

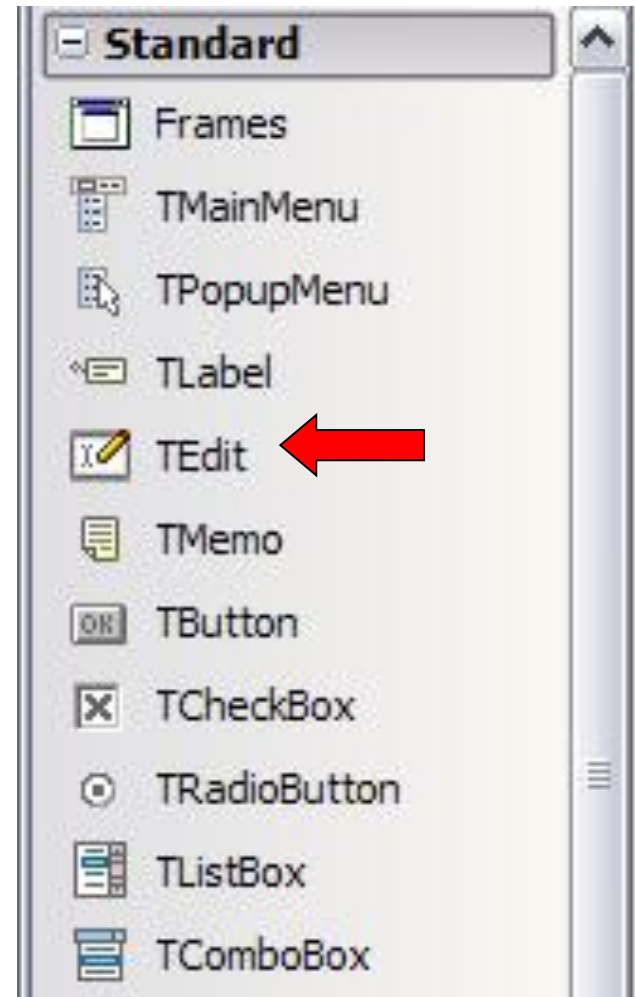
Entrada de Datos

- **Edit, LabelEdit** y EditMask.
- Listas de strings - Clase TString y TStringList.
 - **Memo**, RichEdit
 - **ComboBox.**
- Selección de opciones
 - **ListBox, CheckBox** y CheckListBox.
 - **RadioButton – RadioGroup.**
 - ColorBox
- Ingreso mediante barras (scrollBar, TrackBar).
- Ingreso de Fechas: DateTimePicker, MonthCalendar
- Paleta Dialogs (OpenDialog, SaveDialog, ColorDialog, PrintDialog)

Estas son las descritas en este documento

Componente Edit

- Permite ingresar una línea de texto (Propiedad **text**).
- **Otras propiedades**
 - MaxLength
 - PasswordChar
 - CharCase



Componente Edit

- **Propiedad MaxLength**

- Permite indicar la **cantidad máxima de caracteres** que contendrá la propiedad **text**.
- Note que no tiene nada que ver con la cantidad de caracteres que se ven dentro del casillero.
- Si vale cero indica que no tiene límite.

- **Propiedad PasswordChar**

- El carácter indicado en esta propiedad será utilizado para reemplazar la entrada de datos



Componente Edit

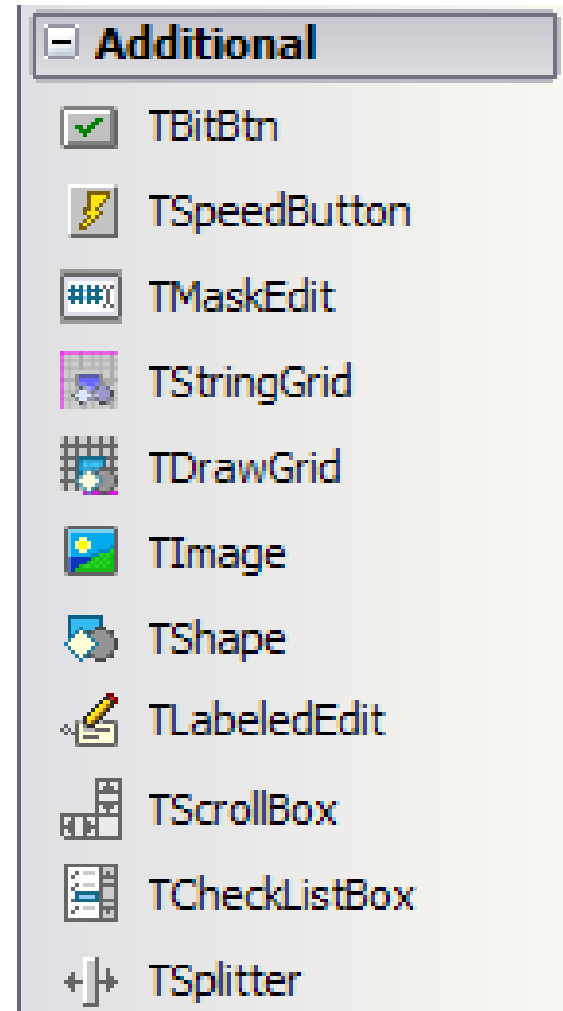
- **Propiedad CharCase**
 - Convierte la entrada de datos a mayúsculas o minúsculas.

Constante	Conversión realizada
ecNormal	No convierte
ecLowerCase	Todas las mayúsculas a minúsculas
ecUpperCase	Todas las minúsculas a mayúsculas

