StartUp

☑ Show Title

Name: Product Details

Description: Product Details

Caption:

Icon URL: product.png

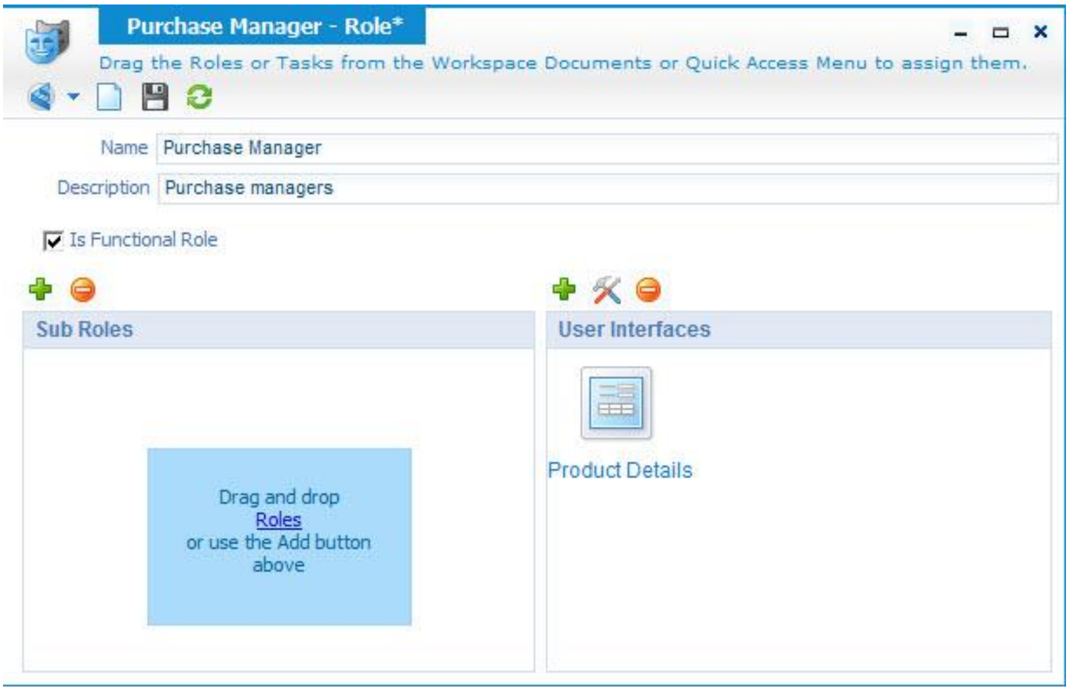
6. **Save** the form.
7. Close the form

4.1.4 Assign User Interface to a Role

In this part you will assign a user interface to a role.

1. In the *Workspace Document* navigate to *Roles*.
2. Open the *Purchase Manager* role.

3. Click the add button (✚) at *User Interfaces* panel to select the *Product Details* form:



Purchase Manager - Role*

Drag the Roles or Tasks from the Workspace Documents or Quick Access Menu to assign them.

Name: Purchase Manager

Description: Purchase managers

☑ Is Functional Role

Sub Roles

Drag and drop Roles or use the Add button above

User Interfaces

Product Details

4. **Save** the role and close.
5. In the *Workspace Documents* tree, right click the folder Roles and select *Publish to Organization.*

The related Cordys Web dir icon and the user interface will be published automatically as well.

4.1.5 Accessing a User Interface at Runtime

In this part you will assign a role to a user so that the user can work with the assigned user interfaces.

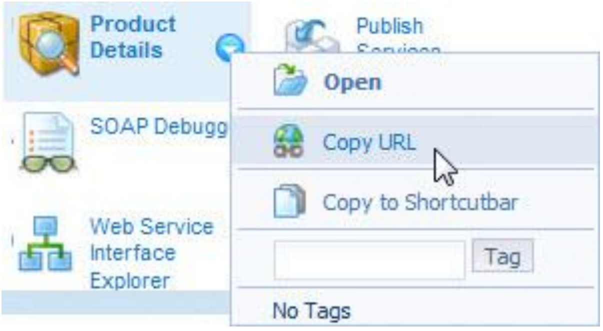
1. Open the *User Manager* (User Manager).
2. Assign yourself the role *Purchase Manager*.
3. Open the *My Application App Palette*.
4. You should now see the *Product Details* (Product Details) artifact.
5. Open the form and check the proper functioning of the user interface.

4.2 Using Forms Standalone

In this exercise you will see how you can use user interfaces outside the Cordys environment. You will simulate usages of a portal by directly using the URL of a user interface.

1. Expand *My Application App Palette*.
2. Hoover the mouse above the *Product Details* artifact.

3. Click the blue arrow and select *Copy URL*:



4. Copy the URL.
5. Open a new Browser or tab and paste the copied URL in the address bar, provide user credentials when asked.

At the end of the URL there is a variable Organization=xxxx is this organization name mandatory?
http://cordysserver/cordys/com/company1/myapplication/Product Details.caf**?organization=o%3DOrganization1%2Ccn%3Dcordys%2Ccn%3Ddef aultInst%2Co%3Dcordys.com**

____

____

6. Study the URL.
7. Change the URL so it displays the *Customer Information* instead of the *Product Details*.
NOTE: You can just type the space in the URL.

4.3 Make Changes Available to SCM

In this exercise you will make your developed content/changes available to your team members by sending the changes to the SCM application.

You should only make changes available after you have tested these to work correctly. This is to ensure that your team members' work is not affected when they incorporate your changes.

NOTE
This only applies when your workspace is created using an SCM application.

1. If closed, open the Workspace Documents.
2. Click **Make Changes Available to Others** (📝) in the toolbar.
3. Review the modified content.

4. Provide as comment **User Interfaces at Runtime**.
5. Click **Make Available**.

5. WS-AppServer Integration

When you build your web services using WS-AppServer to be able to add business logic, this logic can be used in your user interfaces as well. In this exercise you will study the integration between WS-AppServer and your forms.

5.1 Prerequisites

You must have completed the module *Developing Web Services* in order to complete this exercise. When you did not perform that module you can just ignore this exercise and continue with the next exercise.

5.2 Creating the Form

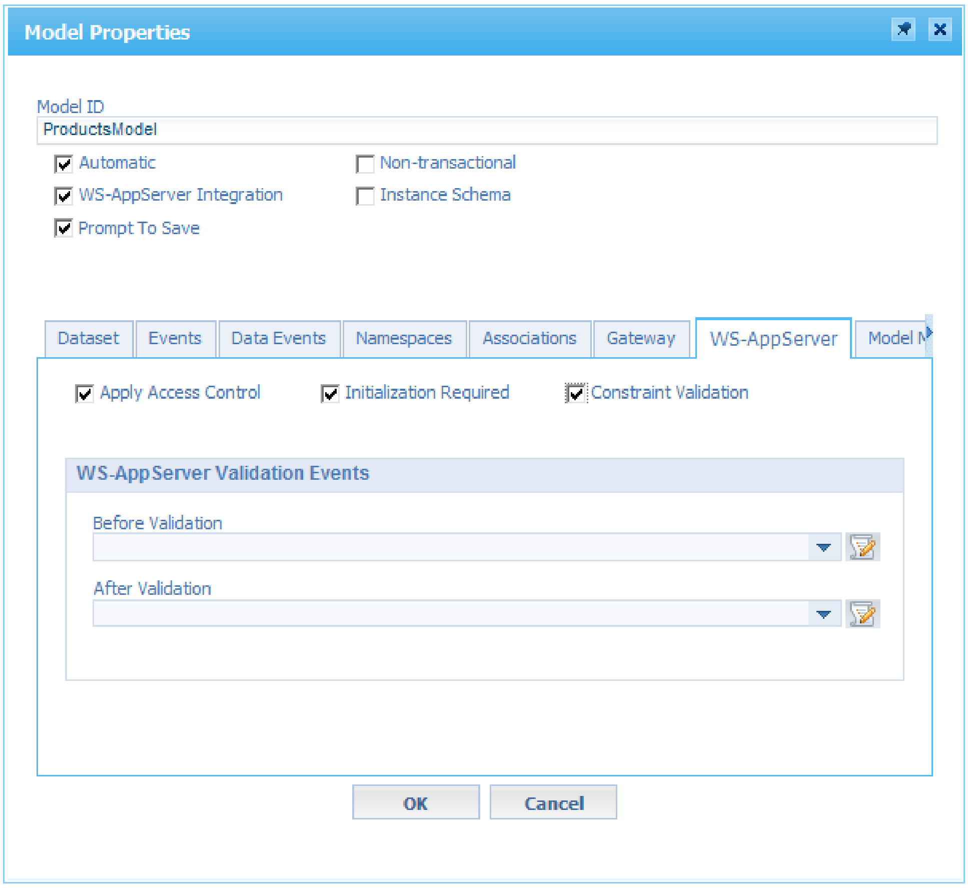
1. Create a new *User Interface* with the following details:

Field	Value
Name	Product Details with Logic
Description	Product details with WSAppServer logic
Location	My Application Project/User Interfaces/com/companyX/myapplication

5.3 Adding Web Service Operations to the Form

1. In the Workspace tab navigate to *Web Services → Northwind → Basic*.
2. Drag and Drop *GetProductsObject* in to the form editor.
3. Check only *Generate UI for Output messages*.
4. Click **OK**.
5. Open the *ProductsModel* properties.
6. Check *WS-AppServer Integration*.

7. On the *WS-AppServer* tab page, check *Apply Access Control, Initialization Required* and *Constraint Validation*.



Model Properties

Model ID: ProductsModel

☑ Automatic
☐ Non-transactional
☑ WS-AppServer Integration
☐ Instance Schema
☑ Prompt To Save

☑ Apply Access Control
☑ Initialization Required
☑ Constraint Validation

WS-AppServer Validation Events

Before Validation:

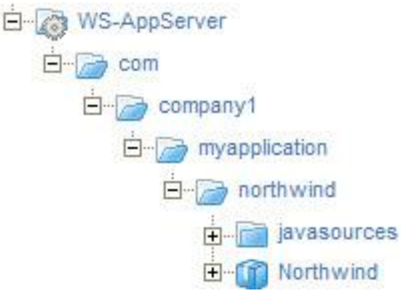
After Validation:

8. Click **OK**.
9. **Save** the form.
10. Preview the form.
11. Click **Insert** (✚).
12. Notice all fields are empty.
13. Close the preview without saving the new product.

5.4 Adding Static Business Logic

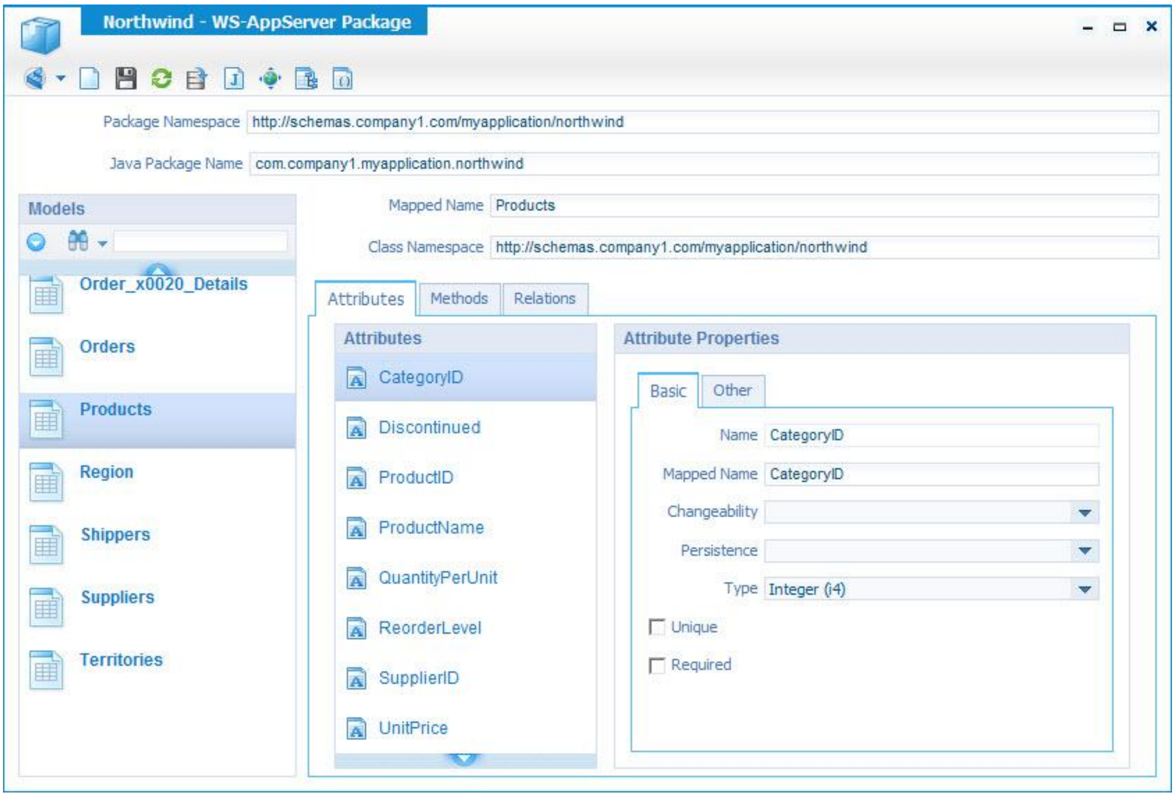
In this part you will add validation- and initialization constraints to the *Products* class in your WS-AppServer package.

1. In the *Workspace Documents* navigate to *WS-AppServer* → *com* → *companyX* → *myapplication* → *northwind*.



2. Double click the *Northwind* package to open it.

3. In the *Models* groupbox select the *Products* Model:



Northwind - WS-AppServer Package

Package Namespace: http://schemas.company1.com/myapplication/northwind

Java Package Name: com.company1.myapplication.northwind

Mapped Name: Products

Class Namespace: http://schemas.company1.com/myapplication/northwind

Name: CategoryID

Mapped Name: CategoryID

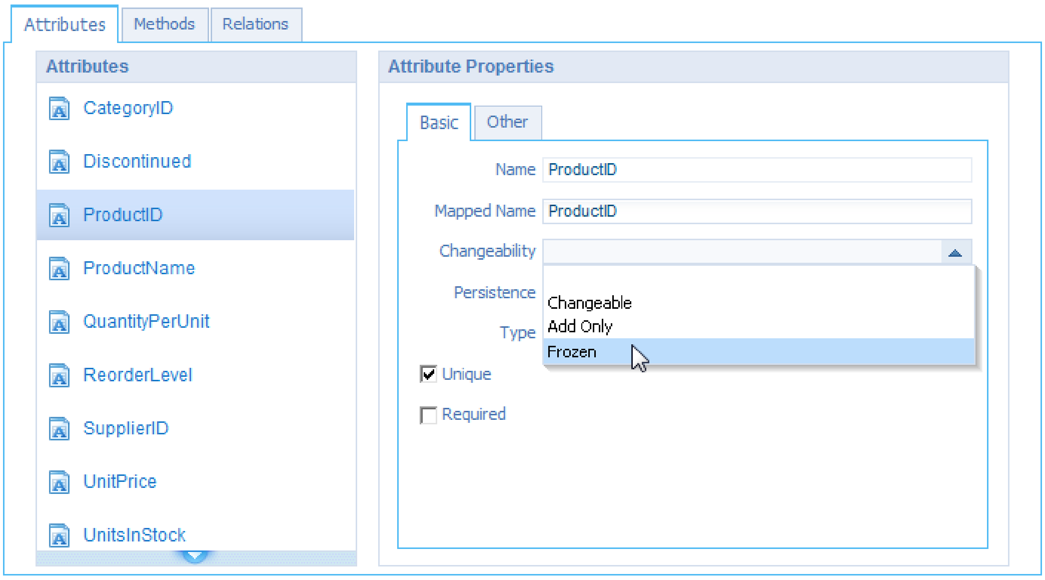
Type: Integer (i4)

