

PNMsoft Knowledge Base

Sequence User Guides

UX Studio Defining Controls

March 2014 Product Version 7.0 and above



© 2014 PNMsoft All Rights Reserved

This document, including any supporting materials, is owned by PNMsoft Ltd and/or its affiliates and is for the sole use of the PNMsoft customers, PNMsoft official business partners, or other authorized recipients. This document may contain information that is confidential, proprietary or otherwise legally protected, and it may not be further copied, distributed or publicly displayed without the express written permission of PNMsoft Ltd. or its affiliates.

PNMsoft UK 38 Clarendon Road Watford Hertfordshire WD17 1JJ

Tel: +44(0)192 381 3420 • Email: info@pnmsoft.com • Website: www.pnmsoft.com

Microsoft Partner

Gold Application Development



TABLE OF CONTENTS

Advanced Form Controls and Operations	1
Adding a Grid to a Form	1
Creating a Custom Grid Add/Edit Record Form	10
Defining Filtered Grids	11
Grid Features	11
Defining an Advanced Combo Box	13
Defining Filtered (Interconnected) Combo Boxes	14
Combo Box Caching Items	16
Form Table Editing	17
Managing Multiple Views	18
Loading Form Sub Views on Demand	20
Passing Parameters from Main Form to Sub View	21
Save Command Options	24
Implementing Validations	26
Validation Groups	27
Example of Compare Validator	27
Required Field Validator	28
Using Ajax Update Panels	28



Advanced Form Controls and Operations

Note: The guide assumes you are using the Options > Basic view. Some features in the guide require the Advanced view, and are noted as such in their section.



Basic View

Adding a Grid to a Form

Grids enable you to implement a multi-row form, such as an inventory list. There are several methods of adding a grid to a form. Below, we will describe one method. The video below shows an alternative method:

Adding a Grid to a Form using UI Generation – 5 min. Video

To add a grid to a form:

1. In the UX Studio, from the **Toolbox > Data** section, click the **Grid** control. A grid is added to the canvas.



Design Views Data Debug Form1 - DefaultView* Image: Save Refresh Options Validate Toolbox C Form1 G Image: Save Refresh Options Validate Toolbox C Form1 G Image: Save Refresh Options G Image: Save Refresh G G Image: Save Refresh G G Image: Save Refresh G G Image: Save Refresh C G G Image: Save Image: Save G Refresh C G Image: Save Image: Save Image: Save G G G G Image: Save Image: Save Image: Save Image: Save G G G G G Image: Save Image: Save Image: Save Image: Save G G G G G G G G G G G G G G G G G <th>Properties General) (ID) (accessibility Tab Index Appearance</th> <th>Grid1</th>	Properties General) (ID) (accessibility Tab Index Appearance	Grid1
Save Refresh Sive Sive Toolbox Validate Toolbox Image: Sive Sive Sive Sive Sive Sive Sive Sive	Properties General) (ID) Accessibility Tab Index Appearance	Grid1
Toolbox C F Text Form1 A Label Form description If Text Box Label Is Button Label Is Submit Button Label Is Data Picker Label Time Picker Grid Taskc Other All Time Picker Grid Taskc Owner Network Batabound Col0 Databound Col0 Databound Col2 No records to display. Effects Schema	Properties General) (ID) Accessibility Tab Index Appearance	Grid1
P Text A Label G G G G G G G G G G G G G G G G G G G	General) (ID) Accessibility Tab Index Appearance	Grid1
A Label Form description If Image: Text Box Label Image: Text Box Image: Mumeric Text Box Label Image: Text Box Image: Submit Button Label Image: Text Box Image: Submit Button Label Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box Image: Text Box	(ID) Accessibility Tab Index Appearance	Grid1
I Text Box I Label	Accessibility Tab Index Appearance	0
Image: Submit Button Label Transmit Button Image: Submit Button Label Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button Image: Submit Button <td< td=""><td>Tab Index</td><td>0</td></td<>	Tab Index	0
Es Batton Esbeit Esbei Label L	Appearance	
Examine Button		
Is Date Picker Ti Time Picker Ti Date and Time Picker Ti Date and Time Picker Ti Date and Time Picker Ti Databound Col0 Databound Col0 Databound Col1 Databound Col2 Refresh Schema Ti bit Upload Ti b	Grid Lines	None 🗸
Time Picker To Date and Time Picker To Month and Year Picker Combo Box To File Upload To File Upload To File Upload	Show Footer	
Month and Year Picker Databound Col0 Databound Col1 Databound Col2 Refresh Schema Combo Box No records to display. Viauto-generate Columns at Colamas and Colamas and Colamas and Colamas and Colamas and Colamas at Colam		
Databound Col0 Databound Col1 Databound Col2 Refresh Schema Viauto-cenerate Columns at Viauto-cenerate Columns at	(None)	~
File Upload		
	at Runtime	
V Check Box	mn at Runtime	
8 Radio Button List * Mandatory Fields □ Auto-generate Delete Columnation	lumn at Runtime	
I Text Area		
🚳 Sub View Command Item Display	Тор	~
a Standard Edit Mode	EditForms	~
A HyperLink General Features		
j⊴ Image		
Validation Enable Sorting		
Les Requirée rieie validator		
Enable Grouping		
Edit Templates		
Contraction of the second seco	Width	646px
Toolbox UL Generation Design Solid Source Stream TABLE TRUTH TABLE TRUTH Source Solid Source Source Solid Source Source Solid Source Source Solid Source Source Source Source Solid Source Solid Source S		

