TMS FNC Object Inspector

DEVELOPERS GUIDE

January 2022

Index

Getting Started 3
Availability 3
Hierarchy 4
Assigning an object 4
Reading / Writing properties 6
Collection / list editor 8
Datasource 9

Getting Started

You can already start using the TTMSFNCObjectInspector by dropping an instance on the form and assign a value to the Object property (public). The object inspector will read the properties and display them inside the list represented with a name and value column.

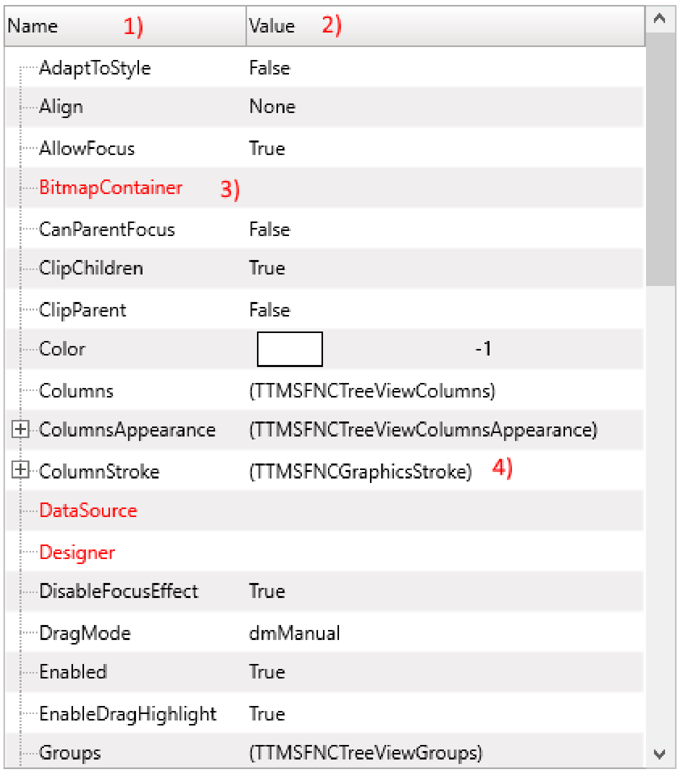
The TTMSFNCObjectInspector inherits from TTMSFNCTreeView, please see the TTMSFNCTreeView documentation for more information.

Availability

Supported frameworks and platforms
- VCL Win32/Win64
- FMX Win32/Win64, MacOS-X
- LCL Win32/Win64, Linux

Supported IDE's
- Delphi XE7 and C++ Builder XE7 or newer releases
- Lazarus 1.4.4 with FPC 2.6.4 or newer official releases.

Hierarchy



Name	Value
AdaptToStyle	False
Align	None
AllowFocus	True
BitmapContainer	
CanParentFocus	False
ClipChildren	True
ClipParent	False
Color	⬜ -1
Columns	(TTMSFNCTreeViewColumns)
⊞ ColumnsAppearance	(TTMSFNCTreeViewColumnsAppearance)
⊞ ColumnStroke	(TTMSFNCGraphicsStroke)
DataSource	
Designer	
DisableFocusEffect	True
DragMode	dmManual
Enabled	True
EnableDragHighlight	True
Groups	(TTMSFNCTreeViewGroups)

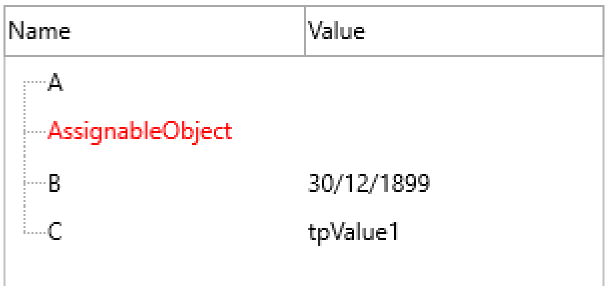
1) The Name column, showing the name of the property
2) The Value column, showing the value / class name of the property
3) An assignable property, shown in red. Can be used to assign an object to a property of a specific class. Typically used for non-visual components, such as TPopupMenu or TTMSFNCBitmapContainer.
4) An expandable property. The object inspector lists the full hierarchy of the object.

Assigning an object

The TTMSFNCObjectInspector not only supports visual and non-visual components but supports any kind of object. You can easily create your own class and pass this as an object to the object inspector. To do so, create a class inside your unit. For this purpose, we have created a TTestClass with a set of properties of different types.

