



XIX Edizione, 2020

LiveBindings

Brutto anatroccolo
o
cigno incompreso?



Maurizio Del Magno
Developer



Lev@nte software



i-ORM
DJSON

github.com/mauriziodm/iORM
github.com/mauriziodm/DJSON



mauriziodm@levantesw.it

mauriziodelmagno@gmail.com



facebook.com/maurizio.delmagno
iORM + DJSON (group)



Membro fondatore

eInvoice4D

<https://github.com/delphiforce/eInvoice4D>



LiveBindings

Brutto anatroccolo
o
cigno incompresso?



speaker: Maurizio Del Magno



LIVEBINDINGS

BRUTTO ANATROCCOLO O CIGNO INCOMPRESO?

PRINCIPALE DIFETTO

- Lento
- Migliorato nel tempo (pre Sydney)
- Ulteriore sensibile miglioramento in Delphi 10.4 Sydney

LIVEBINDINGS

- Qualcosa che permette di collegare il valore di una proprietà di un oggetto qualsiasi ad un'altra proprietà di un altro oggetto qualsiasi
- Unico strumento di binding in Firemonkey (FMX)
- **Disponibile anche per progetti VCL**

E' UN CONCETTO NUOVO?

- Il concetto di associazione e interazione tra oggetti non è nuovo
 - Componenti TDataSet & DBAware
 - Meccanismo di owning di TComponent e classi derivate
- Classi progettate specificamente per collaborare tra loro in modo rigido
- Implementano proprietà e metodi concordati in precedenza
- Disponibilità per un numero limitato di classi
- **La programmazione DataSet oriented non è OOP**
 - Dov'è il singolo punto dove mettere la business logic?
 - La manutenibilità diventa facilmente un problema
 - Codice difficilmente testabile (*unit test*)

LIVEBINDINGS: FA DI PIÙ?

- Si può collegare qualsiasi cosa a qualsiasi altra cosa
- Anche oggetti non progettati per collaborare tra loro
- Non si è limitati a classi/componenti/oggetti specifici
- Si possono bindare TDataSet anche a controlli non DBAware (anche VCL)
- Si possono bindare **oggetti “vivi”** a qualunque controllo
- **Reale programmazione OOP** anche in un ambiente RAD
 - Le classi del domain model includono anche la logica di business
 - Basta codice in OnBefore... OnAfter...
 - Classi facilmente testabili (*unit test*)
 - Codice più manutenibile e riutilizzabile
 - Interi grafi di oggetti, anche complessi, come sorgente di dati

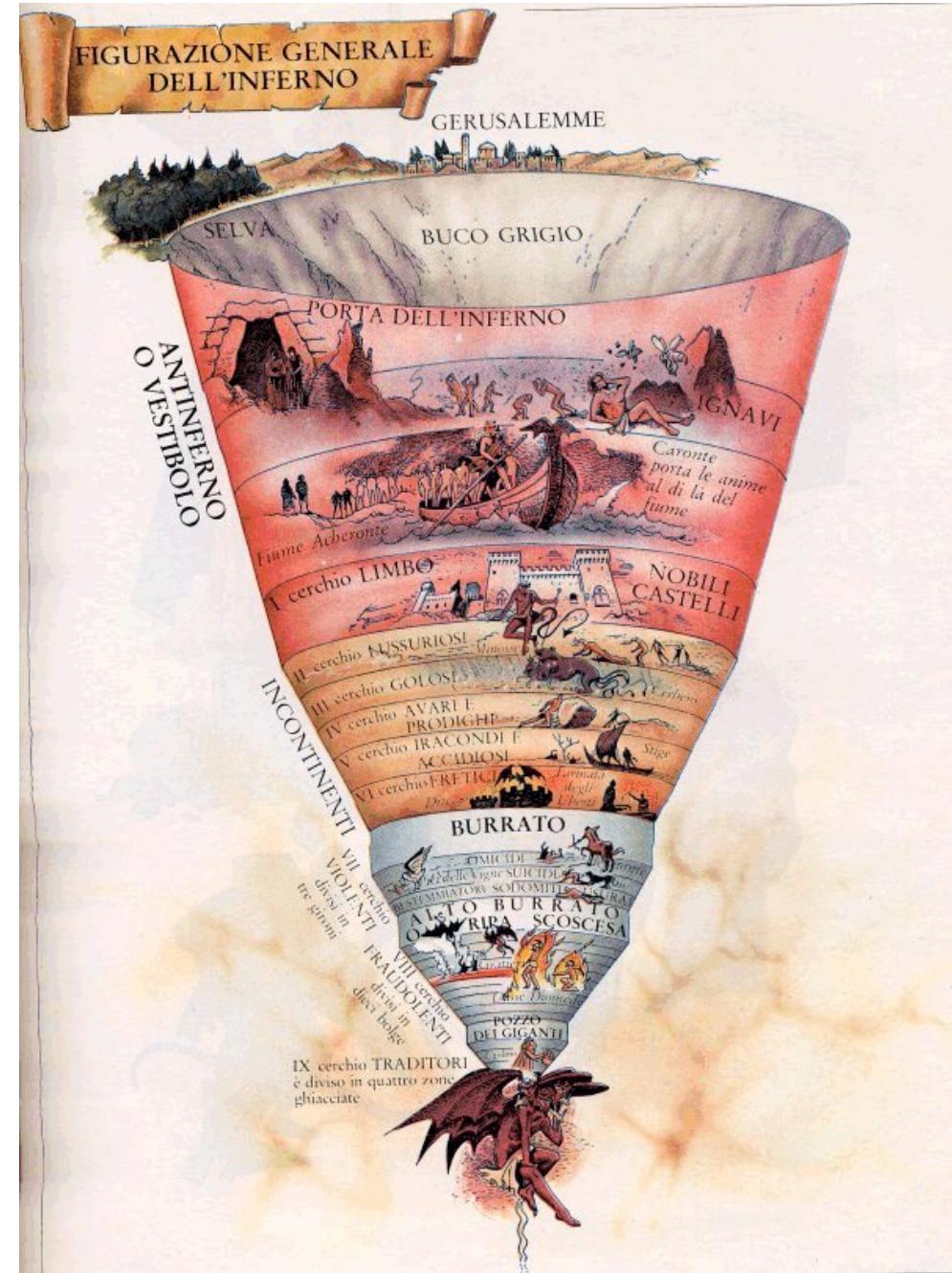
DOMANDE...

- Cosa permette di fare?
- E' customizzabile?
- Come estendere LiveBindings per adattarlo alle mie esigenze?
- Esistono librerie che estendono le sue funzionalità?

LIVEBINDINGS

COME È FATTO?

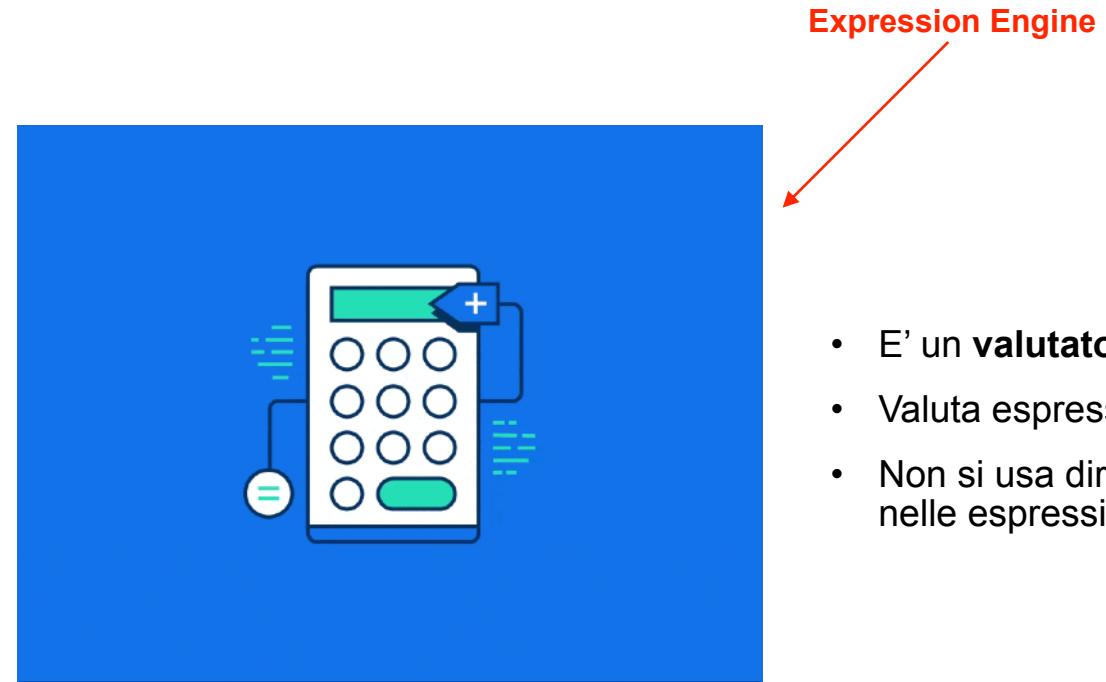




BASICS

- Expression engine
- Relational expressions
- Scope
- Output converters

EXPRESSION ENGINE

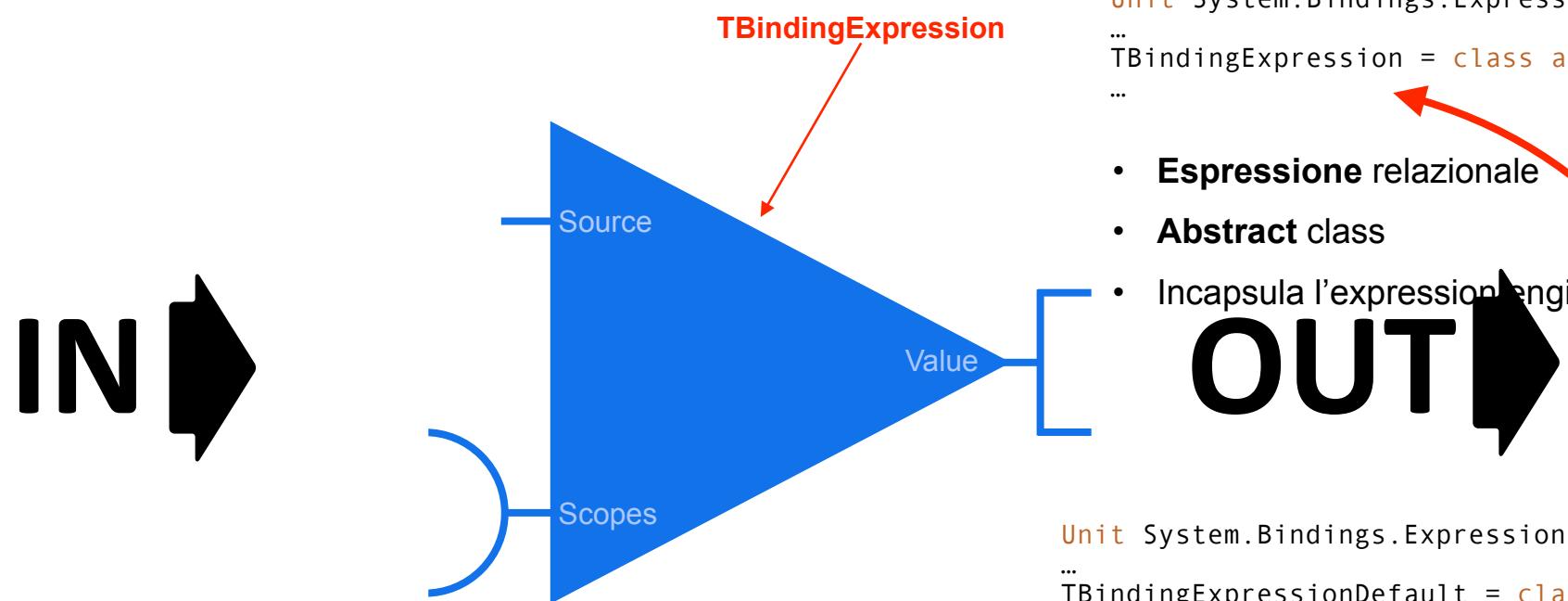


Expression Engine

- E' un **valutatore di espressioni**
- Valuta espressioni e **restituisce un risultato**
- Non si usa direttamente, sempre encapsulato nelle espressioni stesse

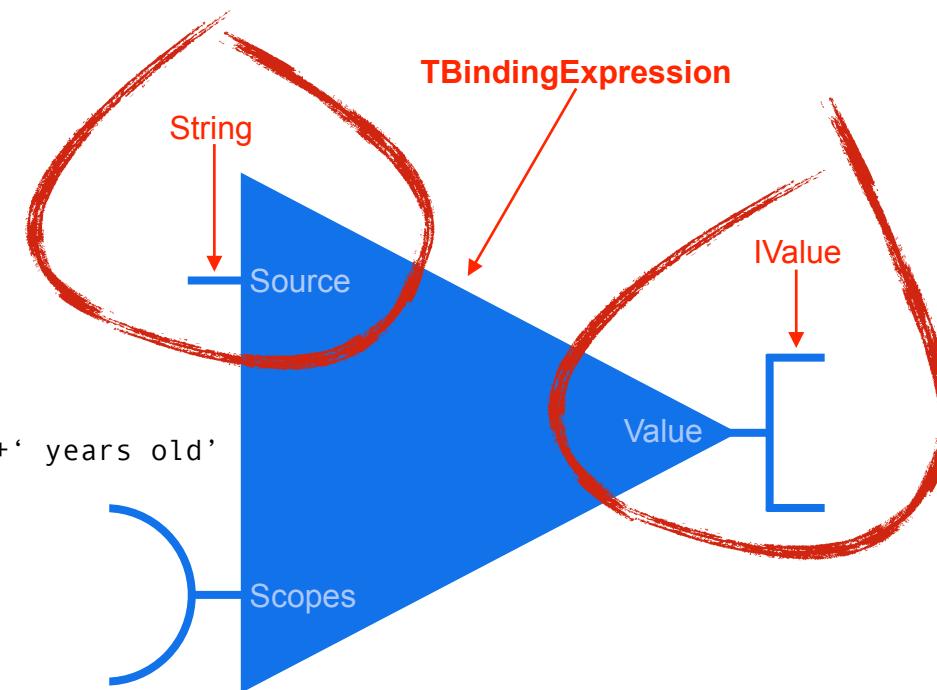
TBINDINGEXPRESSION

TBINDINGEXPRESSION



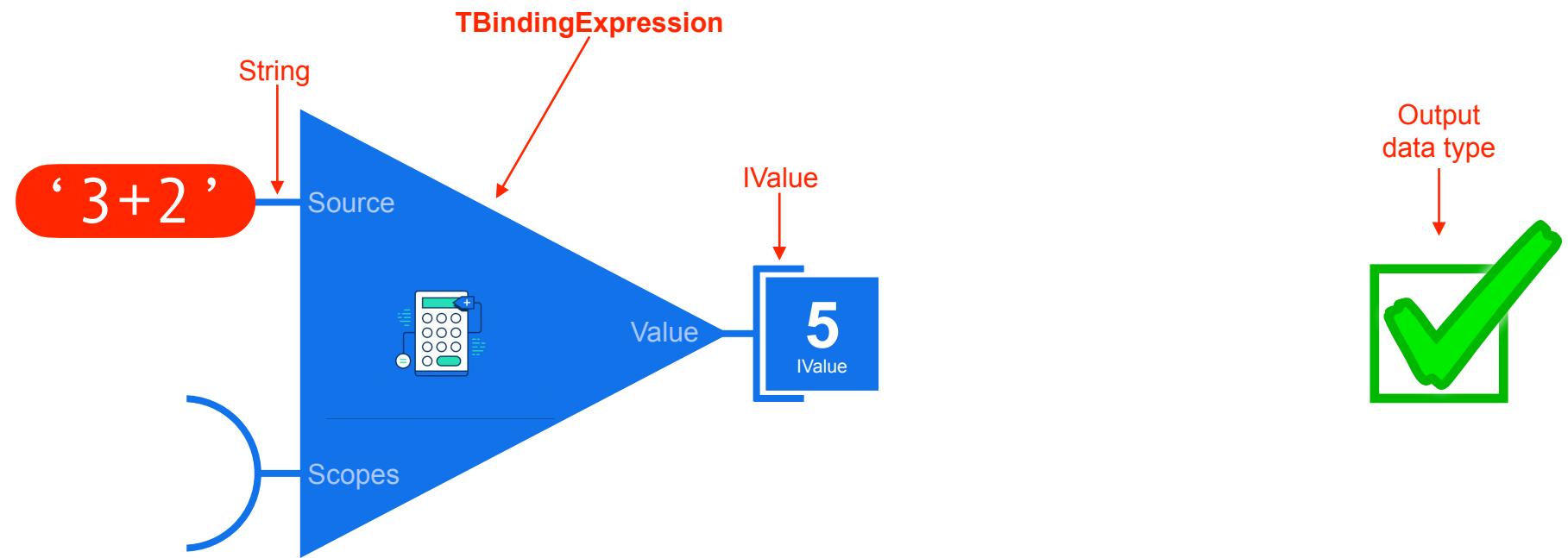
TBINDINGEXPRESSION

- **Source expression:** (relational expression)
 - `'(6+2)/(5-3)*-1'`
 - `'Sono le ore ' + ToString(10)`
 - `Person.FullName+' is '+ToString(Person.Age)+' years old'`



- ```
Unit System.Bindings.EvalProtocol;
...
IValue = interface
 function GetType: PTypeInfo;
 function GetValue: TValue;
end;
```
- Ritorna il **risultato** dell'espressione
  - Si basa su **TValue**
  - Può contenere **qualsiasi tipo** di valore
    - *String, Integer, TDateTime, Boolean*
    - *Object, Interface, Record, Array*
    - *Enumerated*

