

Extending The IDE With Debugger Visualizers And The Open Tools API

NIGEL TAVENDALE
www.allthingssyslog.com



Code on GitHub

<https://github.com/ntavendale/DebuggerVisualizers>



Open Tools API

Access to IDE through COM interfaces.

Use any language that supports COM.

IDE extensions can be compiled into packages.

IDE extensions can be compiled into DLLs (with sharemem) independent of IDE version.

Packages are loaded with the package manager.

DLLs are loaded using Registry entries.

HKEY_CURRENT_USER\Software\Embarcadero\BDS\<IDE_Version>\Experts

Native Tools API

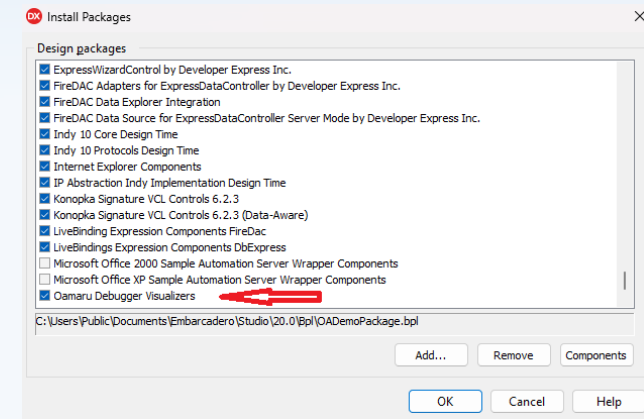
Direct access to IDE Objects (ex. TMainMenu)

Use Delphi (not a bad thing).

IDE extensions can be compiled into packages.

IDE extensions can be compiled into DLLs (with sharemem).

DLLs need runtime packages for IDE version they are built with.



What's In The Open Tools API?

COM Services.

Implemented by the IDE for you to work with when using it.

Gives you access to the properties and state of the various parts of the IDE.

COM Interfaces.

Forms, wizards etc in a package that the IDE uses to access your functionality.

Your Open Tools API Project must implement these.



IOTAServices

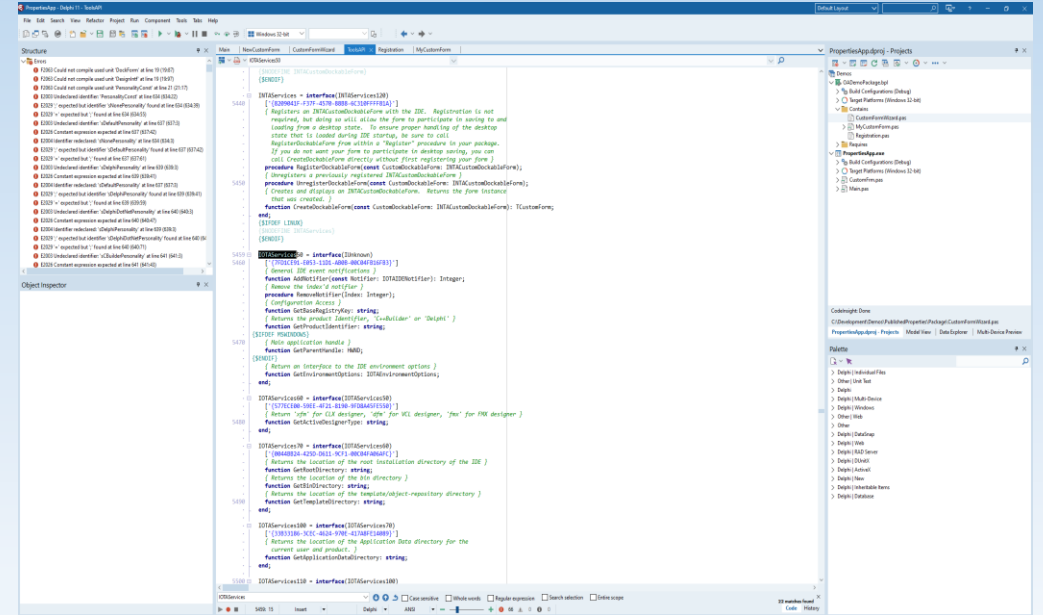
IDE installation information.

Preferred UI languages.

Startup Directories.

Application Data Directories.

