Combo

Creates, removes, or updates a combo box drop down.

Usage

```
Set_Property(OLECtrlEntID, "OLE.Combo[CtrlId]", Array)
```

Values

Array has the following structure:

Pos	Name	Type	Description	Default
<1>	Show Combo Drop Down	Boolean	Flag indicating whether or not combo box drop down exists	0
<2>	**Configuration**			
<2, 1>	Columns List	Multivalue	@TM delimited list of column headings	""
<2, 2>	Column Settings List	Multivalue	@TM delimited list of column settings	""
<2, 3>	Data	Multivalue	Combo box data	""
<2, 4>	Master Column	Integer	The column containing the return values	1
<2, 5>	Autofill	Boolean	If true, then autofill is enabled	1
<2, 6>	Case Sensitive Autofill	Boolean	If true, then autofill is case sensitive	0
<2, 7>	Number of Visible Rows	Integer	The number of viewable rows at a time	0
<2, 8>	Fire OnOptionClick	Boolean	If true, fire the OnOptionClick when the user presses the drop down button	0
<2, 9>	Filtered Autofill	Boolean	If true, reduce the list to only those items that partially match the current contents	1
<2,10>	Show Drop Down Immediately	Boolean	If true, shows the drop down as soon as the user enters the control	1
<2, 11>	LIST Format	Boolean	If true, then the Data is in LIST format instead of ARRAY format	0
<2,12>	Autofill Column(s)	Integer	The column(s) against which autofill is performed	Master Column
<2,13>	Hide When Empty	Boolean	If true, then the dropdown hides when the user clears the editline	0
<2,14>	Normal Navigation	Boolean	If true, then the up/down/pgup/pgdown keys move the editline cursor when the drop down is hidden	0
<2,15>	Show Only When Empty	Boolean	If true and Show Immediately is true, then the dropdown only shows immediately when the editline's contents are empty	0
<2,16>	No Show Autofill	Boolean	If true, then the drop down does not appear during autofill	0
<2,17>	Clear Selection on Backspace or Delete	Boolean	If true, then the currently selected drop down item will be deselected when the user presses backspace or delete	0
<2,18>	Show Popup While Read Only	Boolean	If true, then the user can click the drop down button and select an item but still cannot type in the cell	0
<2,19>	Show Popup While Navigating	Boolean	If true and Normal Navigation is false, then pressing up/down/pgup/pgdown shows the popup	0
<2,20>	Always Tab Out on Enter	Boolean	If true, then pressing Enter will tab out of the cell even if the drop down was visible	1
<2,21>	Dropdown Threshold	Integer	The number of rows, at minimum, that must be visible within the dropdown for it to appear below the cell	0
<2,22>	Dropdown Font	Font	The font used to render the drop down items	Tahoma, 8pt
<2,23>	Limit to List	Boolean	Determines if the editline's contents must be limited to the values in the dropdown.	0
<2,24>	Match Anywhere	Boolean	Determines if autofill will also match what you type to any substrings within text, instead of just at the beginning of it.	0
<2,25>	Row Height	Integer	Sets the height of each row in the drop down.	14

Indices

Index	Description
CtrlId	Identifies a subclassed control

Remarks

Applies To: EDITLINE, EDITBOX

The Combo property can be used to add an efficient and feature rich combo box drop down to an OI EDITLINE control. The user interacts with the drop down just as if it were an OI COMBOBOX control. Whenever you turn this property on, the OptionButton is set to 1 to show an option button. The option button is then used to show the drop down.

The **Columns List** parameter is an @TM delimited list of header captions. Leave this blank if you don't want column headers to appear at all.

The **Column Settings List** parameter lets you specify column alignment and width; it's an @TM delimited list of these values, and each element of the list has the structure Alignment:@STM:Width. The alignment sets the column's data justification and can be "Left", "Center", or "Right". You can optionally use "L", "C", and "R" respectively. The width can be used to limit the width of a column (if you expect it to contain long strings of data). If you want your columns to be autowidth, omit the Fixed Width. If you want your columns to dynamically resize during filtered autofill, set the width to "Dynamic" or "DYN".

