# UXMultipleSelectionComboBox Class

## Remarks

**UXMultipleSelectionComboBox** is a queryable combo box control derived from the UXDataComboBox which is specially designed to support multiple selection. All features in UXPageableComboBox, such as multiple column, sorting, and paging feature, are also supported in this control. To learn more about these features, see UXDataComboBox Overview and UXPageableComboBox Overview.

## Example

The following code shows how to use **UXMultipleSelectionComboBox** control in single column.

**XAML**

```
<Intersoft:UXMultipleSelectionComboBox SearchResult="{Binding Customers}"
        FilterDescriptors="{Binding LookUpDescriptor.FilterDescriptors, Mode=TwoWay}"
        SortDescriptors="{Binding LookUpDescriptor.SortDescriptors, Mode=TwoWay}"
        PageDescriptor="{Binding LookUpDescriptor.PageDescriptor}"
        DisplayMemberPath="ContactName">
        <Intersoft:UXMultipleSelectionComboBox.DataContext>
                <ViewModels:CustomerViewModel/>
        </Intersoft:UXMultipleSelectionComboBox.DataContext>
</Intersoft:UXMultipleSelectionComboBox>
```

Notice that if you use the paging feature, you need to do special handling in your **ViewModel**. See UXPageableComboBox to learn how to handle this behavior.
To enable multiple columns feature, simply add the following code to your **UXMultipleSelectionComboBox** control markup.

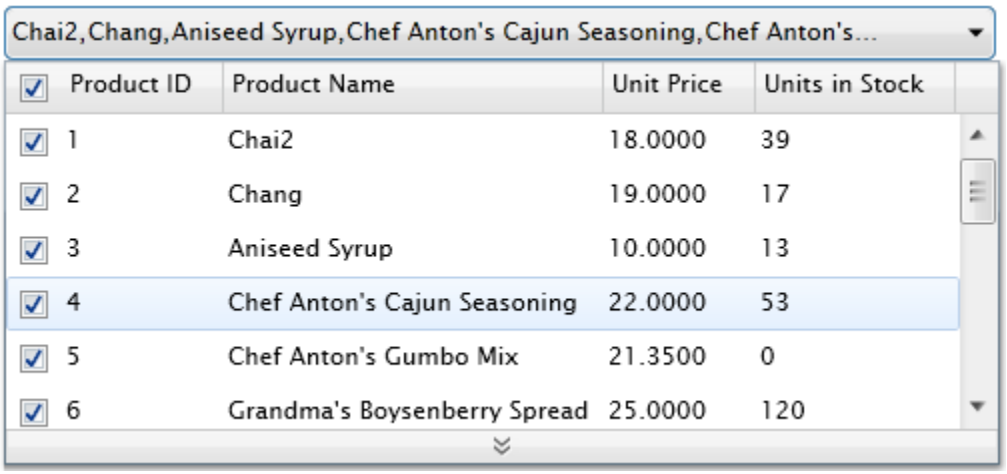
**XAML**

```
<Intersoft:UXMultipleSelectionComboBox.Columns>
        <Intersoft:UXDataComboBoxTextColumn Header="Customer ID"
    Binding="{Binding CustomerID}"
                DisplayMode="Image" ImageHeight="64" ImageWidth="64"
    ImageStretch="Fill" ImageBinding="{Binding PhotoPath}"/>
        <Intersoft:UXDataComboBoxTextColumn Header="Contact Name"
    Binding="{Binding ContactName}"/>
        <Intersoft:UXDataComboBoxTextColumn Header="Company Name"
    Binding="{Binding CompanyName}"/>
        <Intersoft:UXDataComboBoxTextColumn Header="Country"
    Binding="{Binding Country}"/>
</Intersoft:UXMultipleSelectionComboBox.Columns>
```

To learn more about Columns in **UXMultipleSelectionComboBox**, see UXPageableComboBox Columns Overview.