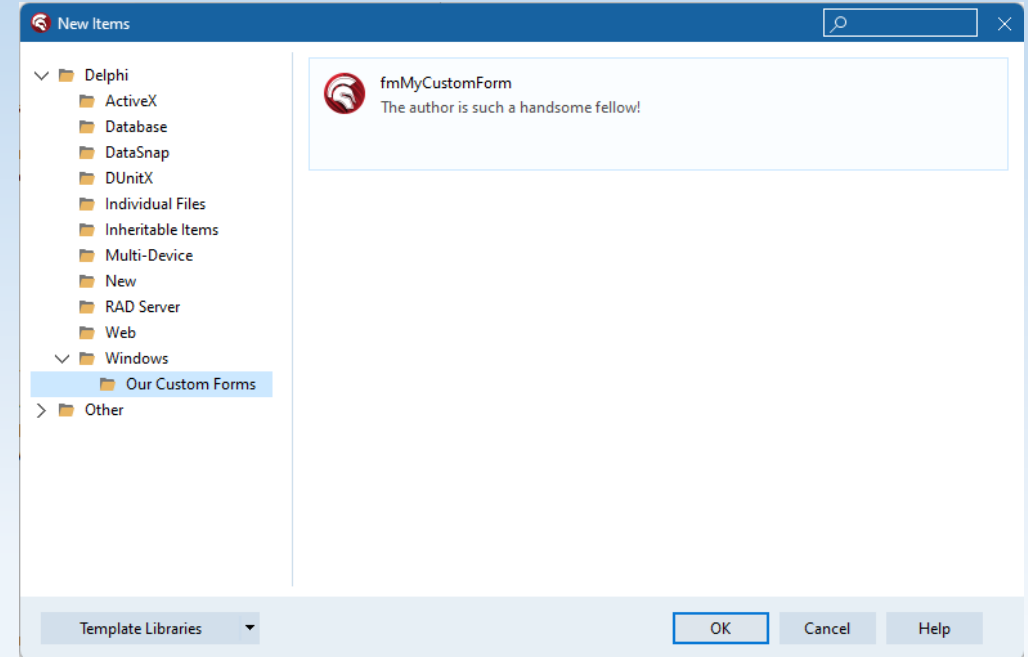
Base Registry Key.



IOTAGalleryManager

Add your own custom categories in the New Items dialog.

Assign your own custom forms/units/file types to your categories.



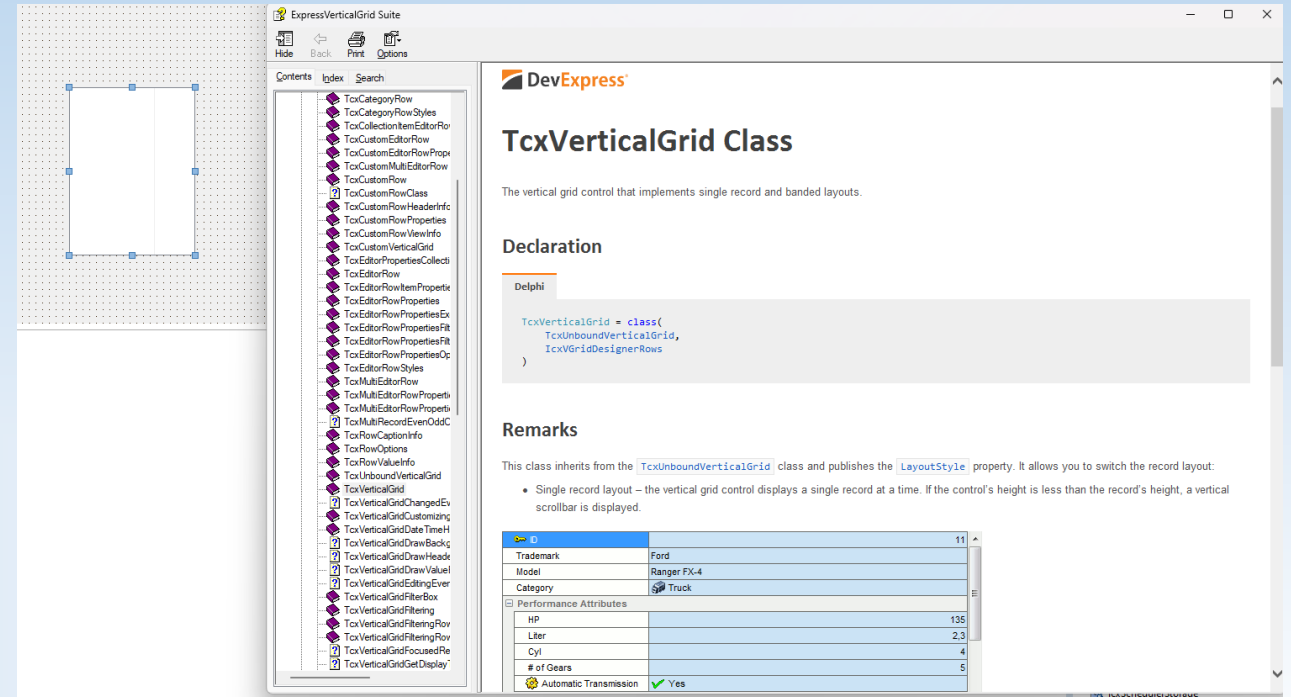
You must implement the IOTAGalleryCatetory interface.