□ Unique

□ Required

4. On the *Attributes* tab select the *ProductID*.

5. Change the *Changeability* to **frozen**.



Name: ProductID

Mapped Name: ProductID

☑ Unique

□ Required

6. Do the same for *UnitsInStock*.

7. Set the field *ProductName* as *Required*.

8. Select the *UnitPrice* and go to the *Other* tab.

9. Set *Min Inclusive* to **0**.

10. For the *ReorderLevel* set the *initial* value to **5**.

11. For the *Discontinued* set the *initial* value to **0** (0 = false, 1 = true).

12. **Save** the *Northwind* package.

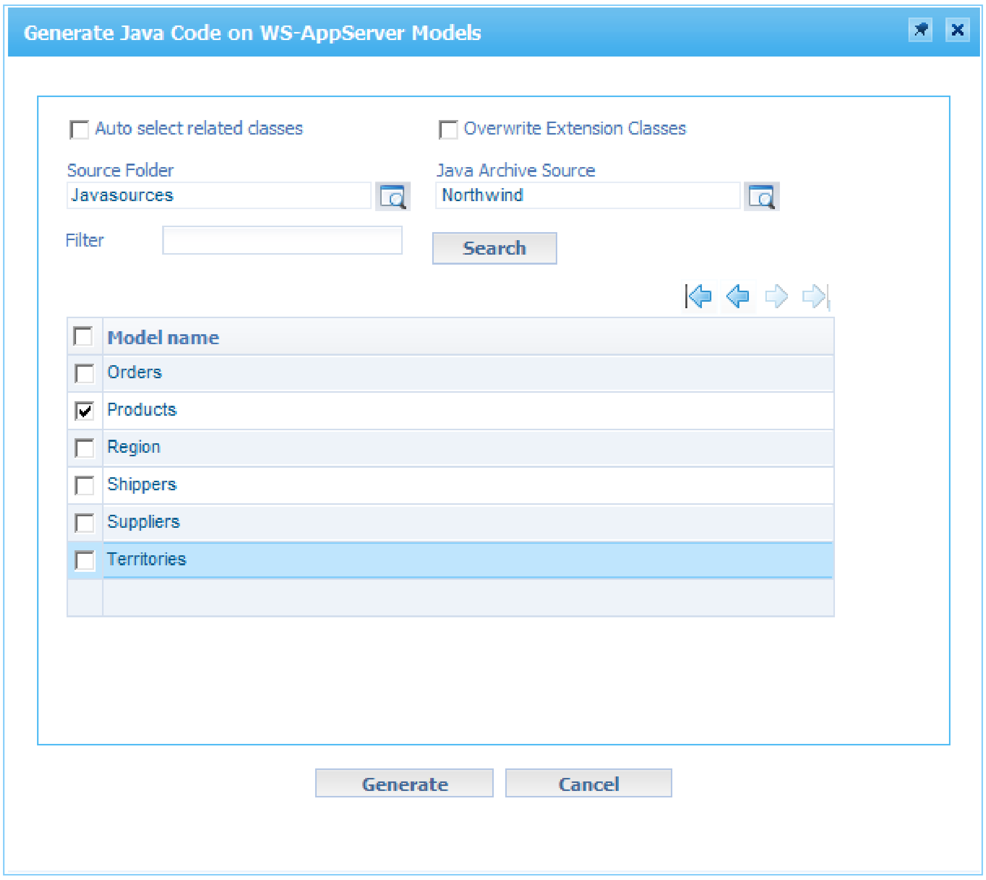
13. Click **Generate Java Code** (J) in the toolbar.

14. Deselect *Auto select related classes*.

15. Make sure *Overwrite Extension Classes* is UNCHECKED.

> **NOTE**
> The extension class might contain business logic you have created before and therefore should not been overwritten.

16. Check the *Products* model only:



Generate Java Code on WS-AppServer Models

□ Auto select related classes

□ Overwrite Extension Classes

Source Folder: Javasources

Java Archive Source: Northwind

Filter:

□	Model name
□	Orders
☑	Products
□	Region
□	Shippers
□	Suppliers
□	Territories

17. Click **Generate**.

18. Use the **Quick access** menu to *Publish* the package.

19. Close the *Northwind* package.

5.5 Viewing a Form with Business Logic

In this part you will see the effect of the logic added in the WS-AppServer code.

1. Return to the *Product Details with Logic* form.

2. Preview the form.

What is the effect of the logic while viewing a product?

____

____

3. Test the implemented business logic while inserting a new product.

   What is the advantage of working with server side logic?

   ____

   ____

5.6 Make Changes Available to SCM

In this exercise you will make your developed content/changes available to your team members by sending the changes to the SCM application.

You should only make changes available after you have tested these to work correctly. This is to ensure that your team members’ work is not affected when they incorporate your changes.

> **NOTE**
> This only applies when your workspace is created using an SCM application.

1. If closed, open the Workspace Documents.
2. Click **Make Changes Available to Others** (📝) in the toolbar.
3. Review the modified content.
4. Provide as comment **User Interfaces and WS-AppServer integration**.
5. Click **Make Available**.

6. User Interfaces in Business Processes

When you use user interfaces as a task in a process the data of the user interface become accessible in the message map of the process. Data entry forms usually are based on transactional web services, but in a process you can also use data which is not bound to a (transactional) web service to process data that have to be provided by human actors.

In this chapter you will create the same forms that will be used in the module workflow to be implemented in a business process.

6.1 Creating a Form with Non-transactional Data

In this exercise you will build a small form that will be used by the finance department to check the financial status of the customer.

Note that the form you create will use non-transactional data, which means that the data does not originate from a backend application, but from the process itself.

6.1.1 Create the User Interface

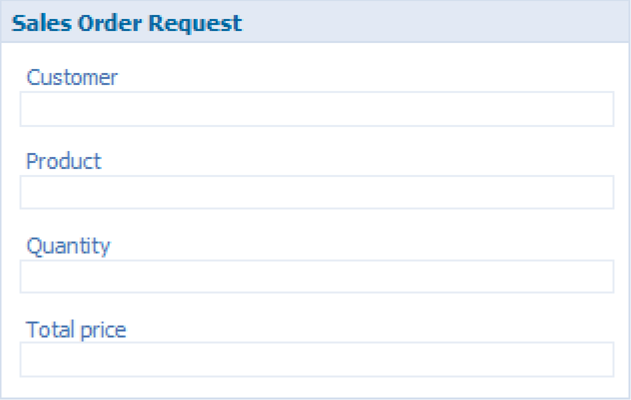
1. If closed, open the *Workspace Documents.*
2. Create a *new User Interface* with the following details:

Field	Value
Name	Customer Order Financial Check
Description	Check financial status for customer order
Location	My Application Project/User Interfaces/com/companyX/myapplication

6.1.2 Creating the Layout

1. From the *Toolbox* drag and drop a *Groupbox* into the form.
2. Drag and drop 4 *Output* fields into the *GroupBox*.
3. Click in the labels of the fields and rename them as follows:

Field	Value
Groupbox1	Sales Order Request
Output1	Customer
Output2	Product
Output3	Quantity
Output4	Total price


Sales Order Request

Customer

Product

Quantity

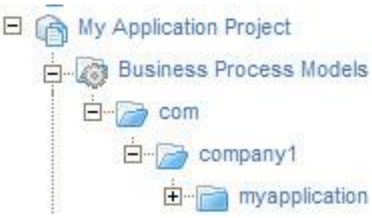
Total price

4. **Save** the form.