Add Grid to Canvas

2. Open the Grid's easy menu, select **Query Name > New Query** and create a query (e.g. a Table) for the grid data.

>	Grid Tasks	
Refresh	Query Name	(None) DefaultView
512	Refresh Schema	<new query=""></new>
	☑ Auto-generate Columns at	Runtime
	Auto-generate Edit Column	n at Runtime
	Auto-generate Delete Colu	mn at Runtime
	Edit Columns	
	Command Item Display	Тор 🗸
	Edit Mode	EditForms 🗸
	General Features	
	✓ Enable Paging	
	Enable Sorting	

New Query



[2] New	Query Wizard W	/ebpage Dialog	x
Creat Select	e New Query T the datasource fo	ype r the new query	
	Table Service	Lookup Table	Stored Procedure
		(DK Cancel

Create New Query

🧿 Tab	le Data Source Webpage	Dialog	x
Tabl	e Properties		?
	O Current Workflow		
	O External Table		
	New		
	Selected Table	UACT3e9a9d91c136403194d6f5da35c9f019	
	Name	GridTable	
	Key Name	fidIActId	
	Secondary Key Name		
	Connection String		
		OK Cancel	

Table Properties

3. From the top ribbon, open the Data Manager (select **Data > Data Model**), select to edit the grid's table and add columns (Data Fields) for the grid:



	5						
GridT	able_				Properties		
	Name	Туре		~	General		ľ
~		Text (String)	× ^		Can be blank (Nullable)		
	Quantity	Whole Number (Int3	× ×		Data Source	Date	
	Date ×	Date	× ×		Default Value		
					Display Name	Date	
					Is Primary Key		
				>			
Add	New Field						

Data Fields

4. From the Grid **Properties > Data editing** section, select: AllowAutomaticDeletes, AllowAutomaticInserts, AllowAutomaticUpdates

v
v
V

Grid Properties > Data Editing Section

- 5. From Grid's Easy Menu, select:
 - AutoGenerateDeleteColumn, to give the user a delete grid row option.
 - AutoGenerateEditColumn, to give the user an edit grid row option.



Grid Easy Menu



 (Optional) Modify the location of the Add new record button of the grid from the Easy Menu, by selecting options from the Command Item Display field. You can also modify the Edit Mode for grid rows (In Place, Edit Forms, Popup) here:

Grid Tasks		
Query Name	GridTable	~
Configure Where Parameters		
Configure Order By Paramet	ers	
Refresh Schema		
Auto-generate Columns a	at Runtime	
🗹 Auto-generate Edit Colur	nn at Runtime	
☑ Auto-generate Delete Co	lumn at Runtime	
Edit Columns		
Command Item Display	Bottom	~
Edit Mode	EditForms	~
	Grid Tasks Query Name <u>Configure Where Parameters</u> <u>Configure Order By Parameters</u> <u>Configure Order By Parameters</u> <u>Matto-generate By Parameters</u> <u>Auto-generate Columns as</u> <u>Auto-generate Edit Columns</u> <u>Edit Columns</u> <u>Command Item Display</u> <u>Edit Mode</u>	Grid Tasks Query Name GridTable Configure Where Parameters Configure Order By Parameters Refresh Schema ✓ Auto-generate Columns at Runtime ✓ Auto-generate Edit Column at Runtime ✓ Auto-generate Delete Column at Runtime Edit Columns Command Item Display Edit Mode

Command Item Display

- 7. (Optional) From the easy menu, select:
 - Enable Paging.
 - Enable Sorting
 - Enable Filtering
 - Enable Grouping

to enable end users to perform paging, sorting, filtering and grouping of the grid data.

General Features	
🗹 Enable Paging	
Enable Sorting	
Enable Filtering	
Enable Grouping	
Edit Templates	

User Grid Data Manipulation Settings

To edit the grid display columns, (for example to add the columns you added as Data Fields and to remove the columns you do not want to display, e.g. fldID, fldiwfID, etc.):

1. Open the Grid easy menu and select **Edit Columns**.

>	Cold Tealer	
C Refresh	Grid Tasks	
No Neirean	Query Name	GridTable 🗸
Date	Configure Where Parameters	
	Configure Order By Parameter	rs
	Refresh Schema	
	✓ Auto-generate Columns at	Runtime
	Auto-generate Edit Colum	n at Runtime
	Auto-generate Delete Colu	ımn at Runtime
	Edit Columns	

Edit Grid Columns



The Grid Editor appears.

Columns Double click a column fron column.	n the "	Available Columns" list to a	add it to your	grid. Use the Prope	erties Pane to config	ure your
Available Columns	:	Selected Columns		Appearance		•
Attachment		fldId (Numeric)		FooterText		
Button		ldIActId (Numeric)		HeaderImageUrl		
Calculated CheckBox		dAIId (Numeric) column (Bound)		HeaderText	Quantity	
ClientDelete ClientSelect		column1 (Bound) column2 (Bound)		Behavior		
DateTime DragDrop				AllowFiltering	✓	
DropDown				AllowSorting	✓	
HTMLEditor				DataFormatString		
Image	•			Display	✓	
Masked Numeric			×	FooterAggregateFo		
Rating RowIndicator				Groupable	✓	
Template				GroupByExpressio		
				ReadOnly		
				Reorderable	✓	
				Resizable	✓	~
				Convert this co	lumn to a Template	column
1						

Grid Editor

- 2. To add a column from the Available Columns list, select the column name and click OR double-click the column name. It moves to the Selected Columns list.
- 3. To delete a column from the Selected Columns list, click the column name and click
- 4. To move the position of the columns, click the column name and use the 🔸 buttons.
- 5. You can edit column properties in the Properties area on the right.
- Add several Bound columns and connect them to the data fields you added. You must enter the name of the column in the Data Field property of each column. Also change the Display property to the name of the column header you wish to display, and the DataType property to the correct type.



olumns	om the	"Available Columns" list to a	ıdd it to your	grid. Use the Prop	perties Pane to configur	e your							
Available Columns Attachment Bound Button Calculated CheckBox		Selected Columns fldId (Numeric) fldIWfId (Numeric) fldAIId (Numeric) fldAIId (Numeric) column (Bound)		Groupable GroupByExpress ReadOnly Reorderable									
ClientDelete ClientSelect DateTime DragDrop DropDown EditCommand HTMLEditor HyperLink Image	•	column1 (Bound) column2 (Bound)	column1 (Bound) column2 (Bound)	•	Resizable ShowSortIcon SortExpression Visible								
Masked Numeric Rating RowIndicator Template											Data Aggregate DataField MaxLength	None Quantity 0	~
				Misc AndCurrentFilter AndCurrentFilter Convert this o	Fu NoFilter Va column to a Template co	► ►							

Adding Data Field Columns

7. (Optional) Click **Convert this column to a Template column**, to reuse the column's settings.