# TBINDINGEXPRESSION



# TBINDINGEXPRESSION SAMPLE

```
var
 Expr: TBindingExpression;
 ExprResult: IValue;
begin
 Expr := TBindingExpressionDefault.Create(nil); ← Creazione istanza
 try
 Expr.Source := '5 + 3 * (2 + 4)'; ← Definizione dell'espressione
 Expr.Compile([], [], []);
 ExprResult := Expr.Evaluate; ← Valutazione
 Writeln(ExprResult.GetValue.ToString);
 finally
 Expr.Free;
 end;
end;
```

# TBINDINGEXPRESSION SAMPLE

```
var
 Expr: TBindingExpression;
 ExprResult: IValue;
begin
 Expr := TBindingExpressionDefault.Create(nil);
 try
 Expr.Source := '5 + 3 * (2 + 4)';
 Expr.Compile([], [], []);
 ExprResult := Expr.Evaluate;
 WriteLn(ExprResult.GetValue.ToString);
 finally
 Expr.Free;
 end;
end;
```

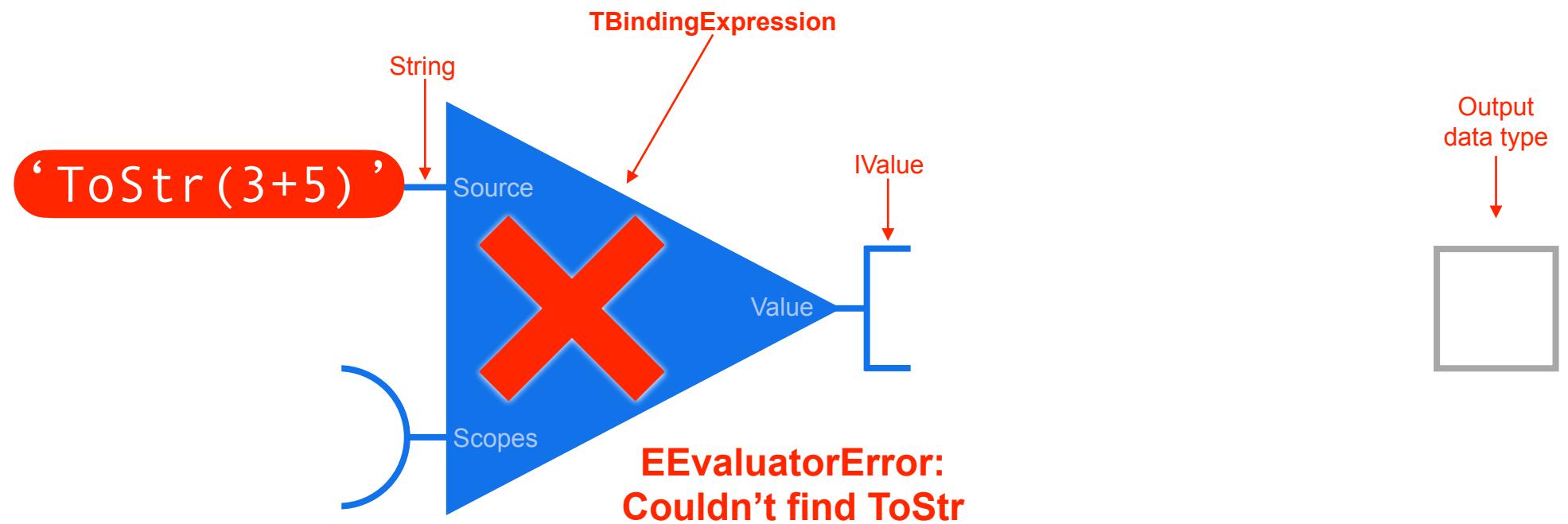
overloaded → Expr.Compile([], [], []); ← Array of IScope

Array of TComponents (*accessibile con il nome specificato nella proprietà "Name"*)

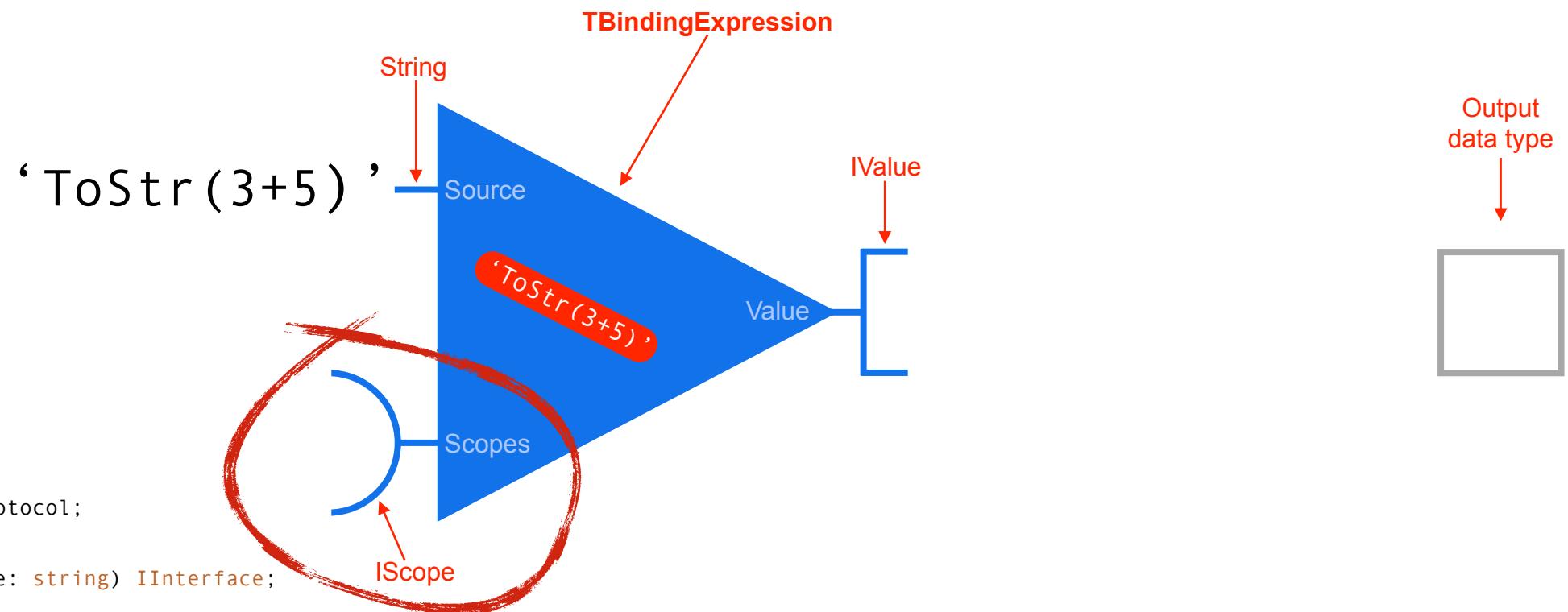
Array of TBindingAssociation

# ISCOPE

## ISCOPE



# ISCOPE



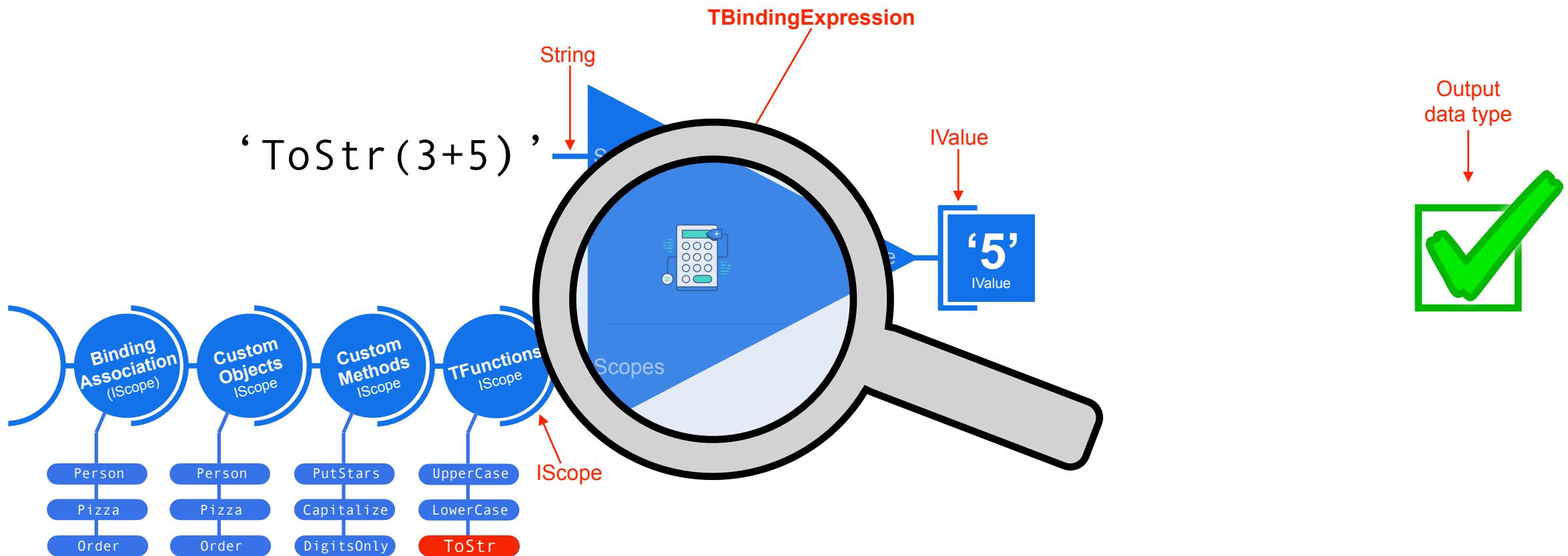
```

Unit System.Bindings.EvalProtocol;
...
IScope = interface
 function Lookup(const Name: string) IInterface;
end;

```

- Determina la **visibilità** di “qualcosa” nel **contesto dell’espressione** (*lookup multilivello*)
- L’expression engine vede **solo IScope** (*Metodi, oggetti, componenti, variabili, costanti ecc.*)
- Ogni scope è associato a un **nome**
- Gli scope sono **gerarchici**
- **Customizzazione**, adattamento dell’engine alle mie esigenze

## ISCOPE



# ISCOPE - TBINDINGMETHODSFACTORY

- Unit: System.Bindings.Methods (*static class*)
- Collezione di **metodi predefiniti** (*UpperCase, LowerCase,ToStr,Format,IfThen...*)
- Possibilità di registrare propri **metodi custom**
- Fornisce **IScope** per rendere visibili all'expression engine alcuni o tutti i metodi registrati

# TBINDINGMETHODSFACTORY SAMPLE

```
TBindingMethodsFactory.RegisterMethod(← Registra nuovo metodo (parametro TMethodDescription)
 TMethodDescription.Create(MakeInvokable(← Wrap dell'anonimo method in una IInvokable (parametro TInvokableBody)
 function(Args: TArray<IValue>): IValue ← Anonimous method (attenzione ai parametri)
 var
 StrValue: String; ← Crea una TMethodDescription (1° parametro IInvokable)
 begin
 StrValue := Args[0].GetValueAsString;
 Result := TValueWrapper.Create(CapitalizeString(StrValue)); ← Wrap del risultato in un IValue
 end),
 'Capitalize', // Method identifier
 'Capitalize', // Method name
 'Utilities', // Unit name where the method is defined
 True, // Enabled
 'Capitalize the first letter of every word in the string', // Long method description (Hint)
 nil // Method's platform (TComponent=VCL; TFMXComponent=FMX; nil=both)
)
);
```

Esempio: 1.3

# ISCOPE - CUSTOM METHOD (NO FACTORY)

- E' possibile aggiungere metodi custom senza la factory (*TCustomMethodsFactory*)
- **TBindings helper class** (*System.Bindings.Helper*)
  - class function TBindings.CreateMethodScope(const MethodName: string; InvokableMethod: IInvokable): IScope;

```
var
 LScope: IScope;
begin
 LScope := TBindings.CreateMethodScope('PutStars',
 MakeInvokable(← Wrap dell'anonimo method in una IInvokable (parametro TInvokableBody)
 function(Args: TArray<IValue>): IValue ← Anonimous method (attenzione ai parametri)
 begin
 Result := TValueWrapper.Create('***' + Args[0].GetValueAsString + '**');
 end
)
);
 ...
end;
```

Crea uno IScope per il metodo (2° parametro IInvokable)

Wrap dell'anonimo method in una IInvokable (parametro TInvokableBody)

Anonimous method (attenzione ai parametri)

Wrap del risultato in un IValue

Esempio: 1.4

# ISCOPE - OBJECTS

- function **WrapObject(AObject: TObject): IScope;** (*System.Bindings.ObjEval*)
- Wrap dell'oggetto in un IScope

```
var
 LScope: IScope;
begin
 LScope := WrapObject(TPerson.Create('Mario', 'Rossi', '01/07/1980'));
 ...
end;
```

Esempio: 1.5

# ISCOPE - TDICTIONARYSCOPE

- Gli IScope sono **gerarchici**
  - In realtà l'espressione riceve sempre un solo IScope (root scope)
  - Il root scope ha poi una serie di IScope innestati
  - Gli IScope innestati possono a loro volta contenere altri IScope ulteriormente innestati
  - Una **gerarchia** di IScope con un meccanismo di **Lookup** (*multilivello*)
- **TDictionaryScope** è uno IScope (*implementa l'interfaccia IScope - System.Bindings.EvalSys*)
- Contiene una **collezione** di IScope ognuno associato a un **nome**
- La chiave del dictionary è anche il nome con il quale quello scope sarà visibile nell'espressione

# ISCOPE - TDICTIIONARYSCOPE SAMPLE

```
var
 LPerson: TPerson;
 LPersonScope: IScope;
 LDictionayScope: TDictionayScope;
begin
 LPerson := TPerson.Create('Maurizio', 'Del Magno', '22/10/1970');
 LPersonScope := WrapObject(LPerson); ← Wrap di LPerson in un IScope
 LDictionayScope := TDictionayScope.Create; ← Creazione del TDictionayScope
 LDictionayScope.Map.Add('Person', LPersonScope);
 ...
end;
```

Aggiunge lo scope alla collezione associandogli un nome

# TBINDINGASSOCIATION

- Unit: System.Bindings.Expression;
- E' un record
- Associazione di un oggetto a un nome
- L'oggetto sarà “visibile” nel contesto dell'espressione con quel nome
- Dietro le quinte viene comunque creato un IScope

```
TBindingAssociation = record
public
 RealObject: TObject;
 ScriptObject: String;
 constructor Create(ARealObject: TObject; const AScriptObject: String);
end;
```