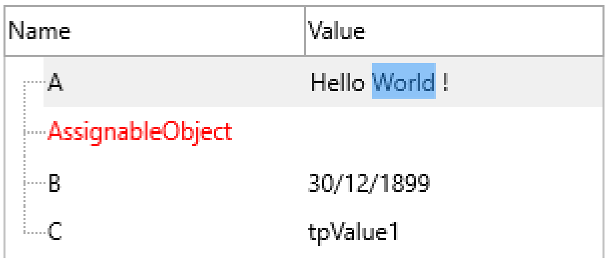
```
type
  TTestProperty = (tpValue1, tpValue2, tpValue3);

  TTestClass = class
  private
    FB: TDate;
    FC: TTestProperty;
    FA: string;
    FObject: TComponent;
  published
    property A: string read FA write FA;
    property B: TDate read FB write FB;
    property C: TTestProperty read FC write FC;
    property AssignableObject: TComponent read FObject write FObject;
  end;
```



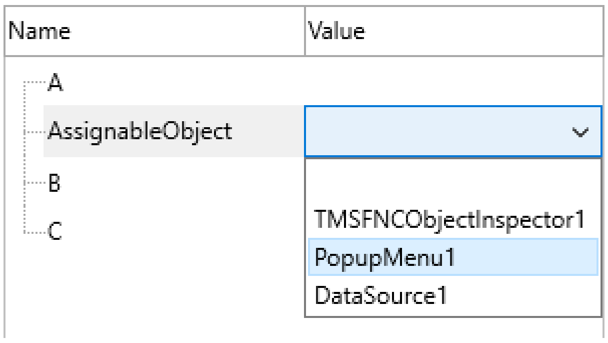
Name	Value
A	
AssignableObject	
B	30/12/1899
C	tpValue1

The property A is a string property that simply shows an editable field with plain text.



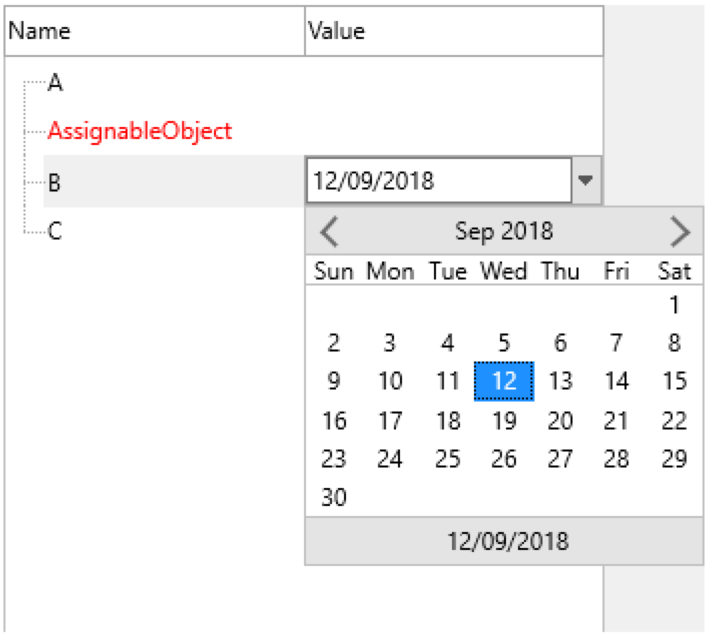
Name	Value
A	Hello World !
AssignableObject	
B	30/12/1899
C	tpValue1

The AssignableObject property is colored in red, which indicates it can be assigned with an instance of an object that is found on the main form, of that specific class type.

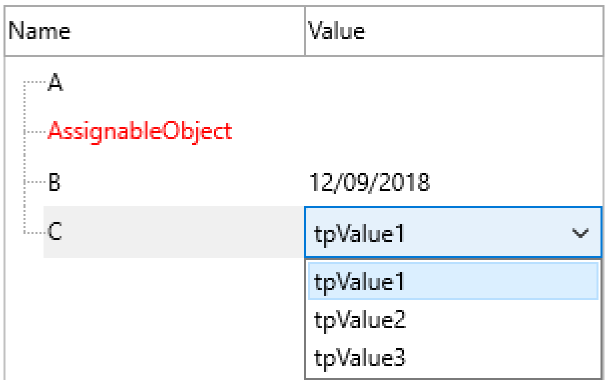


Name	Value
A	
AssignableObject	
B	
C	

The property B is a TDate property that can show an inplace date picker editor.



The property C is a propert of type TTestProperty, which shows a combobox with the possible values.

