



T210 Customized Forms and Master-Detail Relationship

Dmitrii Naumov

ISV Solution Architect

Timing and Agenda

July 19, 2022 -10 AM -11:30 AM

Day 1

Lesson 1.1: Adding Custom Fields

July 20, 2022 -10 AM -11:30 AM

Day 2

Lesson 1.2: Configuring the UI—Self-Guided Exercise

Lesson 2.1: Defining a Master-Detail Relationship

Timing and Agenda

July 21, 2022 -10 AM -11:30 AM

Day 3

Lesson 2.2: Defining the Business Logic

Lesson 3.1: Adding a New Tab

July 22, 2022 -10 AM -11:30 AM

Day 4

Lesson 4.1: Calculating Field Values

Lesson 4.2: Inserting a Default Detail Record

Day 1



Part 1: Custom Fields (Stock Items Form)

Lesson 1.1: Adding Custom Fields

Learning Objectives

In this lesson, you will learn how to do the following:

- Add a custom column to an Acumatica ERP database table
- Add a custom field to an Acumatica ERP data access class
- Add the control for the custom field to the form

Figure: Custom elements to be added to the Stock Items form

Stock Items

NOTES ACTIVITIES FILES CUSTOMIZATION TOOLS

New Record

← 📄 📁 ↶ + 🗑️ 📄 K < > >| ⋮

* Inventory ID: Product Workgroup:
Item Status: **Active** Product Manager:
Description:

GENERAL PRICE/COST WAREHOUSES VENDORS ATTRIBUTES PACKAGING CROSS-REFERENCE GL ACCOUNTS

ITEM DEFAULTS

* Item Class:

Type: **Finished Good**

Repair Item

Repair Item Type:

Valuation Method: **Standard**

* Tax Category:

* Posting Class:

Auto-Incremental Value:

Country Of Origin:

WAREHOUSE DEFAULTS

Default Warehouse:

UNIT OF MEASURE

* Base Unit: Divisible Unit

* Sales Unit: Divisible Unit

* Purchase Unit: Divisible Unit

Weight Item

↻ + ×

* From Unit	Multiply/Divid	Conversion Factor	To Unit

Figure: Customization menu

Stock Items

New Record

NOTES ACTIVITIES FILES CUSTOMIZATION TOOLS ▾

← ↻ 📄 ↺ + 🗑️ 📄 ▾ ⏪ < > ⏩ ⋮

* Inventory ID:

Item Status:

Product Workgroup:

Product Manager:

Description:

GENERAL PRICE/COST WAREHOUSES VENDORS ATTRIBUTES PACKAGING >>

ITEM DEFAULTS

* Item Class: ✎

Type:

Valuation Method:

* Tax Category: ✎

* Posting Class: ✎

Auto-Incremental Value:

Country Of Origin:

WAREHOUSE DEFAULTS

Default Warehouse: ✎

Select Project...
Inspect Element (Ctrl+Alt+Click)
Edit Project...
Manage Customizations...

Figure: Element Properties dialog box

The screenshot shows a software interface for creating a new record for 'Stock Items'. The main window has a title bar with 'Stock Items' and 'New Record'. Below the title bar are navigation icons and a menu bar with 'NOTES', 'ACTIVITIES', 'FILES', 'CUSTOMIZATION', and 'TOOLS'. The main content area is divided into sections: 'GENERAL', 'PRICE/COST', 'PACKAGING', 'ITEM DEFAULTS', and 'WAREHOUSE DEFAULTS'. The 'Element Properties' dialog box is open, displaying the following information:

- Control Type: Tab
- Data Class: [InventoryItem](#)
- View Name: ItemSettings
- Business Logic: InventoryItemMaint

At the bottom of the dialog, there are three buttons: 'CUSTOMIZE', 'ACTIONS', and 'CANCEL'.

Figure: Suppression of the error in a comment

The screenshot displays the Visual Studio IDE with a C# code file named `InventoryItemExtensions.cs`. The code defines a namespace `PX.Objects.IN` and a public class `InventoryItemExt` that inherits from `PXCacheExtension<PX.Objects.IN.InventoryItem>`. A red squiggly line under the class name indicates an error. A context menu is open over the class name, with the option "Suppress the PX1016 diagnostic with Acuminator" selected. A sub-menu is also open, showing "in a comment" as the chosen option. The error list at the bottom shows the error `PX1016` with a description: "A DAC extension must include the public static IsActive method with the bool return type. Extensions which are constantly active re performance. Suppress the error if you need the DAC extension to be constantly active." The error list also shows the error is suppressed in the code, with a green highlight on the comment: `// Acuminator disable once PX1016 ExtensionDoesNotDeclareIsActiveMethod extension should be constant`. The code snippet below the error list shows the class definition: `public class InventoryItemExt : PXCacheExtension<PX.Objects.IN.InventoryItem>`. The status bar at the bottom indicates "Error List - Current Document (InventoryItemExtensions.cs)".

```
23 namespace PX.Objects.IN
24 {
25     public class InventoryItemExt : PXCacheExtension<PX.Objects.IN.InventoryItem>
26     {
27         UsrRepairItem
34     }
35 }
```

Mark the type as sealed
Suppress the PX1016 diagnostic with Acuminator ▶ in a comment ▶
Suppress the PX1011 diagnostic with Acuminator ▶ in the Acuminator suppression file
Suppress or Configure issues ▶

✘ PX1016 A DAC extension must include the public static IsActive method with the bool return type. Extensions which are constantly active re performance. Suppress the error if you need the DAC extension to be constantly active.

Lines 24 to 26

```
{
// Acuminator disable once PX1016 ExtensionDoesNotDeclareIsActiveMethod extension should be constant
public class InventoryItemExt : PXCacheExtension<PX.Objects.IN.InventoryItem>
{
```

Preview changes

100 % 1 1

Error List - Current Document (InventoryItemExtensions.cs)

Figure: Fix of the warning

```
25 // Acuminator disable once PX1016 ExtensionDoesNotDeclareIsActiveMethod extension should be constantly active
26 public class InventoryItemExt : PXCACHEExtension<PX.Objects.IN.InventoryItem>
27 {
28     #region UsrRep
29     [PXDBBool]
30     [PXUIField(DisplayName = "Inventory Item Active")]
31     public virtual bool? UsrRepairItem { get; set; }
32     #endregion
33     public abstract class usrRepairItem : PX.Data.BQL.BqlBo
34     {
35     }
36 }
```

Warning: PX1011 Because multiple levels of inheritance are not supported for PXCACHEExtension, the derived type can be marked as sealed

Fixes:

- Mark the type as sealed
- Suppress the PX1011 diagnostic with Acuminator
- Suppress or Configure issues

Preview changes

Fix all occurrences in: Document | Project | Solution

PhoneRepairShop

- PhoneRepairShop_Code
- Properties
- References

InventoryItemExtension
RSSVDevice.cs
RSSVRepairService.cs
a
per
Constants.cs
Messages.cs

C# RSSVDeviceMaint.cs

Figure: The Type node in the control tree

Screen Editor: IN202500 (Stock Items)

EDIT ASPX PREVIEW CHANGES ...

LAYOUT PROPERTIES ATTRIBUTES EVENTS ADD CONTROLS **ADD DATA FIELDS** VIEW ASPX

Data View: Inventory Item(ItemSettings)

CREATE CONTROLS NEW FIELD ALL VISIBLE **CUSTOM**

<input type="checkbox"/>	Used	Field Name	Control
<input type="checkbox"/>	<input type="checkbox"/>	UsrRepairItem (Repair Item)	CheckBox

The control tree on the left shows the following structure:

- DataSource: InventoryItemMaint
 - Form: Item
 - Tab: ItemSettings
 - General
 - Column
 - Template ID
 - Group
 - Item Class
 - Type**
 - Is a Kit
 - Valuation Method
 - Tax Category
 - Posting Class
 - Lot/Serial Class
 - Auto-Incremental Value
 - Country Of Origin
 - Form: CurySettings_InventoryItem
 - Merge
 - [Layout Rule]
 - Column
 - Subitems
 - Price/Cost
 - Manufacturing
 - Warehouses

Figure: The added control

Screen Editor: IN202500 (Stock Items)

EDIT ASPX PREVIEW CHANGES ...