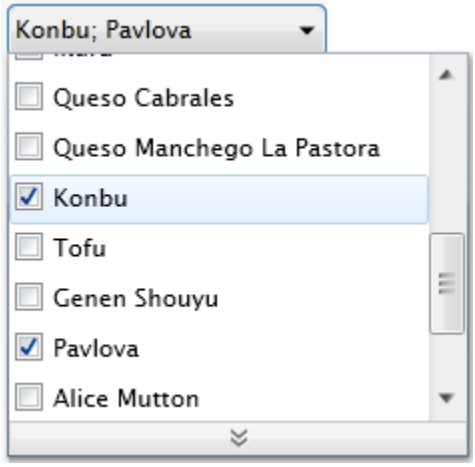
### Working with CheckBoxHeaderVisibility Property

Often times, users need to check-all or uncheck-all the records. Of course, it will be a waste of time if users have to check or uncheck each record in the list. To address this scenario, UXMultipleSelectionComboBox already includes a built-in check-all or uncheck-all feature. To enable this feature, simply set the **CheckBoxHeaderVisibility** property to **true**. The control will automatically generate a checkbox in the column header area. Note that this time-saving feature can only be used in non-editable mode with multiple columns enabled.




## Working with InputSeparator Property

By default, selected items displayed on the non-editable combo box control is splitted with a comma character. If you want to change this behavior, you must set the **InputSeparator** property with a string or the character you desired.




The following code change the separator value into a semicolon character which followed by a space character. It will produce results like the following image:

**XAML**

```
<Intersoft:UXMultipleSelectionComboBox SearchResult="{Binding Customers}"
        FilterDescriptors="{Binding LookUpDescriptor.FilterDescriptors, Mode=TwoWay}"
        SortDescriptors="{Binding LookUpDescriptor.SortDescriptors, Mode=TwoWay}"
        PageDescriptor="{Binding LookUpDescriptor.PageDescriptor}"
        DisplayMemberPath="ContactName" InputSeparator="; ">
        <Intersoft:UXMultipleSelectionComboBox.DataContext>
                <ViewModels:CustomerViewModel/>
        </Intersoft:UXMultipleSelectionComboBox.DataContext>
</Intersoft:UXMultipleSelectionComboBox>
```

## Definition

public class UXMultipleSelectionComboBox : UXDataComboBox, IMultipleSelection

## Summary

The following table summarizes the members exposed in this class.

### Public Constructors

| | |
|---|---|
| UXMultipleSelectionComboBox Constructor() | Initializes a new instance of the UXMultipleSelectionComboBox class. |

### Public Properties

| | |
|---|---|
| AutoFilterSelectedItems | Gets or sets a value indicating whether the selected items will be filtered automatically. |
| CheckBoxHeaderVisibility | Gets or sets the checkbox header visibility. |
| CheckBoxStyle | Gets or sets the style applied to the checkbox element. |
| InputSeparator | Gets or sets the input separator. |
| IsTextSearchEnabled | Not applicable. |
| KeyMemberPath | Gets or sets a value that represents the properties path used to filter the selected items when the UXMultipleSelectionComboBox is enabled. |
| RemoveItemCommand | Gets or sets command that will be invoked when removing an item. |
| SelectedItemConverter | |
| SelectedItems | Gets or sets the selected items |
| SelectedItemsCollection | Gets selected items collection. |
| SelectionInputControlStyle | Gets or sets the selection input control style. |

### Protected Properties

| | |
|---|---|
| ChangeSelection | Internal usage. |
| Product | Product Info. |
| SuspendCollectionChanged | Internal usage. |

### Fields

| | |
|---|---|
| AutoFilterSelectedItemsProperty | Identifies the AutoFilterSelectedItems dependency property. |
| CheckBoxHeaderVisibilityProperty | Identifies the CheckBoxHeaderVisibility dependency property. |
| CheckBoxStyleProperty | Identifies the CheckBoxHeaderVisibility dependency property. |
| InputSeparatorProperty | Identifies the InputSeparator dependency property. |
| KeyMemberPathProperty | Identifies the KeyMemberPath dependency property. |
| RemoveItemCommandProperty | Identifies the RemoveItemCommand dependency property. |
| SelectedItemsProperty | Identifies the SelectedItems dependency property. |
| SelectionInputControlStyleProperty | Identifies the SelectionInputControlStyle dependency property. |