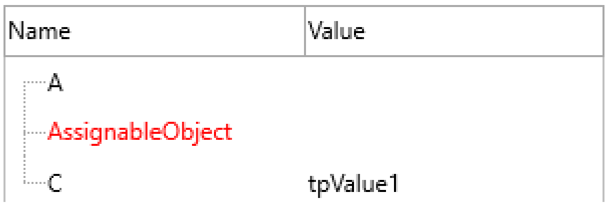


Reading / Writing properties

With the above sample, the values that are present on the object are all listed inside the object inspector. Sometimes, it might be necessary to hide values from the user, or change the value that is writing to the object property, therefore the object inspector exposes a set of events for reading.

The event OnReadProperty can be used to hide or show the value inside the object inspector. For example, if we want to hide property B, we can write the following code:

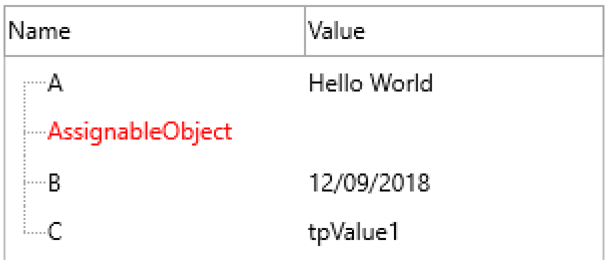
```
procedure TForm1.TMSFNCObjectInspector1ReadProperty(Sender, AObject: TObject;
  APropertyInfo: PPropInfo; APropertyName: string; APropertyType: TTypeKind;
  var ACanRead: Boolean);
begin
  ACanRead := (APropertyName <> 'B');
end;
```



Name	Value
A	
AssignableObject	
C	tpValue1

If you want to display the property, but want to change the value instead, you can use the OnReadPropertyValue event. Below is a sample that shows how to change the default value shown in the object inspector. This can be helpful to predefine a value for a specific property, or manipulate the value that is being read from the property itself.

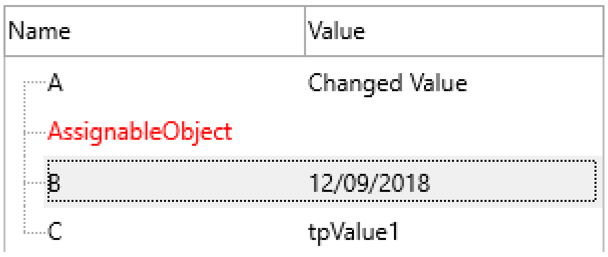
```
procedure TForm1.TMSFNCObjectInspector1ReadPropertyValue(Sender,
  AObject: TObject; APropertyInfo: PPropInfo; APropertyName: string;
  APropertyType: TTypeKind; var APropertyValue: string; var ACanRead: Boolean);
begin
  if APropertyName = 'A' then
    APropertyValue := 'Hello World';
end;
```



Name	Value
A	Hello World
AssignableObject	
B	12/09/2018
C	tpValue1

Whenever a property value is being changed, the OnWritePropertyValue is being triggered. This event can be used to change the value, or prevent to value from being written to the property. Below is a sample that demonstrates how to change the value being written to the property.

```
procedure TForm1.TMSFNCObjectInspector1WritePropertyValue(Sender,
  AObject: TObject; APropertyInfo: PPropInfo; APropertyName: string;
  APropertyType: TTypeKind; var APropertyValue: string; var ACanWrite: Boolean);
begin
  if APropertyName = 'A' then
    APropertyValue := 'Changed Value';
end;
```



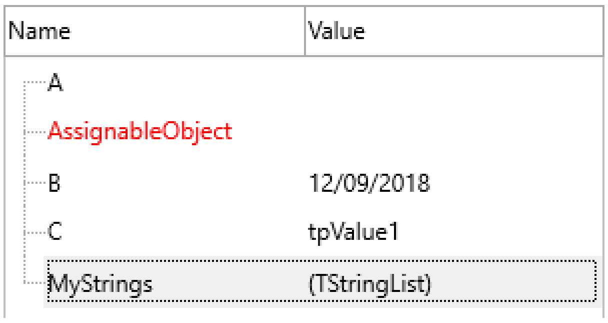
Name	Value
A	Changed Value
AssignableObject	
B	12/09/2018
C	tpValue1

Collection / list editor

Whenever a collection / list is detected, the object inspector can show the collection or list inside a popup editor. Below is a sample when we add a TStringList to TTestClass.