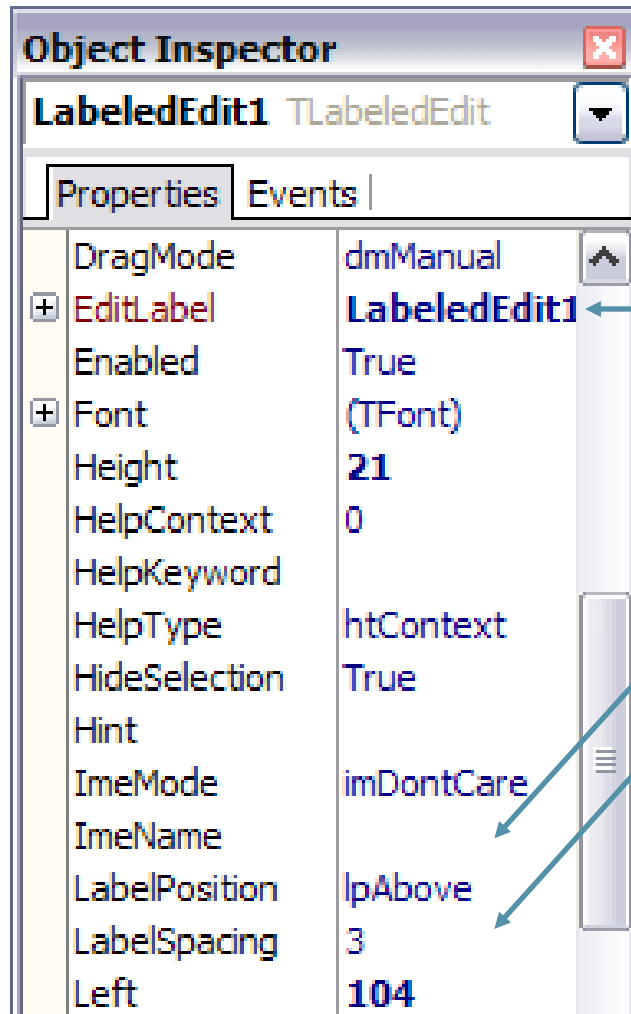
Valor por defecto

Componente LabeledEdit

- Es la unión de los componentes **Label** y **Edit**



Componente LabeledEdit



- **Propiedades**

- EditLabel (acá van los datos del Label)
- LabelPosition
- LabelSpacing

Listas de Strings

- Aparecen en varios componentes

Memo

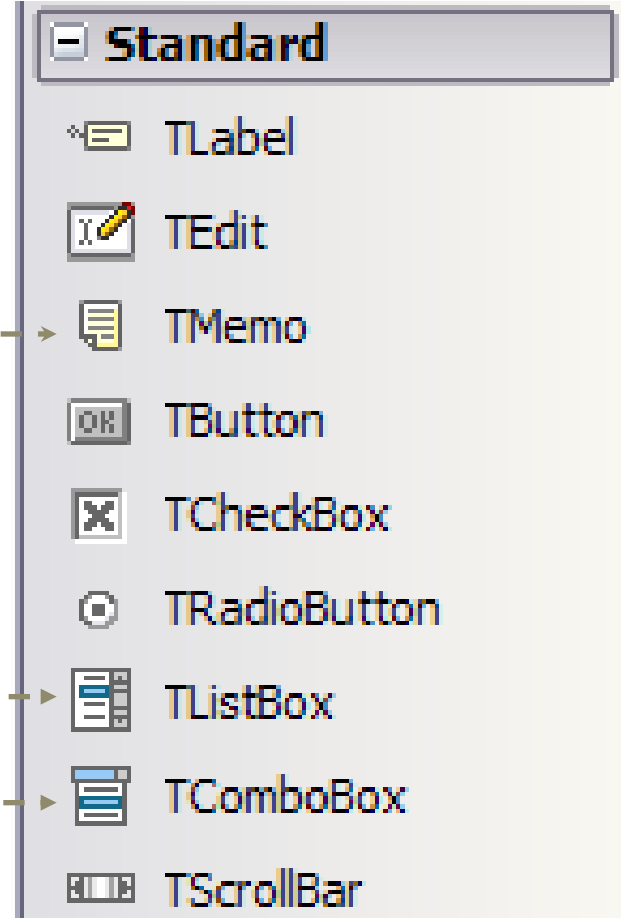
Permite ingresar varias líneas de texto.

ListBox

Permite visualizar una lista de strings y seleccionar uno o varios.

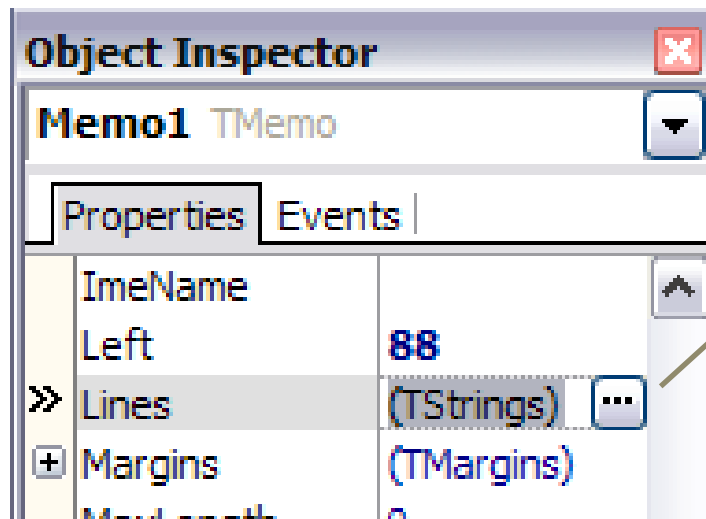
ComboBox

Permite seleccionar una opción de una lista de opciones o ingresarla desde teclado.



Listas de Strings

- La clase **TStrings** provee operaciones tales como:
 - Almacenar o borrar los strings en una lista.
 - Reordenar los strings de la lista.
 - Leer o guardar la lista en un archivo.



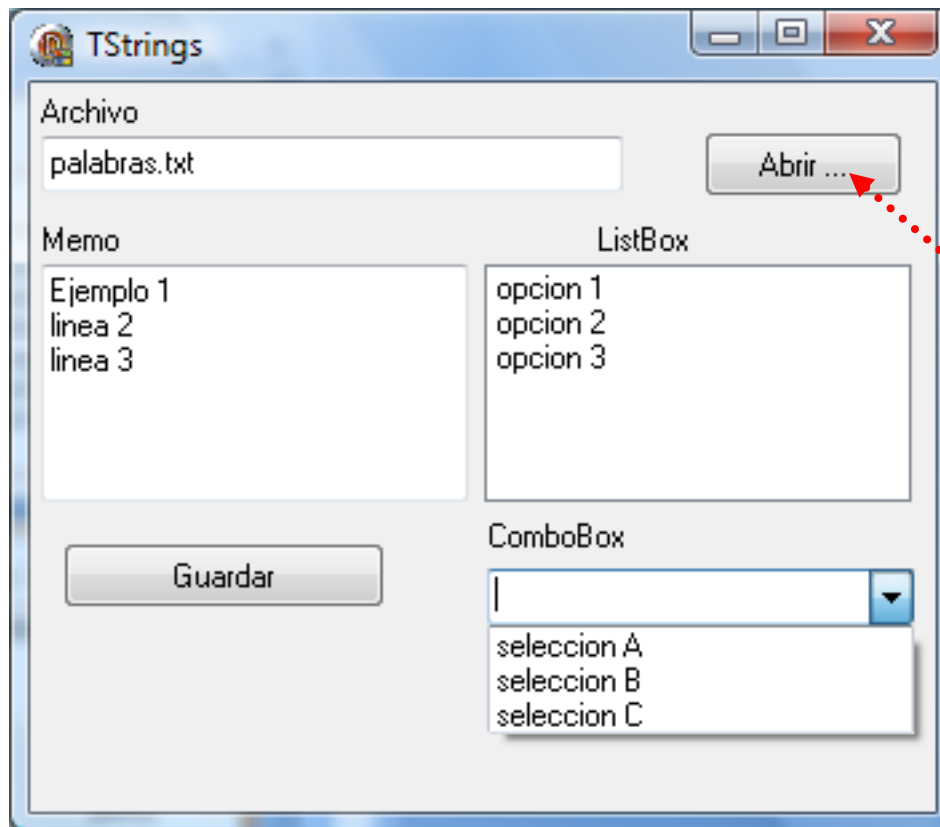
Las componentes que utilizan listas de strings permiten acceder al editor de líneas desde el Inspector de Objetos

Listas de Strings

- Vamos a ver como
 - Cargar y salvar listas de strings.
 - Gestión de los elementos de la lista :
 - Incorporación de elementos.
 - Acceso a los elementos de la lista.
 - Borrado de elementos.
 - Búsqueda de un elemento dentro de la lista.
 - Modificando la posición de un elemento dentro de la lista.

