Bookmark Property

Applies To

GTList control**, GTCombo** control

Description

This property returns or sets the current location in the database that the List is pointing to.

Usage

object.**Bookmark** [= *bookmark obj*]

The **Bookmark** property has these parts:

Part	Description
object	An object expression that evaluates to one of the controls in the Applies To list.
bookmark obj	A bookmark object specifying the location in the database that the ListIndex is at.

Remarks

This property is only valid when the control is bound to a datasource. This dictates the currently focused item in the list.

See Also

ListIndex property**, SelBookmarks** Property Array

Example

The following code uses the **Bookmark** property to access data in the current row:

```
Info = GTList1.ListItems(GTList1.Bookmark).Text
```

SelBookmarks Property Array

Applies To

GTList control

Description

This property returns or sets the current location in the database that the List is pointing to.

Usage

object.**SelBookmarks(***Index***)** [= *bookmark obj*]

The **SelBookmarks** property has these parts:

Part	Description
object	An object expression that evaluates to one of the controls in the Applies To list.
Index	This is an index that is 0 or greater and not larger than the **SelCount** property.
bookmark obj	A bookmark object specifying the location in the database that the ListIndex is at.

Remarks

This property is only valid when the control is bound to a datasource. This dictates the selected items in the list. When **MultiSelect** is set to *None* then this array only contains on item. When **MultiSelect** is set to *Simple* or *Extended* then this array has **SelCount** items in it.

See Also

Bookmark Property Array, **SelCount** Property, **MultiSelect** Property

Example

The following code uses the **SelBookmarks** property to access data in the selected rows.

```
For x = 0 to GTList1.SelCount - 1
    Info = GTList1.ListItems(GTList1.SelBookmarks(x)).Text
    '*********Use Info some how********
Next x
```

Virtual Mode

What is Virtual Mode?

GTList and GTMask can both be used in Virtual Mode. This mode is active when the

control is not bound to a datasource and the Virtual Property is set to True.

When in Virtual Mode, the control does not store the items that are being displayed internally. You have to maintain the data in some manor whether it is in a database, memory, file, or being generated on the fly. The control requests items when its needs them via the ListItemDataRequest Event.

The advantages of this mode arethat you are fully in control of the data, load time is increased since there is no data to be loaded, and run-time performance is based on the efficency of you providing the data.

Using Virtual Mode

Turning Virtual Mode On: Place a GTList (GTCombo can also be used for this entire section) on the Form. Set the Virtual Property to True. Set the VirtualItemCount property to the number of items to be displayed in the list. This number may be anything up to 2 Billion. We will use 100.

Setting Up The Columns: For the control to create the ListItem object that is passed to the ListItemDataRequest Event, you must define one ColumnDef object in the ColumnDefs collection for every column to be displayed in the list. To do this, right click on GTList. Then click on DataList Designer.

In the DataList Designer, click the *Add Column* button three times. This will give us three columns in the list.

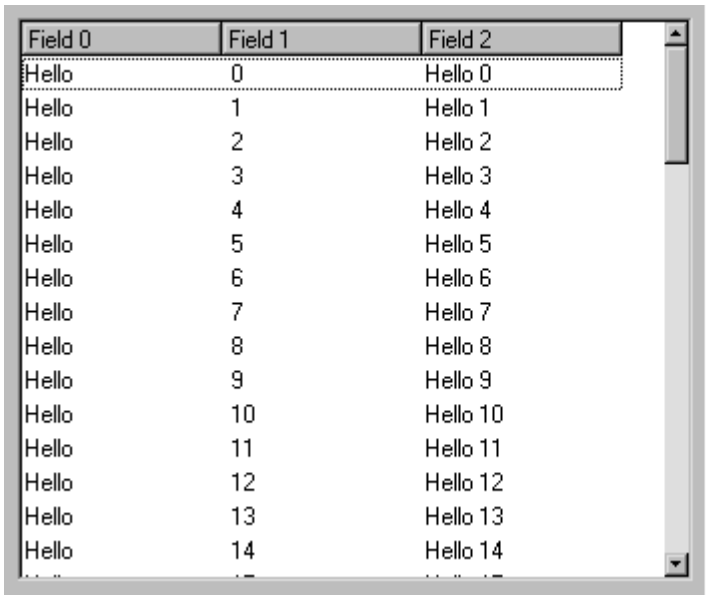
Exit DataList Designer.

Using ListItemDataRequest: Event: Place the following code in the ListItemDataRequest Event:

```
ListItem.SubItems(0).Text = "Hello"
ListItem.SubItems(1).Text = Str$(Index)
ListItem.SubItems(2).Text = ListItem.SubItems(0).Text + ListItem.SubItems(1).Text
```

This code will place the string “Hello” into the first column of every row, the row number into the second column of every row, and a concatinated string consosting of the string “Hello” and the rw number. For example, the three columns of row 5 will contain “Hello”, “5”, and “Hello5” respectively.

Running the Sample: Run the sample. The resulting output from this program should look something like this:



Field 0	Field 1	Field 2
Hello	0	Hello 0
Hello	1	Hello 1
Hello	2	Hello 2
Hello	3	Hello 3
Hello	4	Hello 4
Hello	5	Hello 5
Hello	6	Hello 6
Hello	7	Hello 7
Hello	8	Hello 8
Hello	9	Hello 9
Hello	10	Hello 10
Hello	11	Hello 11
Hello	12	Hello 12
Hello	13	Hello 13
Hello	14	Hello 14