# TBINDINGASSOCIATION SAMPLE

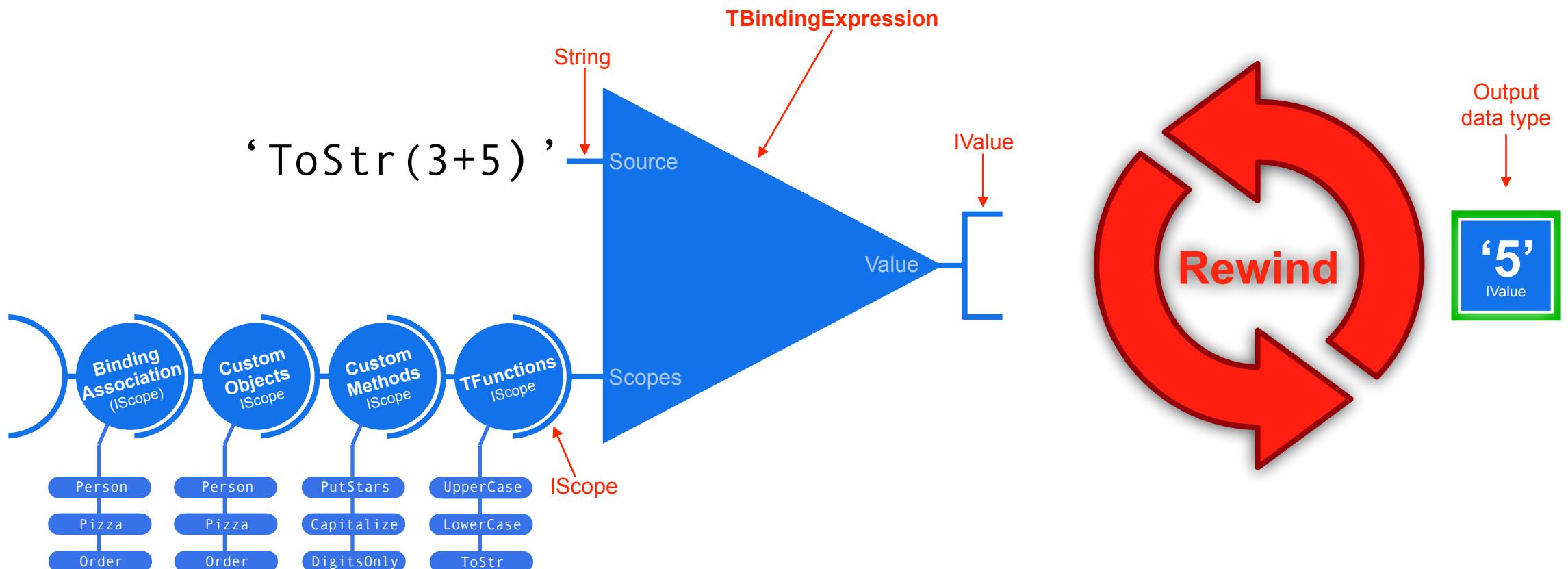
```
var
 LExpr: TBindingExpression;
 LPerson: TPerson;
begin
 LPerson := TPerson.Create('Mario', 'Rossi', '14/08/1972');
 ...
 LExpr.Compile([], [TBindingAssociation.Create(LPerson, 'Person')]. []);
 ...
end;
```

or

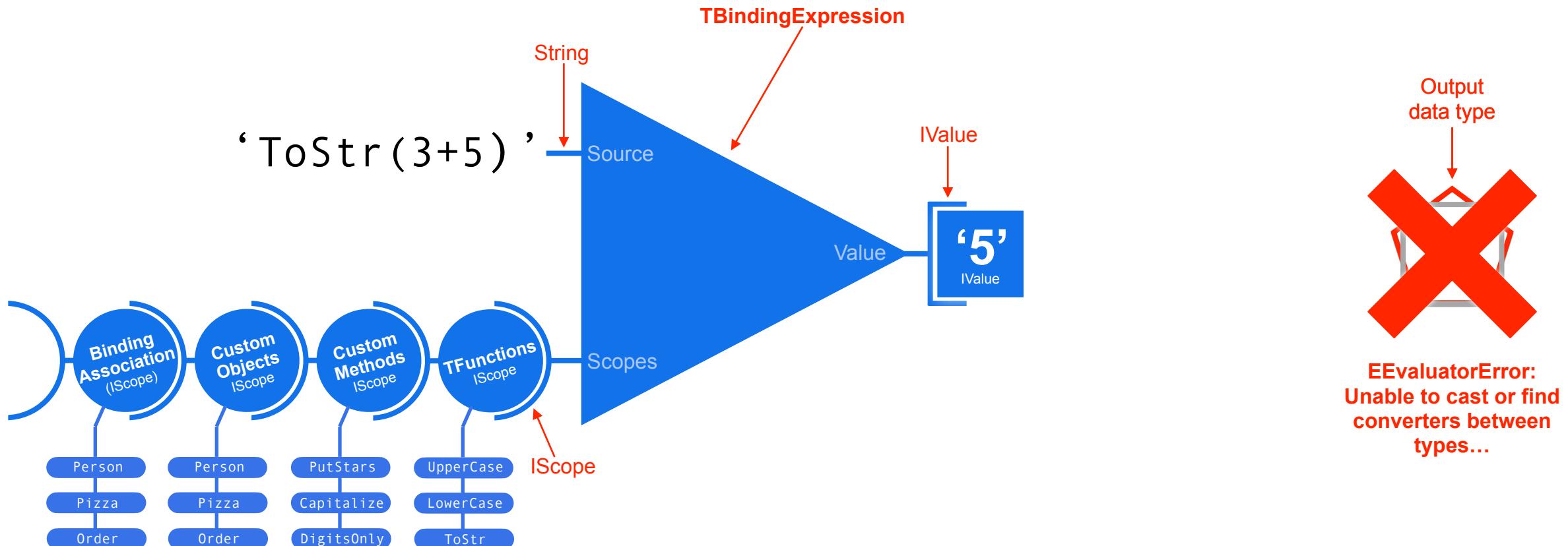
```
var
 LExpr: TBindingExpression;
begin
 ...
 LExpr.Associations.Add(TPerson.Create('Mario', 'Rossi', '14/08/1972'), 'Person');
 ...
end;
```

# OUTPUT CONVERTERS

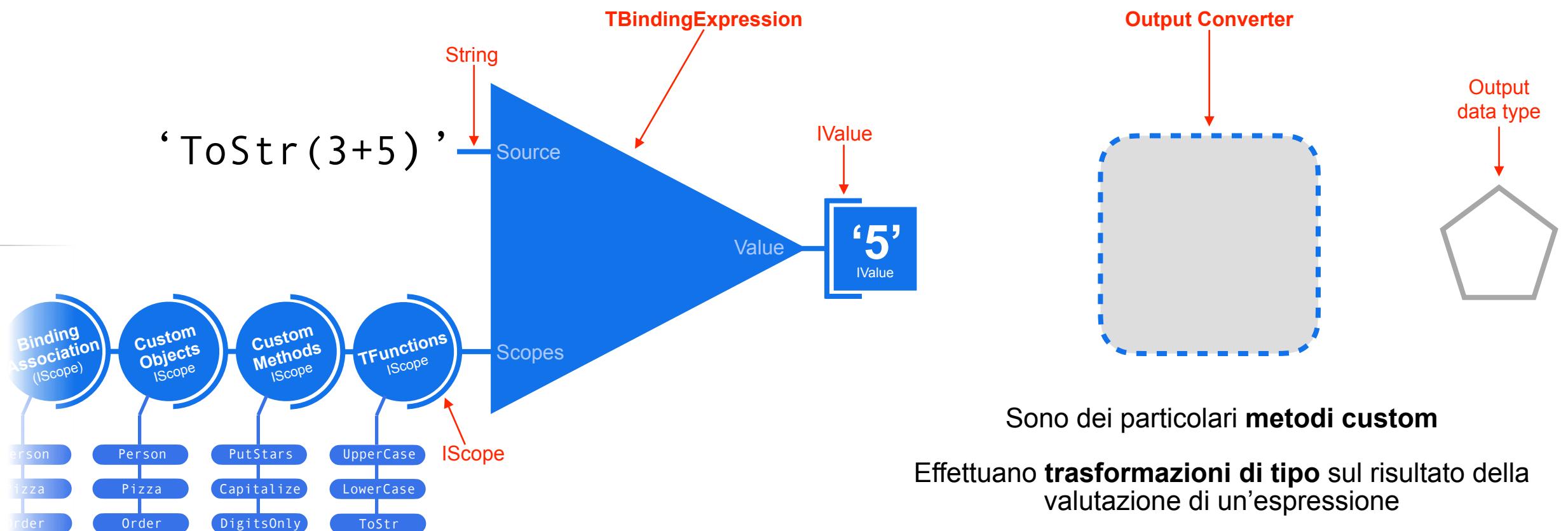
# OUTPUT CONVERTERS



# OUTPUT CONVERTERS



# OUTPUT CONVERTERS



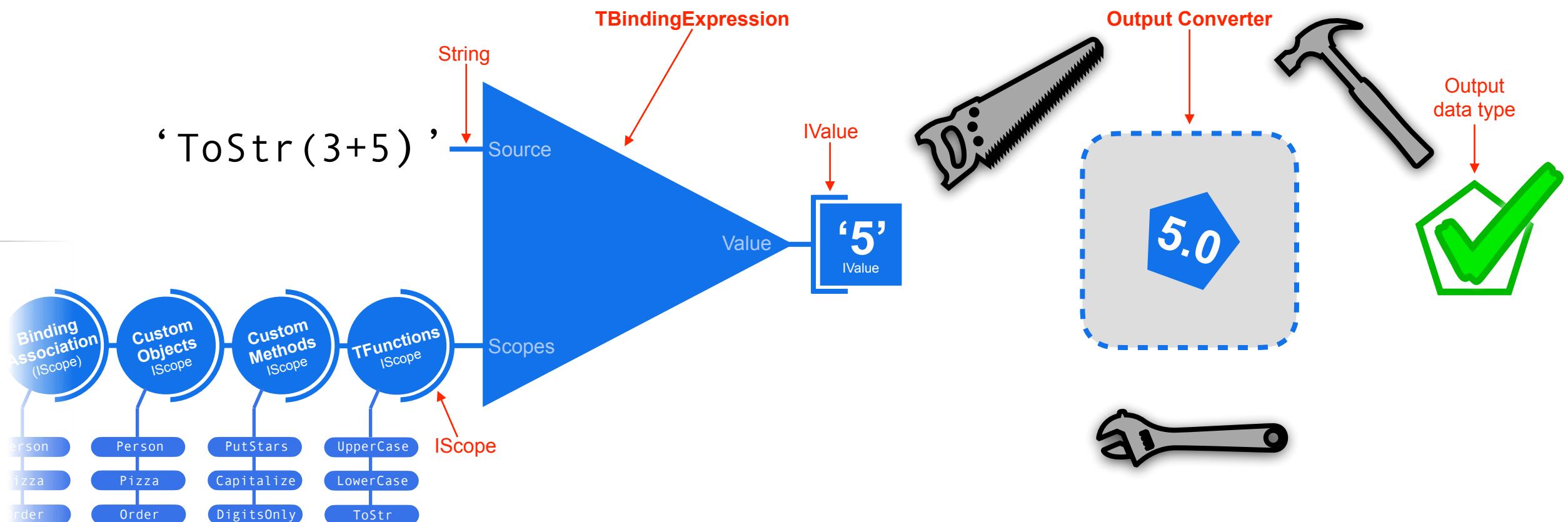
Sono dei particolari **metodi custom**

Effettuano **trasformazioni di tipo** sul risultato della valutazione di un'espressione

Non vengono inseriti esplicitamente nelle espressioni

Altro grado di **customizzazione**

# OUTPUT CONVERTERS



# OUTPUT CONVERTERS SAMPLE

```
TValueRefConverterFactory.RegisterConversion(TypeInfo(Integer), TypeInfo(String),
TConverterDescription.Create(
procedure(const InValue: TValue; var OutValue: TValue)
begin
 OutValue := InValue.AsString;
end,
'IntToStr', // Method identifier
'IntToStr', // Method name
'U.Converters', // Unit name where the method is defined
True, // Enabled
'Convert an integer into a string', // Long method description (Hint)
nil // Converter platform (TComponent=VCL; TFMXComponent=FMX; nil=both)
)
);
```

# HIGH LEVEL COMPONENTS

# HIGH LEVEL COMPONENTS

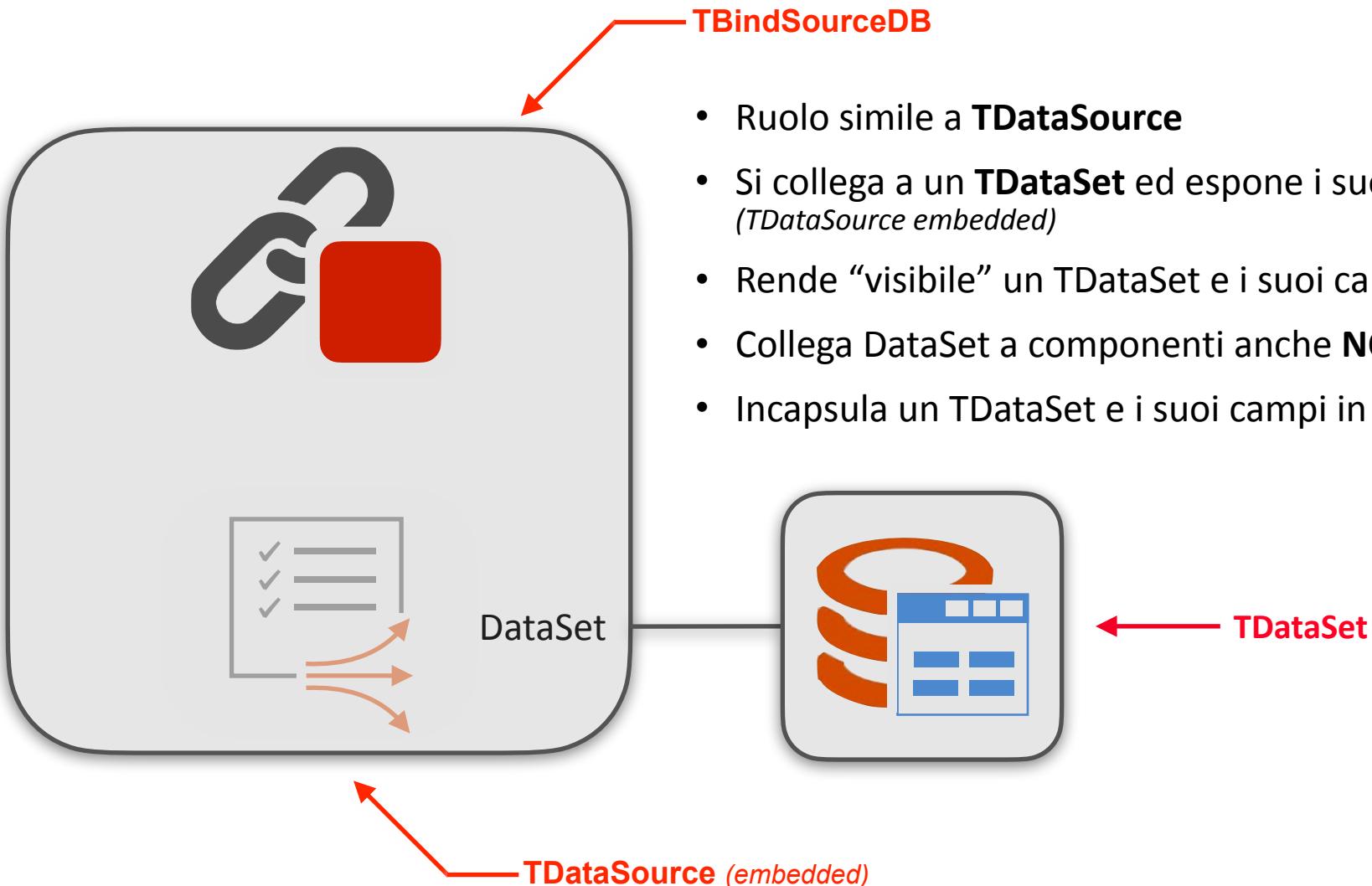
- **TBindingList**
  - Mantiene una lista Binding Classes (espressioni) su una form
  - Methods & Output Converters collections
  - Expression editor
- **Binding classes**
  - Binding Expressions (*TBindExpression*, *TBindExprItems*, **NOT** *TBindingExpression*)
  - List LiveBindings
  - Link LiveBindings
  - Quick Bindings
- **Bind sources**
  - *TBindSourceDB*
  - *TPrototypeBindSource*
  - (*TAdapterBindSource*)

# BIND SOURCES

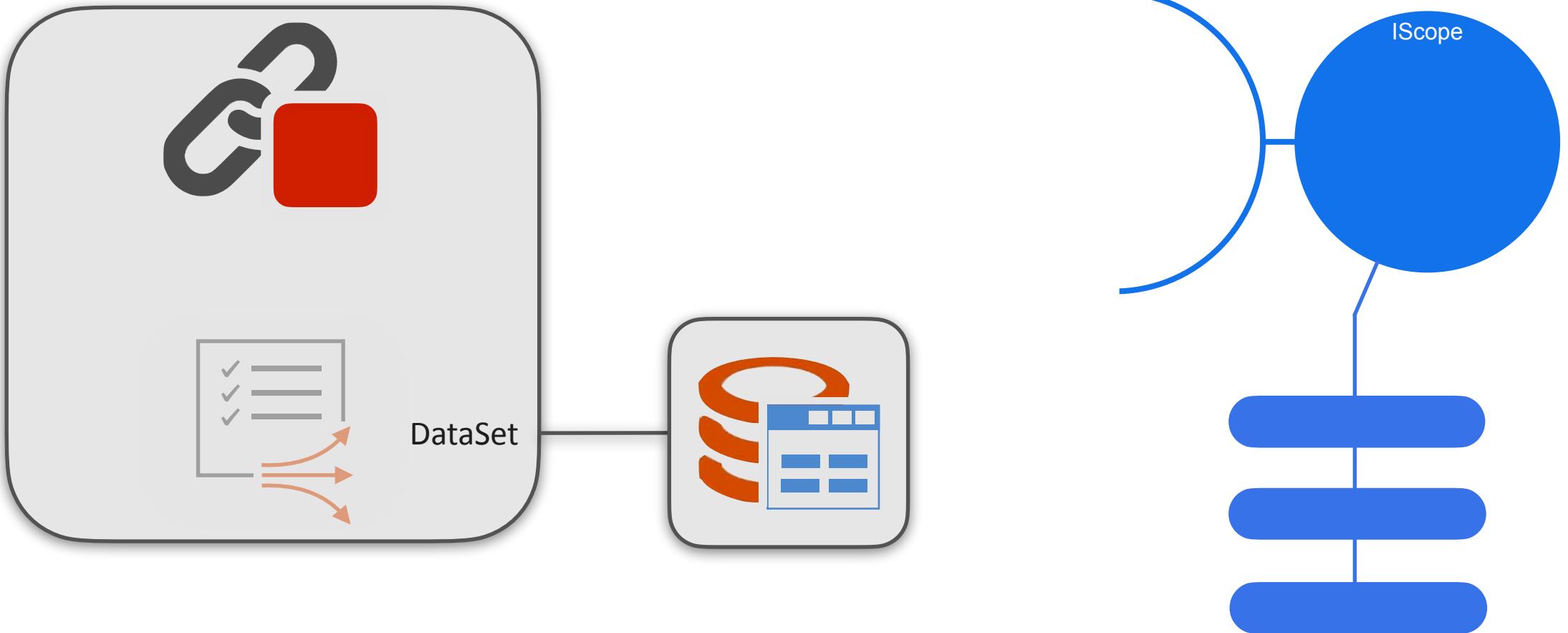
# BIND SOURCES

- Fonte di dati
- Qualcosa che espone all'expression engine dei dati provenienti da:
  - DataSet (*DB*)
  - Oggetti (*singoli o liste*)

