

Top Ten Forms Editing Techniques

Document ID: Q000073

Last Revised On: October 03, 2008

This article applies to the following:

Product Version: IssueNet 4.0 and later

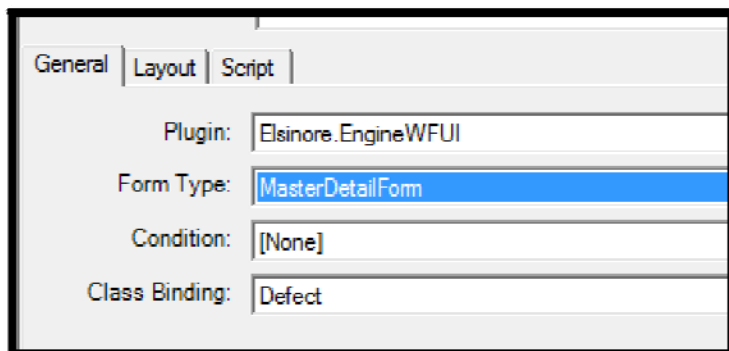
Component(s): Architect

Solutions(s): All

The IssueNet forms editor provides precise control over the appearance of forms and the options users have for entering and editing information. However, the number of available options can obscure some of the more important forms editing features and techniques. This tech tip assembles a top ten list of forms editing techniques that address our users most frequently asked questions.

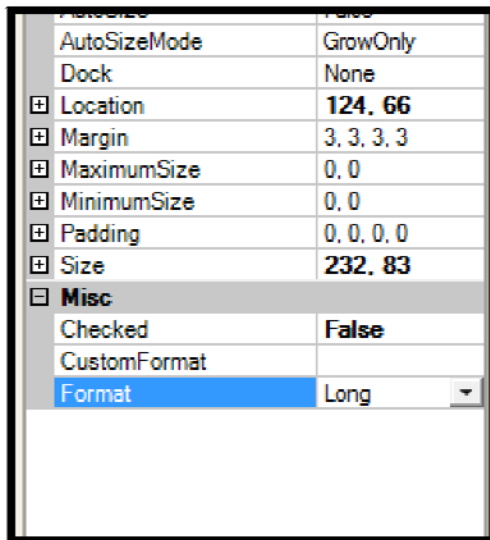
When building a new form use the **Form Type** property to choose a pre-defined form template.

When creating a new form you don't need to create the entire form layout from scratch. IssueNet offers number of form templates you can use as starting points for your own forms. To select a form template change the value of the **Form Type** field on the **General** tab of the new form. Some form types such as **ObjectFormBase** or **MasterDetailForm** provide basic blank forms that are ideal for highly customized forms. Other form templates such as Issue or Task allow you to easily make more incremental changes to completed forms. Choose the form type which is most similar to the form you wish to create and you will save significant time and effort.



Change the format of date fields if you don't want the text display of the day and month included in the date.

By default, date fields display with a long format which includes the text display of the day, e.g., Friday, October 3, 2008. If you want to omit the text display set the display format to **Short**. You can also set custom formats.



Forms scale. So, don't forget to set the anchoring for your controls.

When you are editing a form, the design time representation of the form is almost always a different dimension than the runtime version. Based on the user's screen resolution or preference for tabbed or non-tabbed MDI display options, the form will likely be significantly larger or smaller than the representation of the form in the editor. To allow the form to grow and shrink, but also maintain a consistent and pleasing layout, the IssueNet forms editor allows you to set anchoring options for the form that determine how the controls will move and change dimension as a form changes dimension.

There are many different ways to set anchoring options based on the overall layout of the form. The best way to understand the option is to open an existing form and review the anchoring properties of controls on forms that have sizes and locations similar to controls on your own forms. Pay particular attention to forms that have multiple rows of controls side by side. If the size, location, or visibility of a control does not look like it does in the designer, anchoring is the likely cause.