Cargando y Salvando listas de Strings

- Métodos : **LoadFromFile** y **SaveToFile**



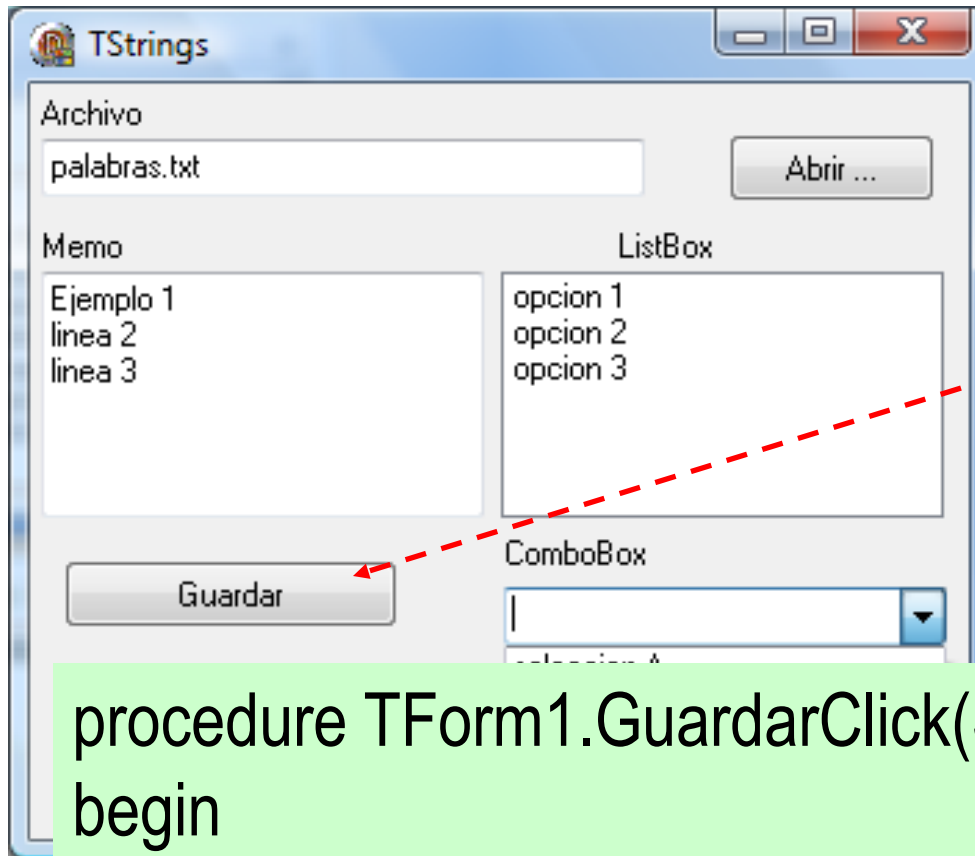
Cargar en el Memo,
en el ListBox y en el
ComboBox, el
contenido del
archivo

LoadFromFile.dpr

Cargando y Salvando listas de Strings

```
procedure TForm1.AbrirClick(Sender: TObject);  
Var  
    NomArch : String;  
begin  
    NomArch := Edit1.Text;  
    Memo1.Lines.LoadFromFile( NomArch );  
    ListBox1.Items.LoadFromFile(NomArch );  
    ComboBox1.Items.LoadFromFile(NomArch );  
end;
```

Cargando y Salvando listas de Strings



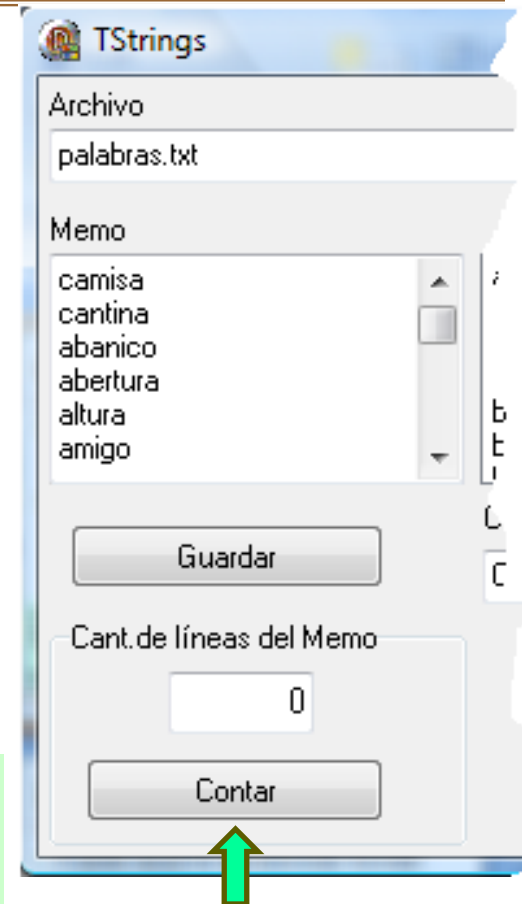
Guarda el
contenido del
Memo en el
archivo

```
procedure TForm1.GuardarClick(Sender: TObject);  
begin  
    Memo1.Lines.SaveToFile(Edit1.Text);  
end;
```

Cantidad de elementos de la lista

- La propiedad **Count** representa la cantidad de líneas de la lista.
- Ejemplo:
 - Memo1.Lines.**Count**
 - ListBox1.Items.**Count**

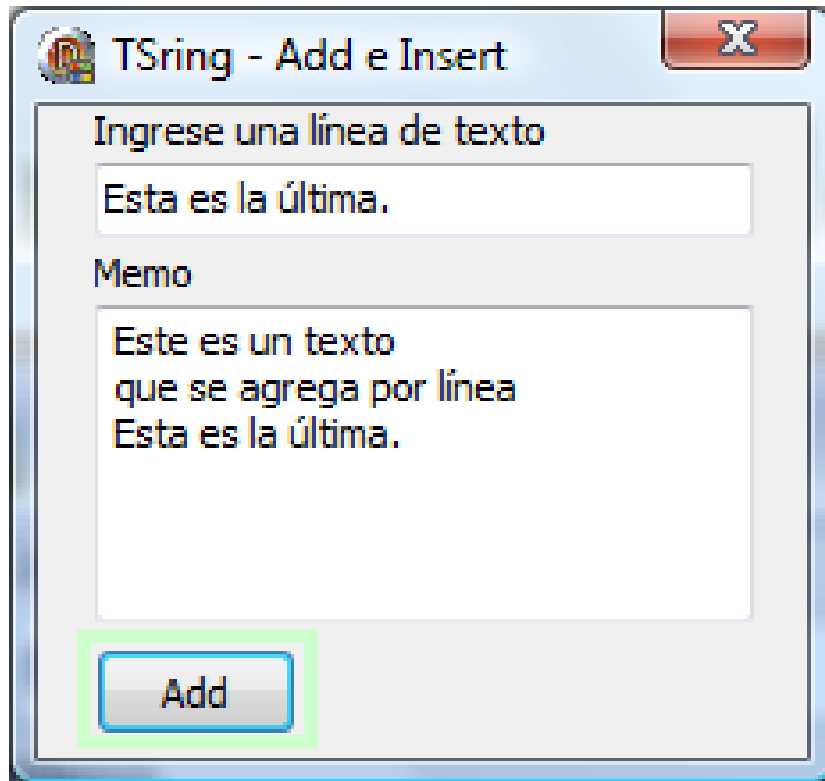
```
procedure TForm1.Button1Click(Sender: TObject);  
begin  
    Edit1.Text := IntToStr ( Memo1.Lines.count ) ;  
end;
```



Incorporando elementos a la lista

- Puede hacerse de distintas formas:
 - **Add**
 - agrega un elemento al final.
 - **Insert**
 - agrega un elemento en una posición determinada.
 - **AddStrings**
 - agrega una lista de strings en otra

Incorporando elementos a la lista



TSring - Add e Insert

Ingrese una línea de texto

Esta es la última.