6.1.3 Using the Form in a Process

In this exercise, you will add the user interface to a business process.

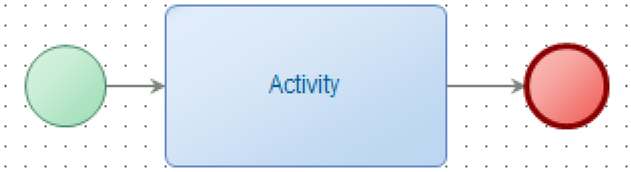
1. Navigate to *Business Process models → com → companyX → myapplication*.



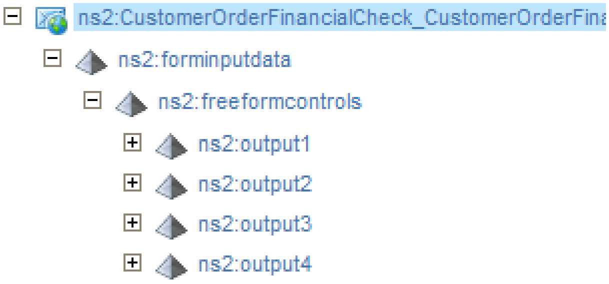
2. Right click the *myapplication* folder and create a new document of type *Business Process Model*.
3. **Save** the process with the following values:

Field	Reference
Name	FormDemo
Description	Demo using forms in processes
Location	Prefilled with: My Application Project /Business Process Models/com/companyX/myapplication

4. Right click in the processes and select *Model Properties* (F8).
5. Change the Namespace to:
   http://schemas.companyX.com/myapplication/salesprocesses
6. Create the following process design:



7. Use the *Insert* or *Workspace* tab to drag and drop the *Customer Order Financial Check* user interface on the activity.
8. Go to the *Message Map* tab.
9. Select the *Customer Order Financial Check* activity.
10. Expand the *forminputdata* node:



What needs to be done based on the presented structure?

____

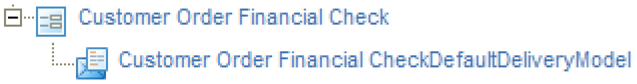
____

5. **Save** the process.
6. Minimize the process and leave it open for later use.

6.1.4 Viewing the Delivery Model

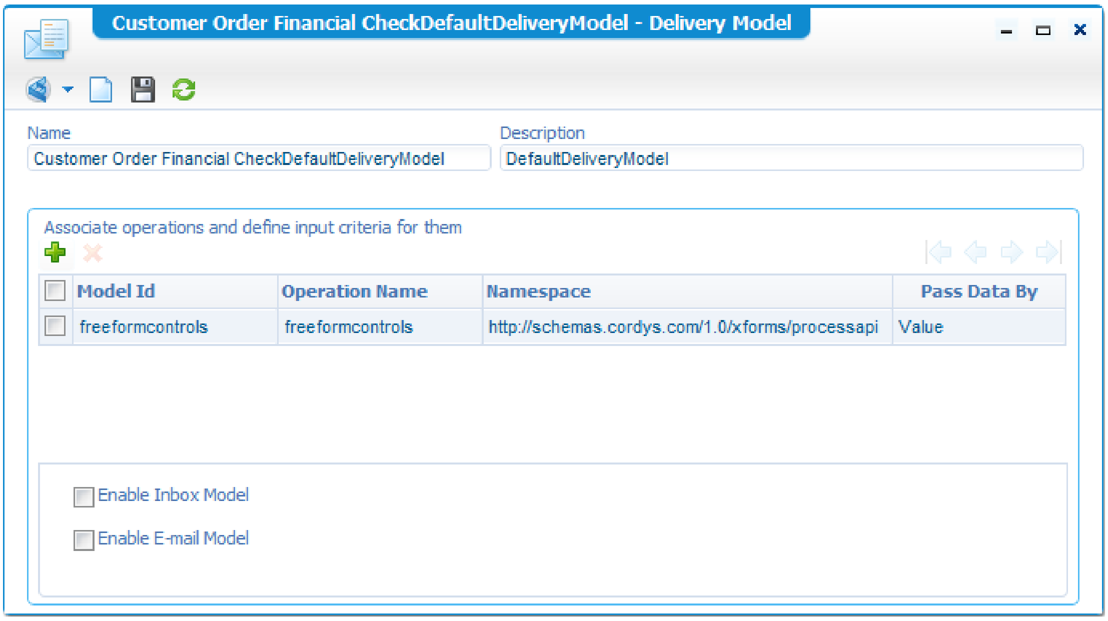
As soon as you drag and drop a user interface in a process (or case model) a delivery model is automatically created. In this part, you will review the created delivery model.

1. In the *Workspace Documents*, navigate to *User Interfaces → com → companyX → myapplication*.
2. Expand the *Customer Order Financial Check* form.
3. You will see the created *Customer Order Financial CheckDefaultDeliveryModel*:



Customer Order Financial Check
- Customer Order Financial CheckDefaultDeliveryModel

4. Open the *Delivery Model*:



Customer Order Financial CheckDefaultDeliveryModel - Delivery Model

Name: Customer Order Financial CheckDefaultDeliveryModel

Description: DefaultDeliveryModel

Associate operations and define input criteria for them

	Model Id	Operation Name	Namespace	Pass Data By
□	freeformcontrols	freeformcontrols	http://schemas.cordys.com/1.0/xforms/processapi	Value

□ Enable Inbox Model

□ Enable E-mail Model

The delivery model contains one message named *freeformcontrols*, which you may recognize from the input message in the message map.

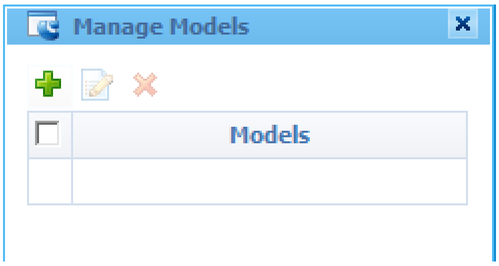
5. Close the delivery model without making changes.

6.1.5 Creating a Model in the form

In this part you will create a model that contains the data in the form instead of using the free form controls. You will reuse the same form to see the effect of the changes in the delivery models as well in the process.