IOTAHelpServices

Query IDE for information about help system.

Determine if IDE help understands Key Words.



The screenshot displays the ExpressVerticalGrid Suite IDE. On the left, a visual grid editor shows a simple rectangular grid. The main window is titled 'ExpressVerticalGrid Suite' and contains a 'Contents' pane on the left listing various classes and properties. The central pane shows the 'TcxVerticalGrid Class' documentation from DevExpress. The documentation includes a declaration in Delphi, remarks about inheritance and properties, and a table of performance attributes.

TcxVerticalGrid Class

The vertical grid control that implements single record and banded layouts.

Declaration

Delphi

```
TcxVerticalGrid = class(  
    TcxUnboundVerticalGrid,  
    TcxVGridDesignerRows  
)
```

Remarks

This class inherits from the `TcxUnboundVerticalGrid` class and publishes the `LayoutStyle` property. It allows you to switch the record layout:

- Single record layout – the vertical grid control displays a single record at a time. If the control's height is less than the record's height, a vertical scrollbar is displayed.

ID		11
Trademark	Ford	
Model	Ranger FX-4	
Category	Truck	
Performance Attributes		
HP		135
Liter		2.3
Cyl		4
# of Gears		5
Automatic Transmission	Yes	

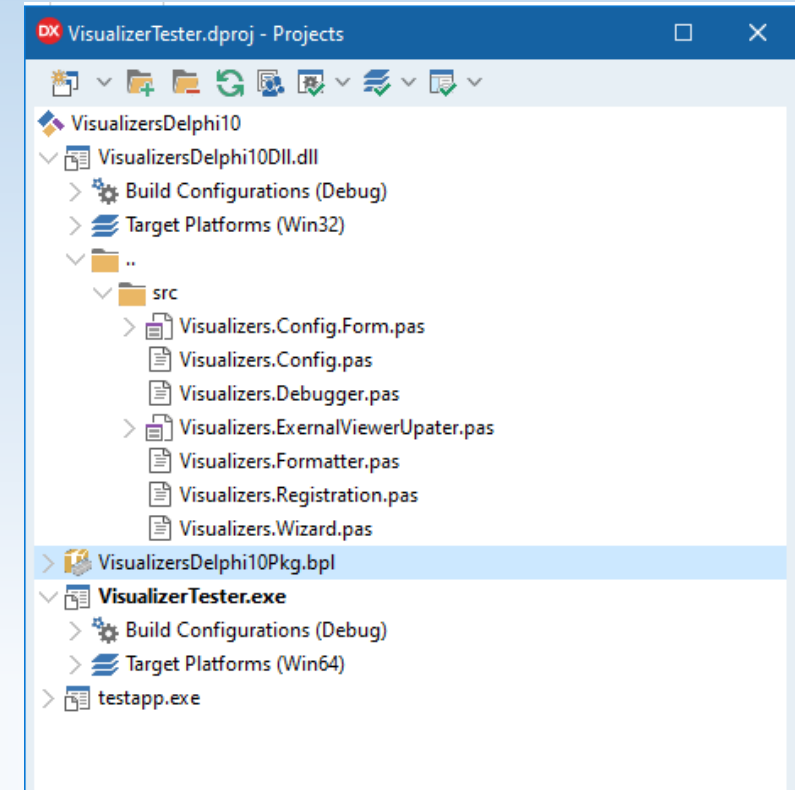


IOTAModuleServices

Access Project Group, Projects, Files

Register Custom Modules

- Forms with custom published properties in Object Inspector
- Creating Custom File Types (yaml, json).