Memo

Este es un texto
que se agrega por línea
Esta es la última.

Add

AddInsert.dpr

```
procedure TForm1.AgregarClickSender: TObject);  
begin  
    Memo1.Lines.Add(Edit1.Text);  
end;
```

Incorporando elementos a la lista

TSring - Add e Insert

Ingrese una línea de texto

Esto se insertará en 2do. lugar

Memo

Este es un texto
Esto se insertará en 2do. lugar
que se agrega por línea
Esta es la última.

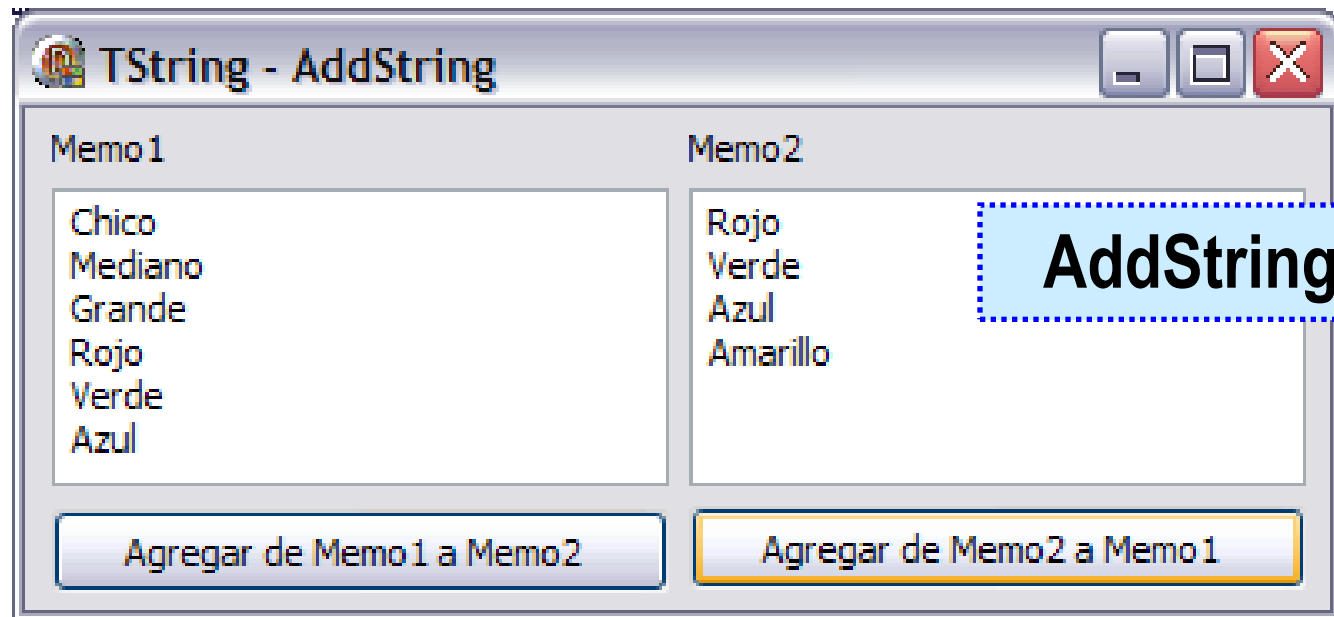
Add Insertar en 1

AddInsert.dpr

Número entre 0 y la cantidad de elementos de la lista - 1

```
procedure TForm1.InsertarClick(Sender: TObject);  
Var indice :Integer;  
begin  
    indice := StrToInt(EdPosi.text);  
    Memo1.Lines.Insert(indice, Edit1.text);  
end;
```


Incorporando elementos a la lista



```
procedure TForm1.BtnM2aM1Click(Sender: TObject);  
Begin  
    { Agrega las líneas del Memo2 al Memo1 }  
    Memo1.Lines.AddStrings(Memo2.Lines);  
end;
```

Accediendo a un elem.de la lista

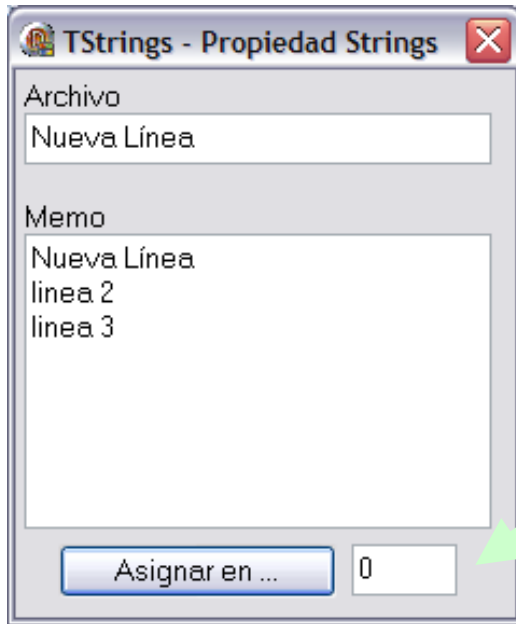
- Para acceder al contenido de la lista puede utilizarse la propiedad **Strings** que consiste de un arreglo con cada uno de los elementos (líneas) indexados a partir de cero.

Ej: Memo1.Lines.strings[0] := 'Primer string';



Objeto TString

Accediendo a un elem.de la lista



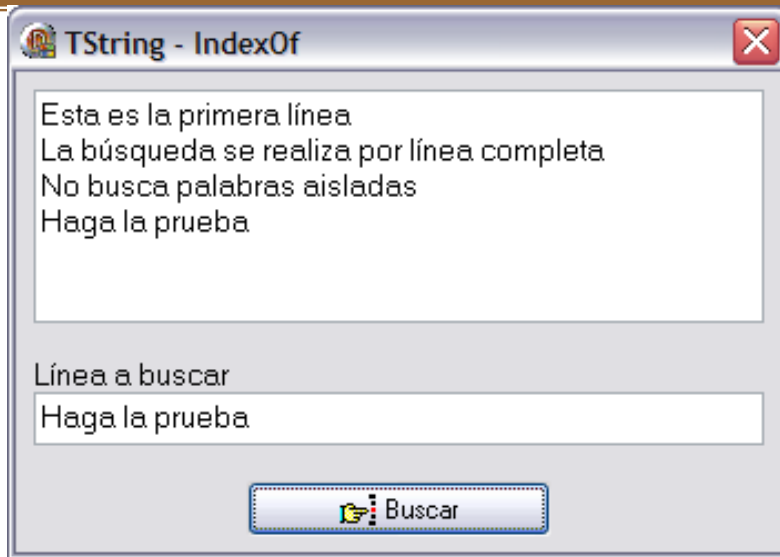
Número entero entre 0
y la cantidad de líneas
de la lista - 1.

```
procedure TForm1.InsertarClick(Sender: TObject);  
Var nro :Integer;  
begin  
    nro := StrToInt(indice.text);  
    Mem1.Lines.strings[nro] := Edit1.Text;  
end;
```

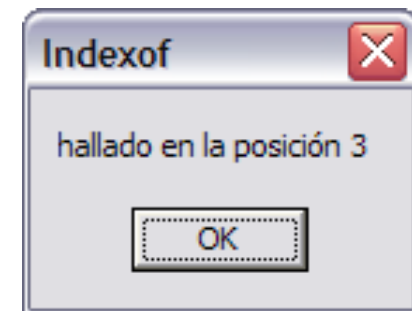
Buscando un string en la lista

- Ubicando un string dentro de la lista
 - El método **IndexOf** retorna la posición del string recibido como parámetro dentro de la lista.
 - Si no lo encuentra retorna -1 .
 - La coincidencia es exacta. Si se busca una coincidencia parcial debe hacerse en forma manual iterando sobre la lista.