Now your grid should include the columns you have selected.

label				
label				
label				
+ Add	d new record			€ Refresh
	Item	Category	Quantity	Date
	No records to	display.		

Grid Columns



Now you are ready to run or debug your grid form:

Process #3414		Request Invoice						
Activity Properties	^	Request In Enter invoice i	nvoice tems					
Permission Full Con	ntrol	Name	ame Ron Samson					
Workflow Properties	^	Date		13/03/20	14			
Created 25/03/2	2014	Item	Qty	Cost	Date			
Created By Eli Stutz	z	Item: Desk						
Last Updated 25/03/2	2014	Qty: 18						
Permission Pull Con		Date: 20/03/	2014	Ê				
Start		~ ×						
Request Invoice		Laptop	23	455	11/03/2014 00:00:00	×	-	
		+ Add new	record	41	19/05/2014 00:00:00	2	Refresh	
		* Mandatory Fiel	lds					

Grid in Flowtime

To add a dropdown list as one of your grid columns:

- 1. In the Data Model (**Data > Data Model**), create or select a Lookup Table query for the data of the dropdown list.
- 2. Change the view to advanced user view (from the top ribbon, select **Design > Options.** Select **Advanced**).

UX Studio Options Webpage Dialog	×
UX Studio Options	
Advanced ✓ Automatic Data Source Management Auto Register with Data Manager Show Advanced Property Grid Mode Show Advanced Toolbox Display Non-visual Controls	:
	OK Cancel

Advanced View

- 3. From the **Toolbox > Data** section click **Data Source**. A Data Source is added to the form.
- 4. In the **Entity Data Source > Properties** pane, in the **QueryName** property, select the Lookup Table query (e.g. Categories).



Data		
Expressions		
OrderBy		
Query Name	Categories	~
Where		
where		

Query Name Property

5. Select **Edit Columns** from the Grid's Easy Menu. Select the dropdown column, and in the DataSourceID attribute, enter the value of the data source you added in step 3.

editor olumns	rom the	"Available Columns" list to a	dd it to your	arid. Use the Prope	rties Page to confi	
Available Columns Attachment Bound Button Calculated CheckBox ClientDelete ClientDelete ClientDelete DragDrop BrogDown EditCommand HTMLEditor HyperLink Image Masked Numeric Rating RowIndicator Template		Selected Columns column (Bound) column1 (Bound) column2 (Bound) column3 (Bound) column3 (Bound)		ReadOnly Reorderable Resizable ShowMoreResultsE ShowSortIcon SortExpression Visible Data AllowAutomaticLoz DataField DataSourceID ListDataMember ListTextField ListValueField Misc AndCurrentFilterFu Convert this col		× ie column

For Example : DataSourceID="EntityDataSource2"

DataSourceID



Creating a Custom Grid Add/Edit Record Form

By default, the grid automatically includes an **Add New Record** button, which opens a form where the user can enter a new record by entering the values of the grid display columns. The edit record form also displays these columns.

Alternative, you can create a custom form for a user to add new/edit records of the grid.

To create a custom add/edit record form that contains different fields than the display columns:

1. From the Grid's easy menu, click Edit Templates.

>	Grid Tasks								
esn	Query Name	GridTable	\checkmark						
T	Configure Where Parameters								
	Configure Order By Parameter	<u>s</u>							
	Refresh Schema								
	Auto-generate Columns at Runtime								
	Auto-generate Edit Column at Runtime								
	☑ Auto-generate Delete Column at Runtime								
	Edit Columns								
	Command Item Display	Тор	\checkmark						
	Edit Mode	EditForms	~						
	General Features								
	✓ Enable Paging								
	Enable Sorting								
	Enable Filtering								
_	Enable Grouping								
	Edit Templates								

Edit Templates

2. From the dropdown list, select MastertableViewEditForm.

Grid Tasks	MasterTableView
Display	NoRecordsTemplate
End Templat	ePagerTemplate
	CommandItemTemplate
	Nestedviewiempiate
	EditItemTemplate
	MasterTableViewEditForm
	MasterTableView: Column 0 - Item
	MasterTableView: Column 1 - Category
S	ub MasterTableView: Column 2 - Quantity
	MasterTableView: Column 3 - Date

Edit Form

3. Edit your custom form, adding values for the fields you want the user to complete when adding a new record/editing a record of the grid. Ensure that each input control is bound to your field in the data source.

Note: For ease of development, you can create a form in your view and copy the markup to MasterTableViewEditForm.



Defining Filtered Grids

The UX Studio enables you to create a filtered grid (filtered with fields on the same form), using a Where Parameter wizard. <u>Click here</u> for how to use this feature.

Grid Features

You can enable several additional grid features by editing the grid markup.