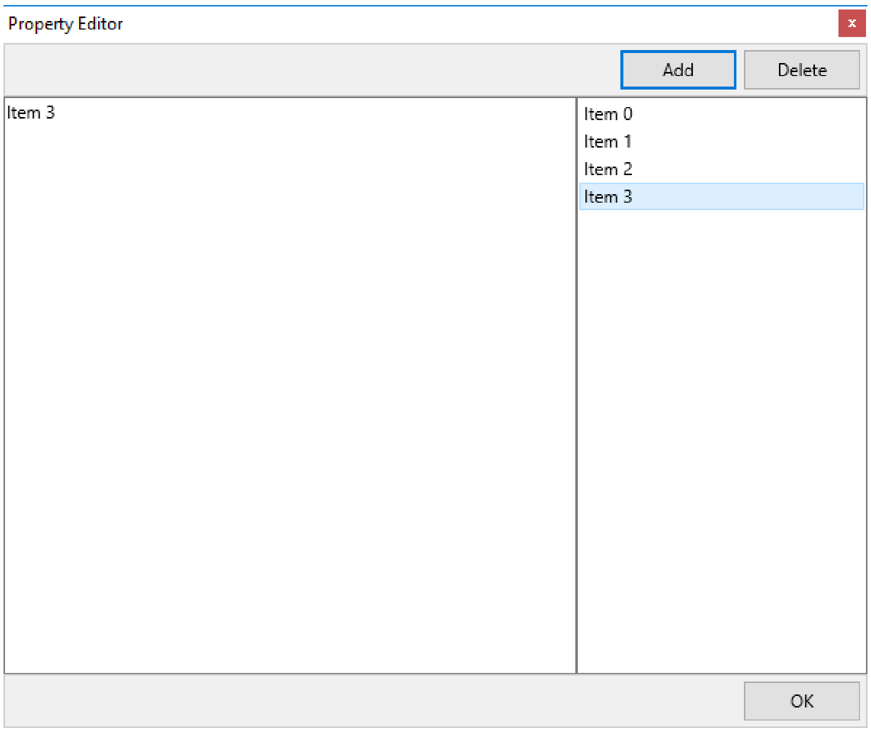
```
type
  TTestProperty = (tpValue1, tpValue2, tpValue3);

  TTestClass = class
  private
    FB: TDate;
    FC: TTestProperty;
    FA: string;
    FObject: TComponent;
    FMyStrings: TStringList;
  public
    constructor Create;
    destructor Destroy; override;
  published
    property A: string read FA write FA;
    property B: TDate read FB write FB;
    property C: TTestProperty read FC write FC;
    property AssignableObject: TComponent read FObject write FObject;
    property MyStrings: TStringList read FMyStrings;
  end;
```



Name	Value
A	
AssignableObject	
B	12/09/2018
C	tpValue1
MyStrings	(TStringList)

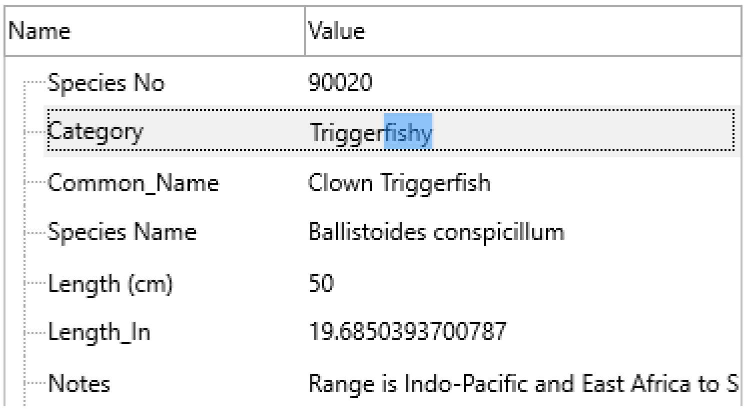
Click on the MyStrings value column to start the editor. The editor will show the values inside the editor. At the left side a memo is shown which can be used to change the value.



When editing a TCollection, the memo is replaced by a TTMSFNCObjectInspector instance, that allows editing the properties of the TCollectionItem instance, selected from the list at the right side. The changes that are being made to the properties are directly applied to the collection.

Datasource

The TTMSFNCObjectInspector is also capable of showing the fields and fields data from the active record. It also changes the value and posts it directly to the dataset as well as updates automatically when a value in the dataset field changes. Below is a sample that demonstrates this with the biolife.xml file assigned to a TClientDataSet.



Name	Value
Species No	90020
Category	Triggerfishy
Common_Name	Clown Triggerfish
Species Name	Ballistoides conspicillum
Length (cm)	50
Length_In	19.6850393700787
Notes	Range is Indo-Pacific and East Africa to S

Changing a field value will automatically post the value to the dataset. There are equivalent events that can be used to read or write the field value: OnReadDBField and OnReadDBFieldValue and OnWriteDBFieldValue. These events are only triggered when a DataSource is assigned.