LAYOUT PROPERTIES ATTRIBUTES EVENTS ADD CONTROLS **ADD DATA FIELDS** VIEW ASPX

Data View: Inventory Item(ItemSettings)

CREATE CONTROLS NEW FIELD ALL VISIBLE **CUSTOM**

<input type="checkbox"/>	Used	Field Name	Control
<input type="checkbox"/>	<input checked="" type="checkbox"/>	UsrRepairItem (Repair Item)	CheckBox

DataSource: InventoryItemMaint

- Form: Item
- Tab: ItemSettings
 - General
 - Column
 - Template ID
 - Group
 - Item Class
 - Type**
 - Repair Item**
 - Is a Kit
 - Valuation Method
 - Tax Category
 - Posting Class
 - Lot/Serial Class
 - Auto-Incremental Value
 - Country Of Origin
 - Form: CurySettings_InventoryItem
 - Merge
 - [Layout Rule]
 - Column
 - Subitems
 - Price/Cost
 - Manufacturing

Figure: The Repair Item check box

Stock Items

NOTES ACTIVITIES FILES CUSTOMIZATION TOOLS ▾

New Record

← ↻ 📄 ↶ + 🗑️ 📄 ▾ ⏪ < > ⏩ ⋮

* Inventory ID: 🔍

Item Status: ▾

Product Workgroup: 🔍

Product Manager: 🔍

Description:

GENERAL PRICE/COST WAREHOUSES VENDORS ATTRIBUTES **PACKAGING** >>

ITEM DEFAULTS

* Item Class: 🔍 ✎

Type: ▾

Repair Item

Valuation Method: ▾

* Tax Category: 🔍 ✎

* Posting Class: 🔍 ✎

Auto-Incremental Value:

Country Of Origin: 🔍

WAREHOUSE DEFAULTS

Figure: The list of the project items

Edit Project Items

GET PACKAGE

Object Name	Type	Description	Created By	Creation Date	Last Modified By	Last Modified On
~/pages/in/in202500.aspx	Page		admin admin	10/12/2021	admin admin	10/12/2021
~/pages/rs/rs201000.aspx	Page		admin admin	10/8/2021	admin admin	10/8/2021
~/pages/rs/rs202000.aspx	Page		admin admin	10/8/2021	admin admin	10/8/2021
BiniPhoneRepairShop_code.dll	File		admin admin	10/8/2021	admin admin	10/8/2021
InputDataIRSSVDevice.csv	File		admin admin	10/8/2021	admin admin	10/8/2021
InputDataIRSSVRepairService.csv	File		admin admin	10/8/2021	admin admin	10/8/2021
Pages\RS\IRS201000.aspx	File		admin admin	10/8/2021	admin admin	10/8/2021
Pages\RS\IRS201000.aspx.cs	File		admin admin	10/8/2021	admin admin	10/8/2021
Pages\RS\IRS202000.aspx	File		admin admin	10/8/2021	admin admin	10/8/2021
Pages\RS\IRS202000.aspx.cs	File		admin admin	10/8/2021	admin admin	10/8/2021
Serviceed Devices	GenericInquiryScreen		admin admin	10/8/2021	admin admin	10/8/2021
InventoryItem	Table		admin admin	10/12/2021	admin admin	10/12/2021
Serviceed Devices	SiteMapNode		admin admin	10/8/2021	admin admin	10/8/2021
Repair Services	SiteMapNode		admin admin	10/8/2021	admin admin	10/8/2021

Source

```

<PXTabItem Text="General" ParentId="phG_tab_Items#0" TypeFullName="PX.Web.UI.PXTabItem">
  <Children Key="Template">
    <AddItem>
      <PXCheckBox TypeFullName="PX.Web.UI.PXCheckBox">
        <Prop Key="Virtual ApplyStylesheetSkin" />
        <Prop Key="ID" Value="CstPXCheckBox1" />
        <Prop Key="DataField" Value="UsrRepairItem" />
      </PXCheckBox>
    </AddItem>
    <PXCheckBox DataField="KitItem" OriginalIndex="5" />
  </Children>
</PXTabItem>
</Page>

```

Figure: The Repair Item Type box

Stock Items

New Record

NOTES ACTIVITIES FILES CUSTOMIZATION TOOLS ▾

← ↻ 📄 ↶ + 🗑️ 📄 ▾ ⏪ < > ⏩ ⋮

* Inventory ID: 🔍 Product Workgroup: 🔍
Item Status: **Active** ▾ Product Manager: 🔍
Description:

GENERAL PRICE/COST WAREHOUSES VENDORS ATTRIBUTES PACKAGING CROSS-REFERENCE GLACCOUNTS >>

ITEM DEFAULTS

* Item Class: 🔍 ✎
Type: **Finished Good** ▾
 Repair Item
Repair Item Type: ▾
Valuation Method: **Standard** ▾
* Tax Category: 🔍 ✎
* Posting Class: 🔍 ✎
Auto-Incremental Value:
Country Of Origin: 🔍

UNIT OF MEASURE

* Base Unit: 🔍 ✎ Divisible Unit
* Sales Unit: 🔍 ✎ Divisible Unit
* Purchase Unit: 🔍 ✎ Divisible Unit
 Weight Item

🔄 + ×

* From Unit	Multiply/Divid	Conversion Factor	To Unit
-------------	----------------	-------------------	---------

WAREHOUSE DEFAULTS

Default Warehouse: 🔍 ✎

Figure: The generation of the event handler

Screen Editor: IN202500 (Stock Items)

EDIT ASPX PREVIEW CHANGES ...

LAYOUT PROPERTIES ATTRIBUTES **EVENTS** ADD CONTROLS ADD DATA FIELDS VIEW ASPX

Data Class: **PX.Objects.IN.InventoryItem**
Field Name: UsrRepairItem
Business Logic: PX.Objects.IN.InventoryItemMaint::ItemSettings

ADD HANDLER VIEW SOURCE

Event	Handled in Source	Customized
Control	<input type="checkbox"/>	<input type="checkbox"/>
RowSelecting	<input type="checkbox"/>	<input type="checkbox"/>
RowSelected	<input type="checkbox"/>	<input type="checkbox"/>
FieldSelecting	<input type="checkbox"/>	<input type="checkbox"/>
RowInserting	<input type="checkbox"/>	<input type="checkbox"/>
RowInserted	<input type="checkbox"/>	<input type="checkbox"/>
RowUpdating	<input type="checkbox"/>	<input type="checkbox"/>
RowUpdated	<input checked="" type="checkbox"/>	<input type="checkbox"/>
RowDeleting	<input type="checkbox"/>	<input type="checkbox"/>
RowDeleted	<input type="checkbox"/>	<input type="checkbox"/>
FieldDefaulting	<input type="checkbox"/>	<input type="checkbox"/>
FieldUpdating	<input type="checkbox"/>	<input type="checkbox"/>
FieldVerifying	<input type="checkbox"/>	<input type="checkbox"/>
ExceptionHandler	<input type="checkbox"/>	<input type="checkbox"/>

Figure: The CommitChanges property

Screen Editor: IN202500 (Stock Items)

EDIT ASPX PREVIEW CHANGES ...

LAYOUT PROPERTIES ATTRIBUTES EVENTS ADD CONTROLS ADD DATA FIELDS VIEW ASPX

DataSource: InventoryItemMaint
Form: Item
Tab: ItemSettings
General
Column
Template ID
Group
Item Class
Type
Repair Item
Repair Item Type
Is a Kit
Valuation Method
Tax Category
Posting Class
Lot/Serial Class
Auto-Incremental Value
Country Of Origin
Form: CurySettings_InventoryItem
Merge
[Layout Rule]
Column
Subitems
Price/Cost

Override	Property	Value
Base Properties		
<input type="checkbox"/>	CommitChanges	True
<input checked="" type="checkbox"/>	DataField	UsrRepairItem
<input checked="" type="checkbox"/>	ID	CstPXCheckBox1
<input type="checkbox"/>	Size	
<input type="checkbox"/>	Text	
Ext Properties		
<input type="checkbox"/>	AlignLeft	
<input type="checkbox"/>	AlreadyLocalized	
<input type="checkbox"/>	AutoCallBack	
<input type="checkbox"/>	CheckImages	
<input type="checkbox"/>	Enabled	
<input type="checkbox"/>	FalseValue	
<input type="checkbox"/>	LabelWidth	
<input type="checkbox"/>	RenderStyle	

Indicates whether the control performs commit callback after the value of the control has been changed.