Buscando un string en la lista

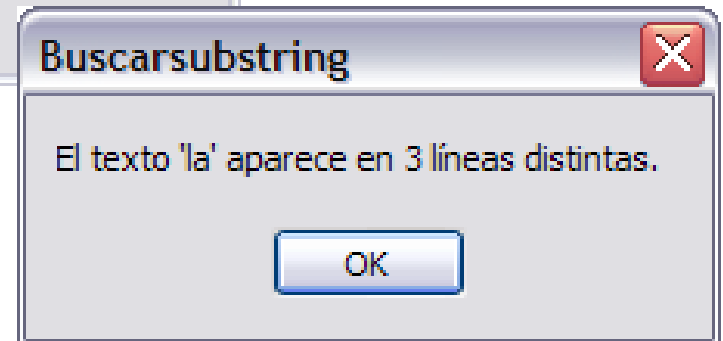
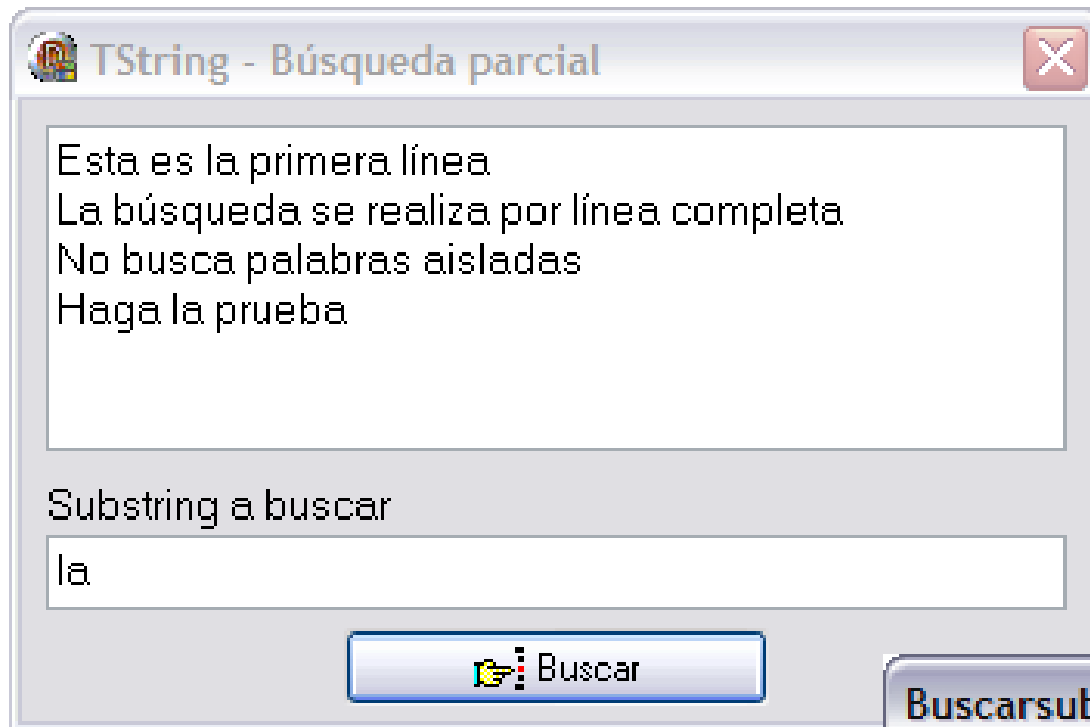


IndexOf.dpr



```
procedure TForm1.BitBtn1Click(Sender: TObject);
Var Indice : integer;
begin
    indice := Memo1.Lines.IndexOf(Edit1.Text);
    if (indice>-1) then ShowMessage('hallado en la
                                   posición ' + IntToStr(indice))
    else ShowMessage('no está');
end;
```

Buscando un substring



Buscando un substring

```
procedure TForm1.BitBtn1Click(Sender: TObject);  
var i, cant: Integer;  
begin  
    cant := 0;  
    for i := 0 to Memo1.Lines.Count - 1 do begin  
        if pos( Edit1.text, Memo1.Lines.Strings[ i ] ) > 0 then  
            cant := cant + 1;  
    end;  
    if cant = 0 then  
        showMessage(Edit1.text + ' no aparece NUNCA.')  
    else showMessage(Edit1.text + ' aparece en ' + intToStr(cant) +  
        ' líneas distintas.');
```

end;

Buscando un substring

```
procedure TForm1.BitBtn1Click;
```

```
var i, cant: Integer;
```

```
begin
```

```
    cant := 0;
```

```
    for i := 0 to Memo1.Lines.Count - 1 do begin
```

```
        if pos( Edit1.text, Memo1.Lines.Strings[ i ] ) > 0 then
```

```
            cant := cant + 1;
```

```
    end;
```

```
    if cant = 0 then
```

```
        showMessage(Edit1.text + ' no aparece NUNCA.')
```

```
    else showMessage(Edit1.text + ' aparece en ' + intToStr(cant) +  
                    ' líneas distintas.');
```

```
end;
```

Las líneas de la lista se acceden de a una comenzando por el índice 0 (cero)

Buscando un substring

```
procedure TForm1.BitBtn1Click(Sender: TObject);  
var i, cant: Integer;  
begin  
    cant := 0;  
    for i := 0 to Memo1.Lines.Count - 1 do begin  
        if pos( Edit1.text, Memo1.Lines.Strings[ i ] ) > 0 then  
            cant := cant + 1;  
    end;
```

- **Función POS**

- Busca un substring en otro y retorna su posición
- Ej : pos('un', 'Esto es un ejemplo') retorna 9.
pos('línea', 'Esto es un ejemplo') retorna 0.

Manejo de strings en una lista

- El método **Move** permite cambiar la posición de los elementos dentro de la lista.