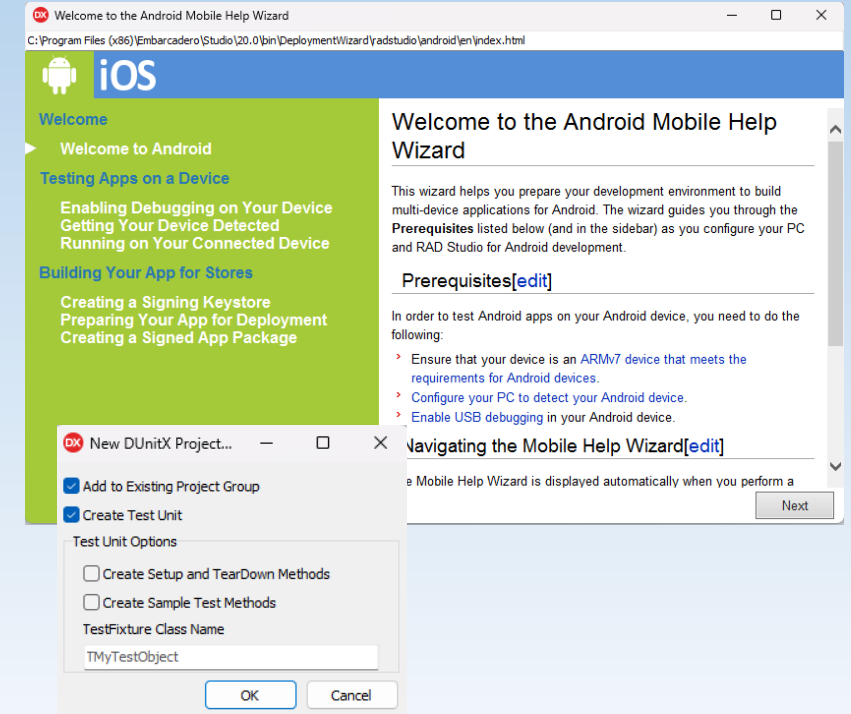


IOTAWizardServices

Menu Wizards. Open Tools API Menu Wizards only accessible through the Help menu.

Every custom module introduced through IOTAModuleServices needs a wizard for users to access it!

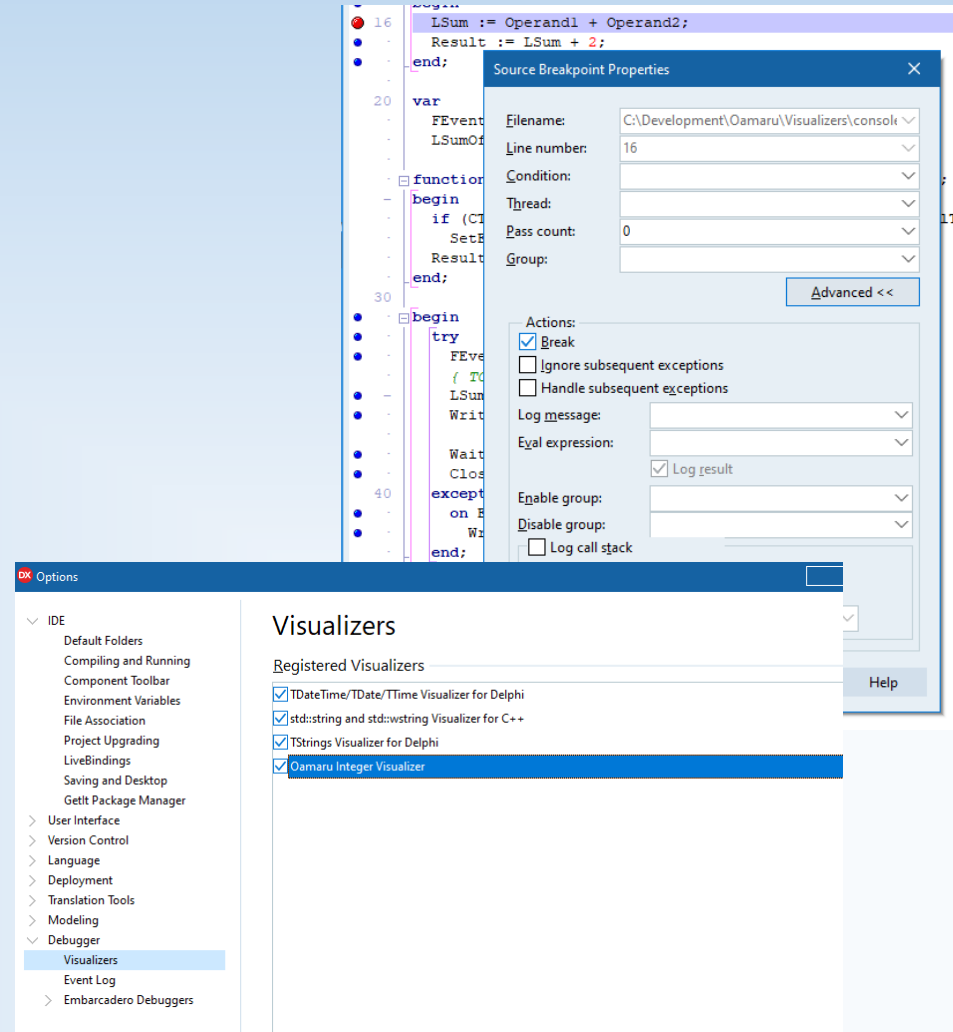
- Form Wizards for custom forms/units.
- Project Wizards for custom project types.
- Repository Wizards for custom Gallery Categories.