Lesson Summary

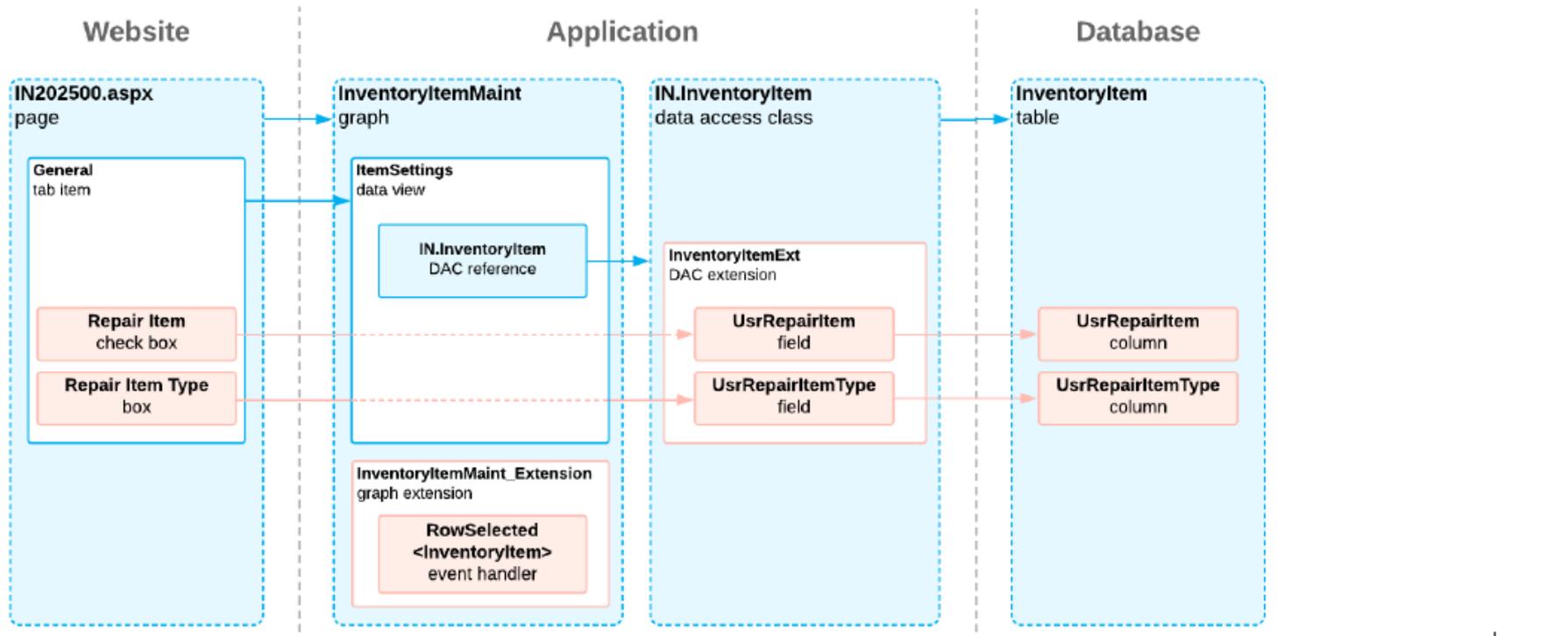
In this lesson, you have learned how to create a control so that you can display on a form a custom field bound to the database. To implement this customization, you have learned how to add the necessary modifications to a customization project and how to publish the project to apply the changes to the system.

As you have completed the lesson, you have added the following elements to the PhoneRepairShop customization project:

- Two column definitions in the InventoryItem table of the database.
- Two custom field declarations in the extension of the IN.InventoryItem data access class (in the PhoneRepairShop_Code extension library).
- Two controls to display the custom fields on the Stock Items (IN202500) form.
- One custom event handler, which you have added to the InventoryItemMaint graph. You have used the RowSelected event handler to configure the UI presentation logic.

Lesson Summary

Addition of New Custom Elements



Day 2

Lesson 1.2: Configuring the UI—Self-Guided Exercise

The screenshot displays the Acumatica user interface. At the top, the header includes the Acumatica logo, a search bar, a refresh icon, the user name 'Yogifon', the date and time '10/13/2021 5:21 AM', a help icon, and the user role 'admin, admin'. The left navigation pane lists various modules: Favorites, Data Views, Phone Repair Shop, Time and Expenses, Finance, Banking, Payables, Receivables, Sales Orders, and Purchases. The main workspace is titled 'Phone Repair Shop' and contains a 'Configuration' section with 'Repair Services' and 'Serviced Devices', and a 'Profiles' section with 'Stock Items' highlighted. The background shows a form with fields for 'REFERENCE', 'GL ACCOUNTS', and 'Divisible Unit' (checked). A blue banner at the bottom states 'Your product is in trial mode. Only two concurrent users are allowed.' and includes an 'ACTIVATE' button.

Figure: Stock Items form in the Phone Repair Shop workspace

Figure: SiteMapNode item for the Stock Items form

Customization Project Editor [Back](#) [Reload](#)

File Publish Extension Library Source Control

PhoneRepairShop ◀ Site Map

SCREENS

- Data Access
- Code
- Files (7)
- Generic Inquiries (1)
- Reports
- Dashboards
- Site Map (3)**
 - Database Scripts (8)
 - System Locales
 - Import/Export Scenarios
 - Shared Filters
 - Access Rights
 - Wikis
 - Web Service Endpoints
 - Analytical Reports
 - Push Notifications
 - Business Events
 - Mobile Application
 - User-Defined Fields
 - Webhooks
 - Connected Applications

RELOAD FROM DATABASE MANAGE SITE MAP

Object Name	Description	Last Modified By	Last Modified On
Repair Services		admin admin	10/8/2021
Serviced Devices		admin admin	10/8/2021
> Stock Items		admin admin	10/13/2021



Part 2: Master-Detail Relationship and Business Logic (Services and Prices Form)

Lesson 2.1: Defining a Master-Detail Relationship

Learning Objectives

In this lesson, you will learn how to do the following:

- Define the master-detail relationship between data
- Implement automatic numbering of detail records

Figure: Services and Prices form

Services and Prices

New Record

NOTES FILES CUSTOMIZATION TOOLS ▾

← ↻ 📄 ↺ + 🗑️ 📄 ▾ ⌂ < > >|

* Service: 🔍 Approximate Price:

* Device: 🔍

REPAIR ITEMS TAB ITEM 2

↻ + × ⏪ ⏩

📄	🔍	📄	Repair Item Type	Required	* Inventory ID	Description	Price	Default
---	---	---	------------------	----------	----------------	-------------	-------	---------