Move(posición original, posición nueva)

Ej: ComboBox1.Items.Move(2,4)

{ mueve el 3er.elem. a la 5ta.posición }

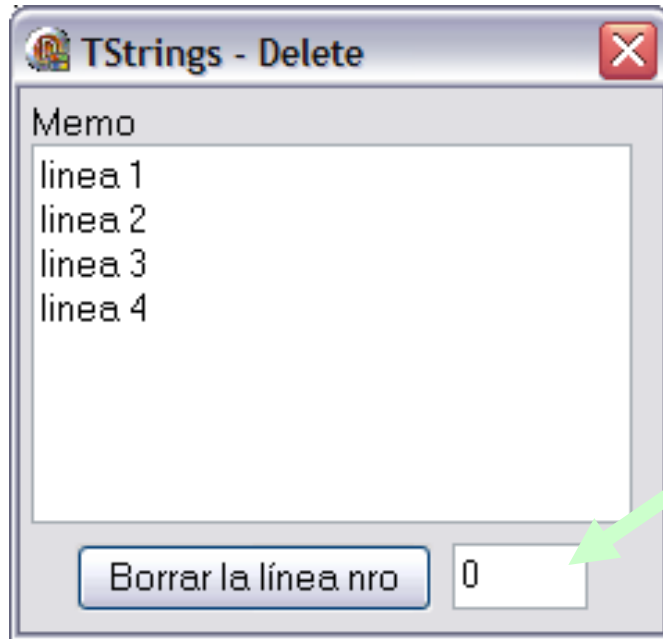
- El método **Delete** borra un elemento de la lista

Delete(posición del elemento a borrar)

Ej: ListBox1.Items.Delete(1)

{ borra el 2do. Elemento de la lista }

Borrando un elemento de la lista



Número entero
entre 0 y la
cantidad de líneas
de la lista - 1.

```
procedure TForm1.InsertarClick(Sender: TObject);  
Var nro :Integer;  
begin  
    nro := StrToInt(indice.text);  
    Memo1.Lines.delete(nro);  
end;
```

Componente Memo

Propiedades

💡 **Lines** → **TString**

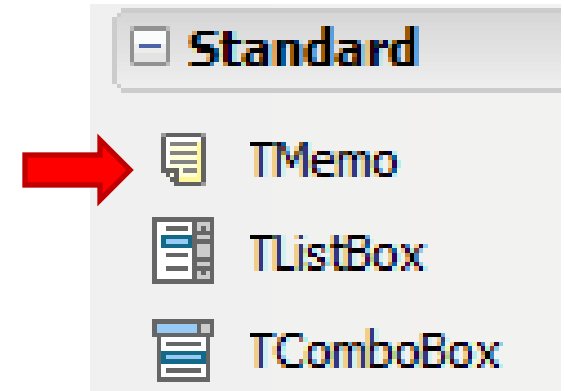
- **Propiedad : Count**
- **Métodos:**

- » **LoadFromFile, SaveToFile**
- » **Add, Insert, AddInsert**
- » **Strings, IndexOf, Move, Delete**

💡 **Text**

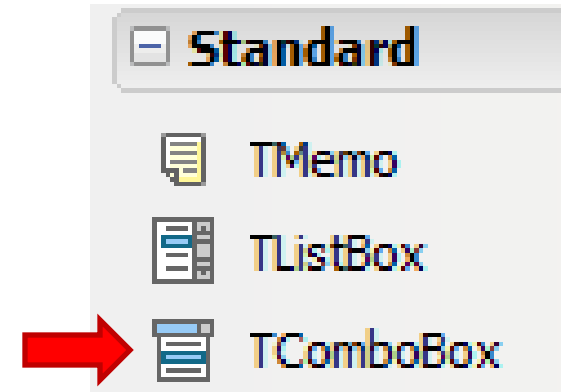
Método

💡 **Clear**



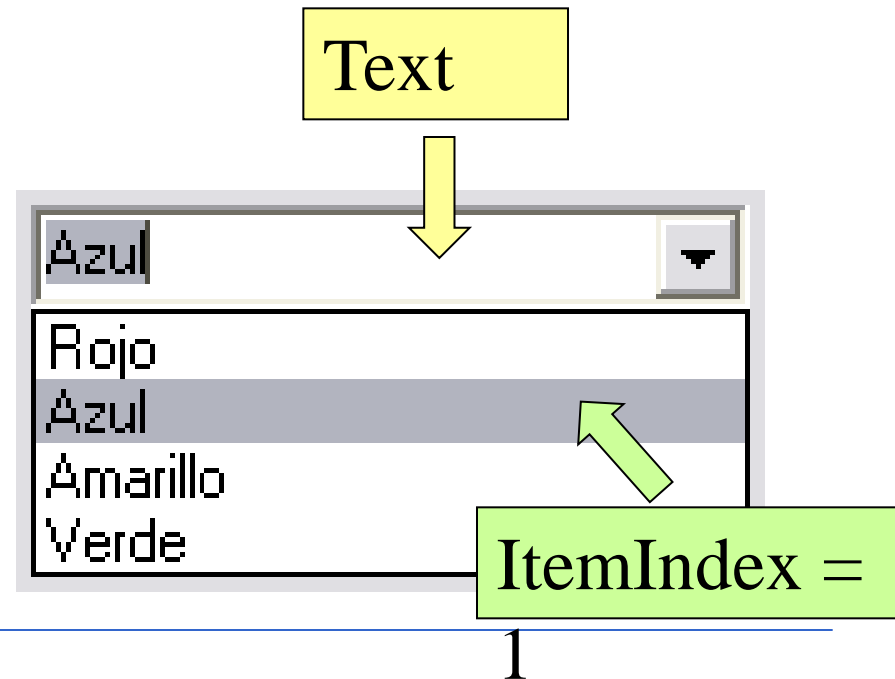
Componente ComboBox

Permite seleccionar una opción de una lista de opciones o ingresarla desde teclado.



Propiedades

- 🔦 Items → **TString**
- 🔦 ItemIndex
- 🔦 Text
- 🔦 Style
- 🔦 Sorted



Componente ComboBox

- **Propiedades**

- **Items** : Lista de opciones (TString).
- **ItemIndex** : Vale -1 si no se ha seleccionado nada o un nro. entre 0 y la cantidad de elementos de la lista de opciones - 1 si alguno fue seleccionado.
- **Text** : contiene el texto ingresado ya sea porque haya sido tipeado o porque fue seleccionado de la lista.
- **Style** : Indica el tipo de ComboBox a utilizar
 - csDropDown : es el valor por defecto
 - csDropDownList : sólo permite seleccionar de la lista. No se puede ingresar por teclado.
 - csSimple : la lista de opciones es siempre visible. Es preciso modificar el tamaño del ComboBox para poder ver la lista.
- Cuando se trabaja con una lista desplegable, la propiedad **DropDownCount** indica la cantidad de elementos visibles de la lista.

Componente ListBox

Permite visualizar una lista de strings y seleccionar uno o varios.