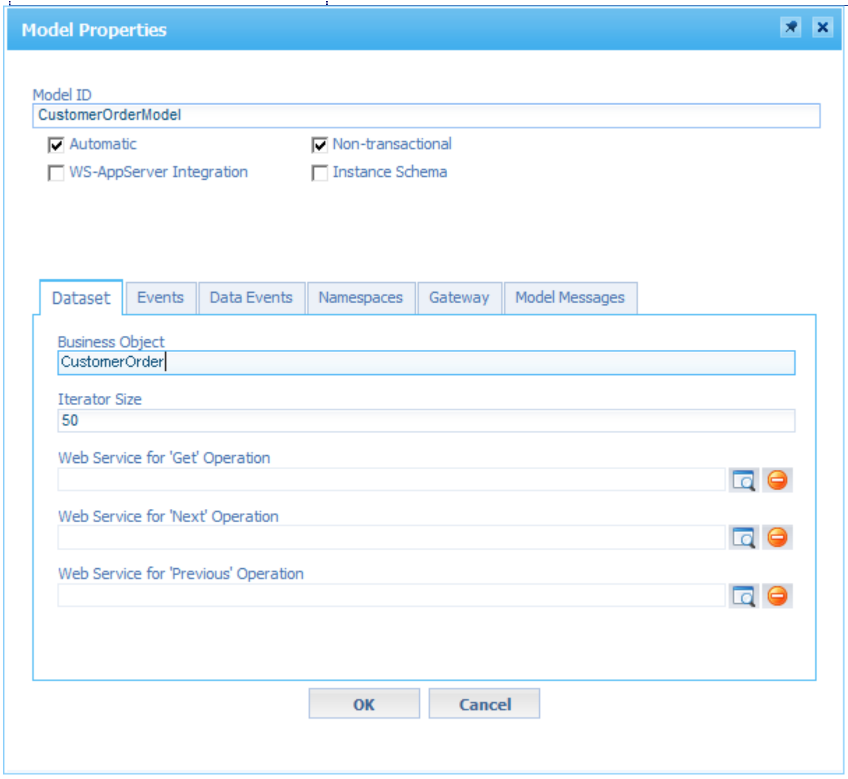
1. If closed open the form *Customer Order Financial Check*.

2. Click **Manage Models** ( ) in the toolbar.
   The *Manage Models* frame is opened at the right:



3. Click **Insert** (✚).
4. Provide the following values:

Field	Value
Model ID	CustomerOrderModel
Automatic	Checked
Non-transactional	Checked
Business Object	CustomerOrder



Model ID: CustomerOrderModel

☑ Automatic ☑ Non-transactional

☐ WS-AppServer Integration ☐ Instance Schema

Business Object: CustomerOrder

Iterator Size: 50

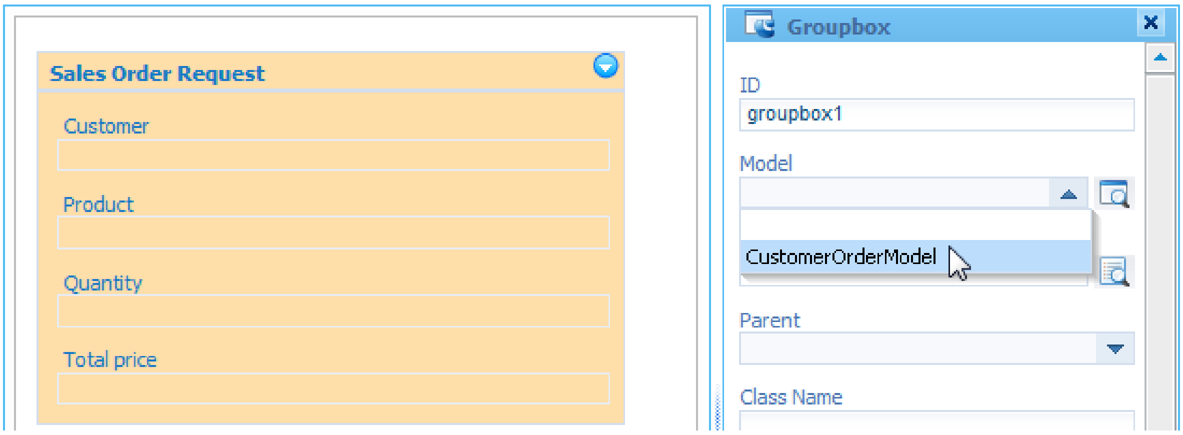
Web Service for 'Get' Operation:

Web Service for 'Next' Operation:

Web Service for 'Previous' Operation:

5. Click **OK**.
6. Open the *Properties* of the *Group Box*.

7. Select the *Model* you have created:



Sales Order Request

Customer

Product

Quantity

Total price

Groupbox

ID: groupbox1

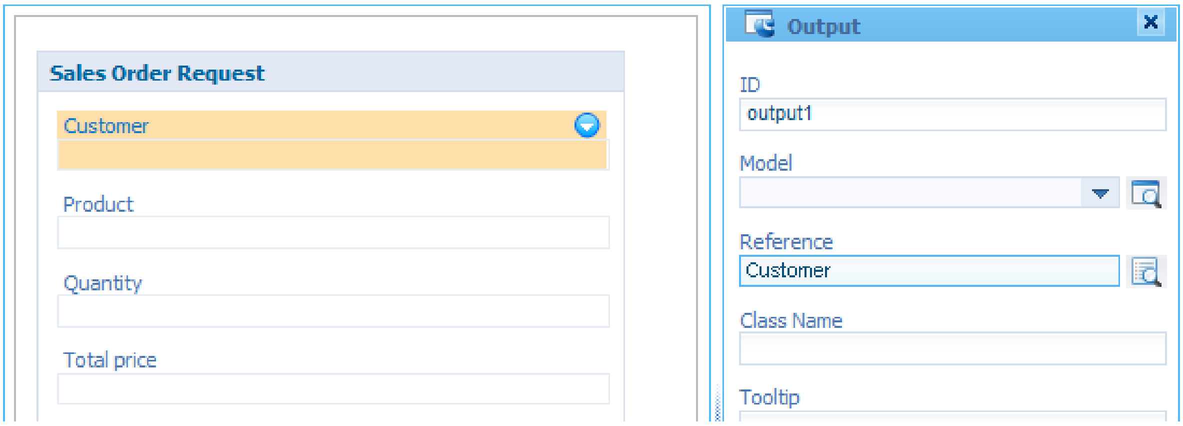
Model: CustomerOrderModel

Parent

Class Name

8. Select the *Customer* field to view the properties.

9. In the *Reference* field specify **Customer**:



Sales Order Request

Customer

Product

Quantity

Total price

Output

ID: output1

Model

Reference: Customer

Class Name

Tooltip

10. Provide the following references for the remaining fields:

Field	Reference
Product	Product
Quantity	Quantity
Total price	TotalPrice

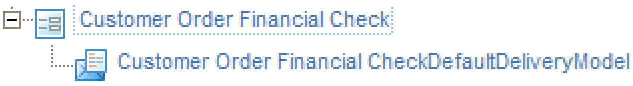
11. **Save** the form and close.

> **NOTE**
> In the properties of the fields you did not change the ID field, normally you should do that as well. To clearly show the difference in the business process model, between using free form fields and using the references via a model, it has been skipped for now.