IOTADebuggerServices

Access to the Debugger

- Add/Remove breakpoints
- Iterate breakpoints in module
- Create breakpoints
- Deploy custom breakpoint editors
- Register/Unregister *Debugger Visualizers*



Visualizers – What Can We Do With Them?

Inspect Variables In The IDE With Value Replacers

Integer

```
begin
  LSum := Operand1 + Operand2;
  Result : Operand1 65535
end;
```

Hex


```
begin
  LSum := Operand1 + Operand2;
  Result : Operand1 🔍 $FFFF
end;
```

Binary

```
begin
  LSum := Operand1 + Operand2;
  Result : Operand1 🔍 1111 1111 1111 1111
end;
```

Inspect Variables In The IDE With An External Viewer

```
begin
  LSum := Operand1 + Operand2;
  Result : Operand1 🔍 $FFFF
end;
```



Operand1 (Word)

Visualizer IDE Info

Value

Value: \$FFFF

Operand1 (Word)	
hex	\$FFFF
dec	65535
oct	177 777
bin	1111 1111 1111 1111



IOTADebuggerVisualizer

Base Interface For Any Visualizer Component

- Name
- Identifier (Unique)
- Description

Number & names of the types the visualizer works with.

```
function GetSupportedTypeCount: Integer;
```

```
procedure GetSupportedType(Index: Integer; var TypeName: String;  
    var AllDescendants: Boolean); overload;
```



IOTADebuggerVisualizerValueReplacer

Basic Interface you need to implement to show your custom values in the IDE during variable inspection.

```
function GetReplacementValue(const Expression, TypeName,  
EvalResult: string): string;
```

- Expression: The name of the variable in the code
- TypeName: Integer, UInt64 etc
- EvalResult: Default value assigned by original IDE visualizer

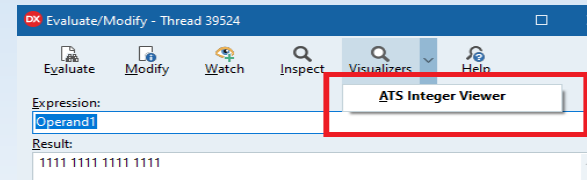


IOTADebuggerVisualizerExternalViewer

Basic interface you need to implement to create show your custom values in your own viewer.

function GetMenuText: **string**;

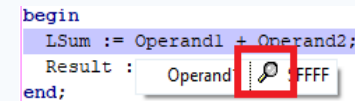
- Returns the Menu text seen on the Visualizers button of the Evaluate/Modify dialog.



function Show(**const** Expression, TypeName, EvalResult: **string**; SuggestedLeft, SuggestedTop: Integer): IOTADebuggerVisualizerExternalViewerUpdater;

- Expression, TypeName & EvalResult are the same as used by the value Replacer
- SuggestedLeft, SuggestedTop: Initial form position based on mouse position in IDE.

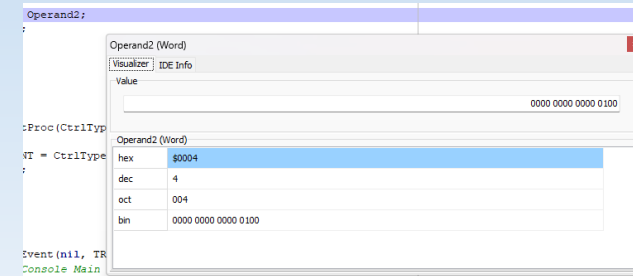
When interface present you see magnifying glass on variable in IDE.



IOADebuggerVisualizerExternalViewerUpdater

Basic interface you need to implement to create show your custom values in your own viewer.