Propiedades

💡 Items → **TString**

💡 ItemIndex

💡 Count

💡 Sorted

💡 Multiselect

💡 SelCount

💡 Selected



Standard

TMemo

TListBox

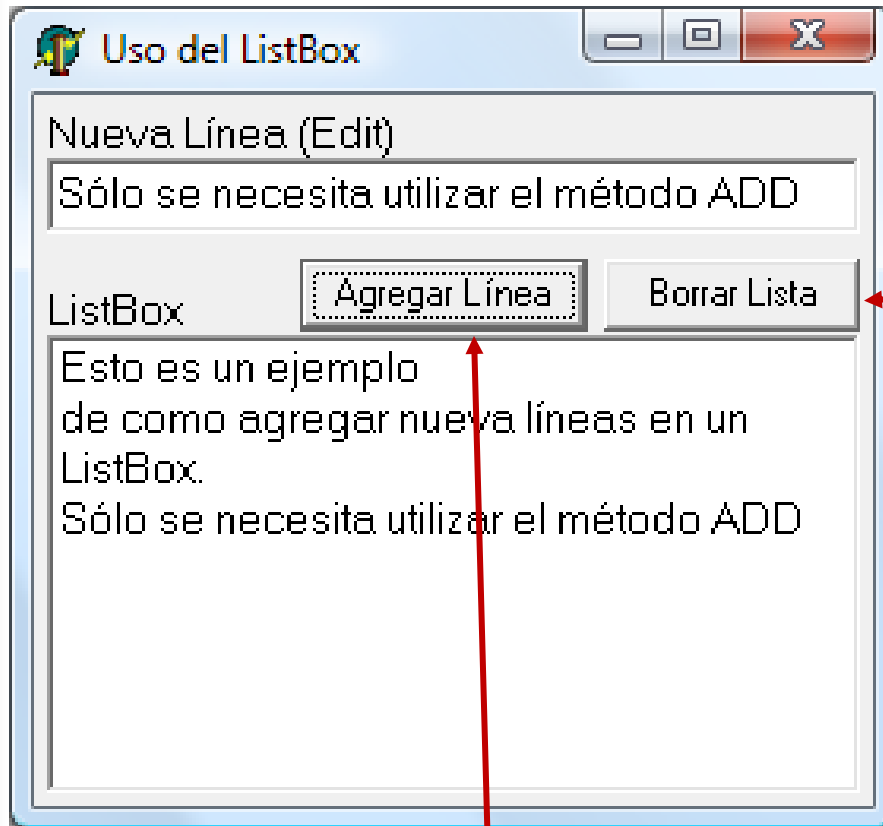
TComboBox

Métodos

💡 Clear

💡 DeletSelected

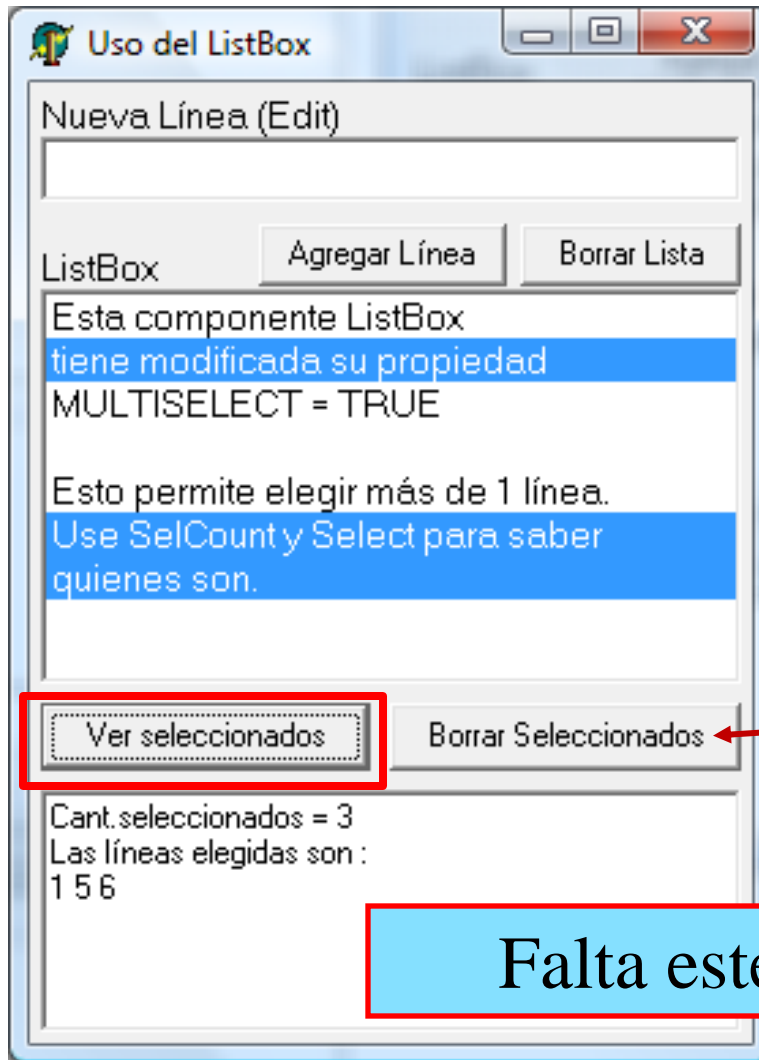
Ejemplo ListBox.dpr



`ListBox1.clear`

`ListBox1.items.add(edit1.text)`

Ejemplo ListBox.dpr



ListBox1.DeleteSelected

Falta este

Botón 'Ver seleccionados'

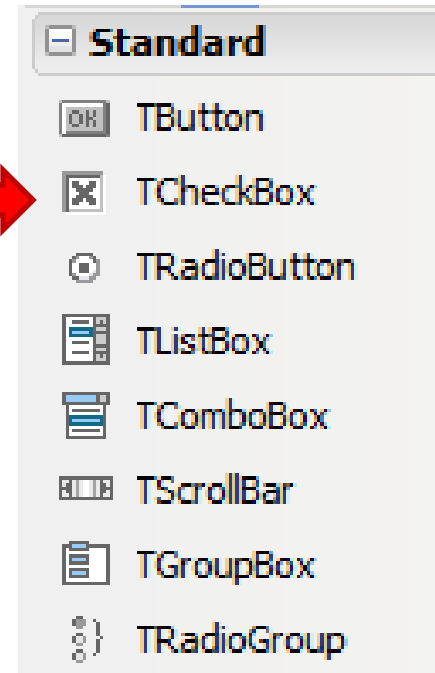
```
procedure TForm1.Button1Click(Sender: TObject);
var i : integer;
    linea : string;
begin
    Memo1.clear;
    if ListBox1.Selcount > 0 then begin
        Memo1.Lines.add( 'Cant.seleccionados =' +
                        IntToStr(ListBox1.Selcount));

        for i :=0 to ListBox1.Count - 1 do
            if ListBox1.Selected[ i ] then
                linea := linea + intToStr( i ) + ' ';

        Memo1.Lines.add('Las líneas elegidas son : ');
        Memo1.Lines.Add(linea);
    end;
end;
```

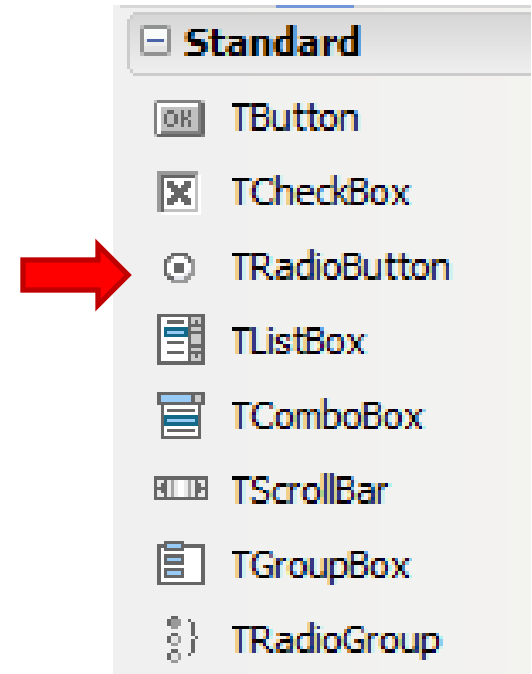
Componente CheckBox

Permite activar o
desactivar una opción.
(Propiedad **Checked**)



Componente RadioButton

- Permite seleccionar una opción.
Propiedad **Checked**.
- Si hay varios botones de radio dentro de la misma componente contenedora sólo uno puede ser seleccionado a la vez.