⏪ < > >|

Tables for the Repair Items Tab of the Services and Prices Form

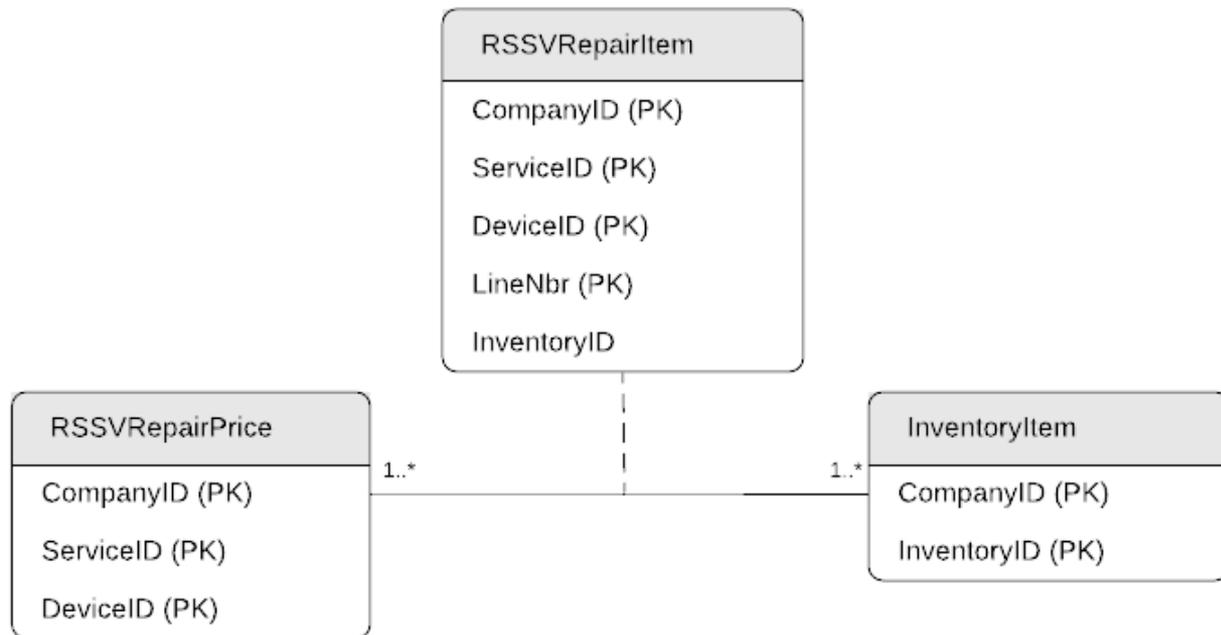


Figure: InventoryID attribute

Source Code CUSTOMIZATION TOOLS ▾

SCREEN ASPX BUSINESS LOGIC DATA ACCESS FIND IN FILES WEBSITE SOURCES

Table Name: 🔍

```
#region Operation
#region OrigPlanType
#region InvtMult
#region InventoryID
public abstract class inventoryID : PX.Data.BQL.BqlInt.Field<inventoryID>
{
    public class InventoryBaseUnitRule :
        InventoryItem.baseUnit.PreventEditIfExists<
            Select<SOShipLine,
                Where<inventoryID, Equal<Current<InventoryItem.inventoryID>>,
                    And<lineType, In3<SOLineType.inventory, SOLineType.nonInventory>,
                    And<confirmed, NotEqual<True>>>>>
            { }
        }
    }
protected Int32? _InventoryID;
[Inventory( Enabled = false)]
[PXForeignReference(typeof(Field<inventoryID>.IsRelatedTo<InventoryItem.inventoryID>))]
public virtual Int32? InventoryID
{
    get
    {
        return this._InventoryID;
    }
    set
    {
        this._InventoryID = value;
    }
}
#endregion
#region IsIntercompany
```

Figure: Two columns in the Summary area

Services and Prices

New Record

NOTES FILES CUSTOMIZATION TOOLS ▾



* Service: 🔍

Approximate Price: 0.00

* Device: 🔍

Lesson Summary

In this lesson, you have learned how to set up a master-detail relationship between data.

To create a master-detail form, you have completed the following actions:

1. Set up the master-detail relationship between data access classes as follows:

- Added the PXDBDefault attribute to the key data fields of the detail DAC that are the keys to the master record. The PXDBDefault attribute provides the default value for the key field of the detail DAC.
- Added the PXParent attribute to the first foreign key data field of the detail DAC. The PXParent attribute enables cascading deletion of detail records on deletion of the master record.

2. Defined two data views that select the master-detail data for the form. You have used the FromCurrent parameter in the detail data view type to select records for a particular master record.

Lesson Summary

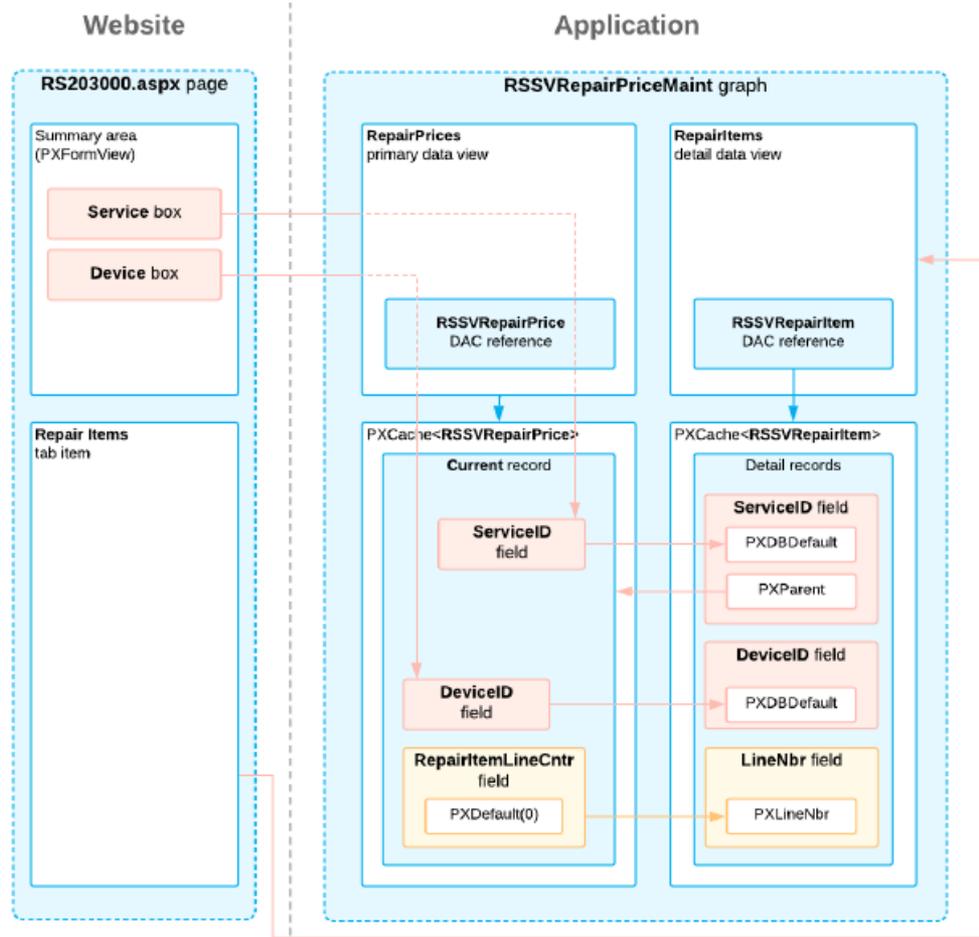
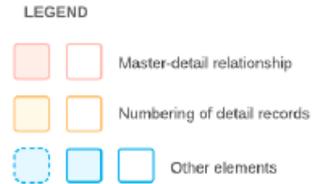
3. Bound the UI controls that display the data on the ASPX page as follows:

- Specified the master data view for the datasource control on the page (in the PrimaryView property of the control).
- Specified the master data view for the form (in the DataMember property of the control).
- Specified the detail data view for the grid (in the DataMember property of the control).

In the lesson, you have also implemented the numbering of detail data records by using the PXLineNbr attribute.

Lesson Summary

Implementation of Master-Detail Relationship and Line Numbering



Day 3

Lesson 2.2: Defining the Business Logic

Learning Objectives

In this lesson, you will learn how to do the following:

- Restrict the possible values of a field by using the PXRestrictor attribute
- Mark localizable messages in code
- Update the fields of the same data record on update of a field of this record
- Update the fields of other records on update of a field
- Learn one of the possible ways to retrieve a data record from the database in code by using the static PXSelectorAttribute.Select<>() method

Figure: Inventory ID selector

Services and Prices
BatteryReplace Nokia3310

NOTES FILES CUSTOMIZATION TOOLS

* Service: BATTERYREPLACE - Battery Approximate Price: 0.00
* Device: NOKIA3310 - Nokia 3310

REPAIR ITEMS TAB ITEM 2

Repair Item Type	Required	Inventory ID	Description	Price	Default
Battery	<input type="checkbox"/>			0.00	<input type="checkbox"/>

SELECT

Inventory ID	Description	Item Class	Item Status	Type
BAT3310	Battery for Nokia 3310	STOCKITEM	Active	Finished Good
BAT3310EX	Extended Battery for Nokia 3310	STOCKITEM	Active	Finished Good

Lesson Summary

In this lesson, you have configured the business logic of the Repair Items tab of the Services and Prices (RS203000) form as follows:

- You have used the `PXRestrictor` attribute to configure a restriction on the values in the Inventory ID column.
- You have used a class with the `PXLocalizable` attribute specified to make the text available for localization.
- You have used the `FieldUpdated` and `FieldDefaulting` event handlers to modify the values of a detail record on update of the column of this detail record. In the event handlers, you have used the `PXSelectorAttribute.Select<>()` method to obtain the stock item record with the inventory ID selected in the updated field. You have also used the `SetValueExt<field>` and `SetDefaultExt<field>` methods to trigger additional events for particular fields.