The example markup below presents a grid with several added features (see the color code in the markup below):

Features:

- Enable Ajax on the grid
- Allow sort on the grid columns
- Set the add record position: Top, Bottom, TopAndBottom
- Set the Edit mode, it can be PopUp, InForm, InPlace
- Edit and delete columns with Images
- Images instead of text in the edit\insert form
- Client side reorder animation
- Center the edit form popup to the center of the screen
- Drop Down column

Markup:

<script type="text/javascript">

var popUp; function PopUpShowing(sender, eventArgs) { popUp = eventArgs.get_popUp();

var gridWidth = sender.get element().offsetWidth;

var gridHeight = sender.get element().offsetHeight;

var popUpWidth = popUp.style.width.substr(0,

popUp.style.width.indexOf("px"));

var popUpHeight = popUp.style.height.substr(0, popUp.style.height.indexOf("px"));

popUp.style.left = ((gridWidth - popUpWidth) / 2 + sender.get element().offsetLeft).toString() + "px";

popUp.style.top

(sender.get_element().offsetTop/2).toString() + "px";

}

</script>

<asp:UpdatePanel ID="UpdatePanel1" runat="server"><ContentTemplate>

<sq:Grid runat="server" ID="Grid2" DataSourceID="Address_DS" GridLines="None" Width="95%" CellSpacing="0" Culture="English (United



Kingdom) " AllowAutomaticDeletes="True" AllowAutomaticInserts="True" AllowAutomaticUpdates="True" AllowTrue OrderBy="Address""> <MasterTableView CommandItemDisplay="Bottom" DataKeyNames="fldId" DataSourceID="Address DS" EditMode="PopUp" AutoGenerateColumns="False"> <CommandItemSettings ExportToPdfText="Export to PDF"></CommandItemSettings> <RowIndicatorColumn Visible="True" FilterControlAltText="Filter RowIndicator column"></RowIndicatorColumn> <ExpandCollapseColumn Visible="True" FilterControlAltText="Filter ExpandColumn column"></ExpandCollapseColumn> <Columns> <sq:GridDropDownColumn DataSourceID="Foreign DS" ListTextField="Foreign" ListValueField="fldId" DataField="Foreign" AllowAutomaticLoadOnDemand="True" HeaderText="Foreign" SortExpression="Foreign" UniqueName="Foreign" FilterControlAltText="Filter Foreign column"></sq:GridDropDownColumn> <sq: GridDropDownColumn DataSourceID="Countries DS " ListTextField="Country" ListValueField="fldId" DataField="Country' HeaderText="Country" SortExpression="Country" UniqueName="Country" FilterControlAltText="Filter Country column"></sq:GridBoundColumn> <sq:GridBoundColumn DataField="Address" HeaderText="Address" SortExpression="Address" UniqueName="Address" FilterControlAltText="Filter Address column"></sq:GridBoundColumn> <sq:GridBoundColumn DataField="ComplementAddress" HeaderText="Complement Address" SortExpression="ComplementAddress" UniqueName="ComplementAddress" FilterControlAltText="Filter ComplementAddress column"></sq:GridBoundColumn> <sq:GridBoundColumn DataField="PostalCode" HeaderText="Postal Code" SortExpression="Postal Code" UniqueName="PostalCode" FilterControlAltText="Filter PostalCode column"></sq:GridBoundColumn> <sq:GridBoundColumn DataField="Locality" HeaderText="Locality" SortExpression="Locality" UniqueName="Locality" FilterControlAltText="Filter Locality column"></sq:GridBoundColumn> <sqr:GridEditCommandColumn ButtonType="ImageButton"</pre> EditImageUrl="~/Shared Resources/images/Orna/Edit.gif" FilterControlAltText="Filter EditCommandColumn column"></sqr:GridEditCommandColumn> <sqr:GridButtonColumn ButtonType="ImageButton" CommandName="Delete" Text="Delete" ImageUrl="~/Shared Resources/images/Orna/Delete.gif" UniqueName="DeleteColumn" FilterControlAltText="Filter DeleteColumn column"></sqr:GridButtonColumn> </Columns> <EditFormSettings>



<EditColumn ButtonType="ImageButton" InsertImageUrl="~/Shared
Resources/images/Orna/AddRecord.gif" UpdateImageUrl="~/Shared
Resources/images/Orna/Update.gif" CancelImageUrl="~/Shared
Resources/images/Orna/Cancel.gif" UniqueName="EditCommandColumn1"
FilterControlAltText="Filter EditCommandColumn column"></EditColumn><///>
<///>

</EditFormSettings>

</MasterTableView>

<FilterMenu EnableImageSprites="False"></FilterMenu>

<ClientSettings AllowDragToGroup="True" AllowColumnsReorder="True" ReorderColumnsOnClient="True" ColumnsReorderMethod="Reorder">

<Selecting AllowRowSelect="True"></Selecting>

<ClientEvents OnPopUpShowing="PopUpShowing"></ClientEvents>

<Animation AllowColumnReorderAnimation="True" AllowColumnRevertAnimation="True"></Animation>

</ClientSettings>

<HeaderContextMenu CssClass="GridContextMenu
GridContextMenu Default"></HeaderContextMenu>

</sq:Grid><sq:BoundControl runat="server" TargetControlID="Grid2" AutoConfig="True"></sq:BoundControl>

</ContentTemplate>

</asp:UpdatePanel>

<sq:EntityDataSource runat="server" ID=" Address_DS " QueryName="Addresses" EnableInsert="True" EnableUpdate="True"></sq:EntityDataSource>

<sq:EntityDataSource runat="server" ID="Countries_DS" QueryName="Countries" ></sq:EntityDataSource>

Defining an Advanced Combo Box

The UX Studio provides advanced control over a combo box's appearance and behavior with a set of properties you can define.

To define an advanced combo box:

- 1. From the top ribbon > **Options**, select **Advanced** mode.
- 2. Click the combo box, and from the Properties tab, select the following options:
 - EnableAutomaticLoadingOnDemand: enable automatic loading of list results.
 - ShowMoreResultsBox: display the Show More Results box.



