



What is the Altium Designer RTL?

Summary

Article
AR0134 (v1.2) Dec 4, 2007

This article reviews what the Altium Designer RTL is and how it can be used in scripts and server projects.

The Altium Designer Run Time Library (RTL) is composed of Application Programming Interfaces (APIs). Many editors in Altium Designer have their own APIs. For example the PCB editor has the PCB API, Schematic editor has the Schematic API, the Project Manager has the Workspace manager API, Altium Designer platform has its own Client API and so on. Each API can compose of object interfaces, classes, routines and enumerated constants.

These Editor specific APIs have their own Object Model. An Object Model is a hierarchical system of Object Interfaces. These object interfaces represent objects in the Altium Designer.

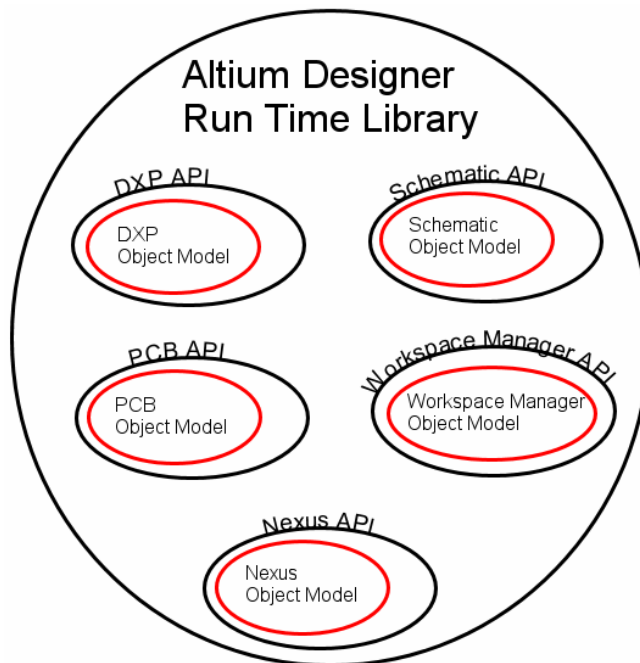


Figure 1: Altium Designer Run Time Library, its APIs and their Object Models

Object Model Hierarchy

Object Interfaces represent design objects that you can use in your scripts and in server code to update data from opened design documents in Altium Designer. Object Interfaces have properties and methods.

- Methods of Object Interfaces are the actions an Object Interface can perform.
- Properties of Object Interfaces represent the data contained in the object that is represented by the interface.

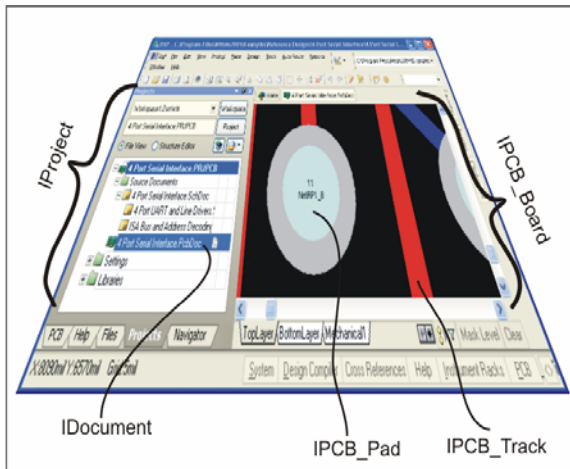
A parent object interface can have child object interfaces and in turn, child object interfaces can act as parent object interfaces and have child object interfaces and so on. This forms an Object Model Hierarchy.

For example, in the Workspace Manager Object Model, the `IWorkspace` Object Interface is the top level interface representing the Workspace manager in Altium Designer. Projects are part of the Workspace Manager, therefore an `IProject` interface is a child interface from the parent `IWorkspace` interface.

Documents are part of a project, therefore the `IProject` interface is the parent interface and `IDocument` is a child interface. Design objects such as sheet symbols, components and buses are child objects of the document object (represented by the `IDocument` interface). This is illustrated in a simple hierarchy example below.

```
IWorkspace
  IProject
    IDocument
      ISheetSymbol
      IComponent
      IBus
```

The figure below is an illustration of the relationship between objects in Altium Designer and the object Interfaces supported by the various Object Models from Altium Designer RTL.



Projects and the corresponding documents are managed by the Workspace Manager. A project open in Altium Designer is represented by the `IPProject` object interface, and the documents from this project are represented by the `IDocument` interfaces.

The PCB documents and PCB design objects are managed by the PCB Editor and its PCB Object Model. The PCB document open is represented by its `IPCBoard` interface and the design objects, for example, the pad object and the track object are represented by `IPCBoard_Pad` and `IPCBoard_Track` interfaces.



For more information on Altium Designer RTL and the object models within, open the *Knowledge Center* panel and navigate to the API Reference documents via **Configuring the System » Scripting in Altium Designer » Altium Designer RTL Reference**.

How is Altium Designer RTL used in scripts?

The scripting engine in Altium Designer has built in PCB, Schematic and Workspace Manager APIs as well as a subset of Borland Delphi's Run Time Library. The PCB, Schematic, Workspace Manager Object Models from the scripting engine enable you to write scripts that act on PCB or Schematic documents or invoke one of the file management routines.