Lesson Summary

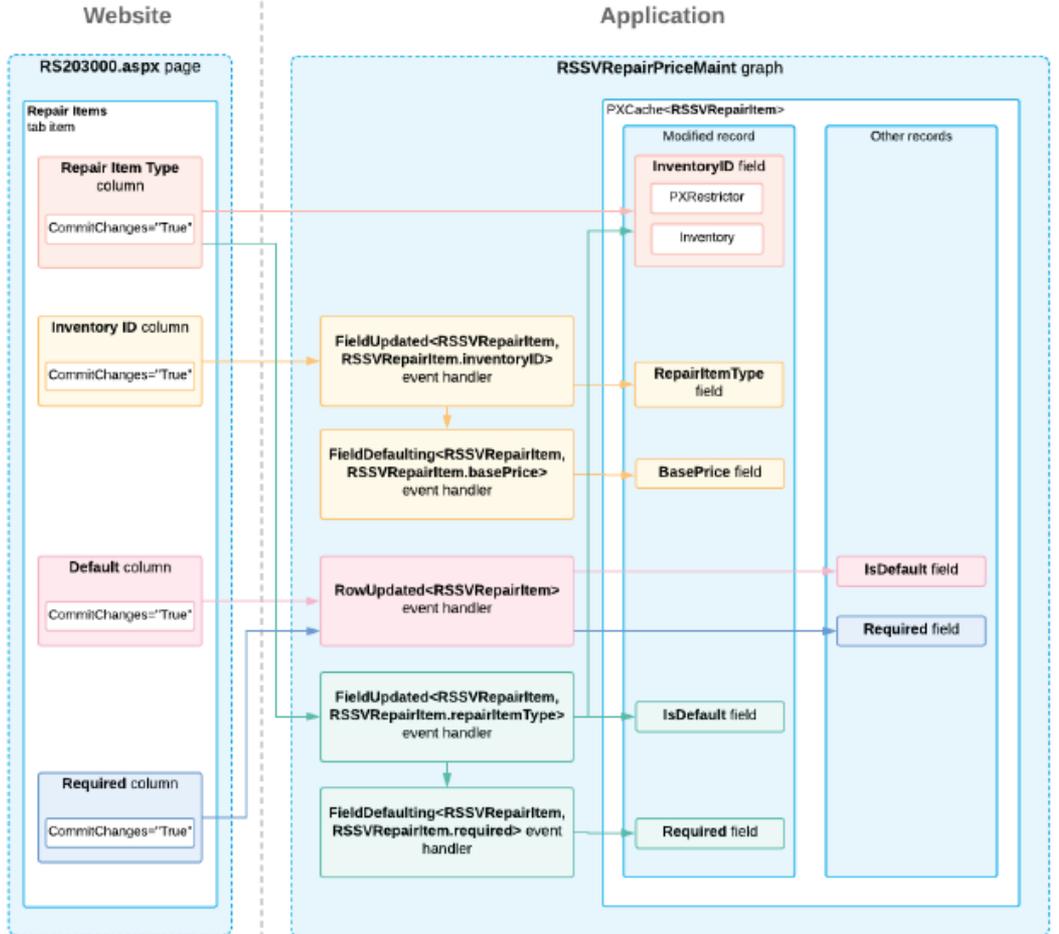
- You have used the RowUpdated event handler so that when particular fields of one detail record are updated, the values in the other detail records are modified. In this event handler, you have used the PXCache.Update() method to update in PXCache the records other than the record for which the event has been raised. You have also used the PXView.RequestRefresh() method to display in the UI the changes in PXCache that you have made in the event handler. You have also used LINQ for the filtering of the records of the data view.

Lesson Summary

Implementation of Business Logic

LEGEND

- The restriction on the **InventoryID** field
- The update of **BasePrice** and **RepairItem** on the update of **InventoryID**
- The update of **IsDefault** in other detail records
- The update of **Required** in other detail records
- The update of **Required** and the clearing of **InventoryID** and **IsDefault** on the update of **RepairItem**
- Other elements



A close-up photograph of a bartender's hand pouring beer from a tap into a glass. The bartender is wearing a checkered shirt and a metal watch. The glass has a logo that says "Devil's Peak" and "BREWING COMPANY". The background is a blurred bar setting.

Part 3: Custom Tab (Stock Items Form)

Lesson 3.1: Adding a New Tab

Learning Objectives

In this lesson, you will learn how to do the following:

- Add a custom data view for an Acumatica ERP form
- Create a custom tab on an Acumatica ERP form
- Conditionally hide a custom tab on an Acumatica ERP form

Figure: The Compatible Devices tab

Stock Items

SCRSGS4 - Screen for Samsung Galaxy S4

NOTES ACTIVITIES FILES CUSTOMIZATION TOOLS

← ↻ 🗑️ 📄 ⌵ ⏪ ⏩ ⏴ ⏵ ⋮

* Inventory ID: Product Workgroup:

Item Status: Product Manager:

Description:

GENERAL PRICE/COST ⚠️ WAREHOUSES VENDORS ATTRIBUTES PACKAGING CROSS-REFERENCE ⚠️ GLACCOUNTS COMPATIBLE DEVICES ⏴

🔄 + × |←| |→|

🗑️ 📄	Device	Description
> 🗑️ 📄	SAMSUNGGS4	Samsung Galaxy S4

⏪ ⏩ ⏴ ⏵

Figure: Addition of the tab item

Screen Editor: IN202500 (Stock Items)

↶ 📄 EDIT ASPX PREVIEW CHANGES ...

The screenshot displays the Acumatica Screen Editor interface for the 'InventoryItemMaint' screen. The left-hand pane shows a tree view of the screen's structure, with the 'Tab: ItemSettings' folder expanded. A red box highlights a 'TAB ITEM' control within this folder. A red arrow points from this control to the 'ADD CONTROLS' tab in the main editor area. The 'ADD CONTROLS' tab is active and shows two columns of controls: 'MAIN CONTAINERS' and 'OTHER CONTROLS'. The 'TAB ITEM' control is highlighted in blue in the 'MAIN CONTAINERS' column. Other controls in the 'MAIN CONTAINERS' column include FORM, TAB, GRID, and POP-UP PANEL. The 'OTHER CONTROLS' column includes PANEL, GROUP BOX, RADIO BUTTON, LABEL, BUTTON, and JAVA SCRIPT. Below these columns is the 'LAYOUT RULES' section, which includes ROW, COLUMN, GROUP, MERGE, and EMPTY RULE.

DataSource: InventoryItemMaint

- Form: Item
 - Tab: ItemSettings**
 - General
 - Subitems
 - Price/Cost
 - Manufacturing
 - Warehouses
 - Vendors
 - Attributes
 - Packaging
 - Cross-Reference
 - Related Items
 - Replenishment
 - Deferral
 - GL Accounts
 - Restriction Groups
 - Description
 - Service Management
 - Sync Status
 - eCommerce
 - TAB ITEM**
- Dialogs

EDIT ASPX PREVIEW CHANGES ...

ADD CONTROLS

MAIN CONTAINERS

- FORM
- TAB
- TAB ITEM**
- GRID
- POP-UP PANEL

OTHER CONTROLS

- PANEL
- GROUP BOX
- RADIO BUTTON
- LABEL
- BUTTON
- JAVA SCRIPT

LAYOUT RULES

- ROW
- COLUMN
- GROUP
- MERGE
- EMPTY RULE

Figure: The Device and Description nodes

Screen Editor: IN202500 (Stock Items)

EDIT ASPX PREVIEW CHANGES ...

LAYOUT PROPERTIES ATTRIBUTES EVENTS ADD CONTROLS **ADD DATA FIELDS** VIEW ASPX

Data View: Device Compatible with Stock Item(Compatible...

CREATE CONTROLS NEW FIELD ALL **VISIBLE** CUSTOM

<input type="checkbox"/>	Used	Field Name	Control
<input type="checkbox"/>	<input type="checkbox"/>	CreatedByID (Created By)	Selector
<input type="checkbox"/>	<input type="checkbox"/>	CreatedByID_Creator_displayName (Created By)	TextEdit
<input type="checkbox"/>	<input type="checkbox"/>	CreatedByID_Creator_Username (Created By)	TextEdit
<input type="checkbox"/>	<input type="checkbox"/>	CreatedByID_description (Created By)	TextEdit
<input type="checkbox"/>	<input checked="" type="checkbox"/>	DeviceID (Device)	Selector
<input type="checkbox"/>	<input type="checkbox"/>	DeviceID_description (Description)	TextEdit
<input type="checkbox"/>	<input checked="" type="checkbox"/>	DeviceID_RSSVDevice_description (Description)	TextEdit
<input type="checkbox"/>	<input type="checkbox"/>	LastModifiedByID (Last Modified By)	Selector
<input type="checkbox"/>	<input type="checkbox"/>	LastModifiedByID_description (Last Modified By)	TextEdit
<input type="checkbox"/>	<input type="checkbox"/>	LastModifiedByID_Modifier_displayName (Last Mo	TextEdit
<input type="checkbox"/>	<input type="checkbox"/>	LastModifiedByID_Modifier_Username (Last Modif	TextEdit

Grid: CompatibleDevices

- Device
- Description**

Levels

Dialogs

Figure: The new tab

Stock Items

NOTES ACTIVITIES FILES CUSTOMIZATION TOOLS ▾

New Record

← ↻ 📄 ↶ + 🗑️ 📄 ▾ ⏪ < > ⏩ ⋮

* Inventory ID: 🔍 Product Workgroup: 🔍

Item Status: Active ▾ Product Manager: 🔍

Description:

GENERAL PRICE/COST WAREHOUSES VENDORS ATTRIBUTES PACKAGING CROSS-REFERENCE GL ACCOUNTS DESCRIPTION COMPATIBLE DEVICES

📄 🗑️ 📄	Device	Description
--------	--------	-------------

Figure: The final layout of the tab

Stock Items

New Record

NOTES ACTIVITIES FILES CUSTOMIZATION TOOLS

← ↻ 📄 ↶ + 🗑️ 📄 ▾ ⏪ < > ⏩ ...