## Public Methods

| | |
|---|---|
| AttachEventHandlers() | Attach built-in event handlers to control templates. Call this method if necessary. |
| DetachedEventHandlers() | Detach built-in event handlers from control templates. Call this method if necessary. |
| InitializeTemplates() | Initializes control templates. |
| IsMultipleSelection() | Gets a value that indicates whether the current action is multiple selection. |
| OnApplyTemplate() | Builds the visual tree for the UXMultipleSelectionComboBox when a new template is applied. |
| Select(int, bool) | Select the item on specified index. |
| Select(int) | Select the item on specified index. |
| Select(object, bool) | Select the specified item. |
| Select(object) | Select the specified item. |
| SelectAll() | Select all items. |
| Unselect(int, bool) | Unselects item at specified index. |
| Unselect(int) | Unselect the item at specified index. |
| Unselect(object, bool) | Unselect the specified item. |
| Unselect(object) | Unselect the specified item. |
| UnselectAll() | Unselect all items. |

## Protected Methods

| | |
|---|---|
| ChangeVisualState(bool) | Change visual state. |
| ClearSelection() | Clear current selection. |
| ClearSelection(object) | Clear current selection. |
| GetContainerForItemOverride() | Creates or identifies the element that is used to display the given item. |
| InitializeSelectedItems() | Initialize selected items collection. |
| InitializeSelection() | Initialize selection. |
| InitializeSelection(DependencyObject, object) | Initialize selection. |
| IsItemItsOwnContainerOverride(object) | Determines if the specified item is (or is eligible to be) its own container. |
| Menu_Opened(object, ISRoutedEventArgs) | Called when the drop down menu is opened. |
| Menu_Repositioned(object, UXPopUpEventArgs) | Called when the drop down menu is repositioned. |
| OnGotFocus(RoutedEventArgs) | Called before element GotFocus event occurs. |
| OnItemsChanged(NotifyCollectionChangedEventArgs) | Called when the value of the ItemsControl.Items property changes. |
| OnItemsSourceChanged(IEnumerable, IEnumerable) | Called when the ItemsSource changes. |
| OnLostFocus(RoutedEventArgs) | Called before element LostFocus event occurs. |
| OnMouseLeftButtonDown(MouseButtonEventArgs) | Called when MouseLeftButtonDown event occurs. |
| OnSelectedIndexChanged(int, int) | Called when the value of the SelectedIndex property changes. |
| OnSelectedItemChanged(object, object) | Called when the value of the SelectedItem property changes. |
| OnSelectedValueChanged(object, object) | Called when the value of the SelectedValue property changes. |

## Public Constructors

public UXMultipleSelectionComboBox()

Initializes a new instance of the UXMultipleSelectionComboBox class.

## Public Properties

public bool AutoFilterSelectedItems { get; set; }

Gets or sets a value indicating whether the selected items will be filtered automatically.
Remarks

> In editable mode, UXMultipleSelectionComboBox includes smart result filtering capability. When this feature is enabled, the selected items will be not be included in the search result list. You can enable this feature by setting the AutoFilterSelectedItems property to true.

public Visibility CheckBoxHeaderVisibility { get; set; }

Gets or sets the checkbox header visibility.

public Style CheckBoxStyle { get; set; }

Gets or sets the style applied to the checkbox element.

public string InputSeparator { get; set; }

Gets or sets the input separator.

public bool IsTextSearchEnabled { get; set; }

Not applicable.

public string KeyMemberPath { get; set; }

Gets or sets a value that represents the properties path used to filter the selected items when the UXMultipleSelectionComboBox is enabled.

public ICommand RemoveItemCommand { get; set; }

Gets or sets command that will be invoked when removing an item.