Design	
(Name)	dateTimePicker1
GenerateMember	True
Locked	False
Modifiers	Private
Layout	
Anchor	Top, Left
AutoScroll	False
AutoScrollMargin	0, 0
AutoScrollMinSize	0, 0
AutoSize	False
AutoSizeMode	GrowOnly
Dock	None
Location	124, 66

Bind labels to a class property to have the label use the property display name.

When using a label you can always simply set the **Text** property of the control to set the text the label will display on the form. However, you can also configure a label to display the display name of a property. By setting labels to use property display names you can simplify form maintenance. If you change the display name of a class property or set a class property override or change a display name at a sub-class level, the text displayed by the labels bound to that property will automatically change. Also notice that by setting the Member Binding to "Value" you can use a label to display the value of a property as opposed to the display name.

TabIndex	0
Visible	True
Data	
MemberBinding	DisplayName
MemberName	Status
Tag	
ValuePrefix	
ValueSuffix	:
Design	
(Name)	StatusLabel
Locked	False
Modifiers	Public
Layout	

You cannot delete inherited controls, but you can hide them.

If you edit a form type which already has controls, you will notice that some controls have an icon with a small blue arrow in the upper left corner. This arrow indicates that the control is inherited as part of the compiled form template. Consequently, you cannot delete these controls from the form. However, you can hide them. To hide a control set the Visibility property to False. So, that it does not interfere with the layout you may also want to reduce the size of the control and move it behind another field to remove it from view.

Text	Status:
TextAlign	MiddleRight
UseMnemonic	True
Behavior	
AutoEllipsis	False
Enabled	True
TabIndex	0
Visible	True
Data	
MemberBinding	DisplayName
MemberName	Status
Tag	
ValuePrefix	

Use the button control and the Actions Open command with ObjectID fields to provide quick access to items in lists.

On all of the standard forms there is a small icon next to any field that contains a list of objects. When you click on the button the form for the item in the list opens. This feature is perfect when you need to view the details of an item referenced on the form you are viewing. For example, the button makes it simple to view and edit the details of a customer specified on a support ticket form.

Submitted By:	Deal, Tanya	▼	
Primary Contact:	Scott, Angus	▼	

You can also use the command button with your own custom fields. Let's suppose you have created a custom class that represents hardware components and you have added an object ID property and field to the Customer Support Ticket class form to track a hardware component for each issue. When you add the combo box that will allow the user to select one of the hardware components you can also add a button control. Once you have added the button control to the form, set the MemberName to the property set for the combo box and set the CommandName to the text "ActionsOpen". Once you have set the MemberName and the CommandName the button will open the object selected in the combo box.

AutoSizeMode	GrowOnly
Dock	None
Location	365, 2
Margin	3, 3, 3, 3
MaximumSize	0, 0
MinimumSize	0, 0
Padding	0, 0, 0, 0
Size	21, 21
Misc	
CommandName	ActionsOpen

Consider using a masked edit control for fields that require data in particular format.

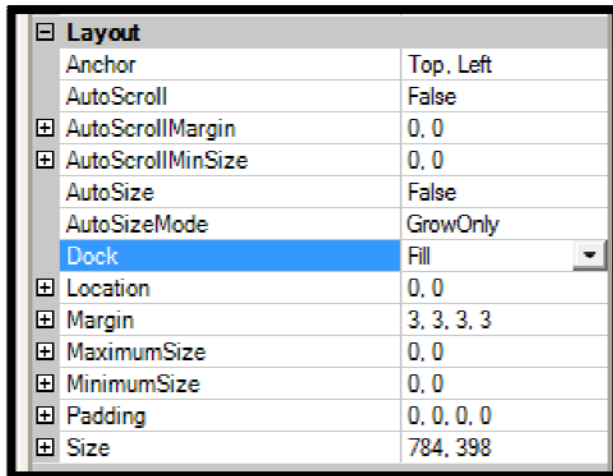
One of the most overlooked but useful controls is the masked edit box. The box allows you to specify control characters that require entered data to be in a particular format. In the screen capture below, the combination of the characters “#” and “.” are used to create an entry field for an I.P. address that requires the correct format. For more detailed information on the use of the masked edit control, refer to the following techtip: [Using the IssueNet Masked Edit Control Box](#)