• **ItemsPerRequest:** define the number of items to display in the drop-down list (e.g. 10 in the figure below).

C UX Studio - Form1 - Windows Internet Explo	rer		
Design Views Data Debug	Form1 - DefaultView*		
Save Refresh Options	Validate		
Toolbox <		Properties	
⊿ Data	Inventory Request	EnableLoadOnDemand	
- Form	Add your inventory items	EnableScreenBoundaryDetection	
🔛 Grid		EnableTextSelection	\checkmark
🔔 Data Source	Name	EnableViewState	\checkmark
Porm Part	Data	EnableVirtualScrolling	
▲ Layout		ErrorMessage	
One Column Table	ID v v	ExpandAnimation	PNMsoft.Rad.Web.UI.ComboBoxAnimationSetti
Two Column Table		ExpandDelay	100
Three Column Table	DataSource - DataSource1	ExpandDirection	Down
Panel	+ Add new record	Filter	None
✓ Editors		HighlightTemplatedItems	
I lext	Item Category Quantity	IsCaseSensitive	
A Label	No records to display.	ItemRequestTimeout	300
im Text Box		Items Per Request	10
In Numeric Text Box	* Mandatory Fields	MarkFirstMatch	
Masked Text Box		MaxLength	0
Date Input	DataSource - Form1DataSource	Minimum Filter Length	0
Button		NoWrap	
Submit Button		OpenDropDownOnLoad	
15 Date Picker		PostBackUrl	
is Time Picker		RegisterWithScriptManager	✓
15 Date and Time Picker		RenderingMode	Full 🗸
Comba Bay		ShowDropDownOnTextboxClick	\checkmark
E Combo Box	L	ShowMoreResultsBox	
		ShowToggleImage	✓
Toolbox UI Generation	Design Split Source sq:Form TABLE TR TD TABLE TR TD sq:ComboBox	<	>

Combo Box Properties



Advanced Combo Box

You can define additional combo box properties from the Properties pane.

Defining Filtered (Interconnected) Combo Boxes

The UX Studio enables you to create filtered (interconnected) combos. For example, a combo of countries and a combo of cities. When you select the country, the combo of cities is repopulated based on the country selected.

To define filtered (interconnected) combo boxes:

- 1. Set the first combo as AutoPostBack.
- 2. Create a function to clear the second combo (e.g. ClearCities in example below).



- Call this function on OnClientSelectedIndexChanging and OnClientKeyPressing events of the first combo.
- 4. Set the second combo as EnableAutomaticLoadOnDemand.
- 5. Add where logic to filter the data source control of the second combo based on the first combo selection (see where expression in example below):
 - For table-based combo boxes, add a "where" expression in the DataSource and a WhereParameters section.
 - For service-based combo boxes, add only a WhereParameters section.
 Define the service ds parameter value as :@[parameter name in WhereParameter] and the Mask = "In".

For Example:

```
<script type="text/javascript">
function ClearCities(sender, eventArgs)
{
var clientId = $sq("[id$=' CitiesComboBox']")[0].id;
var citiesCombo = $find(clientId);
citiesCombo.get items().clear();
citiesCombo.set text("");
}
</script>
<asp:UpdatePanel runat="server"><ContentTemplate>
<sq:ComboBox runat="server"
ID="CountriesComboBox"
DataTextField="Name"
DataValueField="fldId"
AutoPostBack="True"
DataSourceID="CountriesDataSource"
OnClientSelectedIndexChanging="ClearCities"
OnClientKeyPressing="ClearCities"></sq:ComboBox>
<sq:ComboBox runat="server"
ID="CitiesComboBox"
DataTextField="Name"
DataValueField="fldId"
EnableAutomaticLoadOnDemand="True"
DataSourceID="CitiesDataSource">
</sq:ComboBox>
<sq:DataSource runat="server"
ID="CountriesDataSource" QueryName="Countries">
</sq:DataSource>
<sq:DataSource runat="server" ID="CitiesDataSource"
QueryName="Cities"
where="Convert.ToInt32(it["CountryId"])=Convert.ToInt32(@Countr
yId)" >
<WhereParameters>
<asp:ControlParameter ControlID="CountriesComboBox" Name="CountryId" />
</WhereParameters>
</sq:DataSource>
</ContentTemplate>
</asp:UpdatePanel>
```



Combo Box Caching Items

See Combo Box Load On Demand.



Form Table Editing

A table editing easy menu makes form Table editing easy. To add a row or column to your form, simply select the required action from the easy menu:

UX Studio - Initia	ition Form - Windows Inte	ernet Explorer	_	
http://10.10.10.9	9090/administration/Wel	b/UI/UXStudio/Default.aspx?workflowId=a03a3116-075f-49cf-a0e5-e6124cce9e93&activityId=76EB7DF8-7030-4485-8CC6-0D1DBFAAD0A5		
Design Views	Debug	Initiation Form - DefaultView		
Data Model	Lookup Tables Data	Save Refresh Options -		
Γoolbox	<		Properties	
⊿ Data	^	Modify employee details	Misc	
- Form			(ID)	
🛗 Grid			Align	
🚯 Data Source		Process Number : [orocess Number] Creation Date : [Label11] State : Started	BgColor	
▲ Layout		Process Name : [Label5]	Class	
🛄 One Column	Table =	Employee Number : [Employee number] Employee Name :[employee name]	ColSpan	
🛄 Two Columr	Table		Dir	
Three Colun	nn Table	Full Please insert you Table Cell Tasks	NoWrap	
Panel		Insert Row Above	= RunAt	
Editors		Family Nu Insert Row Below Status: Chi Insert Column to the Left	Style	width:40%
A Label		Insert Column to the Right	Title	
Text Box		Nationality : Card Select	VAlign	
📼 Numeric Te	kt Box			
Masked Tex	t Box			
Date Input				
🕞 Button		Foreign Country Address Complement Address Postal Code Locality		
Submit Butt	on	No records to display.		
Date Picker		+ Add new record		
Time Picker			-	
Date and Time	me Picker	Descent Terra		
Month and Y	rear Picker	Vocument Type IVO. Validity Deactivate	-	
oolbox UI Gene	ration	Design Split Source DIV sa:Form DIV TABLE TR TD		
9 0		N 🙆 💁 🔛 🛃		EN 🔺 🛜 07:54

Table Cell Editing



Managing Multiple Views

The UX Studio enables you to create multiple views within the same Template for various platforms (e.g. PC, Mobile). Each of these views access the same data model.