public IValueConverter SelectedItemConverter { get; set; }

Remarks

> The collection of data that has been selected will be assigned to the ISMultipleSelectionControl property. To get the selected items collection or initialize the selected items, you can easily bind the property from your ViewModel to the ISMultipleSelectionControl property of UXMultipleSelectionComboBox. For more details on how to initialize selected items, see How to Initialize Selected Items in UXMultipleSelectionComboBox. With SelectedItemConverter property, it allows you to bind a collection of items where the selected item type is different with the search result type. Note that you should set the SelectedValuePath property of UXMultipleSelectionComboBox for the selected items to work properly. This feature is particularly useful in such a scenario that you want to display the records from a table but you must bind it from another table that logically linked to a foreign key. In this case, you just need to assign the foreign key path to the ISSelectionControl property and UXMultipleSelectionComboBox will do the rest for you. You can see How to Implement Selected Items Converter in UXMultipleSelectionComboBox to learn more about this feature.

public INotifyCollectionChanged SelectedItems { get; set; }

Gets or sets the selected items
Remarks

> The collection of data that has been selected will be assigned to the ISMultipleSelectionControl property. To get the selected items collection or initialize the selected items, you can easily bind the property from your ViewModel to the ISMultipleSelectionControl property of UXMultipleSelectionComboBox. For more details on how to initialize selected items, see How to Initialize Selected Items in UXMultipleSelectionComboBox.

public IList SelectedItemsCollection { get; }

Gets selected items collection.

public Style SelectionInputControlStyle { get; set; }

Gets or sets the selection input control style.

## Protected Properties

protected bool ChangeSelection { get; set; }

Internal usage.

protected ProductInfo Product { get; }

Product Info.

protected bool SuspendCollectionChanged { get; set; }

Internal usage.

## Fields

public static readonly DependencyProperty AutoFilterSelectedItemsProperty

Identifies the AutoFilterSelectedItems dependency property.

public static readonly DependencyProperty CheckBoxHeaderVisibilityProperty

Identifies the CheckBoxHeaderVisibility dependency property.

public static readonly DependencyProperty CheckBoxStyleProperty

Identifies the CheckBoxHeaderVisibility dependency property.

public static readonly DependencyProperty InputSeparatorProperty

Identifies the InputSeparator dependency property.

public static readonly DependencyProperty KeyMemberPathProperty

Identifies the KeyMemberPath dependency property.

public static readonly DependencyProperty RemoveItemCommandProperty

Identifies the RemoveItemCommand dependency property.

public static readonly DependencyProperty SelectedItemsProperty

Identifies the SelectedItems dependency property.

public static readonly DependencyProperty SelectionInputControlStyleProperty

Identifies the SelectionInputControlStyle dependency property.

## Public Methods

public void AttachEventHandlers()

Attach built-in event handlers to control templates. Call this method if necessary.

public void DetachedEventHandlers()

Detach built-in event handlers from control templates. Call this method if necessary.

public void InitializeTemplates()

Initializes control templates.

public bool IsMultipleSelection()

Gets a value that indicates whether the current action is multiple selection.
Return Types

| |
|---|
| Value that indicates whether the current action is multiple selection. True if multiple selection. |

public void OnApplyTemplate()

Builds the visual tree for the UXMultipleSelectionComboBox when a new template is applied.

public void Select(int index, bool multipleSelection)

Select the item on specified index.
Parameters

| | |
|---|---|
| index | Index. |
| multipleSelection | Is multiple selection. |

public void Select(int index)

Select the item on specified index.
Parameters

| | |
|---|---|
| index | Index. |

public void Select(object item, bool multipleSelection)

Select the specified item.
Parameters

| | |
|---|---|
| item | Item. |
| multipleSelection | Is multiple selection. |

public void Select(object item)

Select the specified item.
Parameters

| | |
|---|---|
| item | Item. |

public void SelectAll()

Select all items.

public void Unselect(int index, bool multipleSelection)

Unselects item at specified index.
Parameters

| | |
|---|---|
| index | Index. |
| multipleSelection | Is multiple selection. |

public void Unselect(int index)

Unselect the item at specified index.
Parameters

| | |
|---|---|
| index | Index. |

public void Unselect(object item, bool multipleSelection)

Unselect the specified item.
Parameters

| | |
|---|---|
| item | Item. |
| multipleSelection | Is multiple selection. |

public void Unselect(object item)

Unselect the specified item.
Parameters

| | |
|---|---|
| item | Item. |

public void UnselectAll()

Unselect all items.