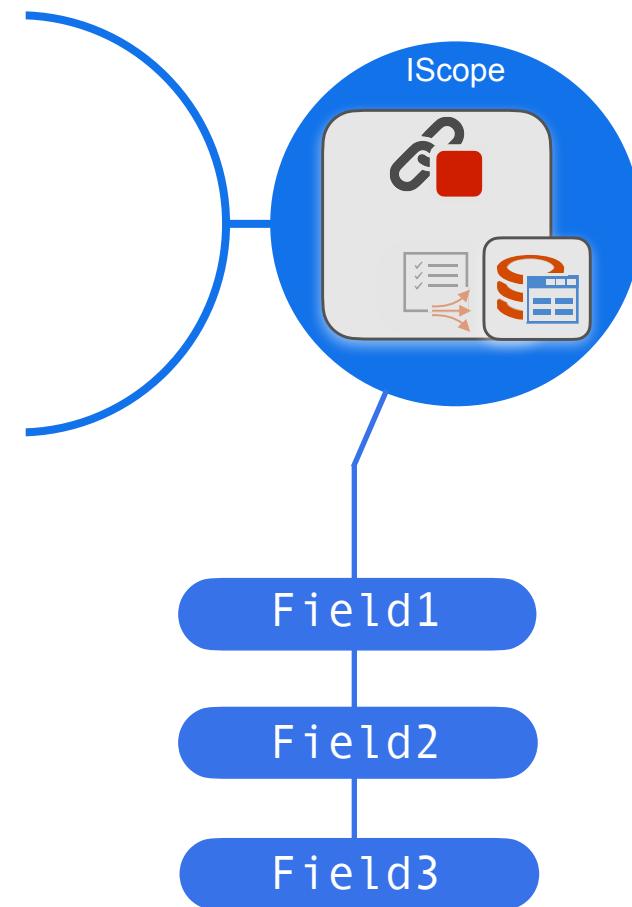
# TBindSourceDB



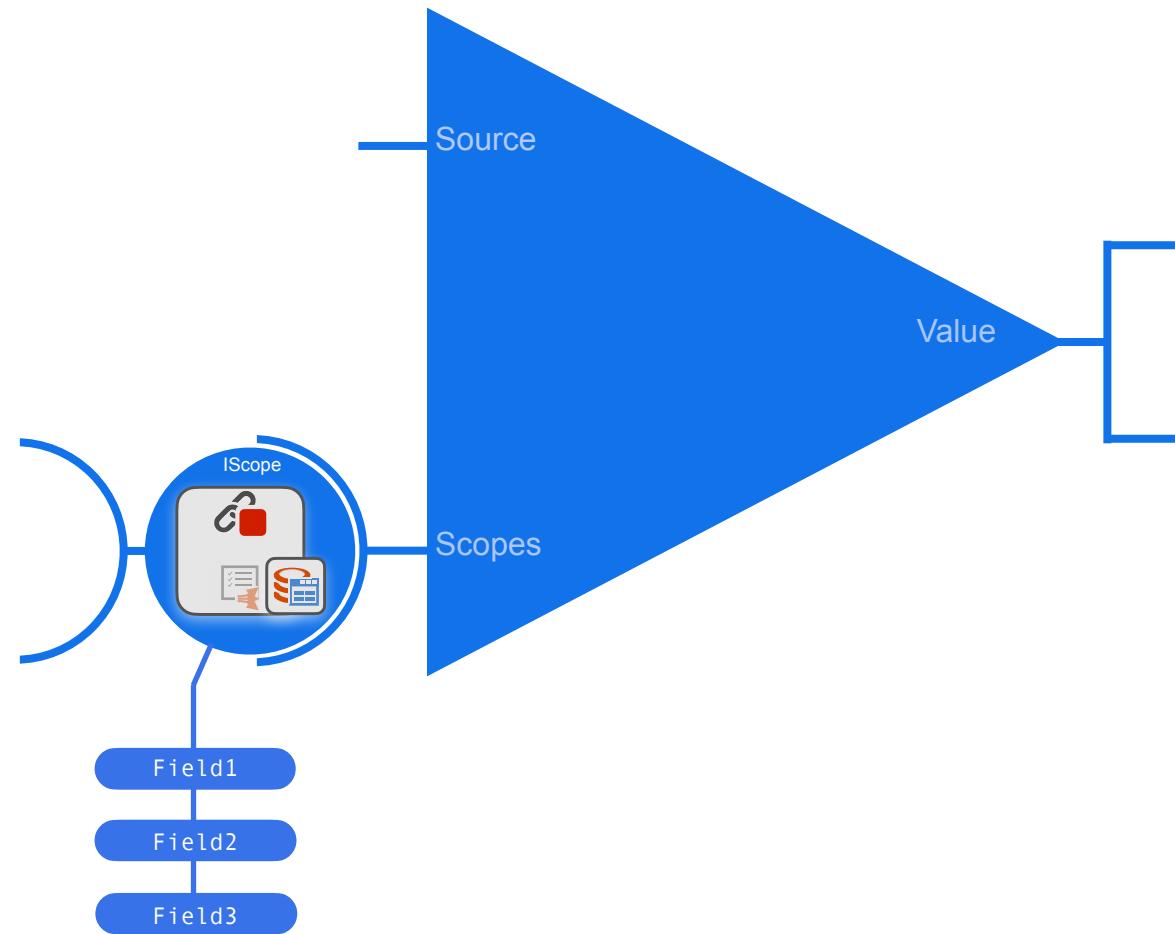
# TBINDSOURCEDB



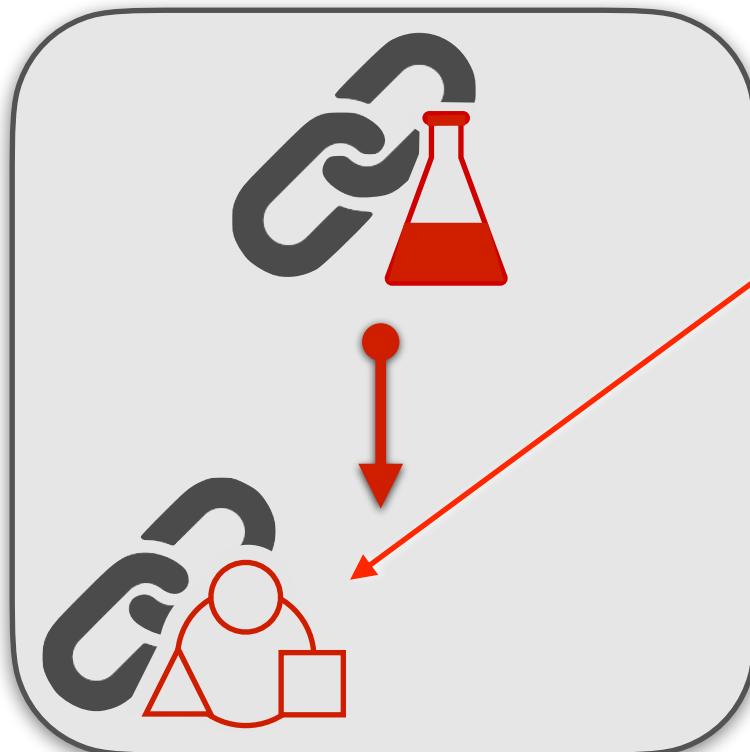
# TBINDSOURCEDB



# TBINDSOURCEDB



# TPROTOTYPEBINDSOURCE



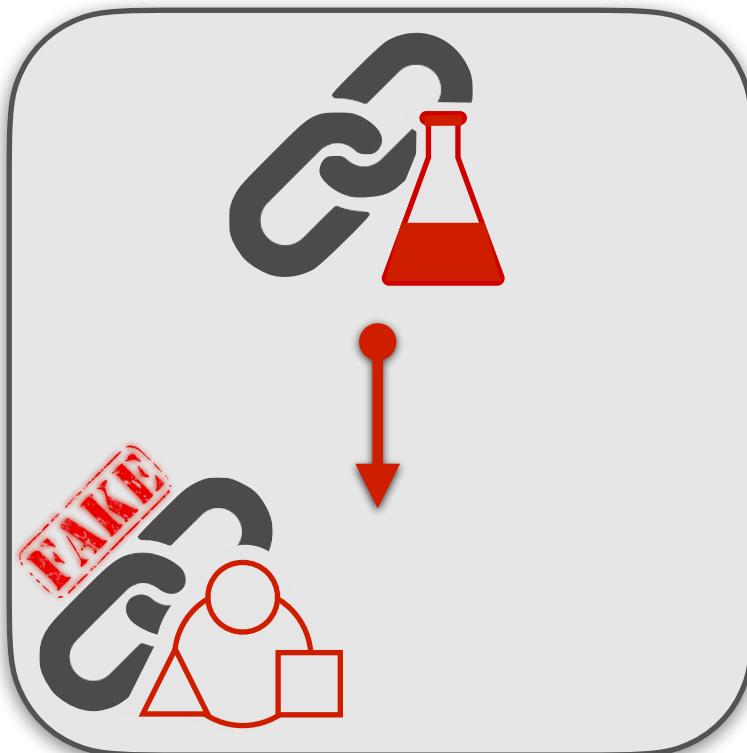
## TPrototypeBindSource

- Ruolo simile a **TDataSource**
- Espone **oggetti** (o liste di oggetti) per il binding

## TBindSourceAdapter

- Ruolo simile a **TDataSet** (*State, Edit, Post, Append, Cancel...*)
- **Adatta** un oggetto (o una lista di oggetti) per essere poi assegnato a un TPrototypeBindSource come fonte di dati
- Load & buffer (*valori di proprietà di oggetti*)
- **Post su oggetti** non su DB (*anche Delete, Append, Cancel...*)
- **Reale OOP** senza perdere le caratteristiche **RAD** di Delphi
- TObjectBindSourceAdapter/TListBindSourceAdapter

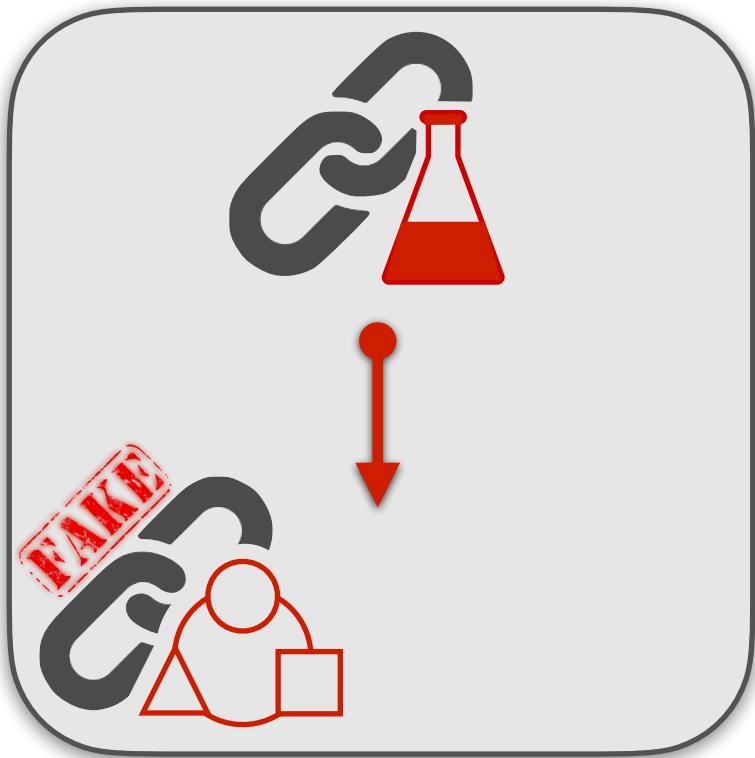
# TPrototypeBindSource



## TPrototypeBindSource

- Ruolo simile a **TDataSource**
- Espone **oggetti** (o liste di oggetti) per il binding
- Permette la **prototipazione** della UI con **dati fake** (*DT*)
- Sviluppo di un prototipo di UI senza avere ancora nemmeno pensato alla fonte dati (*DB?*, *Objects?*) anche prima, o contemporaneamente, lo sviluppo della logica di dominio e/o del DB
- Incapsula un oggetto (o lista?) e le sue proprietà in **IScope**
- A **runtime** dovrò fornire una **fonte di dati reali** che sostituirà il generatore di fake data (evento **OnCreateAdapter**) (*RT*)

# TPROTOTYPEBINDSOURCE

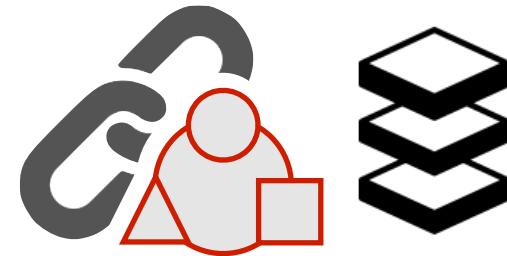


# TPROTOTYPEBINDSOURCE

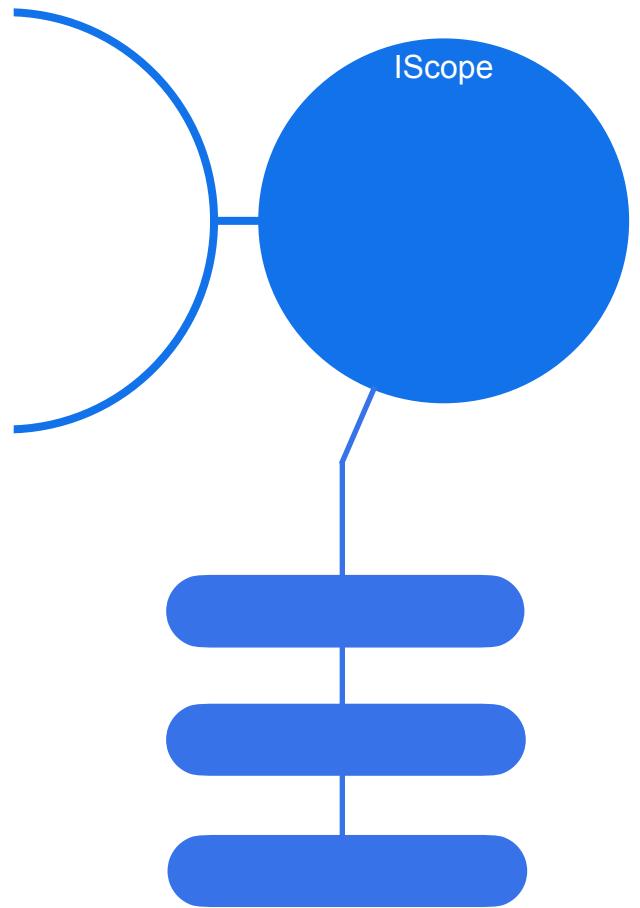
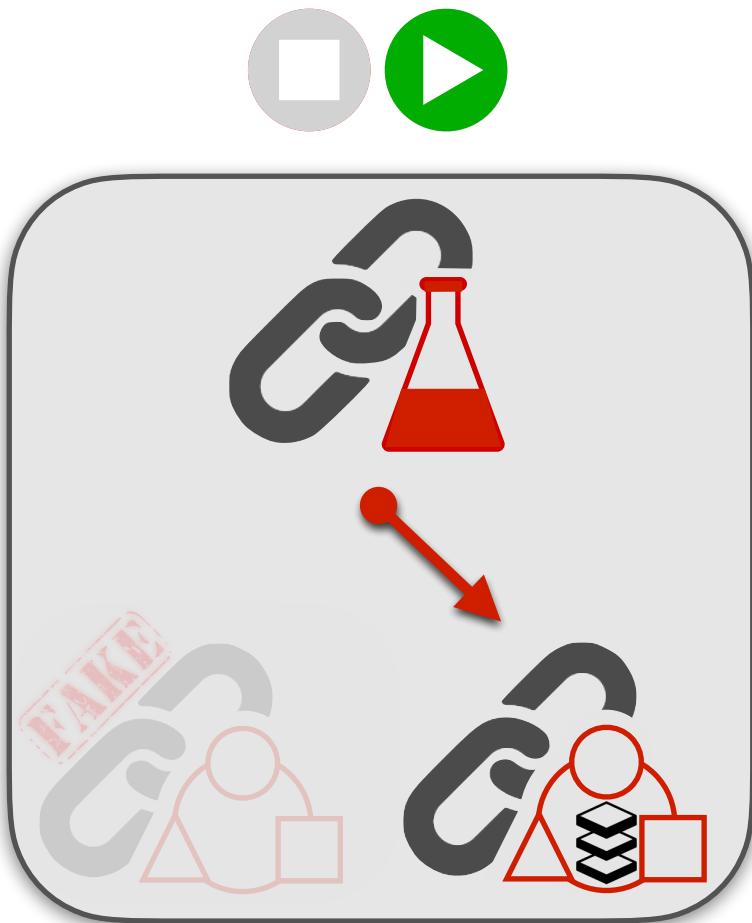


```
MyPersonList: T0bjectList<TPerson>;
```