The Object Models from the Altium Designer RTL is accessible in scripts, so you can code the object names and their method names with appropriate parameter values using one of the several supported scripting languages such as EnableBasic, Visual Basic, Javascript, TCL and as well as commonly used DelphiScript (which is very much like Borland Delphi).

DelphiScript Example

```

Procedure PadCount;
Var
    Board      : IPCBoard;
    Pad        : IPCBoard_Primitive;
    
```

What is the Altium Designer RTL

```
Iterator : IPCB_BoardIterator;
PadNumber : Integer;
Begin
    PadNumber      := 0;
    Board := PCBServer.GetCurrentPCBBoard;
    If Board = Nil Then Exit;
    Iterator      := Board.BoardIterator_Create;
    Iterator.AddFilter_ObjectSet(MkSet(ePadObject));
    Iterator.AddFilter_LayerSet(AllLayers);
    Iterator.AddFilter_Method(eProcessAll);
    Pad := Iterator.FirstPCBObject;
    While (Pad <> Nil) Do
    Begin
        Inc(PadNumber);
        Pad := Iterator.NextPCBObject;
    End;
    Board.BoardIterator_Destroy(Iterator);
    ShowMessage('Pad Count = ' + IntToStr(PadNumber));
End;
```

How is Altium Designer Run Time Library used for Server Development?

The Altium Designer Run Time Library is implemented using Borland Delphi 6™ tool kit. The Altium Designer RTL used for the Server Development is made up of Delphi Compiled Units (files with a *.DCU extension).

These *.DCU files contain compiled Delphi object information which gives you the ability to access to the object oriented structures (for example pad, track and text objects contained on a PCB document). *.DCU files are Delphi compiler-version specific, and are compiled with Borland Delphi 6. Therefore you need the matching Delphi version for your server development.

The units from Altium Designer RTL required in your server project are added to the **Uses** clause in one of the units from this server project. When you want to use a particular RTL function, add the corresponding unit in the *Uses* clause of your server code project.

For example, to use the `MessageRouter_SendCommandToModule` procedure, add `RT_API` unit in the *Uses* clause, or the compiler will report an `Undeclared identifier` error message. The *.DCU files can be found in `\Developer Kit\RTL` subdirectory.

Server Project Example

Uses

```
SysUtils, Windows, Dialogs, Rt_Util, Rt_Types, Rt_Param, Rt_Forms, Rt_Api,
Rt_ClientServerInterface, RT_PCB, RT_PCBProcs;
```

Implementation

```
Procedure DemoSimpleIterator;
```

Var

```
Board      : IPCB_Board;
Pad        : IPCB_Primitive;
Iterator   : IPCB_BoardIterator;
PadNumber  : Integer;
```

Begin

```
PadNumber      := 0;
// retrieve the current board's handle
Board          := PCBServer.GetCurrentPCBBoard;
If Board = Nil Then Exit;
// retrieve the iterator handle
Iterator       := Board.BoardIterator_Create;
Iterator.AddFilter_ObjectSet([ePadObject]);
Iterator.AddFilter_LayerSet(AllLayers);
Iterator.AddFilter_Method(eProcessAll);
// search and count pads
Pad := Iterator.FirstPCBObject;
While (Pad <> Nil) Do
Begin
    PadNumber := PadNumber + 1;
    Pad := Iterator.NextPCBObject;
End;
Board.BoardIterator_Destroy(Iterator);
// Display the count result on a dialog.
ShowMessage('Pad Count = ' + IntToStr(PadNumber));
```

End;

End.

There is additional information on Server Development which is part of the Altium Designer Developer Edition.

Where To Go Next?



Refer to the [Using the Altium Designer RTL](#) document for more information.



Refer to the [Getting Started with Scripting](#) and [Building Script Projects](#) scripting tutorials for more information and examples on how to use scripts in Altium Designer.



Consult the Scripting Resources accessible through the bottom part of the *Knowledge Center* panel in Altium Designer. Navigate to the API Reference documents via **Configuring the System » Scripting in Altium Designer » Altium Designer RTL Reference**.

Revision History

Date	Version No.	Revision
20-Sept-2005	V1.0	New document for Altium Designer
16-Dec-2005	V1.1	Updated for Altium Designer 6.
4-Dec-2007	V1.2	Updated for Altium Designer 6.8

Software, hardware, documentation and related materials:

Copyright © 2007 Altium Limited.

All rights reserved. You are permitted to print this document provided that (1) the use of such is for personal use only and will not be copied or posted on any network computer or broadcast in any media, and (2) no modifications of the document is made. Unauthorized duplication, in whole or part, of this document by any means, mechanical or electronic, including translation into another language, except for brief excerpts in published reviews, is prohibited without the express written permission of Altium Limited. Unauthorized duplication of this work may also be prohibited by local statute. Violators may be subject to both criminal and civil penalties, including fines and/or imprisonment. Altium, Altium Designer, Board Insight, Design Explorer, DXP, LiveDesign, NanoBoard, NanoTalk, P-CAD, SimCode, Situs, TASKING, and Topological Autorouting and their respective logos are trademarks or registered trademarks of Altium Limited or its subsidiaries. All other registered or unregistered trademarks referenced herein are the property of their respective owners and no trademark rights to the same are claimed.