To create a new view:

1. In the UX Studio top ribbon > Views tab, click **New**.

http://10.10.10.152:9090/?workflowId=a15a9ac0-e60a-4326-80b2-3df41d68dc54&activityId=552D8D37-1 -								
Design Views Debug]			Accoun				
DefaultView		Delete	New					
Toolbox	<	*Name		Name is				
▲ Layout	<u> </u>	*0		Class is a				
🛄 One Column Table		Class						
Two Column Table		Approved		Checkbox				
Three Column Table		Start Date						
Panel	E	End Date						
▲ Editors		Submit		~~~~				
A Label								

New View

A new view is opened in the UX Studio Canvas, named View1 (the original view is called DefaultView).

Now you can design a separate view according to your needs.

To toggle between views:

1. In the UX Studio top ribbon > Views tab, select the view of your choice from the dropdown list.



Toggle Between Views

To delete a view:

1. In the UX Studio top ribbon > Views tab, click **Delete**. The selected view is deleted.

Design	Views	Debug			-	
DefaultVie	W		•	Delete	New	
Toolbox				Name		
⊿ Layout	:			al		
🔲 One	e Column T	able		Class		

Delete a View



To add a sub view to the current view

- 1. Create the two views (the main view and the view you wish to add as a sub view see To create a new view, above).
- 2. In the current view, from the Toolbox Editors area, click **Sub View**. A Sub View control is added to the UX Studio Canvas.
- 3. In the Sub View's **Properties** pane, **Misc** section, enter the name of the view you wish to add in the VirtualPath property. Now this view is included as a sub view in the current view.

You can add sub views from the current form, from the other activities in the workflow, or from other workflows. To point Sequence to the correct sub view in other forms, in the sub view properties click ... to select the sub view from another form, or enter the VirtualPath to the subview as follows:

🧉 UX Studio - Request Form - Windows Internet Explorer								
http://10.10.10.5:9090/Administration/Web/UI/U	XStudio/Default.aspx?workflowI	=6920a99a-0653-4b9b-a5c5-8458308ba977&activityId=E1697EE0-48AA-48A6-A312-DEA509CBB068						
Design Views Data Debug		Request Form - DefaultView*						
Save Refresh Options	Salidate							
Toolbox Toble Div Toble Div	Request Form Form description Request Details Enter the request details Date Date Request Priority	Image: Approval Form description Please provide approval Approve of decline this request Name Date Approve Approval		Properties 24 (General) (ID) Behavior EnableViewState Visible Data Expressions Misc VirtualPath	SubView1			
Toolbox UI Generation Des	ign Split Source sq:Form	TABLE TR TD TABLE TR TD sq:SubView						

../../Space_name/wf_name/Form_name/View_name

Adding a Sub View



Loading Form Sub Views on Demand

When your implementation requires showing and hiding sub-views in the parent view, it is recommended to use one sub view with changed virtual path than many sub-view controls.

Using many sub view controls may result in a very large control tree on the server side which will burden the performance.

The following client side markup and server side code achieves this switch upon button click:

Client-side Markup:

```
<%@ Control Inherits="SequenceEx.Forms.Samples.CustomFormControl, ...." %>
<sq:Button runat="server" Text="Switch View" ID="Button1"
OnCommand="OnSwitchButtonCommand" CommandArgument="View2"></sq:Button>
<sq:Button runat="server" Text="Switch View" ID="Button2"
OnCommand="OnSwitchButtonCommand2"></sq:Button>
```

```
<sq:SubView runat="server" ID="SubView1"
VirtualPath="View1"></sq:SubView>
```

Service-side Code:

```
using System.Web.UI.WebControls;
using PNMsoft.Sequence.Forms.Web.UI;
using Sq = PNMsoft.Sequence.Forms.Web.UI.Controls;
namespace SequenceEx.Forms.Samples
{
public class CustomFormControl : FormControl
{
    protected void OnSwitchButtonCommand(object sender, CommandEventArgs e)
    {
           Sq.SubView subview =
           (Sq.SubView)this.TemplateControl.FindControl("SubView1");
           subview.VirtualPath = (string)e.CommandArgument;
    }
    protected void OnSwitchButtonCommand2(object sender, CommandEventArgs e)
    {
           Sq.SubView subview =
           (Sq.SubView)this.TemplateControl.FindControl("SubView1");
           subview.VirtualPath = "View2";
    }
}
}
```



Passing Parameters from Main Form to Sub View

Note: this feature is available from v7.7 and above.

You can pass parameters from a form to its sub view. For example, you have a sub view which displays records of employees, and in the parent form there is a combo box, where the user can select an employee. You want the sub view to display the details of the selected employee.