* Inventory ID: Product Workgroup:
Item Status: **Active** Product Manager:
Description:

GENERAL PRICE/COST WAREHOUSES VENDORS ATTRIBUTES PACKAGING CROSS-REFERENCE GLACCOUNTS DESCRIPTION **COMPATIBLE DEVICES**

↻ + × |←| |→|

📄	🔍	📄	Device	Description
			MOTORRAZR	Motorola RAZR V3
*			<input type="text"/>	<input type="text"/>

⏪ < > ⏩

Figure: ASPX code of the tab

Screen Editor: IN202500 (Stock Items)

PREVIEW CHANGES ACTIONS ▾

VIEW ASPX

- DataSource: InventoryItemMaint
- Form: Item
- Tab: ItemSettings
 - General
 - Subitems
 - Price/Cost
 - Manufacturing
 - Warehouses
 - Vendors
 - Attributes
 - Packaging
 - Cross-Reference
 - Related Items
 - Replenishment
 - Deferral
 - GL Accounts
 - Restriction Groups
 - Description
 - Service Management
 - Sync Status
 - eCommerce
 - Compatible Devices
- Dialogs

```
<px:FXTabItem Text="Compatible Devices">  
  <Template>  
    <px:FXGrid runat="server" ID="CstFXGrid3" SkinID="Details" Width="100%" DataSourceID="ds">  
      <Levels>  
        <px:FXGridLevel DataMember="CompatibleDevices">  
          <Columns>  
            <px:FXGridColumn DataField="DeviceID" Width="140" CommitChanges="True" />  
            <px:FXGridColumn DataField="DeviceID_RSSVDevice_description" Width="280" /></Columns>  
          <RowTemplate>  
            <px:FXSelector runat="server" ID="CstFXSelector4" DataField="DeviceID" AllowEdit="True" /></RowTemplate></px:FXGridLevel></Levels>  
          <AutoSize Enabled="True" MinHeight="200" /></px:FXGrid></Template></px:FXTabItem>
```

Lesson Summary

In this lesson, you have practiced implementing extensions for both the business logic and the user interface of an existing form, and you have learned the following tasks:

- Creating a control container on a form
- Creating the data view in the extension of the graph for the form to provide data for the container
- Conditionally hiding a tab

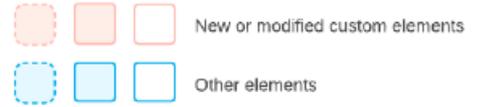
During the lesson, you have added to the customization project the following items:

- On the Stock Items (IN202500) form, the Compatible Devices tab, which contains the Device and Description columns
- The CompatibleDevices data view in the InventoryItemMaint_Extension graph
- The RSSVStockItemDevice DAC

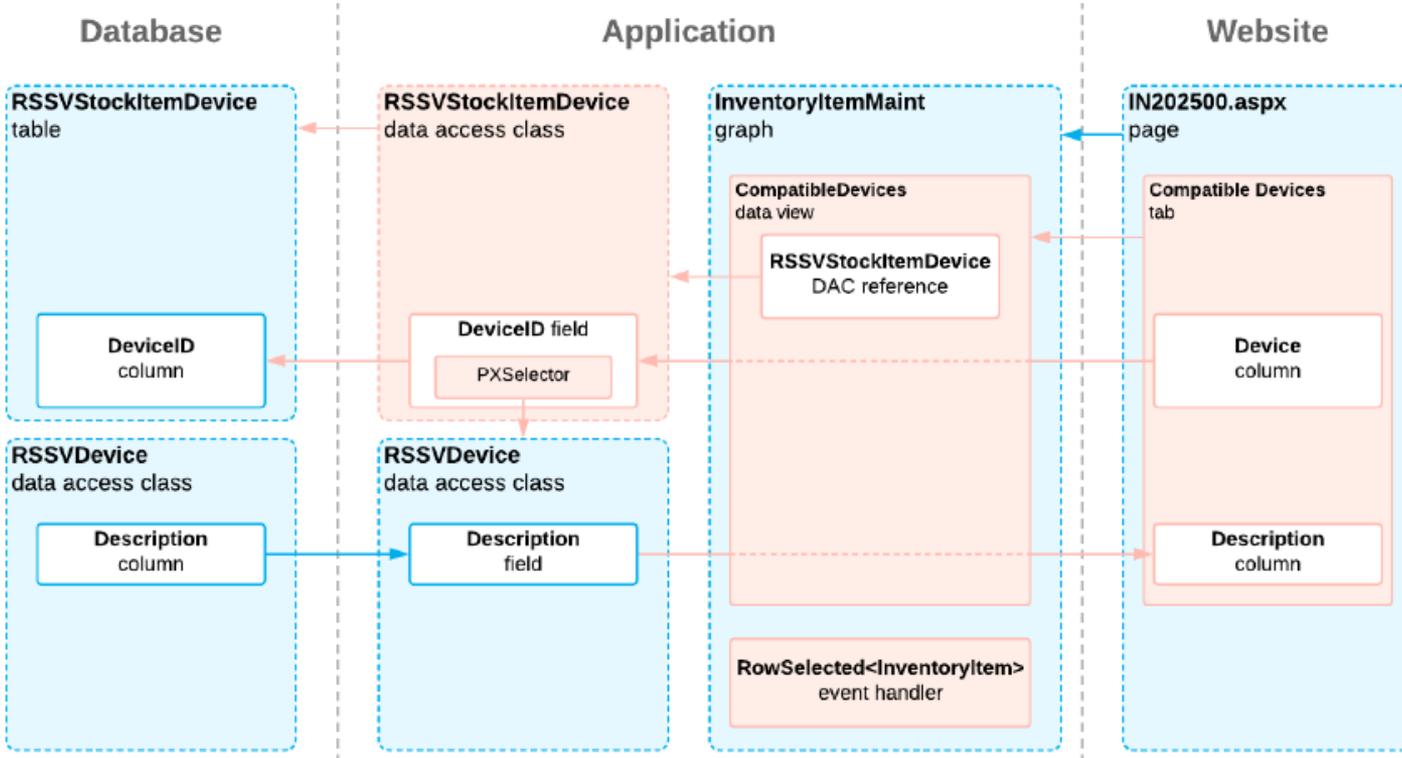
You have also modified the RowSelected<InventoryItem> event handler.