Class implementing this interface is usually a descendent of TForm.

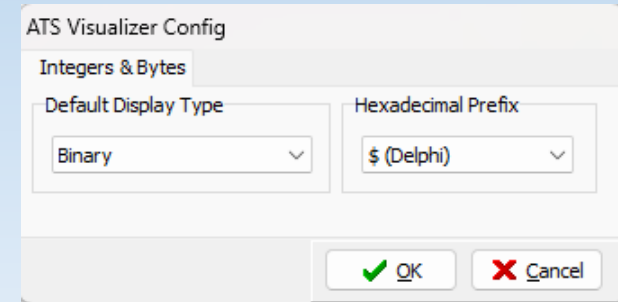
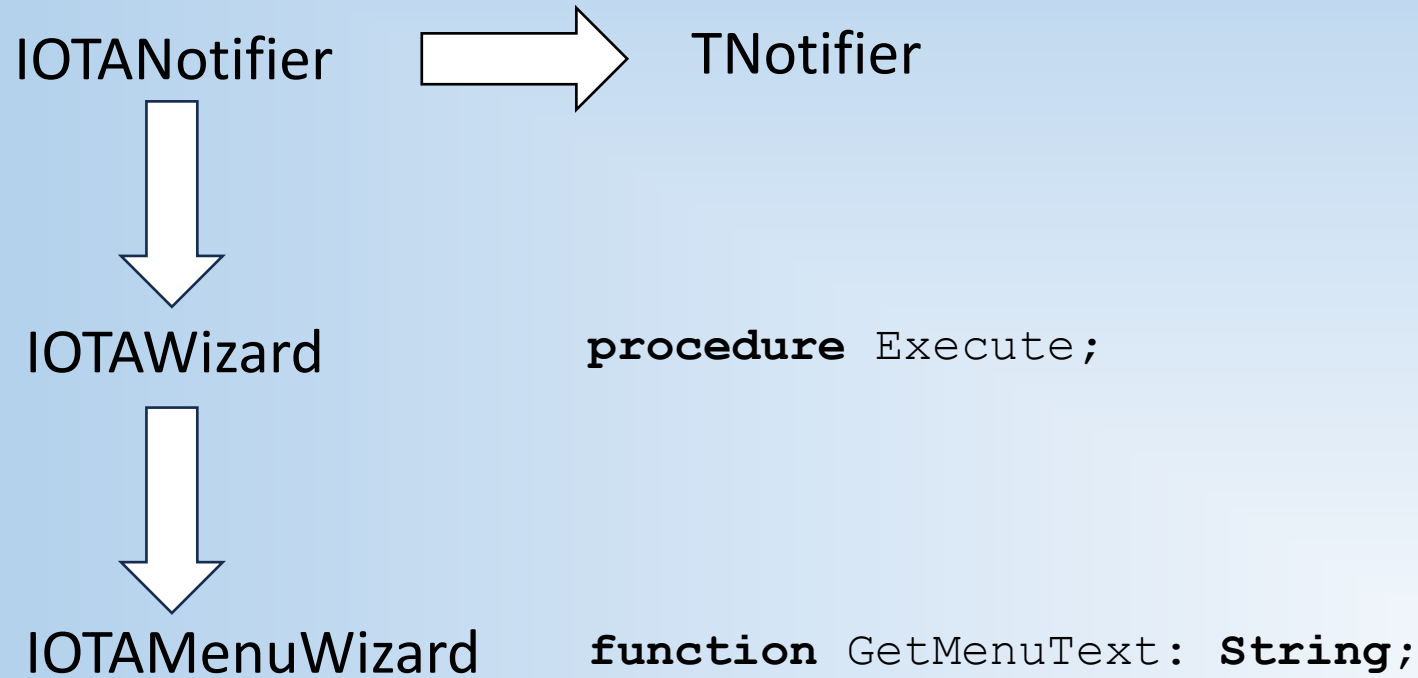


```
procedure CloseVisualizer;
```

```
procedure SetClosedCallback(ClosedProc: TOTAVisualizerClosedProcedure);
```



IOTAMenuWizard



Further Reading

ToolsAPI.pas - Most of the documentation is simply the comments in this file. There are almost no help topics either in the IDE help or online.

David Hoyle's GitHub page: <https://github.com/DGH2112/The-Open-Tools-API-Project>



NIGEL TAVENDALE

nigel.tavendale@allthingssyslog.com

www.allthingssyslog.com

<https://www.linkedin.com/in/nigel-tavendale-2a6a723>