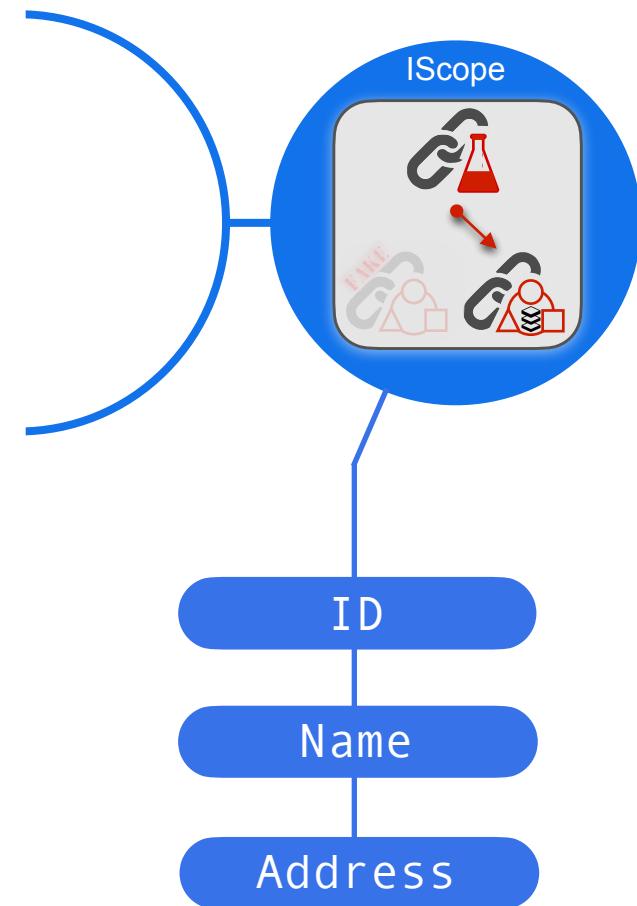
```
procedure TForm1.PBSCustCreateAdapter(Sender: T0bject, var ABindSourceAdapter:
TBindSourceAdapter);
begin
 ABindSourceAdapter := TListBindSourceAdapter<TPerson>.Create(Self, MyPersonList, True);
end;
```



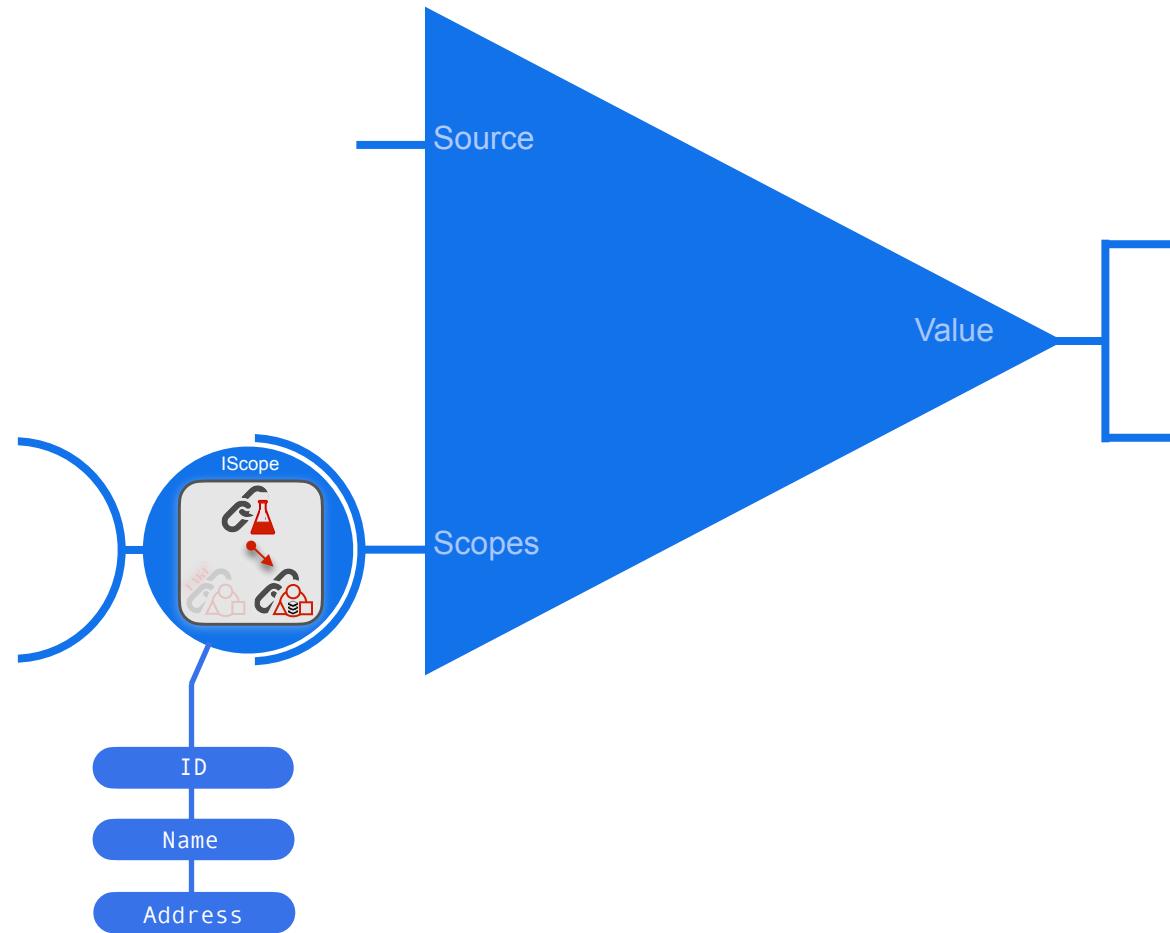
# TPROTOTYPEBINDSOURCE



# TPROTOTYPEBINDSOURCE



# TPROTOTYPEBINDSOURCE



# TBindNavigator



- L'equivalente per LiveBindings del TDBNavigator
- Si collega a un BindSource **direttamente** (*DataSource prop.*)
- Nessuna binding expression

BindNavigator1 TBindNavigator

| Properties            | Events                                   |
|-----------------------|------------------------------------------|
| Align                 | alNone                                   |
| AlignWithMargins      | <input type="checkbox"/> False           |
| Anchors               | [akLeft,akTop]                           |
| ConfirmDelete         | <input checked="" type="checkbox"/> True |
| Constraints           | (TSizeConstraints)                       |
| Ctl3D                 | <input checked="" type="checkbox"/> True |
| Cursor                | crDefault                                |
| CustomHint            |                                          |
| DataSource            | BindSourceDB1                            |
| DragCursor            | crDrag                                   |
| DragKind              | dkDrag                                   |
| DragMode              | dmManual                                 |
| Enabled               | <input checked="" type="checkbox"/> True |
| Flat                  | <input type="checkbox"/> False           |
| Height                | 25                                       |
| HelpContext           | 0                                        |
| HelpKeyword           |                                          |
| HelpType              | htContext                                |
| Hint                  |                                          |
| Hints                 | (TStrings)                               |
| Left                  | 496                                      |
| LiveBindings          | LiveBindings                             |
| LiveBindings Designer | LiveBindings Designer                    |
| Margins               | (TMargins)                               |
| Name                  | BindNavigator1                           |

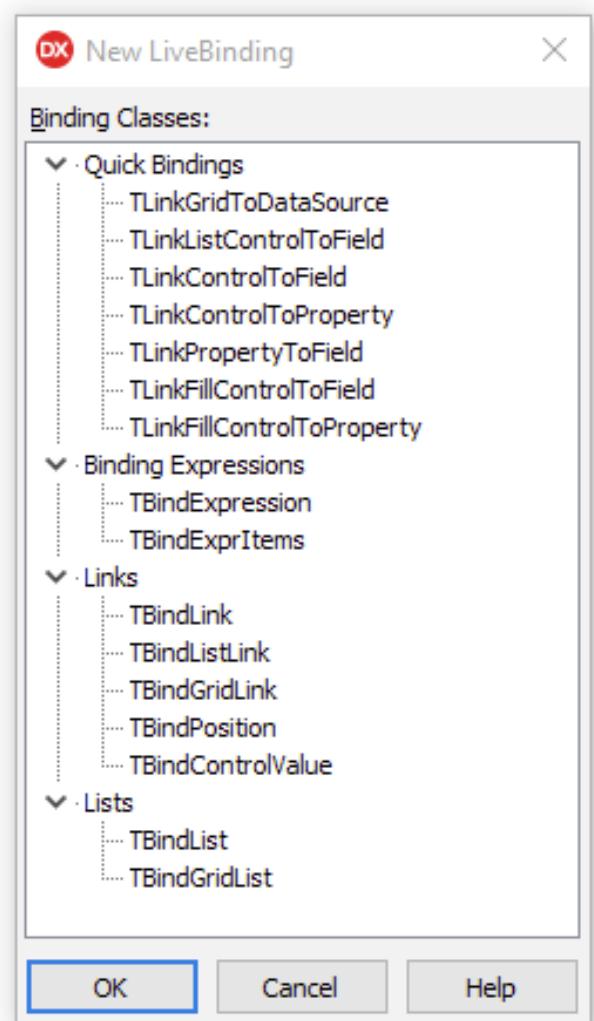
# BINDING CLASSES

## I VERI LIVEBINDINGS

# BINDING CLASSES

I VERI LIVEBINDINGS

- Binding Expressions (*TBindExpression*, *TBindExprItems*, **NOT** *TBindingExpression*)
- List LiveBindings
- Link LiveBindings
- Quick Bindings



# LIVEBINDINGS VS EXPRESSIONS

I VERI LIVEBINDINGS

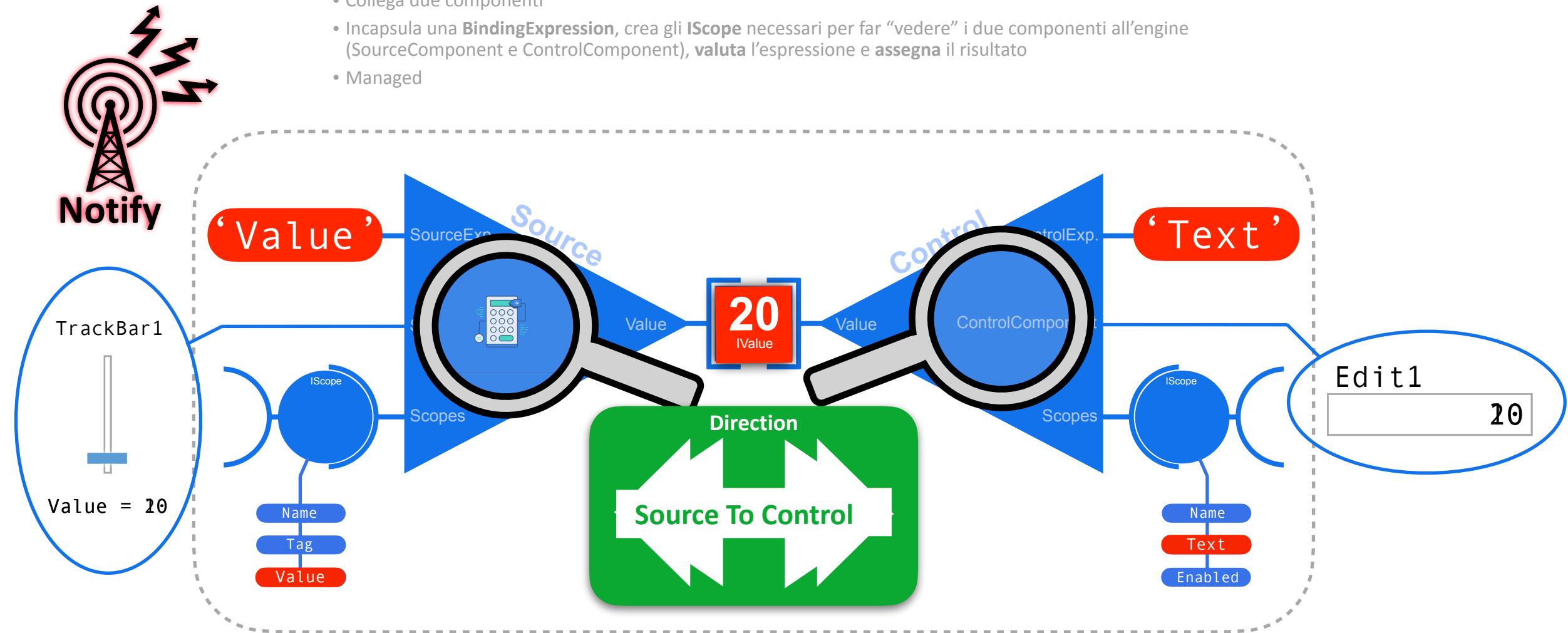
- I LiveBindings sono componenti con **una o più** espressioni
- Le espressioni sono valutate dall'expression engine, poi i LiveBindings usano il risultato per fare qualcosa (*di solito set di una proprietà*)
- Alcuni LiveBindings hanno più **collezioni** di espressioni, possono quindi avere effetto su più proprietà

# TBindExpression

- E' la più semplice delle classi LiveBindings
- Collega due componenti
- Incapsula una **BindingExpression**, crea gli **IScope** necessari per far "vedere" i due componenti all'engine (SourceComponent e ControlComponent), **valuta** l'espressione e **assegna** il risultato
- Managed

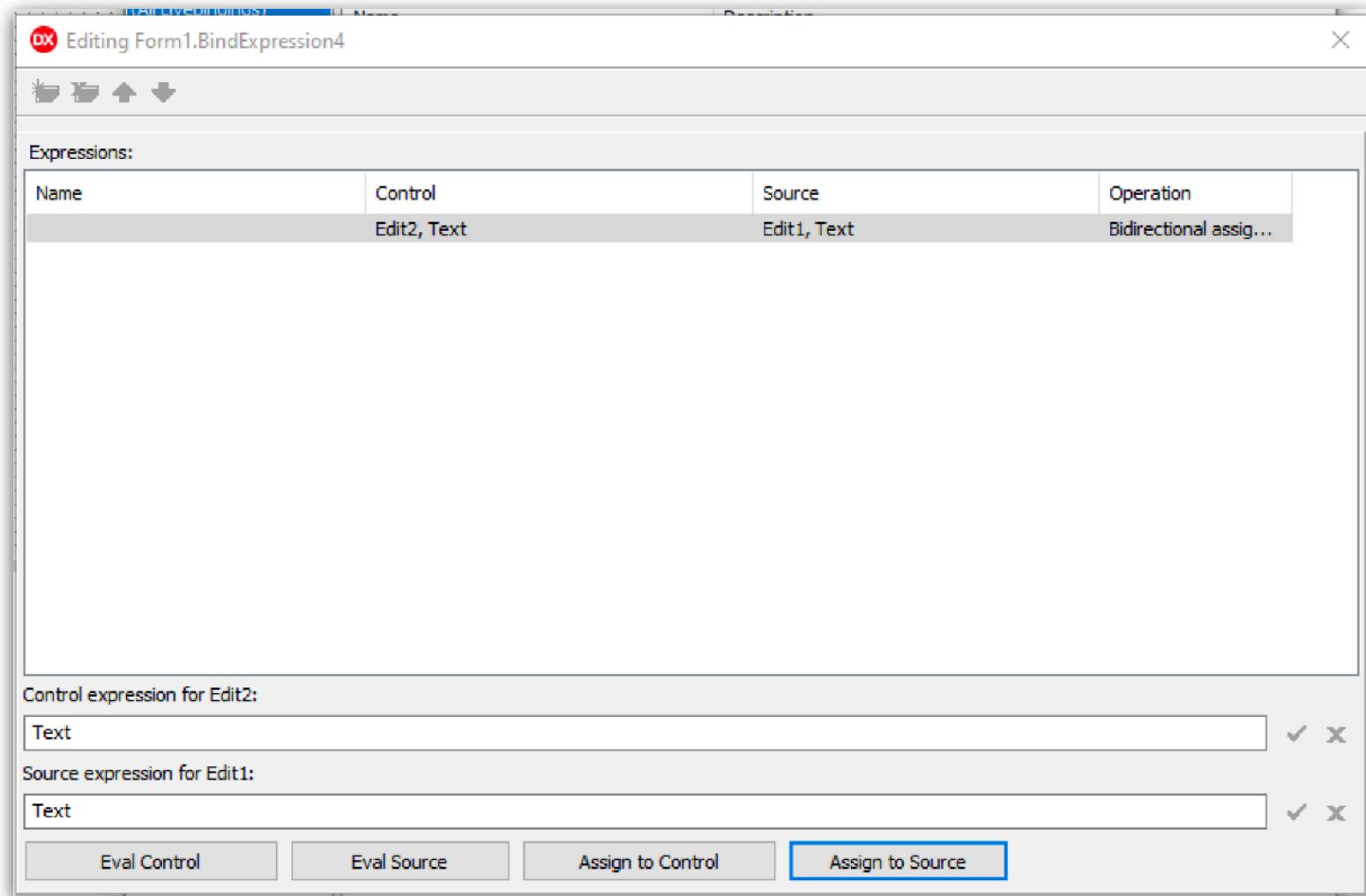
# TBINDEXPRESSION

- E' la più semplice delle classi LiveBindings
- Collega due componenti
- Incapsula una **BindingExpression**, crea gli **IScope** necessari per far "vedere" i due componenti all'engine (SourceComponent e ControlComponent), **valuta** l'espressione e **assegna** il risultato
- Managed