Lesson Summary

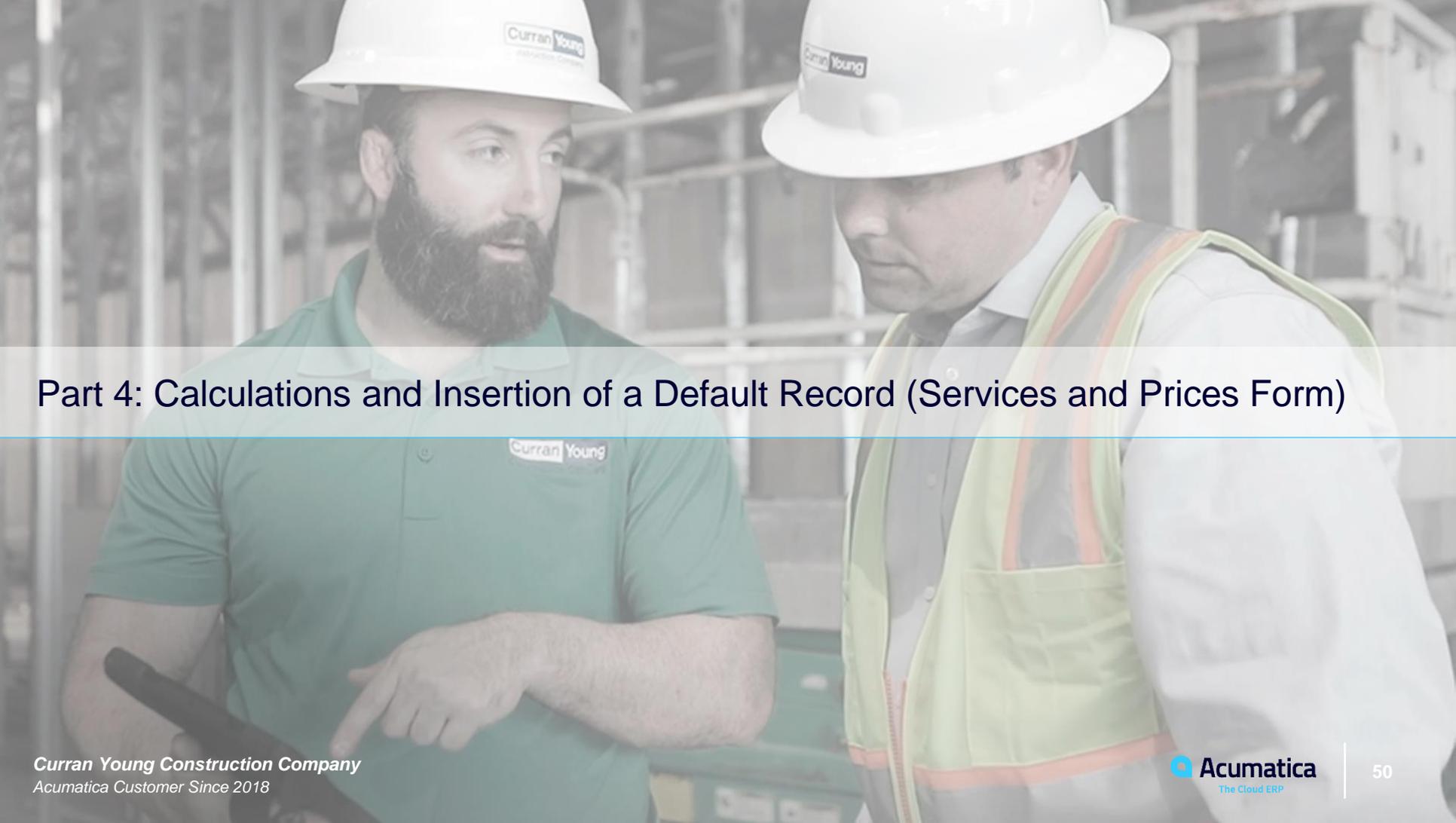
LEGEND



Addition of a Custom Tab



Day 4



Part 4: Calculations and Insertion of a Default Record (Services and Prices Form)

Lesson 4.1: Calculating Field Values

Learning Objectives

In this lesson, you will learn how to use the PXFormula attribute for calculations.

Figure: The Labor tab

Services and Prices

BatteryReplace Nokia3310

NOTES FILES CUSTOMIZATION TOOLS ▾

← ↻ + 🗑️ 📄 ▾ ⏪ < > ⏩

* Service: BATTERYREPLACE - Battery 🔍 Approximate Price: 35.00

* Device: NOKIA3310 - Nokia 3310 🔍

REPAIR ITEMS LABOR

🔄 + × ⏪ 🗑️

📄	🔍	🗑️	Inventory ID	Description	Default Price	Quantity	Ext. Price
>	🔍	🗑️	CONSULT	Consulting service	5.00	1.00	5.00

⏪ < > ⏩

Relationships Between Database Tables

Tables for the Labor Tab of the Services and Prices Form

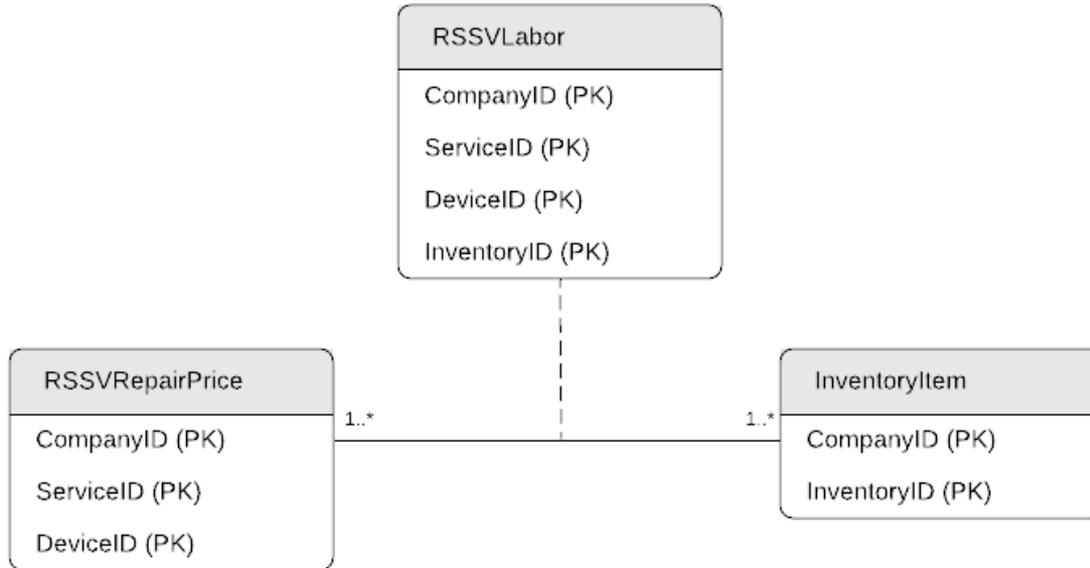


Figure: The calculation of the price

Services and Prices

BatteryReplace Nokia3310

NOTES FILES CUSTOMIZATION TOOLS

← ↻ 📄 + 🗑️ 📄 K < > >|

* Service: BATTERYREPLACE - Battery 🔍 Approximate Price: 35.00

* Device: NOKIA3310 - Nokia 3310 🔍

REPAIR ITEMS LABOR

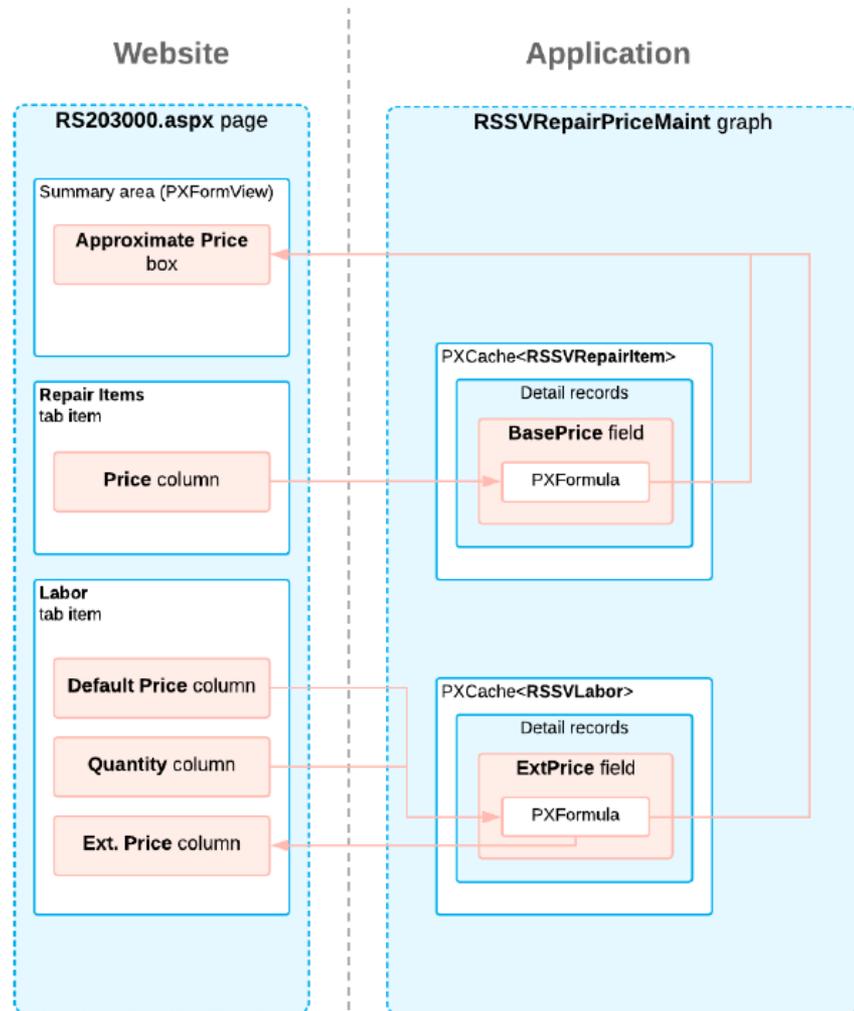
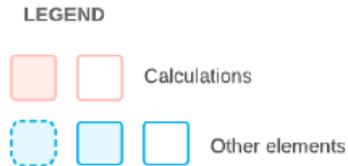
🔄 + × 📏 🗑️

📄	🗑️	📄	Inventory ID	Description	Default Price	Quantity	Ext. Price
↶	🗑️	📄	CONSULT	Consulting service	5.00	1.00	5.00

Lesson Summary

In this lesson, you have implemented the calculation of the value of the Approximate Price box on the Services and Prices (RS203000) form. You have used the PXFormula attribute to configure the calculation.