AcceptsTab	False
AllowPrompt	False
CharacterCasing	Normal
ClipMode	IncludeLiterals
DataGroups	(Collection)
Enabled	True
HideSelection	True
InputMode	OvertypingOnly
Mask	###.###.###.###
MaxLength	32767
MaxValue	7922816251426433
MinValue	0
PasswordChar	
PositionAt	FirstPosition
ReadOnly	False

Use the Dock property when you want a control to fill a region on a form.

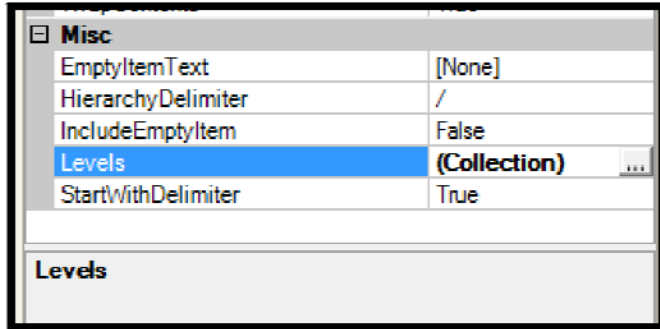
The Dock property will dock a control to a specific region of a container element, such as a tab, on the form. One of the most useful docking options is to set the Dock to Fill. When you set this value for a

large text field or other multiline control, such as a related objects view, the control will fill the entire region. Using the dock property allows you to easily set a control to fill an entire region regardless of the size to the form. It is also important to remember that when a control is docked you cannot resize the docked sides of the control. So, if you select a control on a form, such as a large text field, and cannot resize it, check the Dock property of the control. It is likely that the control is docked to the edges of the container and cannot be resized.



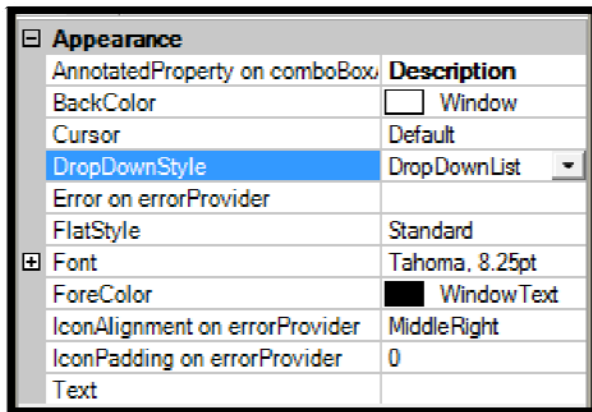
Use the Hierarchy Navigator for linked select lists.

If you need a set of dependent select lists, consider using the Hierarchy Navigator control. This control allows you to create dependent select lists with no scripting. For complete information on using the Hierarchy navigator with examples refer to the following tech tip: [Using the Hierarchy Navigator Control](#). Something to keep in mind when using the Hierarchy Navigator is that if you are using it to replace an existing control, delete the other control or set the member binding to nothing. If the Hierarchy Navigator is bound to the same property as another control the value will reset and exhibit odd behavior when inserting a new record. For example, Elsinore customers have noticed this behavior when replacing the folder field on the Issue form with the Navigator control.



Edit the DropDownStyle property of combo box controls to allow users to type as well as select a value in a combobox.

If you want to have combo box controls which allow users to type in a value as well as select a value from a list, simply toggle the DropDownStyle property of the box from DropDownList to DropDown.



For more information on topics not covered by this Tech Tip please review video tutorials and TechTips at www.elsitech.com. If you have questions beyond the scope of this Tech Tip, you may also contact Elsinore Technical Support Services at support@elsitech.com or 866.866.0034, option 2.