Toolbox	<		Main Form
⊿ Data		Select Employee:	
🗗 Form		Employee Details	
🖽 Grid		Employee Details	
💂 Data Source		ID	[fldEmployeeId]
🟳 Form Part			Sub View
⊿ Layout		First Name	[fldEmpName]
One Column Table		Last Name	[fldEmplastName]
📰 Two Column Table		Last Nume	[idempeasivene]
🛄 Three Column Table		Username	[fldEmpUseName]
Panel			
▲ Editors		Email	[fldEmail]
a _i ⊤ext			· · · · · · · · · · · · · · · · · · ·

Passing Parameters to Sub View

You can do so using the following code:

Main Form:

<%@ Control %>

<sq:Label runat="server" Text="Select Employee:" Width="100px" />

```
<sq:DataSource runat="server" ID="DataSource1"
QueryName="ViewEmployees"></sq:DataSource>
```

```
<sq:ComboBox runat="server" ID="ComboBox1" DataSourceID="DataSource1"
AutoPostBack="true" DataTextField="Name" DataValueField="Id"
EnableAutomaticLoadOnDemand="True" ShowMoreResultsBox="True"
ItemsPerRequest="10"></sq:ComboBox>
```

<sq:SubView runat="server" ID="SubView1" VirtualPath="../UserDetails/DefaultView.ascx" Skin="Default">

<Parameters>

<asp:ControlParameter Name="EmployeeId" ControlID="ComboBox1" DbType="Int32" />

</Parameters>



</sq:SubView>

Sub View:

<%@ Control %>

<sq:Form runat="server" ID="tblEmployeesForm" DataKeyNames="fldEmployeeId"

DataSourceID="tblEmployeesDataSource" ReadOnly="true" CellPadding="0" CellSpacing="0" Height="217px"><ContentTemplate>

```
<h2><sq:Text runat="server" Text="Employee Details"></sq:Text></h2>
```

```
<sq:Label runat="server" Text="ID" />
```

```
<sq:Label runat="server" ID="fldEmployeeId"></sq:Label>
```

<sq:BindableControl runat="server" TargetControlID="fldEmployeeId" DataField="fldEmployeeId"></sq:BindableControl>

```
<sq:Label runat="server" Text="First Name" />
```

<sq:Label runat="server" ID="fldEmpName"></sq:Label>

<sq:BindableControl runat="server" TargetControlID="fldEmpName" DataField="fldEmpName"></sq:BindableControl>



<sq:Label runat="server" Text="Last Name" />

<sq:Label runat="server" ID="fldEmpLastName"></sq:Label>

<sq:BindableControl runat="server" TargetControlID="fldEmpLastName" DataField="fldEmpLastName"></sq:BindableControl>

<sq:Label runat="server" Text="Username" />

<sq:Label runat="server" ID="fldEmpUseName"></sq:Label>

```
<sq:BindableControl runat="server" TargetControlID="fldEmpUseName"
DataField="fldEmpUseName"></sq:BindableControl>
```

<sq:Label runat="server" Text="Email" />

```
<sq:Label runat="server" ID="fldEmail"></sq:Label>
```

<sq:BindableControl runat="server" TargetControlID="fldEmail" DataField="fldEmail"></sq:BindableControl>

</ContentTemplate>

</sq:Form>



<sq:DataSource runat="server" ID="tblEmployeesDataSource"

QueryName="tblEmployees" Where="fldEmployeeId == @fldEmployeeId"><WhereParameters>

<sq:TemplateControlParameter PropertyName="Parameters["EmployeeId"]" Name="fldEmployeeId"></sq:TemplateControlParameter>

</WhereParameters>

</sq:DataSource>

<sq:BoundControl runat="server" TargetControlID="tblEmployeesForm"></sq:BoundControl>

Save Command Options

You can add Submit/Update buttons to your forms which enable the user to save the form data. You must give this button a CommandName in the Properties pane.

The UX Studio includes two save CommandName options:

- Save: saves all the controls in the current view, but not the controls in its sub views.
- SaveAll: saves all controls in this view and in all its sub views.

🦉 UX Studio - Forn1 - Windows Internet Explorer							
Design Views Data Debug	Form1 - DefaultView						
	2						
Save Refresh Options	Validate						
Toolbox <	T					Properties	>
⊿ Layout	Inventory Request					(General)	
One Column Table	Add your inventory items					(ID)	cti08
Two Column Table						Accessibility	
Three Column Table	Name					Tab Index	0
▲ Editors	Date	m				Appearance	
I Text						Text	Submit
A Label	ID					Behavior	
I Text Box				a		Command	
Button	 Add new record 			ið Retresh		Argument	1
Submit Button	Item	Category	Quantity	Date		Command Name	SaveAll
Date Picker	No records to displa	iy.				Enabled	
is Time Picker						Visible	
15 Date and Time Picker	* Mandatory Fields				Submit 0	Visible	
is Month and Year Picker						Layout	
E Combo Box						Height	
. File Upload						wider	
Check Box							
🔠 Radio Button List							
I Text Area							
🎡 Sub View							
▲ Standard							
A HyperLink							
walidation							
a valuation							
Toolbox UI Generation	Design Split Source sq:Form	TABLE TR TD TABLE TR TD	sq:SubmitButton				
						-	

Save CommandName Options – SaveAll

Note: you can add the Submit Button control which comes with default submit properties already configured (command = SaveAll, text = Submit).



You can also control the behaviour upon clicking the save button using the CommandArgument field. The following CommandArgument options are available:

For Forms:

- 0 Save but do not Submit.
- 1 Save and Submit.

For Tasks:

- 1,0 Close the task, Save but do not Submit.
- 1,1 Close the task, Save and Submit.
- 0,0 (this is the same as just writing 0) Save but do not Close the task or Submit it.
- 1,2 Set the action items to be closed when the first user submits the task.

Note: 'Submit' here means redirect to the next activity.