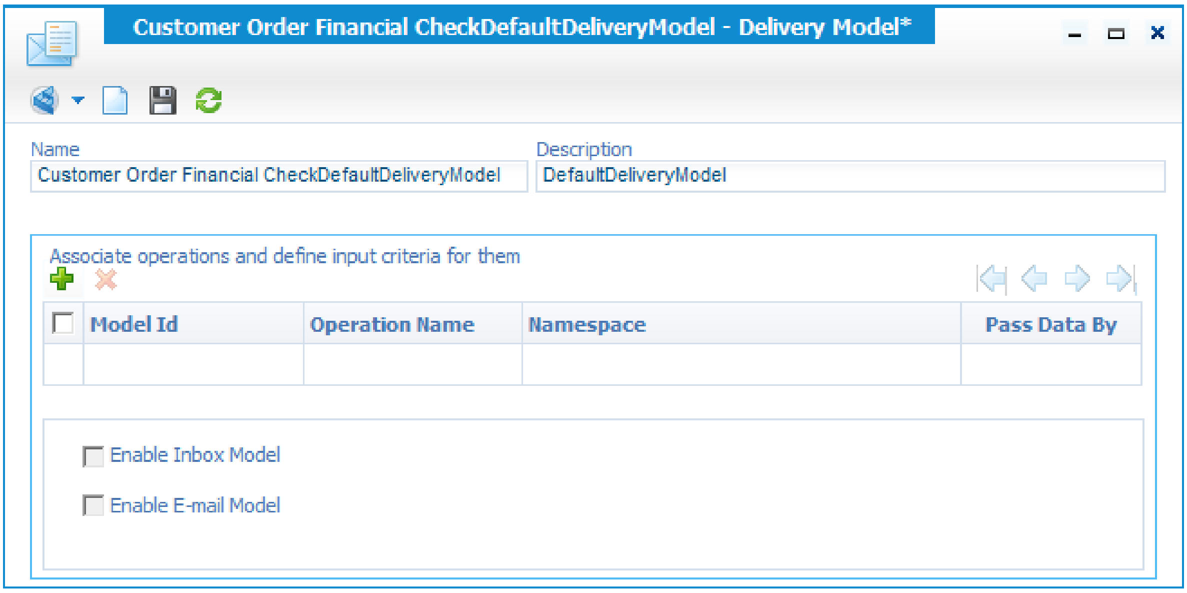
6.1.6 Updating the Delivery Model

When you make changes to a form in the context of models and fields, that is already used in a business process model or a case model it is possible that you have to update the delivery model. In this part you will update the delivery model by switching from freeform fields to the created model.

1. In the Workspace Documents navigate to *User Interfaces → com → companyX → myapplication*.
2. Expand the *Customer Order Financial Check* form.


Customer Order Financial Check
Customer Order Financial CheckDefaultDeliveryModel

3. Open the delivery model.


Customer Order Financial CheckDefaultDeliveryModel - Delivery Model*

Name: Customer Order Financial CheckDefaultDeliveryModel

Description: DefaultDeliveryModel

Associate operations and define input criteria for them

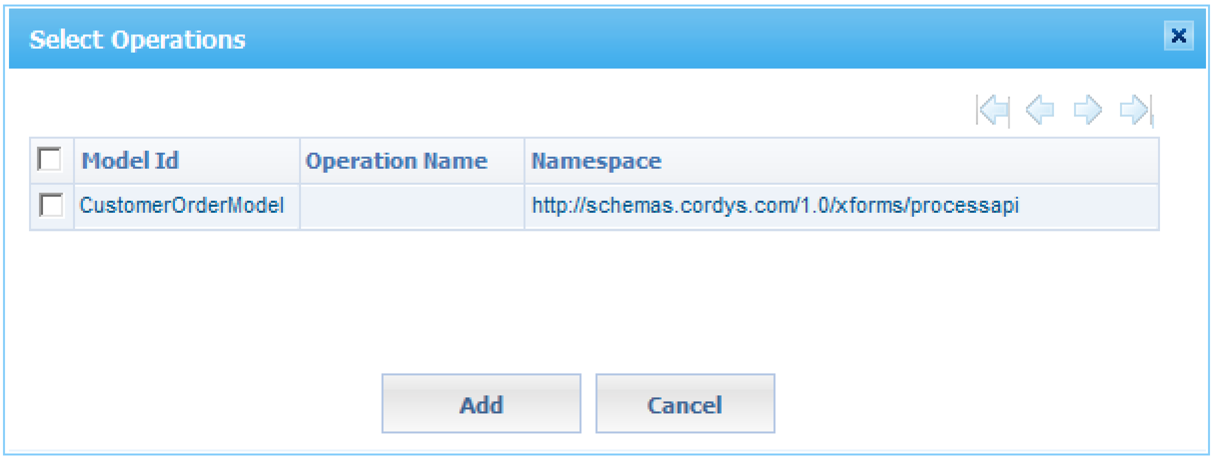
□	Model Id	Operation Name	Namespace	Pass Data By

□ Enable Inbox Model

□ Enable E-mail Model

The *freeformcontrols* model is already removed as there are no freeform fields anymore. As Cordys does not know if you have made changes to this delivery model, new models are not added automatically. You must do this manually, as it is your choice to add models to the delivery model and decide to show them in, for example, the process.

4. Click **add**.
5. Check the *CustomerOrderModel*.



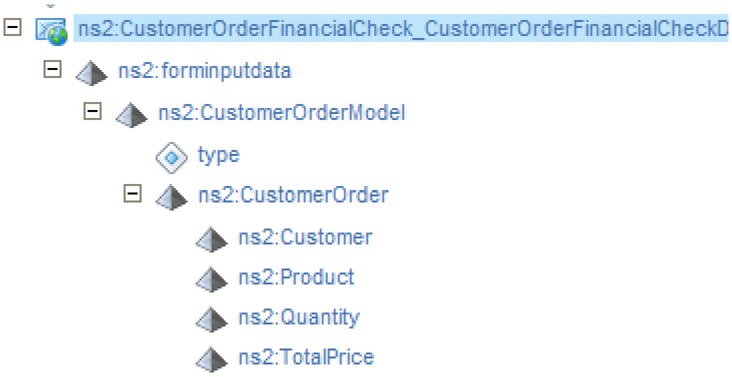
□	Model Id	Operation Name	Namespace
□	CustomerOrderModel		http://schemas.cordys.com/1.0/xforms/processapi

6. Click **Add**.
7. **Save** the delivery model.

Now the delivery model is updated and the form changes will be reflected in the message map of the process as well.

8. Return to the *FormDemo* process that you left open before.
9. Go to the *Message Map* tab.
10. Select the *Customer Order Financial Check* activity.
11. Expand the *forminputdata* node.
12. Notice that it is still using the *freeformcontrols*.

13. Close and Reopen the *FormDemo* business process model.
14. Check the input message for the form again:



- ns2:CustomerOrderFinancialCheck_CustomerOrderFinancialCheckD
  - ns2:forminputdata
    - ns2:CustomerOrderModel
      - type
      - ns2:CustomerOrder
        - ns2:Customer
        - ns2:Product
        - ns2:Quantity
        - ns2:TotalPrice

What is the benefit of using a model in your user interface?

____

____

Why did you have to close and reopen the business process model?

____

____

> **NOTE**
> In the module *Workflow*, you will implement the business process using this user interface. For now, you can create 4 fixed value assignments if you need/want to test the process.

15. Close the *FormDemo* business process model.

6.2 Combining Transactional and Non-Transactional Data

In this part you will build a small form that is used by the purchasing department to check whether the product is in stock. In the workflow module, you will actually use this user interface. The form you create will use a combination of transactional data and non-transactional data:

- Transactional data means that the data is retrieved from a backend application, based on the value coming from the backend and allowing you to automatically update the data as well.
- Non Transactional data

  - Using a model retrieving data from a backend which cannot be updated automatically.
  - Using a model but data is only available in this form but can be exchanged with the form.
- Free form data means that the data is not linked to a model but the value exists as free fields in the form.