Lesson Summary



Lesson 4.2: Inserting a Default Detail Record

Learning Objectives

In this lesson, you will learn how to add a default detail record to the grid.

Figure: The Warranty tab

Services and Prices

ScreenRepair SamsungGS4

NOTES FILES CUSTOMIZATION TOOLS ▾

← ↻ ↺ + 🗑️ 📄 ⏪ < > ⏩

* Service: SCREENREPAIR - Screen Rr ⌵ Approximate Price: 50.00
* Device: SAMSUNGGS4 - Samsung G ⌵

REPAIR ITEMS LABOR WARRANTY

🔄 + × ⏪ ⏩

📄	🔗	📄	Contract ID	Description	Duration	Duration Unit	Contract Type
>	🔗	📄	DFLTWARNT	Default warranty	1	Year	Expiring

< < > >

Relationships Between Database Tables

Tables for the Warranty Tab of the Services and Prices Form

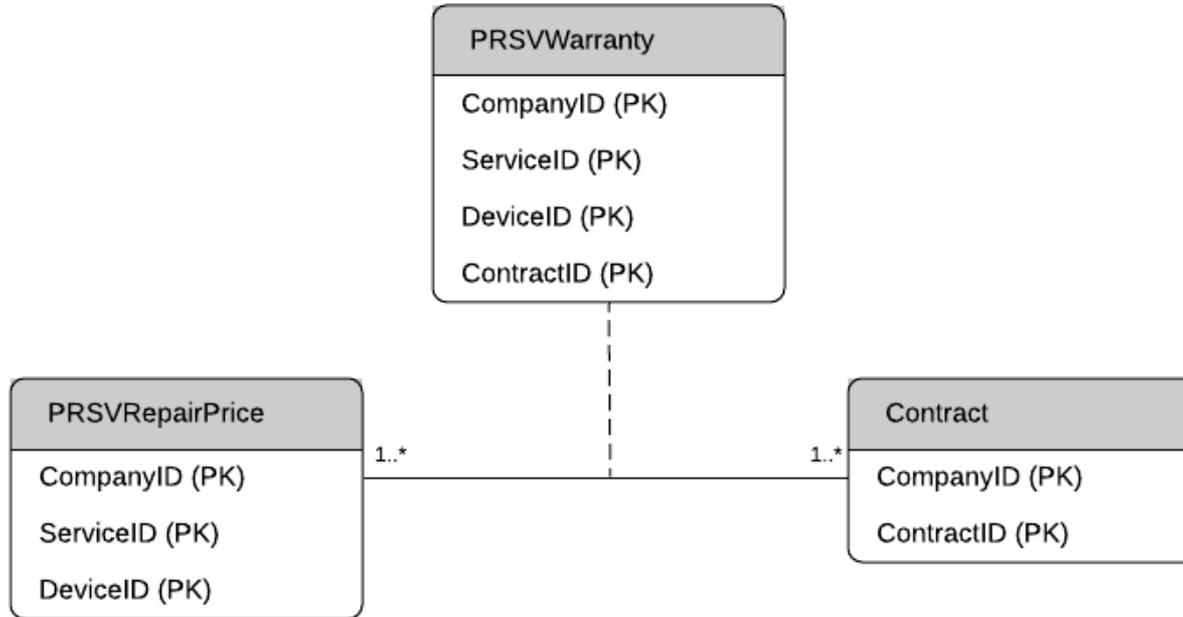


Figure: The error on the form

Services and Prices

ScreenRepair SamsungGS4

NOTES FILES CUSTOMIZATION TOOLS

← ↻ ⏪ ⏩ + 🗑️ 📄

* Service: SCREENREPAIR - Screen Re ⌵ Approximate Price: 0.00

* Device: SAMSUNGGS4 - Samsung G ⌵

REPAIR ITEMS LABOR **WARRANTY**

↻ + × ⏪ ⏩

Contract ID	Description	Duration	Duration Unit	Contract Type
DFLTWARRNT	Default warranty	1	Year	Expiring

The default warranty cannot be deleted.

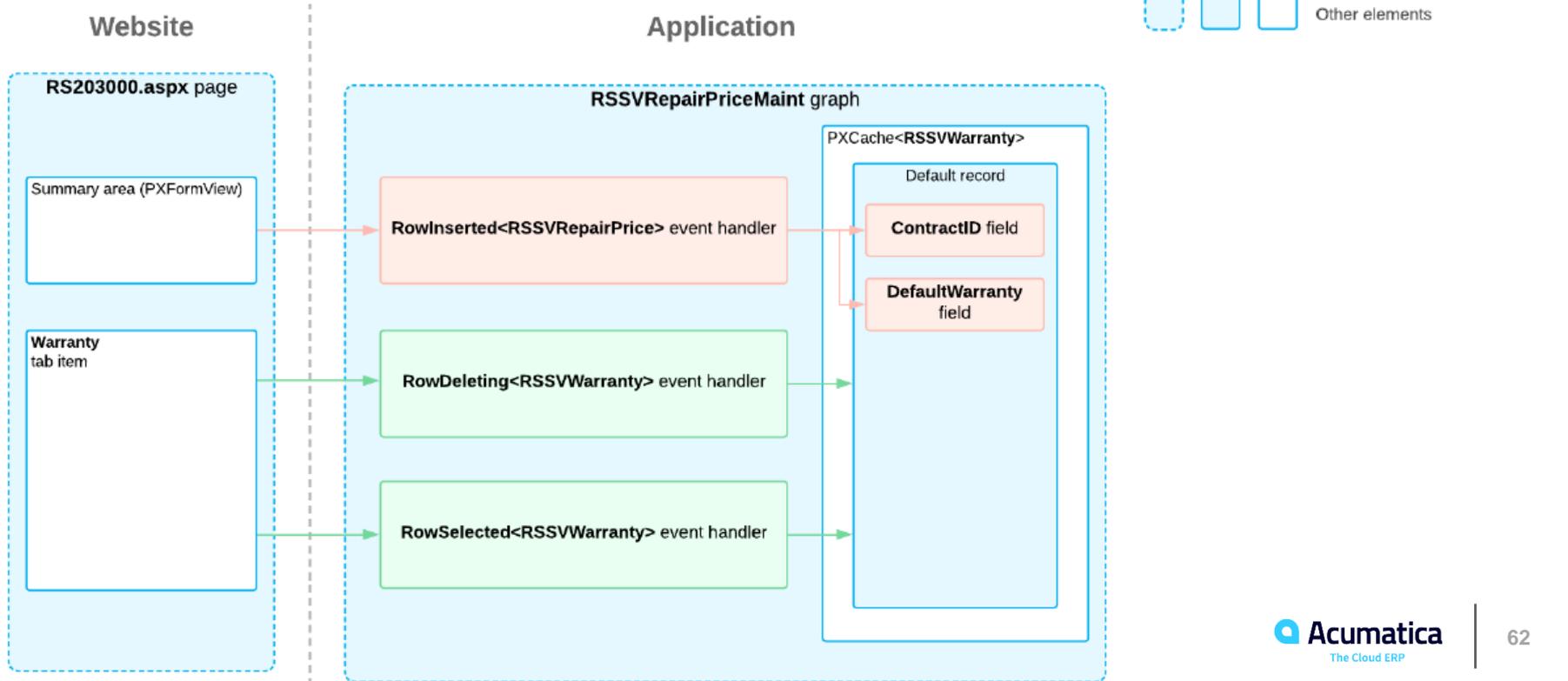
⏪ < > ⏩

Lesson Summary

In this lesson, you have learned how to add a default detail record to the grid. To add a default record, you have used the `RowInserted` event handler for the parent DAC. You have also defined the UI presentation of the default detail record in the `RowSelected` and `RowDeleting` event handlers for the child DAC.

Lesson Summary

Implementation of the Presentation Logic and the Insertion of the Default Detail Record



No Reliance

This document is subject to change without notice. Acumatica cannot guarantee completion of any future products or program features/enhancements described in this document, and no reliance should be placed on their availability.

Confidentiality: This document, including any files contained herein, is confidential information of Acumatica and should not be disclosed to third parties.



Thank you

Dmitrii Naumov