Implementing Validations

Sequence Forms use out-of-the-box ASP.NET functionality for validations. ASP.NET contains validation controls and infrastructure which you can use to perform validations for the form. The simplest use for validators is for defining required fields for a user form:

ocess #749		Account Details	
Activity Pro	perties 🔅	*Name	Alan Waldman
Created	29/03/2012	*Class	Class is a required field.
Permission	Full Control	Approved	Checkbox
		Start Date	21/03/2012
Workflow P	roperties 🌼	End Date	29/03/2012
Workflow Name	Validation Notice	Submit	
Created	29/03/2012	Submit	
Created By	System Administrator		
Last Updated	29/03/2012		
Permission	Full Control		
Start	tails	•	

Required Field

Sequence provides advanced validators as well. The following validation controls are available from the UX Studio Toolbox:

- **Required Field Validator:** provides an error message if the user did not enter a value for this field.
- Range Validator: provides a range for validation.
- **Regular Expression:** provides a regular expression validator.
- Compare Validator: compares the user value with the provided value.
- **Custom Validator:** enables you to provide custom code for client validation or server side validation.
- Validation Summary: use this control to organize all validation messages into one summary.

To add a validator for a control:

- 1. From the top ribbon > **Options**, select **Advanced** mode.
- 2. From the **Toolbox**, click the Validator of your choice (e.g. Required Field Validator).

It is placed on the Form Canvas.

- 3. In the Properties Pane:
 - Enter the id of the control you want to validate (e.g. Textbox1) in the **ControlToValidate** property.
 - Enter the error message in the **ErrorMessage** property (e.g. 'Name is required. Please enter your name').



http://10.10.10.152:9090/?workflowId=a1	5a9ac0-e60a-4326-80b2-3df4	1d68dc54&activityId=	552D8D37-1 - Windows Internet Explor	er		_	
esign Views Debug			Account Details - DefaultView*				
iteliaad Contraction Contracti							
olbox <	*Name		Name is a required field.		Р	roperties	
Literal	*Class		Class is a required field.		(General)	
Calendar	Approved	Chaskboy			((ID)	RequiredFieldValid
⊨ File Upload	Start Date				A	ccessibility	
Grid View	Start Date				4	AccessKey	
Data List	End Date				1	FabIndex	0
Repeater	Submit				A	ppearance	_
List View					E	BackColor	
• Wizard					E	BorderColor	
Xml					E	BorderStyle	NotSet
Multi View					E	BorderWidth	
Panel						LISSCIASS	Chattin
 <] Place Holder						Display	Classic
View						ErrorMessage	Class is a required
alidation						-one	
Required Field Validator						ForeColor	Red
Range Validator						lext	
Require	ed Field Validator				В	lehavior ControlToValidato	TaytBay2
Compare Validator	BoundControl - BoundC	Control 1				EnableClientScript	
Custom Validator	DataSource - DataSour	ce1					
Validation Summary						Enabled	
jax Extensions					E	EnableTheming	M
) Timer					E	EnableViewState	V
Update Panel					I	InitialValue	
Update Progress					S	SetFocusOnError	
	<u> </u>				5	SkinID	
script://Required Field Validator	Design Split Source sq:	Form TABLE TR TD sq:	RequiredFieldValidator		F	Properties Data N	1odel

Adding Validator Control

4. Define additional properties for the validator (e.g. compare values, operators).

Each validation control can be configured, enabling you to choose how to show the validation, e.g. on the form itself or in a message box. You can also write your own validation controls.

Validation Groups

You can add validation controls to validation groups. Each control in this group will be validated according to the settings of the group.

Example of Compare Validator

You can add a compare validator to compare two fields on the form, for example, comparing if the end date is greater or equal to the start date:

Toolbox <				Properties	
🔛 Image Map 🔷	"Name	RequiredFieldValidator		TabIndex	0
abi Hidden Field	*Clace			Appearance	
Literal		RequiredFieldValidator		BackColor	
📅 Calendar	Approved	Checkbox		BorderColor	
堶 File Upload	Start Date			BorderStyle	NotSet 💌
🚰 Grid View	End Date			BorderWidth	
📴 Data List	Submit	k		CssClass	
Repeater				Display	Static 💌
🖽 List View				ErrorMessage	CompareValidator
*+ Wizard				Font	
🔣 Xml				ForeColor	Red
C Multi View				Text	
Panel				Behavior	
🖂 Place Holder				ControlToCompare	DatePicker1
🔁 View				ControlToValidate	DatePicker2
▲ Validation				CultureInvariantVal	
Required Field Validator				EnableClientScript	
🙄 Range Validator 🗮	BoundControl - BoundControl1			Enabled	V
Regular Expression Validator	Deter Courses Deter Course (EnableTheming	▼
🔁 Compare Validator	DataSource - DataSource1			EnableViewState	V
Sustom Validator	CompareValidator			Operator	GreaterThanEc 💌
Validation Summary				SetFocusOnError	
 Ajax Extensions 				SkinID	
🖔 Timer			I	ToolTip	
🖉 Update Panel 🔫				Turne	String -
Toolbox UI Generation	Design Split Source asp:CompareValidator			Properties Data M	odel

Compare Validator



Required Field Validator

You can easily set a field as required. To do so, from the easy menu click the

Add Required Field Validator link. This adds a red asterisk beside the field and enforces it as required before submit. You can change the validation text as required by clicking the asterisk and editing its properties on the right.

Account Requ Form description	est
Label	TextBox Tasks
Label	Add Required Field Validator
Label	
Label	

Add Required Field Validator

Using Ajax Update Panels

Use AJAX and Update Panels to refresh the page wisely. You may be limited in how often to refresh the page, or which parts of the form to refresh. Therefore, use AJAX technology extensively, as this will improve performance and user experience, and will reduce the weight of the page.

To use AJAX update panels – with Sequence combo boxes, grids or sub views:

1. In the markup, wrap your controls with the following tag:

```
<asp:UpdatePanel runat="server" class="myStyle"><ContentTemplate>
----Controls----
</ContentTemplate> </asp:UpdatePanel>
```