6.2.1 Create the User Interface

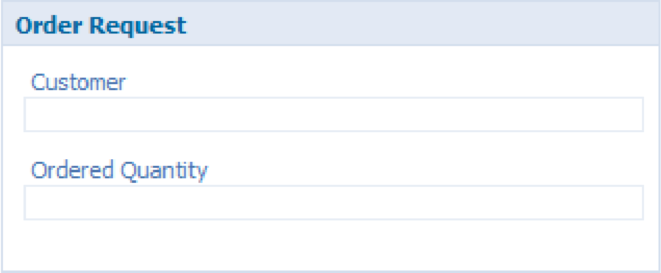
1. Create a new *User Interface* with the following details:

Field	Value
Name	Product Claim Ordered Quantity
Description	Product claim ordered quantity
Location	My Application Project/User Interfaces/com/companyX/myapplication

6.2.2 Creating the Layout

1. Drag and drop a *Groupbox* into the form.
2. Drag and drop 2 *Output* fields into the *Groupbox*.
3. Rename the labels of the fields as follows:

Field	Value
Groupbox1	Order Request
Output1	Customer
Output2	Ordered Quantity



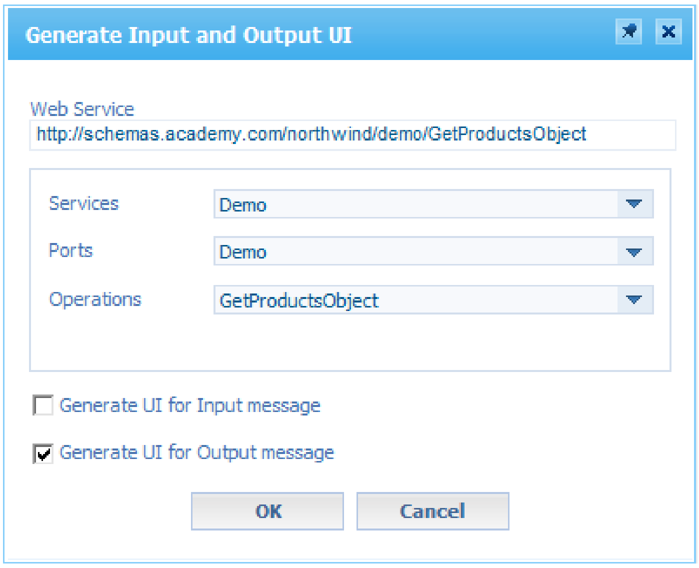
4. Create a new model with the following details:

Field	Value
Model ID	OrderModel
Automatic	Checked
Non-transactional	Checked
Business Object	Order

5. Link the model to the *groupbox*.
6. Give the two fields the *Reference* **Customer** and **OrderedQuantity**.

7. In the *Workspace* tab, navigate to *Runtime References → Web Services → Northwind.Demo*.

8. Drag and drop the *GetProductsObject* into the form.



Generate Input and Output UI

Web Service: http://schemas.academy.com/northwind/demo/GetProductsObject

Services: Demo

Ports: Demo

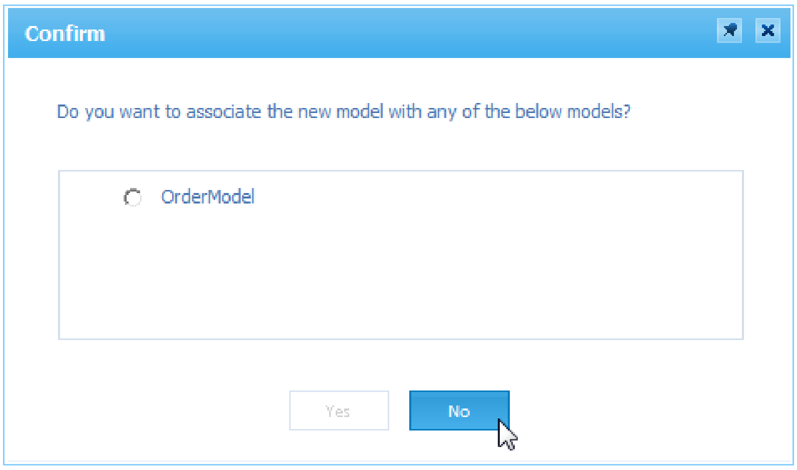
Operations: GetProductsObject

□ Generate UI for Input message

☑ Generate UI for Output message

9. Make sure *Generate UI for Input message* is **unchecked**. The input data for the product will come from the process.

10. Click **OK**.



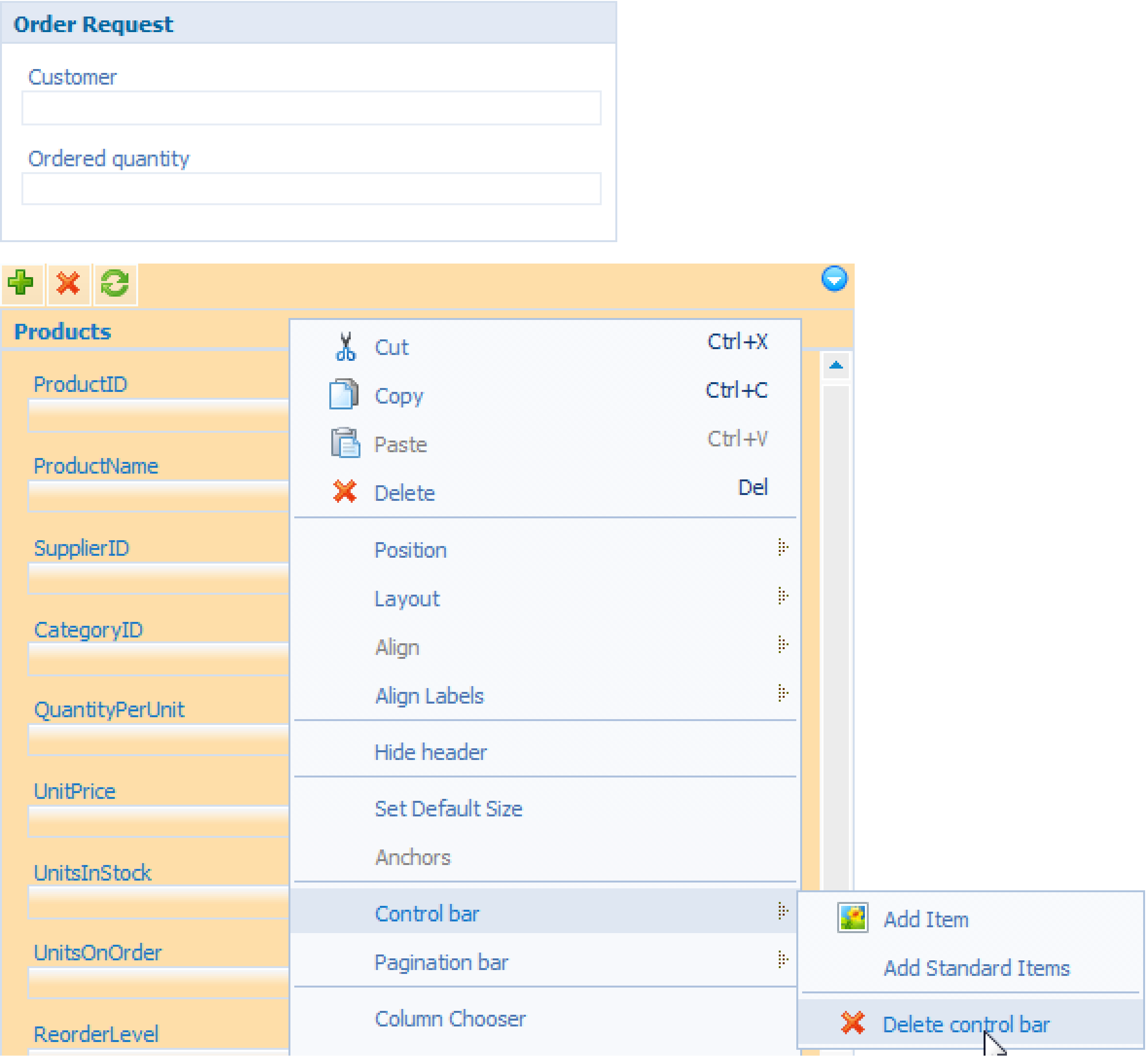
Confirm

Do you want to associate the new model with any of the below models?