# EXPRESSION EDITOR

TBindExpression



# TBindExprItems

BINDING EXPRESSIONS

- Può contenere più di una BindExpression (*tipicamente da 1 a 3*)
- **2** collezioni di BindExpressions:
  - **Format collection:** (*Espressioni che assegnano un valore quando il componente è attivo*)
  - **Clear collection:** (*Espressioni che assegnano un valore quando il componente viene disabilitato*)
- Managed

# EXPRESSION EDITOR

TBINDEXPRITEMS

Editing Form1.BindExprItems1

\* X Up Down

Collections:

- (All Collections)
- Format**
- Clear

Expressions:

| Name      | Control                 | Source                                 |
|-----------|-------------------------|----------------------------------------|
| Format[0] | Edit1, Text             | TrackBar1, 'Rotation: ' +ToStr(Roun... |
| Format[1] | Edit1, DigitsOnly(Text) | TrackBar1, Value                       |

Control expression for Edit1:

DigitsOnly(Text)

Source expression for TrackBar1:

Value

Eval Control    Eval Source    Assign to Control    Assign to Source

# CUSTOM OUTPUT CONVERTER (ESEMPIO)

# LIST LIVEBINDINGS

# LIST LIVEBINDINGS

- Lavorano con componenti basati su Liste
- Unidirezionali (*source to control only*)
- Unmanaged
- Usano i BindSource
- **TBindList**
- **TBindGridList**

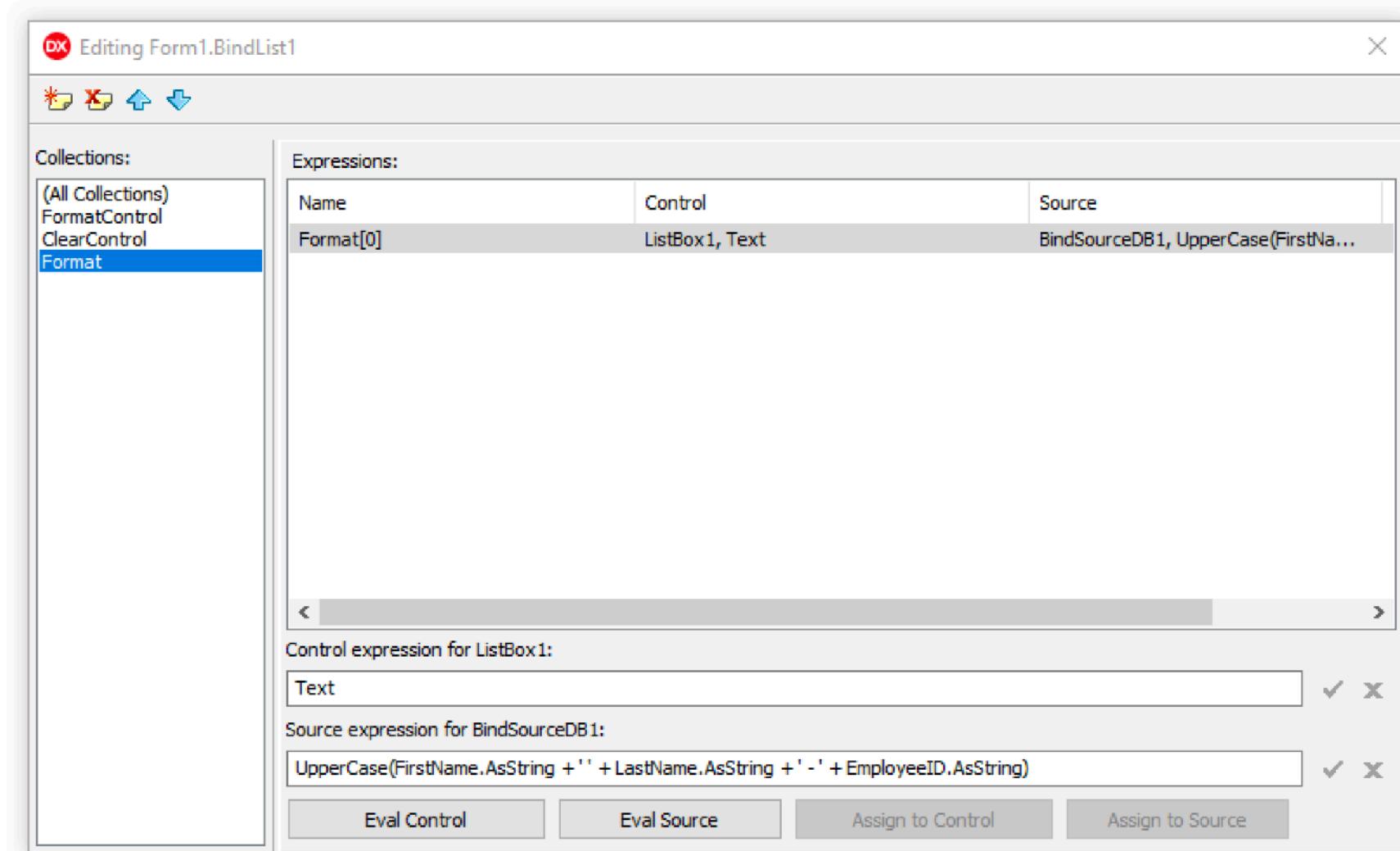


# TBINDLIST

- Target: ComboBox, ListBox, ListView
- 3 collezioni di BindExpressions:
  - **Format** (*specificamente sulla proprietà TStrings*)
  - **FormatControl** (*per il ControlComponent quando è attivo*)
  - **ClearControl** (*per il ControlComponent quando viene disattivato*)
- Può contenere molte espressioni per assegnare diverse proprietà, sia della TStrings che del ControlComponent

# EXPRESSION EDITOR

TBINDLIST



Esempio: 4

# TBINDGRIDLIST

- Target: Grids (*liste di liste*)
- Le griglie sono componenti più complessi, servono molte più espressioni
- 3 collezioni di BindExpressions:
  - **FormatControl** (*per inizializzare il ControlComponent all'attivazione*)
  - **ClearControl** (*per finalizzare il ControlComponent alla disattivazione*)
  - **Columns** (*in realtà è una collezione di collezioni di collezioni di espressioni*)
    - **ColumnName** (*FieldName*)
      - **ColFormat** (*target: colonna*)
      - **CellFormat** (*target: cella*)

# EXPRESSION EDITOR

TBINDGRIDLIST

Editing Form1.BindGridList1

Collections: Expressions:

| Name             | Control                                    | Source                   |
|------------------|--------------------------------------------|--------------------------|
| FormatControl[0] | StringGrid1, Visible                       | BindSourceDB1, True      |
| FormatControl[1] | StringGrid1, ColWidths[0]                  | BindSourceDB1, 25        |
| FormatControl[2] | StringGrid1, ColCount                      | BindSourceDB1, 4         |
| FormatControl[3] | StringGrid1, Owner.acEnableDisable.Capt... | BindSourceDB1, 'Disable' |

Control expression for StringGrid1:  
Visible

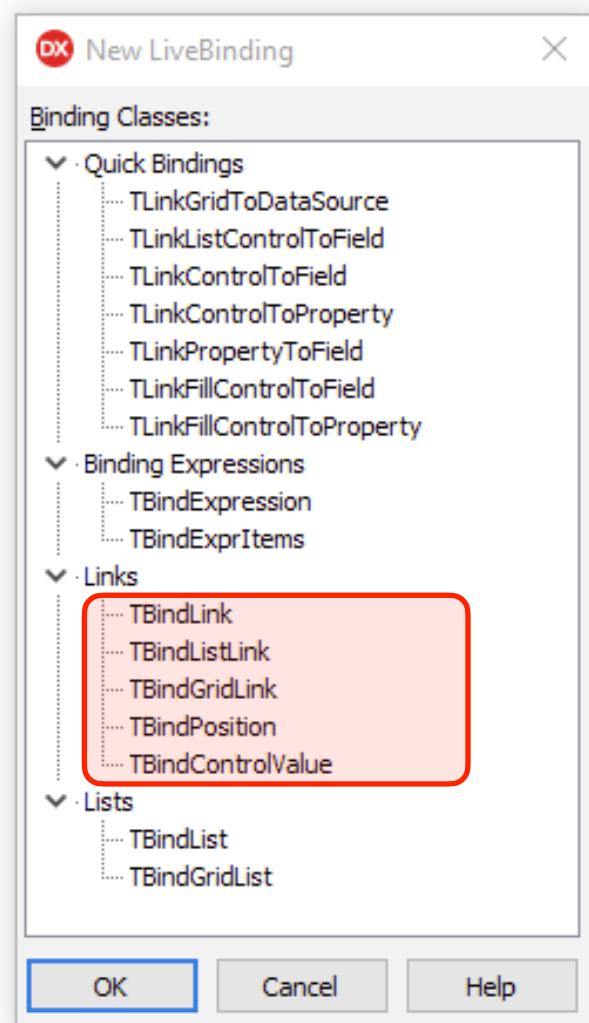
Source expression for BindSourceDB1:  
True

Esempio: 5

# LINK LIVEBINDINGS

# LINK LIVEBINDINGS

- Come i ListLiveBindings usano i BindSource
- Bidirezionali (*read/write*)
- Unmanaged
- Rendono possibile un comportamento simil-DBAware  
(*ma con qualunque componente*)
- 4 componenti:
  - **TBindLink**
  - **TBindPosition**
  - **TBindListLink** (*simile a TBindList*)
  - **TBindGridLink** (*simile a TBindGridList*)



# TBINDLINK

- Collega un controllo single-field (*TEdit*) con un altro componente (*tipicamente un BindSource*)
- Più specificamente: possono collegare controlli che implementano le interfacce **IEditLinkObserver** oppure **IEditGridLinkObserver**
- **3 collezioni di BindExpressions:**
  - **Format collection:** (*Espressioni che assegnano un valore quando il componente è attivo*)
  - **Clear collection:** (*Espressioni che assegnano un valore quando il componente viene disabilitato*)
  - **Parse collection:** (*Espressioni che assegnano un valore al SourceComponent, ControlToSource*)

# EXPRESSION EDITOR

TBINDLINK

Editing Form1.BindLink1

(All Collections) Format Parse Clear

| Name      | Control      | Source                             | Operation         |
|-----------|--------------|------------------------------------|-------------------|
| Format[0] | EditID, Text | BindSourceDB1, CustomerID,AsString | Assign to control |
| Parse[0]  | EditID, Text | BindSourceDB1, CustomerID,AsString | Assign to source  |
| Clear[0]  | EditID, Text | , 'No data to display'             | Assign to control |

Control expression for EditID:  
Text

Source expression for BindSourceDB1, CustomerID:  
AsString

Eval Control    Eval Source    Assign to Control    Assign to Source

# TBindPosition

- E' il più limitato dei 4 Link LiveBindings
- Solo per componenti che “esprimono” un concetto di **posizione relativa**
- I componenti devono implementare l’interfaccia **IPositionObserver**
- **3** collezioni di BindExpressions:
  - **PosControl:** (*assegnano una posizione relativa al ControlComponent basandosi su una proprietà ordinal del SourceComponent - Es. TStringGrid, TScrollBar*)
  - **PosClear collection:** (*Finalizzano il ControlComponent in una posizione “neutrale” alla disattivazione del LiveBinding*)
  - **PosSource collection:** (*come la PosControl ma ControlToSource*)
- E' l'asterisco del LiveBindingDesigner, mantiene sincronizzate le posizioni relative del SourceComponente e del ControlComponent

# EXPRESSION EDITOR

TBINDPOSITION

Editing Form1.BindPosition1

Collections:

- (All Collections)
- PosControl
- PosSource
- PosClear

Expressions:

| Name          | Control                  | Source                              | Operation         |
|---------------|--------------------------|-------------------------------------|-------------------|
| PosControl[0] | ScrollBar1, Position     | BindSourceDB1, Math_Max(0, RecN...) | Assign to control |
| PosControl[1] | ScrollBar1, Max          | BindSourceDB1, RecordCount - 1      | Assign to control |
| PosSource[0]  | ScrollBar1, Position + 1 | BindSourceDB1, RecNo                | Assign to source  |

Control expression for ScrollBar1:

 ✓ ✕

Source expression for BindSourceDB1:

 ✓ ✕

# TBINDLISTLINK

- Simile al TBindList ma bidirezionale
- La bidirezionalità comporta le necessità di ulteriori collezioni di espressioni
- **6** collezioni di BindExpressions:
  - **FormatControl** collection
  - **ClearControl** collection
  - **Format** collection (*specificamente per la proprietà TStrings*)
  - **Parse** collection (*come Format ma ControlToSource*)
  - **PosControl** collection (*sincronizzazione posizione SourceToControl*)
  - **PosSource** collection (*come PosControl ma ControlToSource*)

# EXPRESSION EDITOR

TBINDLISTLINK

Editing Form1.BindListLink1

(All Collections) FormatControl ClearControl Format PosControl PosSource Parse

| Name          | Control               | Source                             | Operation         |
|---------------|-----------------------|------------------------------------|-------------------|
| Format[0]     | ListBox1, Text        | BindSourceDB1, CompanyName.AsSt... | Assign to control |
| PosControl[0] | ListBox1, ItemIndex   | BindSourceDB1, RecNo-1             | Assign to control |
| PosSource[0]  | ListBox1, ItemIndex+1 | BindSourceDB1, RecNo               | Assign to source  |

Control expression for ListBox1:  
Text

Source expression for BindSourceDB1:  
CompanyNameAsString + ', ' + CityAsString

Eval Control    Eval Source    Assign to Control    Assign to Source

# TBindGridLINK

- Simile al TBindGridList ma bidirezionale
- E' il più complesso dei LinkLiveBindings (*ma con le conoscenze acquisite dovrebbe essere facile capire*)
- 5 collezioni di BindExpressions:
  - **FormatControl** collection
  - **ClearControl** collection
  - **PosControl** collection (*sincronizzazione posizione SourceToControl*)
  - **PosSource** collection (*come PosControl ma ControlToSource*)
  - **Columns** (*in realtà è una collezione di collezioni di collezioni di espressioni*)
    - **ColumnName** (*FieldName*)
      - **ColFormat** (*target: colonna*)
      - **CellFormat** (*target: cella*)
      - **CellParse** (*come CellFormat ma ControlToSource*)