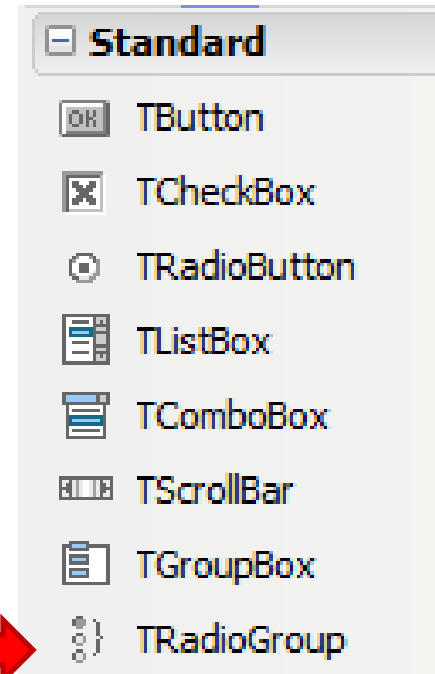
Componente RadioGroup

Permite agrupar
componentes **RadioButton**

- **Propiedades**

(para acceder a los RadioButton)

- **Items** → **TString**
- **ItemIndex**



Ejemplo Encuesta.dpr

Encuesta de la Secretaría de Turismo

Apellido y Nombre
Juan Pablo Zarate

Lugares que conoce

<input type="checkbox"/> Tren de la nubes	<input type="checkbox"/> Cataratas del Iguazú
<input type="checkbox"/> Península de Valdés	<input checked="" type="checkbox"/> El Faro (Ushuaia)
<input checked="" type="checkbox"/> Glaciar Perito Moreno	<input checked="" type="checkbox"/> El Valle de la Luna
<input checked="" type="checkbox"/> El Talampayá	

Hotelería (en general)

☒ Excelente
☐ Muy Buena
☐ Buena
☐ Regular
☐ Mala

Precios de las Excursiones

☒ Muy caras
☐ Precios razonables
☐ Baratas

Medio de transporte habitual

☒ Avión
☐ Auto
☐ Micro

LabeledEdit

GroupBox

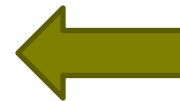
CheckBox

RadioGroup
Items → TString
ItemIndex

RadioButton

Opciones seleccionadas

- Agregar un Memo donde se muestren las opciones seleccionadas



Encuesta de la Secretaría de Turismo

Apellido y Nombre
Juan Pablo Zarate

Lugares que conoce

<input type="checkbox"/> Tren de la nubes	<input type="checkbox"/> Cataratas del Iguazú
<input type="checkbox"/> Península de Valdés	<input checked="" type="checkbox"/> El Faro (Ushuaia)
<input checked="" type="checkbox"/> Glaciar Perito Moreno	<input checked="" type="checkbox"/> El Valle de la Luna
<input checked="" type="checkbox"/> El Talampaya	

Hotelería (en general)

☐ Excelente
☐ Muy Buena
☐ Buena
☐ Regular
☐ Mala

Precios de las Excursiones

☐ Caros
☒ Razonables
☐ Baratos

Medio de transporte habitual

☐ Avión
☐ Auto
☐ Micro

Resumen de la información ingresada

Juan Pablo Zarate
Lugares que conoce:
Glaciar Perito Moreno
El Faro (Ushuaia)
El Valle de la Luna
El Talampaya
No opina sobre Hotelería (en general)
Precios de las Excursiones - Razonables
No eligió medio de transporte

Opciones Seleccionadas

The image shows a software window titled "Micro". Inside the window, there is a section titled "Resumen de la información ingresada" (Summary of entered information). Below this title, the following text is displayed:

Juan Pablo Zarate
Lugares que conoce:
Glaciar Perito Moreno
El Faro (Ushuaia)
El Valle de la Luna
El Talampaya
No opina sobre Hotelería (en general)
Precios de las Excursiones - Razonables
No eligió medio de transporte



```
procedure TForm1.BitBtn1Click(Sender: TObject);
```

```
var i, cant : integer;
```

```
    Opcion : TCheckBox;
```

```
    OpcionR : TRadioButton;
```

```
begin
```

```
    Memo1.Clear;
```

```
    Memo1.Lines.add(LabeledEdit1.Text);
```

```
    cant := 0;
```

```
    for i := 0 to GroupBox1.ControlCount - 1 do begin
```

```
        Opcion := TCheckBox(GroupBox1.Controls[i]);
```

```
        if Opcion.Checked then begin
```

```
            Memo1.Lines.Add(' ' + Opcion.Caption);
```

```
            cant := cant + 1;
```

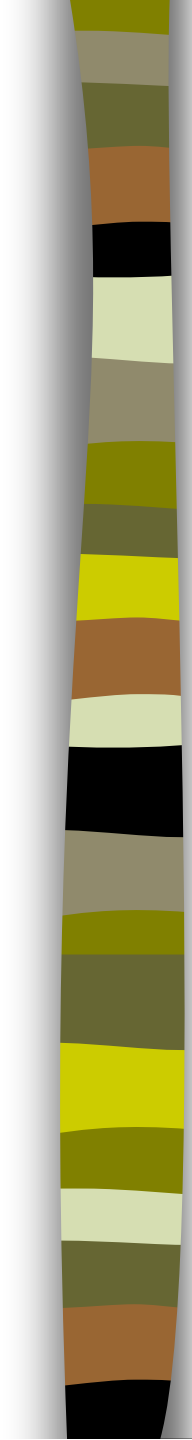
```
        end;
```

```
    end;
```

```
    if cant=0 then Memo1.Lines.add('No conoce ningún lugar')
```

```
        else Memo1.Lines.Insert(1,'Lugares que conoce:');
```

OnClick del botón



```
procedure TForm1.BitBtn1Click(Sender: TObject);
var i, cant : integer;
    Opcion : TCheckBox;
    OpcionR : TRadioButton;
begin
    ...

    Memo1.Lines.Add( QueEligio(RadioGroup1));
    Memo1.Lines.Add( QueEligio(RadioGroup2));

    // Recorrer el formulario buscando los botones de radio y
    // agregar al Memo el medio de transporte elegido o un texto
    // fijo si no eligió nada.

end;
```

OnClick del botón
(cont)

QueEligio es una función declarada en
la parte privada de la clase del
Formulario

Function QueEligio

Dentro de la
clase del
Formulario

private

{ Private declarations }

Function QueEligio(Grupo: TRadioGroup):String;

En implementation

```
Function TForm1.QueEligio(Grupo: TRadioGroup):String;  
begin  
  if Grupo.ItemIndex = -1 then  
    QueEligio := 'No opina sobre '+Grupo.Caption  
  else QueEligio := Grupo.Caption+' - '+  
    Grupo.Items.Strings[Grupo.ItemIndex];  
end;
```