RemAll - 2014 . Last vers.1.0.3 by efsoft

As promised - Part 3

So far we have examined the management of items created by code that is essential to judge them valid for the use that we make of them.
Now becomes primary how to load data into the lists in consideration that are repeatedly created and deleted as needed.
The read data in the database are organized by interposing a separator between them in a single string that is run by a few lines of code driven by a for/next or Do./Loop.
The code , which also found already shown in my previous posts, is this, assuming that the string is as follows.

```
Dato = "Field1 ! Field2 ! Field3 ! Field4 ! - ! - ! .......! - ! "            '( ! separator)

Dim Lst As Label
Lst.Initialize ( "Label" )
Activity.AddView ( Lst , 5dip ............. )
For k = 0 To 9
Tx = Dato.IndexOf("!")
LG.Label = Dato.SubString2(0,Tx)
NDato = Dato.SubString2(Tx+1, Dato.Length) '(remaining string list)
Dato = NDato
Next
```

Please note that the Dato string is removed by writing the individual labels of a list. So it can not be reused without re-creating it ora using anytime a clone **(myDato=Dato)**

Going forward we can see how, using a different element (EditText) also created by code, you can simulate the direct writing of data in a dynamic list .
Despite being a dynamic element it is commonly used as an input box for the use of the Search button . However, as you can create dynamic EditText in any position you want.
And it is this ability that allowing the overlap to each element of a list by the user impresione to write to the list by filling out the contents of the record in full page instead of using some kind of command line.
The code to do this is like this and is activated using the event LongClick of the label that is commonly used only with the Click event :

```
sub Label_LongClick

CreaEdit

Dim Lst As Label
Lst.Initialize ("Label")
Lst = Sender
KT = Lst.Top
KL = Lst.Left
EditText.Top = KT
EditText.Left = KL
EditText.Visible = True
EditText.BringToFront
qq = Activity.NumberOfViews
IME.ShowKeyboard (EditText)
```

Note that each time a EditText is created in the desired position, the program read the index that indicates the number of the Views .
As soon as you complete the last label a specific key to confirm the data will be shown.

Oops i forgot to clarify that to put the EditText on the correct position the code under the sub Crealist, when you call it, allows to select the position that you need.

```
Sub CreaEdit
Dim E As EditText
E.Initialize("EditText")
E.Visible = False
....................
...................
E.ForceDoneButton = True
Select Case True
Case PosEd = 1
Activity.AddView(E, Albl1.Left, Albl1.Top + 30Dip ,LG.Width ,LG.Height)
Case PosEd = 0
Activity.AddView(E, Albl1.Left, Albl1.Top ,Label4.Width ,LG.Height)
Case PosEd = 4
Activity.AddView(E, Albl1.Left, Albl1.Top ,LG.Width ,LG.Height)
End Select
.........................
```

Note that E is initialized as visible= false because will be the editing command Long Click to make visible it each time.
The index PosEd instead will be applied by the label click so adjusting the measures of the EditText to the Label.

The confirmation will create a string of contents divided by separators which as said above, after deleting the EditText and the label will re-create the list loaded with the new data, giving the

impression and confirmation of the boot record.
Take note that the procedure also applies to the modification of a single record, and allows you to have the correct data instantly as soon as you press Enter / Done on the keyboard
The operation also uses colors to highlight the action in progress.
Each label becomes white when waiting to be written and this is up at the end of compilation of all the labels.
When you confirm the list back to the blue background and the text to the current size. (using EditTest text is a few points less).
Too bad when you have to confirm each inserimeto and is not allowed to correct what has already been inserted.
Will this be the new frontier to reach, but I do not think it's easy.
at the next

efsoft, Dec 19, 2013

As promised - Part 4

We have reached the end of this long message posted only to report what could be done using a variety of dynamic elements (i.e. created via code instead of the designer)
We have mentioned how to create them, some tricks to handle them, and the different opportunities provided by the use of some of them together.
There is one an ability of which a such list is missing: to substitute (for my use) a List View standard and to be able to scroll the pages.
It is clear that if an archive contains more elements than the list of labels used by the program i have to add to the list the ability to scroll.

To achieve this we use again the string with elements and separators; in this case the elements are the titles (1° field) of the Records.
The sample is below
Arc = Record1, Record2, Record3, Record4 .... Record10, Record11, Record12, etc ...

The number of records found in the 'selected archive (dd) you can easily derive when you create the above string.
From this value will be derived the number of pages (of 10 labels) that you can get from the string and the remainder (less than 10) will be completed to that value by adding to the string enough empty elements **(-, -, -)**
The code that executes this step is the following

```
Do Until dd <= 0
rs = dd - 10
PagId = PagId + 1
dd = rs
Loop
PagId = Round(PagId)
For    r = 10 To (10 - rs) : FakeStg = FakeStg & "-," : Next
```

When the pages required have been created you have only to load the content of the string into the labels

```
For Pg = 1 To PagId                         '(To create the required pages)
h = 0
Do Until h = 10
Tx = Arc.IndexOf(",")
Item = Arc.SubString2(0, Tx)
Pag(Pg) = Pag(Pg) & Item & ","
Nlist = Arc.SubString2(Tx+1, Arc.Length)
Arc = Nlist
h = h + 1
Loop
Next
```

Then to show in the list, when the above code is applied, send the first Page of the record found

```
Pg = 1
Dato = Pag(Pg) : CreaLio : CreaPan
```

The last line of the above code reveals the last dynamic object applied to RemAll program.
I use this panel as a sort of bar to exchange the various pages created.
This Item is always present at the right side of the Lio list but will be enabled only when an archive with a lot of Records will start the paging code shown above.
Actually i use the panel to control the change of the pages by the touch of the top or the bottom of the panel.
The same result can be obtained with two buttons but i hope to be able in the future to slide the records in the list instead of to scroll the data pages.
The following is the used code for the moment

```
Sub P2_Touch (Action As Int, 0 As Float, V As Float) As Boolean
If Action = Activity.ACTION_DOWN Then
KV = Round(KV)
SelPage
End If
End Sub

Sub SelPage
If KV < P2.Height/2 Then
Pg = Pg - 1 : If Pg < 1 Then Pg = 1
Dato = Pag(Pg)
CreaLio : LN = LN+1
Else
Pg = Pg + 1 : If Pg > PagId Then Pg = PagId
Dato = Pag(Pg)
CreaLio : LN = LN+1
End If
Return True
End Sub
```

I think that there is no other to say except that the code partially shown may be difficult to understand but i hope,that this be usefull to somebody.