The **Data** parameter is the most important as it establishes the drop down list data. It is just like the OI EDITTABLE ARRAY property except it uses @TM instead of @FM and @STM instead of @VM. There is currently no support for sorting, so you'll have to sort this array before you set it. (See the "LIST Format" parameter below for more information on storting Data).

The **Master Column** parameter lets you specify which column contains the values that will be placed in the control when the user selects it. This is also the column that the auto fill uses by default (see "**Autofill Column**" below). To use auto fill, set AutoFill to 1 (although this is 1 by default). Auto fill is case insensitive by default, so set the **Case Sensitive Autofill** to 1 if you need it.

If you suspect that the combo box will have many rows of data, you may use the **Number of Visible Rows** parameter to limit the height of the combo box dropdown list. By default, the list fits as many rows as possible without leaving the screen. This parameter however, will limit the maximum height and display a scroll bar for the user.

In some cases, you may want the combo box dropdown to appear during autofill, but you'd rather show a popup window if the user presses the dropdown button. For such cases, set the **Fire OnOptionClick** parameter to 1 which will fire the OnOptionClick event instead of showing the combo box dropdown list. Note that the user will still see the combo box if they press ALT+DOWN or if it appears when gaining focus (See the "Show Drop Down Immediately" parameter).

The ninth parameter, **Filtered Autofill**, simply enhances the autofill technology by completely hiding items that do not partially match the current autofill, thus making the list smaller and smaller as the user narrows down the search. This conveniently allows the user to more easily select the closest item, especially if your list is not sorted. When this is turned off, autofill still works, but the list never changes its size. If this is turned on and Autofill is off, then the list still filters down as the user types, but no auto selection or filling occurs.

By default, the drop down appears immediately when the user enters the control. If, however, you only want the dropdown to appear once the user starts typing or pressed ALT+DOWN, then set the **Show Immediately** parameter to 0. This way, the user only sees the drop down during autofill. If both the Autofill parameter and this one are set to false, then the user must press ALT+DOWN to make the drop down appear.

The **LIST Format** parameter is a flag allowing you to change how the data array is interpreted. Normally, the third parameter (Data) must be in ARRAY format (@TM delimited columns with @STM delimited rows). By setting this parameter to 1 however, you can pass the Data in LIST Format (@TM delimited rows and @STM delimited columns). This flags also affects the ComboData property. Thus, once you choose the LIST Format, all data interaction is in LIST format.

The **Autofill Column(s)** parameter allows you to specify which column(s) the autofill is performed against. If omitted, then the autofill column is the same as the Master Column. Set this to one or more columns if you want to allow the user to autofill against something other than the Master Column. This list can be comma delimited or @SVM delimited. The order of this list also determines the order that will columns will be searched during autofill.

The next parameter, **Hide When Empty**, determines the behavior of the dropdown when the user clears the editline. By default, once the dropdown is visible it remains so until the user leaves the editline or makes a selection. If you'd rather that the dropdown hides when the user clears the editline, then set this parameter to true. This only has an effect if autofill with filtering is enabled.

The **Normal Navigation** parameter determines how to respond to navigation keys. By default, the up, down, pgup, and pgdown keys make the drop down appear and allow you to navigate the drop down. If you set this parameter to true, then these keys will only navigate the drop down when it is already visible. Otherwise, they move the cursor in the editline or editbox.

The **Show Only When Empty** determines when the drop down appears and is only useful when the Show Immediately parameter is true. By default, the drop down will show immediately whenever the editline gets focus. If you set this parameter to true, then the drop down will show immediately only if the editline is empty. This is useful if you want to prompt users for new values.

The **No Show Autofill** determines whether or not the drop down appears during autofill. Normally, the drop down appears as the user starts to type so they can see the nearest match. If you set this parameter to true, then the drop down will not appear during autofill; the user will have to explicitely show the drop down if they want to see their nearest matches.

The **Clear Selection on Backspace or Delete** flag determines whether or not the currently selected drop down will be deselected when the user presses the backspace or delete key. This is useful when you need to allow for the possibility that an autofill item might match a master column value. This is usually only needed when the autofill column is different from the master column.

The **Show Popup While Read Only** flag, when set, allows the user to click the drop down button to show the combo drop down even though the editline is read only. The user can select an item in the list but will be unable to type within the editline to enter custom data. This is useful if you want to limit the user to the combo list.