## Protected Methods

protected void ChangeVisualState(bool useTransitions)

Change visual state.
Parameters

| | |
|---|---|
| useTransitions | Use transition. |

protected void ClearSelection()

Clear current selection.

protected void ClearSelection(object except)

Clear current selection.
Parameters

| | |
|---|---|
| except | Except specified object. |

protected DependencyObject GetContainerForItemOverride()

Creates or identifies the element that is used to display the given item.
Return Types

| |
|---|
| The element that is used to display the given item. |

protected void InitializeSelectedItems()

Initialize selected items collection.

protected void InitializeSelection()

Initialize selection.

protected void InitializeSelection(DependencyObject element, object item)

Initialize selection.
Parameters

| | |
|---|---|
| element | The container element used to display the specified item. |
| item | The item to display. |

protected bool IsItemItsOwnContainerOverride(object item)

Determines if the specified item is (or is eligible to be) its own container.
Parameters

| | |
|---|---|
| item | The item to check. |

Return Types

| |
|---|
| true if the item is (or is eligible to be) its own container; otherwise, false. |

protected void Menu_Opened(object sender, ISRoutedEventArgs e)

Called when the drop down menu is opened.
Parameters

| | |
|---|---|
| sender | The source object. |
| e | The event argument. |

protected void Menu_Repositioned(object sender, UXPopUpEventArgs e)

Called when the drop down menu is repositioned.
Parameters

| | |
|---|---|
| sender | The source object. |
| e | The event argument. |

protected void OnGotFocus(RoutedEventArgs e)

Called before element GotFocus event occurs.
Parameters

| | |
|---|---|
| e | The data for the event. |

protected void OnItemsChanged(NotifyCollectionChangedEventArgs e)

Called when the value of the ItemsControl.Items property changes.
Parameters

| | |
|---|---|
| e | NotifyCollectionChangedEventArgs that contains the event data. |

protected void OnItemsSourceChanged(IEnumerable oldValue, IEnumerable newValue)

Called when the ItemsSource changes.
Parameters

| | |
|---|---|
| oldValue | |
| newValue | |

protected void OnLostFocus(RoutedEventArgs e)

Called before element LostFocus event occurs.
Parameters

| | |
|---|---|
| e | The data for the event. |

protected void OnMouseLeftButtonDown(MouseButtonEventArgs e)

Called when MouseLeftButtonDown event occurs.
Parameters

| | |
|---|---|
| e | The data for the event. |

protected void OnSelectedIndexChanged(int oldIndex, int newIndex)

Called when the value of the SelectedIndex property changes.
Parameters

| | |
|---|---|
| oldIndex | The old value of the SelectedIndex property. |
| newIndex | The new value of the SelectedIndex property. |

protected void OnSelectedItemChanged(object oldItem, object newItem)

Called when the value of the SelectedItem property changes.
Parameters

| | |
|---|---|
| oldItem | The old value of the SelectedItem property. |
| newItem | The new value of the SelectedItem property. |

protected void OnSelectedValueChanged(object oldValue, object newValue)

Called when the value of the SelectedValue property changes.
Parameters

| | |
|---|---|
| oldValue | The old value of the SelectedValue property. |
| newValue | The new value of the SelectedValue property. |

---

Namespace: Intersoft.Client.UI.Data

Assembly Information: Intersoft.Client.UI.Data, Version 3.0.5000.1

Target Frameworks: Silverlight 5, WPF 4