# EXPRESSION EDITOR

TBINDGRIDLINK

DX Editing Form1.BindGridLink1

Collections: Expressions:

| Name                                      | Control                         | Source                                        | Operation         |
|-------------------------------------------|---------------------------------|-----------------------------------------------|-------------------|
| Columns.RecNo.ColFormat[0]                | StringGrid1, cells[0,0]         | BindSourceDB1, 'RecNo'                        | Assign to control |
| Columns.RecNo.CellFormat[0]               | StringGrid1, cells[0]           | BindSourceDB1,ToStr(RecNo)                    | Assign to control |
| Columns.CompanyName.ColFormat[0]          | StringGrid1, cells[1,0]         | BindSourceDB1, CompanyName, 'Company'         | Assign to control |
| Columns.CompanyName.CellFormat[0]         | StringGrid1, cells[1]           | BindSourceDB1, CompanyName,AsString           | Assign to control |
| Columns.CompanyName.CellParse[0]          | StringGrid1, SelectedText(Self) | BindSourceDB1, CompanyName,AsString           | Assign to source  |
| Columns.ContactTitleAndName.ColFormat[0]  | StringGrid1, cells[2,0]         | BindSourceDB1, 'Contact'                      | Assign to control |
| Columns.ContactTitleAndName.CellFormat[0] | StringGrid1, cells[2]           | BindSourceDB1, ContactTitleAsString + ' - ' + | Assign to control |
| Columns.City.ColFormat[0]                 | StringGrid1, cells[3,0]         | BindSourceDB1, City, 'City'                   | Assign to control |
| Columns.City.CellFormat[0]                | StringGrid1, cells[3]           | BindSourceDB1, City,AsString                  | Assign to control |
| Columns.City.CellParse[0]                 | StringGrid1, SelectedText(Self) | BindSourceDB1, City,AsString                  | Assign to source  |
| Columns.Country.ColFormat[0]              | StringGrid1, cells[4,0]         | BindSourceDB1, Country, 'Country'             | Assign to control |
| Columns.Country.CellFormat[0]             | StringGrid1, cells[4]           | BindSourceDB1, Country,AsString               | Assign to control |
| Columns.Country.CellParse[0]              | StringGrid1, SelectedText(Self) | BindSourceDB1, Country,AsString               | Assign to source  |
| ClearControl[0]                           | StringGrid1, RowCount           | BindSourceDB1, 0                              | Assign to control |
| PosControl[0]                             | StringGrid1, Row                | BindSourceDB1, RecNo                          | Assign to control |
| PosSource[0]                              | StringGrid1, Row                | BindSourceDB1, RecNo                          | Assign to source  |

< >

Control expression for StringGrid1:  
cells[0,0]

Source expression for BindSourceDB1:  
'RecNo'

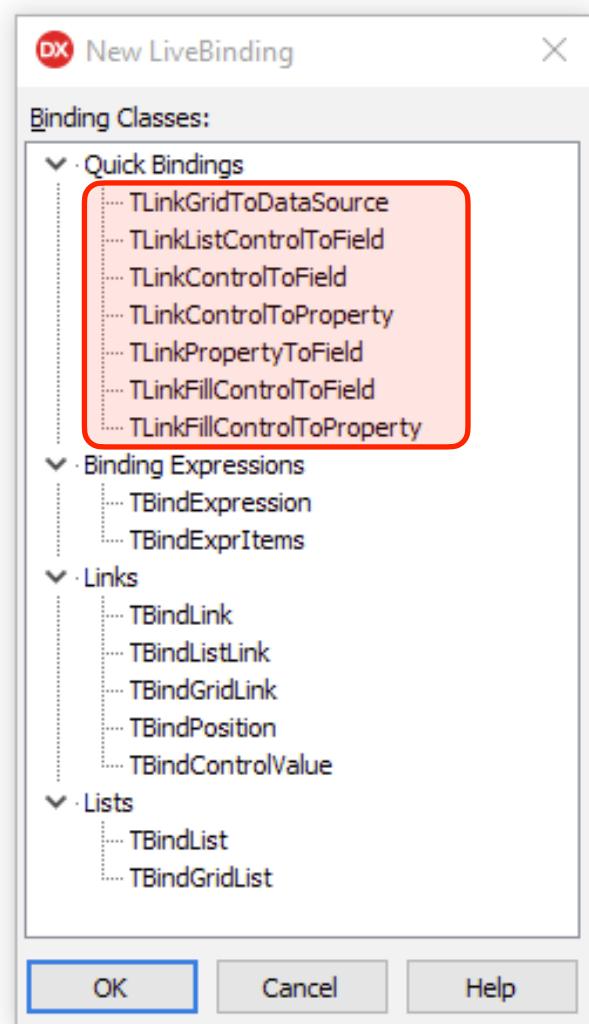
# SELECTEDTEXT METHOD

- Es: **SelectedText(Self)**
- Visto nelle BindExpression delle collezioni **CellParse** (*TBindGridLink*)
- Estrarre il dato editato da un controllo
- Il controllo è specificato dal parametro
- **Self** si riferisce al controllo nel contesto dell'espressione
- Nel caso di una StringGrid si riferisce all'**InPlaceControl** della cella corrente che non sarebbe altrimenti raggiungibile

# QUICK BINDINGS

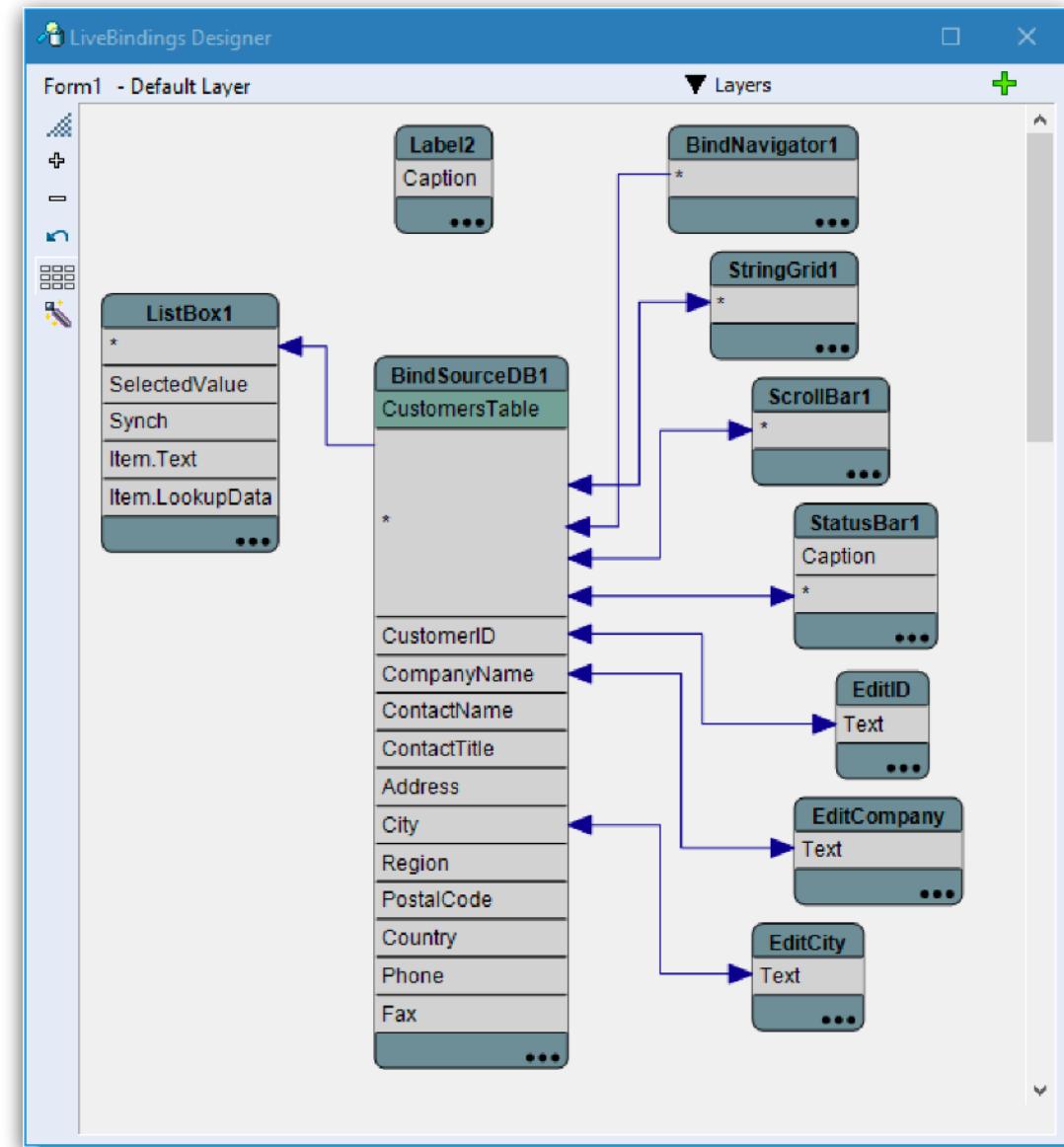
# QUICK BINDINGS

- Incapsulamento di tutte quelle espressioni che scriveremmo nel 95% dei casi
- Espressioni generate automaticamente ma **ReadOnly**
- Il più delle volte collegano:
  - un campo di una sorgente dati a un componente e viceversa (Edit, ComboBox, ListBox, Grids...)
  - una proprietà di un oggetto a un componente e viceversa (Edit, ComboBox, ListBox, Grids...)
  - un componente a un altro componente
- In questi casi userò i Quick Bindings, negli altri (5%)...
- Creati automaticamente dal **LiveBinding Designer** (Visual LiveBindings) o dai **Wizard**
- Rendono comodo e semplice l'uso dei LiveBindings
- Nascondono molta della complessità sottostante



# LIVEBINDINGS DESIGNER

- Capisce quale QuickBinding creare automaticamente
- **Rende** comodo e **semplice** l'uso dei LiveBindings
- **Nasconde** molta della **complessità** sottostante



Esempio: 7

# CUSTOMFORMAT- CUSTOMPARSE

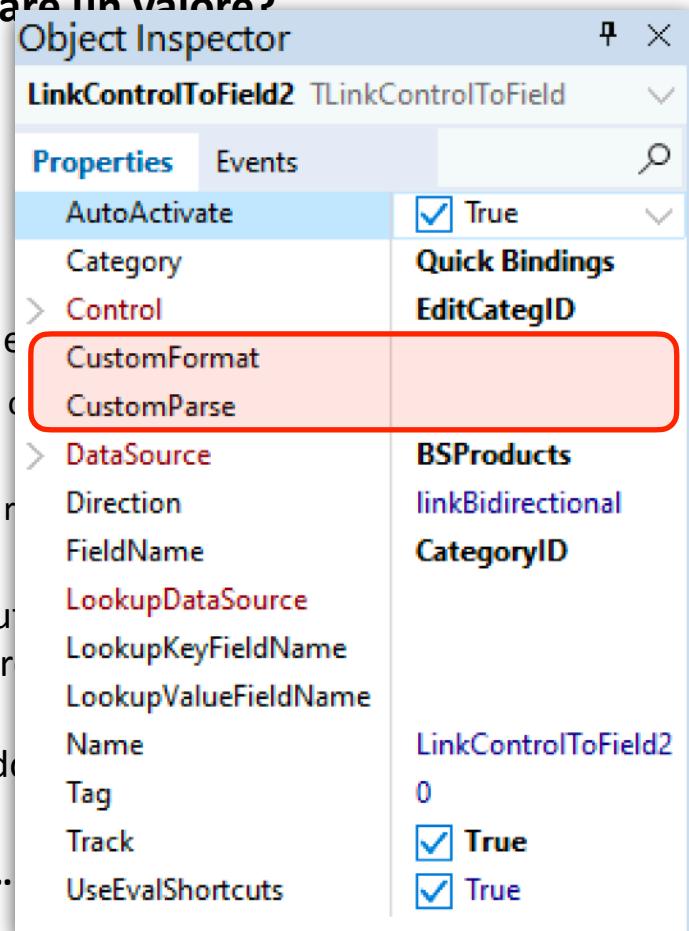
**Nei QuickBindings le espressioni sono ReadOnly, come posso formattare un valore?**

**Se uso TBindSourceDB:**

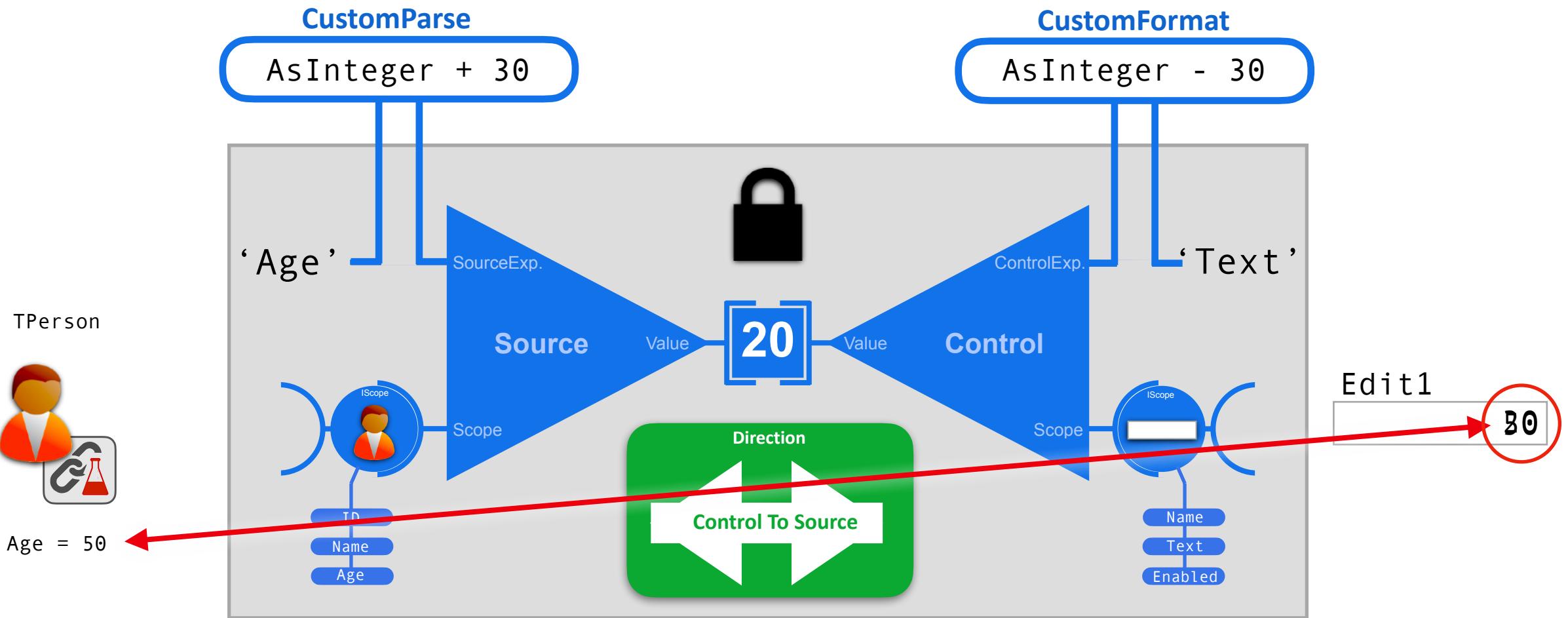
- Posso continuare a usare proprietà come “DisplayFormat” del campo del DataSet

**Altrimenti:**

- Le proprietà CustomFormat e CustomParse permettono di formattare il dato ai due estremi della valutazione dell'espressione.
- **CustomFormat** formatta/modifica il risultato della valutazione dell'espressione prima che venga restituito al ControlComponent (*SourceToControl*)
- **CustomParse** formatta/modifica il dato letto dal ControlComponent prima che venga restituito alla SourceControl (*ControlToSource*)
- **%s** rappresenta il risultato della SourceExpression (*stringa*) che l'engine ha già valutato e passato alla CustomFormat che viene pre-parsata e poi eseguita come ControlExpression. *Esempio: UpperCase(%s)*
- Eventuali altri caratteri “%” usati in funzioni come la “Format” devono essere raddoppiati. *Esempio: Format('%%.2f%%%%%', AsFloat) diventa Format('.2f%%%', 123.5) produce 123,50%*
- Se **%s** è un numero e va formattato (*es: currency*) usare **Self.AsFloat** , **Self.AsInteger** o **Self.AsString**. *Esempio: Format('%%m', Self.AsFloat)*



# CUSTOMFORMAT- CUSTOMPARSE



Esempio: 8

# TPrototypeBindSource MASTER-DETAIL (ESEMPIO)

# LIVEBINDINGS ACTIONS

# LIVEBINDINGS ACTIONS

- Actions che replicano singolarmente le funzionalità del TBindNavigator
  - First
  - Last
  - Prior
  - Next
  - Edit
  - Insert
  - Post
  - Cancel
  - ...