The **Show Popup When Navigating** flag, when set to true, causes the combo drop down to appear when the user presses the up, down, pageup, or pagedown keys. This flag only works when Normal Navigation is off.

The **Always Tab Out on Enter** flag, when set to true, will tab the user to the next control when pressing Enter, even if the drop down was visible. When set to false, pressing Enter while the drop down is visible merely selects the item and keeps focus on the current control, a behavior that better mimicks standard combo box controls. The flag is on by default.

The **Dropdown Threshold** value allows you to customize the behavior of the dropdown as it approaches the bottom of the screen. The dropdown has the effect of being "squished" between the field and screen bottom until fewer rows than the threshold are visible, in which case the dropdown is moved above the field. By default, the threshold is 3 rows. You can increase or decrease this value as desired. A value of 0 means that the dropdown will not appear above the field until no rows are visible. A value of -1 causes the dropdown to appear above the field as soon as it touches the bottom. In other words, -1 disables the "squish" effect.

The **Dropdown Font** value determines the font used to render the items in the dropdown list. It does not affect the font used to render the actual editline control.

The **Limit to List** value determines if the user is prevented from typing anything that does not appear in the dropdown list.

The **Match Anywhere** value modifies autofill behavior slightly. Normally, as the user types, the typed text is matched against all autofill columns (See value <2, 12> above) to find any strings that begin with the same text. When this is set to 1, however, the typed text is allowed to be found anywhere within a string. When this is enabled, note that autofill will only help the user find matching rows but will not autofill the text in the editline unless the match is at the beginning of a string.

The **Row Height** value customizes the height of each row in the combo dropdown.

Example

```
// Subclass my editline control first
CtrlId = @Window:".EDITLINE"
Handle = Get_Property(CtrlId, "HANDLE")
rv = Send_Message(@Window:".OLE_SUBCLASS", "OLE.Subclass", Handle, CtrlId)

// Setup a combo drop down
FirstNames = "Don"  :@STM:"Paul"    :@STM:"Frank":@STM:"Bob"      :@STM:"Kevin"
LastNames  = "Bakke":@STM:"Simonsen":@STM:"Tomeo":@STM:"Fernandes":@STM:"Fournier"
ComboDropDown = ""
ComboDropDown<1>    = 1
ComboDropDown<2, 1> = "First Name":@TM:"Last Name"
ComboDropDown<2, 2> = "L":@STM:100:@TM:"L"
ComboDropDown<2, 3> = FirstNames:@TM:LastNames
ComboDropDown<2, 4> = 2                         ;// column 2 contains the values we care about
ComboDropDown<2, 5> = 1                         ;// auto fill on
ComboDropDown<2, 6> = 0                         ;// case sensitive off
ComboDropDown<2, 7> = 10                        ;// 10 visible rows max
ComboDropDown<2, 8> = 0                         ;// Don't fire the OnOptionClick
ComboDropDown<2, 9> = 1                         ;// Reduce the list to partial matches
ComboDropDown<2, 10> = 0                        ;// Only show the drop down when the user types
ComboDropDown<2, 11> = 0                        ;// Do not use LIST Format
ComboDropDown<2, 12> = 1                        ;// Autofill on first names
ComboDropDown<2, 13> = 1                        ;// Hide dropdown when user clears cell
ComboDropDown<2, 14> = 0                        ;// Let navigation keys show the drop down
ComboDropDown<2, 15> = 0                        ;// Show the drop down regardless of the contents
ComboDropDown<2, 16> = 0                        ;// Show the drop down during autofill
ComboDropDown<2, 17> = 1                        ;// Remove selection when user backspaces/deletes
ComboDropDown<2, 18> = 0                        ;// Show Popup while in read only mode
ComboDropDown<2, 19> = 1                        ;// Show Popup When Navigating
ComboDropDown<2, 20> = 1                        ;// Always Tab Out on Enter
ComboDropDown<2, 21> = -1                       ;// Always show the dropdown above when close to the screen
bottom

Convert "." to ";" in CtrlId
Set_Property(@Window:".OLE_SUBCLASS", "OLE.Combo[":CtrlId:"]", ComboDropDown)
```

See Also

ComboSelPos, ComboRowData, ComboData, ComboDropDown