○ OrderModel

11. Click **No**, as no association between the two models is required.

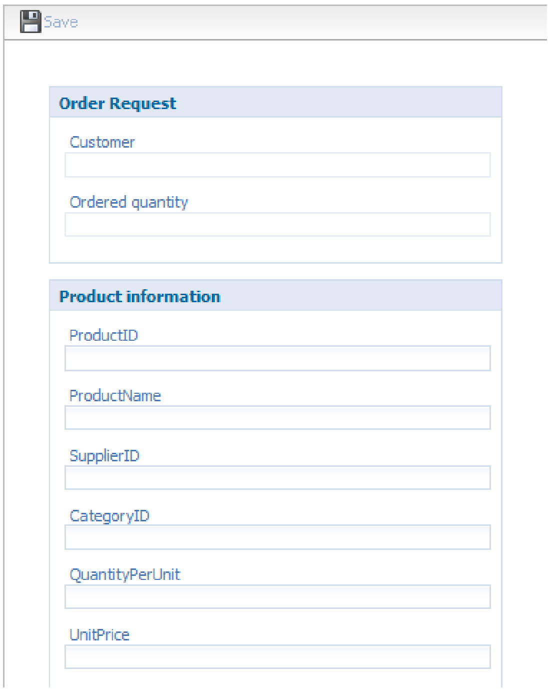
12. Right click the *Products groupbox* and select *Control bar → Delete control bar*.



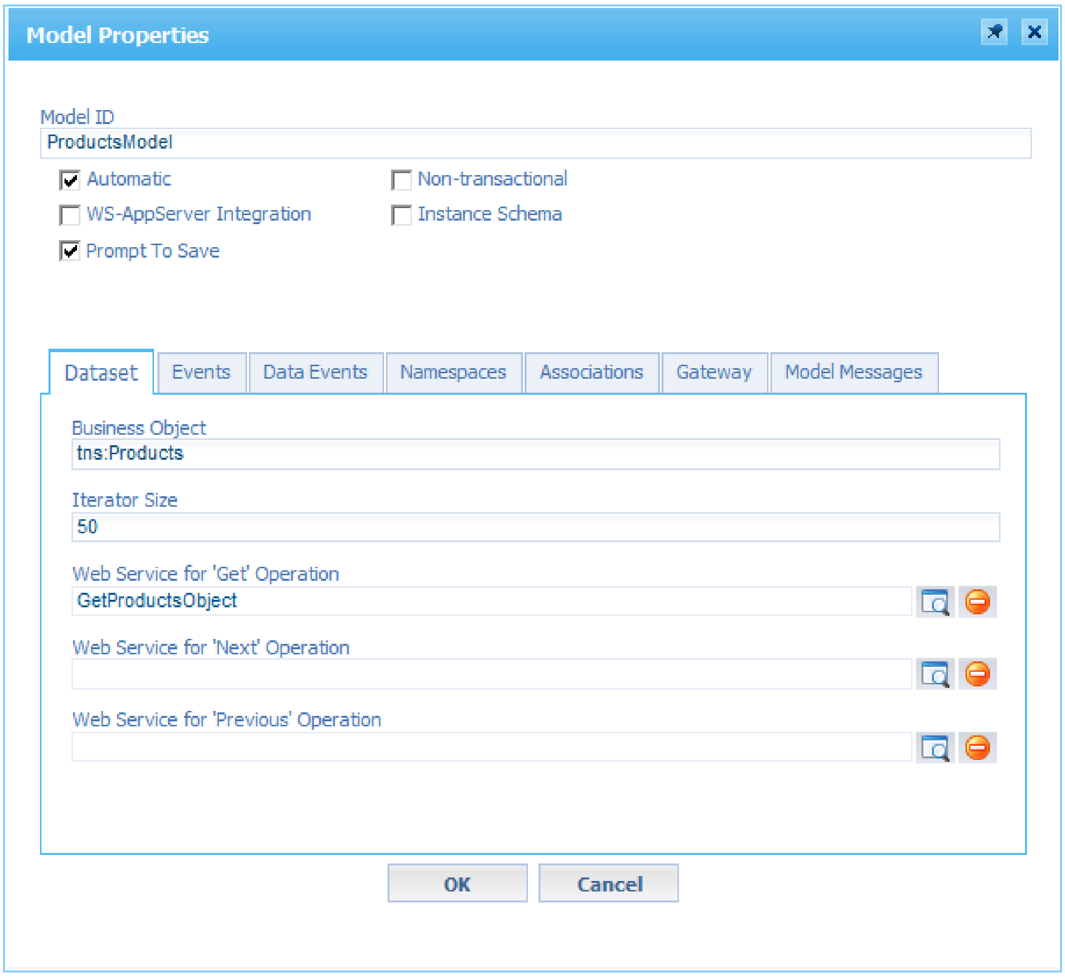
13. Remove the *Pagination bar* in the same way.
You will only use this user interface to display the relevant product data.

14. Change the label of the *Products groupbox* to **Product Information**.

15. Your form should look similar to this:



16. Click **Manage Models** in the toolbar.
17. Click the ProductsModel to open its properties.
18. Clear the Web Service operation for *Next* and *Previous*.



Model Properties

Model ID
ProductsModel

☑ Automatic
□ Non-transactional
□ WS-AppServer Integration
□ Instance Schema
☑ Prompt To Save

Business Object
tns:Products

Iterator Size
50

Web Service for 'Get' Operation
GetProductsObject

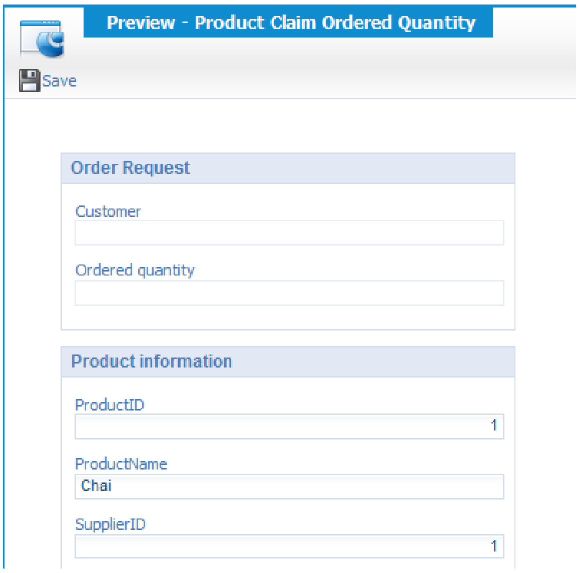
Web Service for 'Next' Operation

Web Service for 'Previous' Operation

19. Click **OK**.
20. **Save** the form.

6.2.3 Previewing the Form

1. Click **Preview** (👁) in the toolbar.
2. The preview displays the product with *ProductID* 1:



Preview - Product Claim Ordered Quantity

Order Request

Customer:

Ordered quantity:

Product information

ProductID: 1

ProductName: Chai

SupplierID: 1

NOTE

In preview mode when an input field for a service is an integer, it is automatically filled with the value 1. So when there is no product number 1 nothing would be displayed. However, when the *Next* and *Previous* operations are also available on the model, automatically the first record is displayed.

3. Close the preview.
4. Close the design form.

6.3 Make Changes Available to SCM

In this exercise you will make your developed content/changes available to your team members by sending the changes to the SCM application.

You should only make changes available after you have tested these to work correctly. This is to ensure that your team members’ work is not affected when they incorporate your changes.

> **NOTE**
> This only applies when your workspace is created using an SCM application.

1. If closed, open the Workspace Documents.
2. Click **Make Changes Available to Others** () in the toolbar.
3. Review the modified content.
4. Provide as comment **User Interfaces in Business Processes**.
5. Click **Make Available**.

7. Learning Report

Achievements

- ❑ I know the concept of User Interfaces.
- ❑ I can create a User Interface.
- ❑ I can associate/link multiple models.
- ❑ I can deploy a User Interface.
- ❑ I can use Cordys User interfaces directly via a URL.
- ❑ I can integrate WS-AppServer web services into my forms.
- ❑ I can use my form in a business process model.

Notes