# I-ORM

# I-ORM

- Interfaced O.R.M.
- Dependency Injection Container
- MVVM framework
- Deep serializer/deserializer (*DJSON*)
- LiveBinding Extensions
  - **ActiveBindSourceAdapters** (*TActiveInterfaceListBindSourceAdapter, TActiveInterfaceObjectBindSourceAdapter, TActiveListBindSourceAdapter, TActiveObjectBindSourceAdapter, TNaturalBindSourceAdapter*)
  - **TioPrototypeBindSource**
  - **TioModelPresenter** (*MVVM*)
  - **TioModelBindSource** (*MVVM*)
  - **TioModelDataSet** (*MVVM*)

# TIOPrototypeBindSource

- In grado di effettuare le chiamate all'ORM autonomamente (*facoltativo*)
- Master-Detail
- AutoLoadData
- AutoPersist
- AutoPost (*funziona*)
- AutoRefresh
- **Selectors**

# DOMANDE?





Maurizio Del Magno  
Developer



Lev@nte software



**i-ORM** [github.com/mauriziodm/iORM](https://github.com/mauriziodm/iORM)  
**DJSON** [github.com/mauriziodm/DJSON](https://github.com/mauriziodm/DJSON)



[mauriziodm@levantesw.it](mailto:mauriziodm@levantesw.it)  
[mauriziodelmagno@gmail.com](mailto:mauriziodelmagno@gmail.com)



[facebook.com/maurizio.delmagno](https://facebook.com/maurizio.delmagno)  
iORM + DJSON (group)



Membro fondatore  
**eInvoice4D**  
<https://github.com/delphiforce/eInvoice4D>



XIX Edizione, 2020

Grazie!!!



Maurizio Del Magno  
Developer



Lev@nte software



**i-ORM** [github.com/mauriziodm/iORM](https://github.com/mauriziodm/iORM)  
**DJSON** [github.com/mauriziodm/DJSON](https://github.com/mauriziodm/DJSON)



[mauriziodm@levantesw.it](mailto:mauriziodm@levantesw.it)  
[mauriziodelmagno@gmail.com](mailto:mauriziodelmagno@gmail.com)



[facebook.com/maurizio.delmagno](https://facebook.com/maurizio.delmagno)  
iORM + DJSON (group)



Membro fondatore  
**eInvoice4D**  
<https://github.com/delphiforce/eInvoice4D>

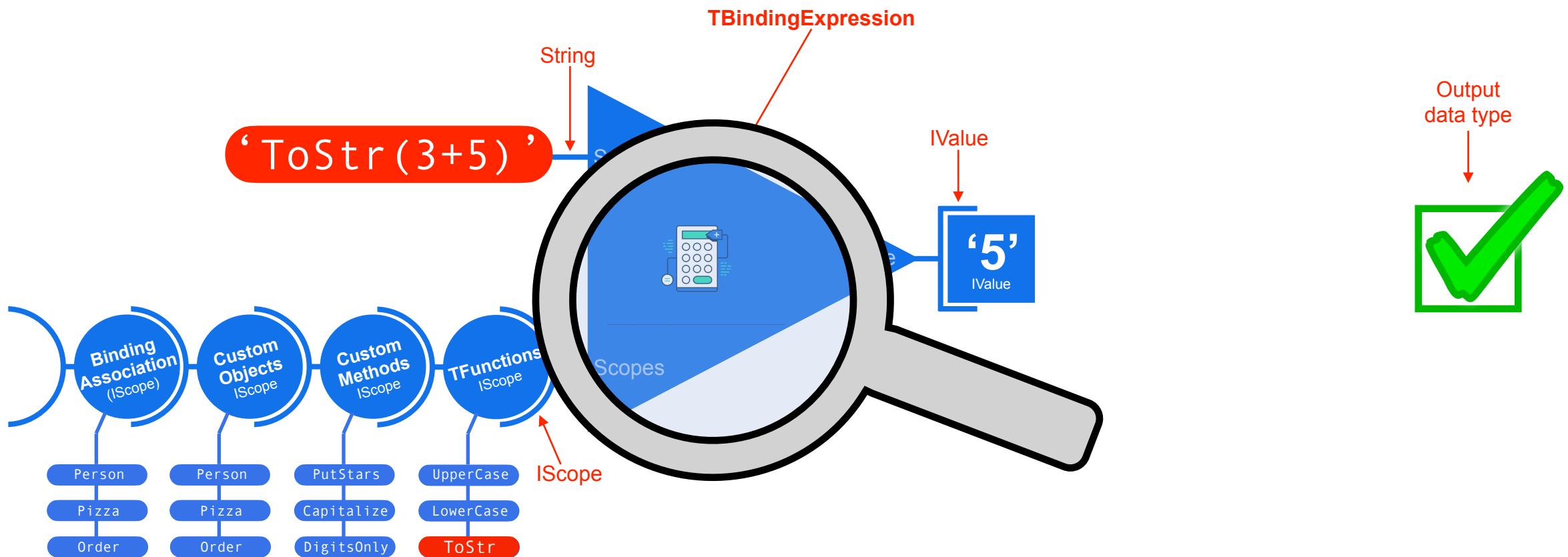


XIX Edizione, 2020

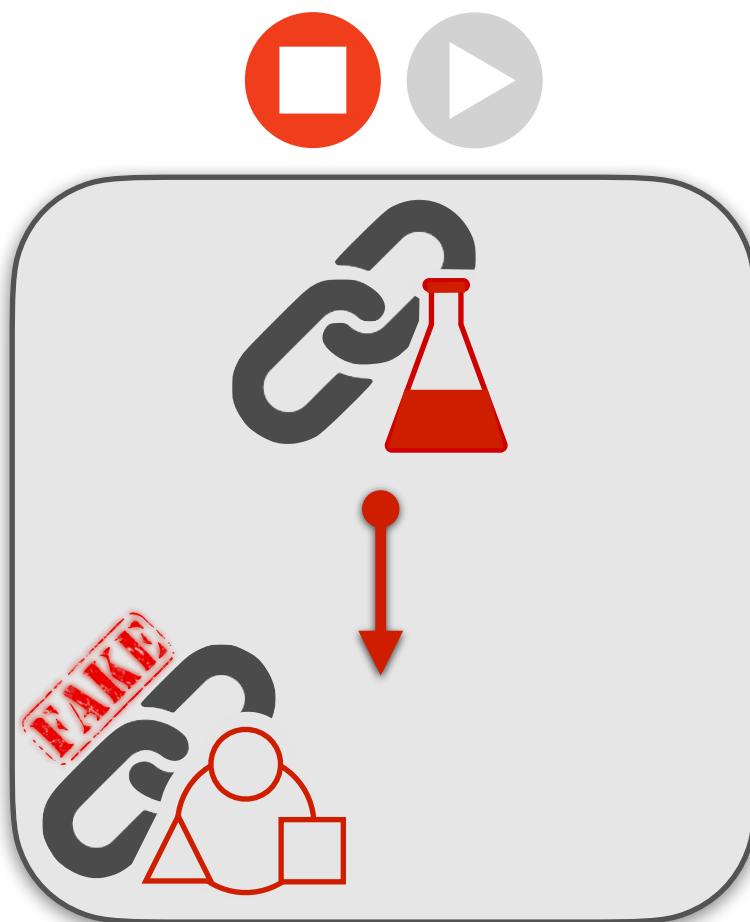
Grazie!!!

# TITLE

## ISCOPE



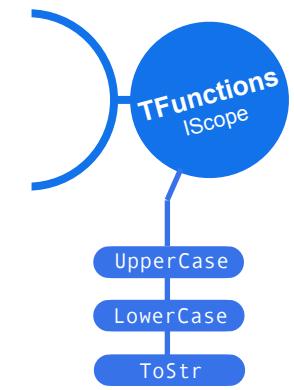
# TPROTOTYPEBINDSOURCE



TBindSourceAdapter

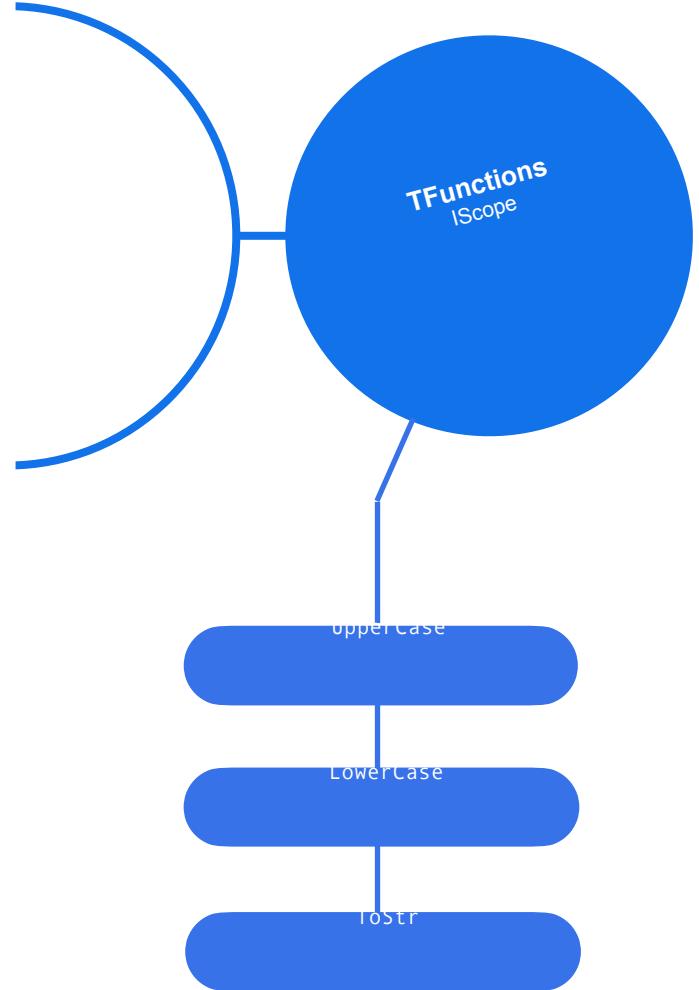
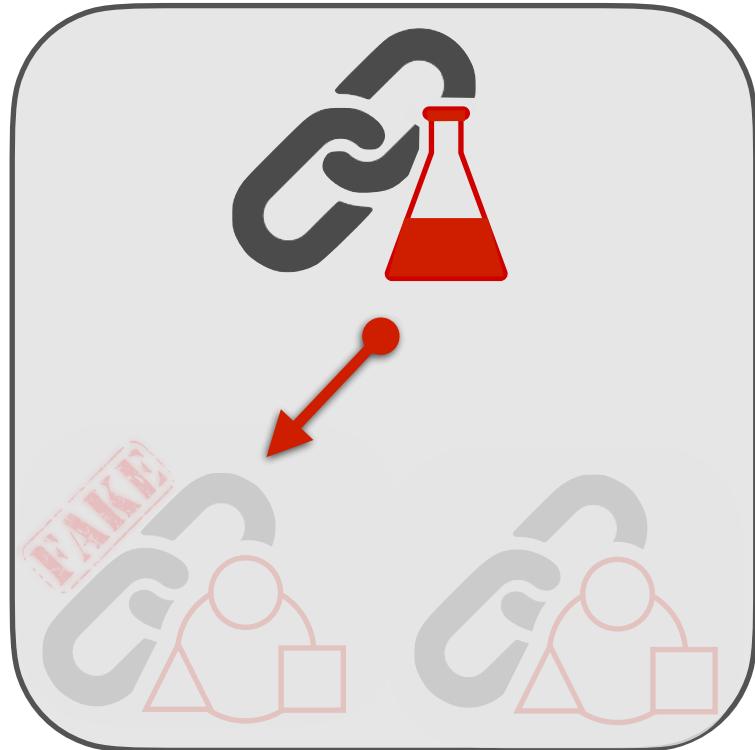


- Permette la prototipazione della UI con dati fake
- Devo definire campi (*fields editor*)
- A runtime dovrò fornire una fonte di dati reali che sostituirà il generatore di fake data (evento **OnCreateAdapter**) (*RT*)
- Sviluppo di un prototipo di UI senza avere ancora nemmeno pensato alla fonte dati (*DB?*, *Objects?*) anche prima, o contemporaneamente, lo sviluppo della logica di dominio e/o del DB
- Una fonte di dati
- Adatta un oggetto (o una lista di oggetti) per essere poi assegnato a un TPrototypeBindSource come fonte di dati
- **Reale OOP** senza perdere le caratteristiche **RAD** di Delphi
- Ruolo simile al TDataSet (*State, Edit, Post, Append, Cancel...*)
- **Post su oggetti non su DB** (*anche Delete, Append...*)
- TObjectBindSourceAdapter/TListBindSourceAdapter



# PROVA

Xxx

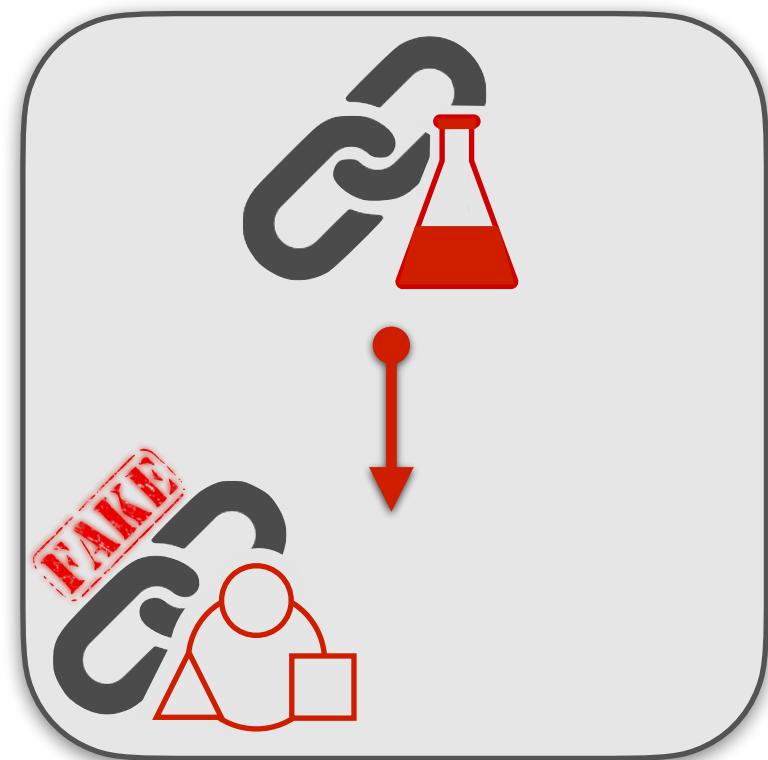


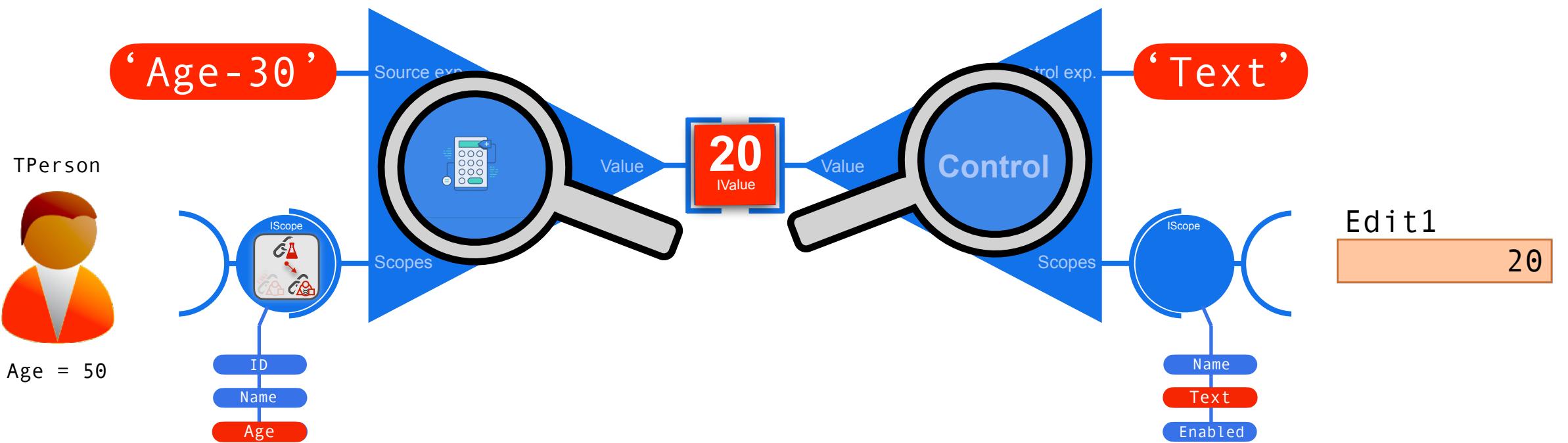
# TPROTOTYPEBINDSOURCE

- Permette la prototipazione della UI con dati fake
- Devo definire campi (*fields editor*)
- A runtime dovrò fornire una fonte di dati reali che sostituirà il generatore di fake data (evento **OnCreateAdapter**) (*RT*)
- Sviluppo di un prototipo di UI senza avere ancora nemmeno pensato alla fonte dati (*DB?*, *Objects?*) anche prima, o contemporaneamente, lo sviluppo della logica di dominio e/o del DB

## TBindSourceAdapter

- Una fonte di dati
- Adatta un oggetto (o una lista di oggetti) per essere poi assegnato a un TPrototypeBindSource come fonte di dati
- **Reale OOP** senza perdere le caratteristiche **RAD** di Delphi
- Ruolo simile al TDataSet (*State, Edit, Post, Append, Cancel...*)
- **Post su oggetti non su DB** (*anche Delete, Append...*)
- TObjectBindSourceAdapter/  
TListBindSourceAdapter





# TBINDEXPRESSION

