



# A Tour of the Scripting System

---

## Summary

Guide

GU0120 (v1.5) May 8, 2007

This guide gives an overview of the Scripting System and how to create and use scripts in Altium Designer.

---

## Exploring the Scripting System

---

Fetching components from an existing schematic document, generating a customized net-list and automating a sequence of commands and so on can be done using scripts in Altium Designer.

The scripting system is object oriented which means you can use the same Altium Designer Run Time Library and Altium Designer Object Models in a script by one of the supported scripting languages.

Features of the scripting system include:

- The ability to choose one of the following scripting languages (DelphiScript, EnableBasic, VBScript, JavaScript and TCL) to write a script in Altium Designer.
- The ability to build dialogs and forms by dropping Visual and Non-visual Components on a script form from the **Tool Palette** panel.
- The ability to reference a design object using DXP Object Interfaces from a script.
- The ability to execute a server process from a script.

## Scripting System Editor

The Script editor is a script editing application within Altium Designer where you can write and execute scripts. You can either create script units or script forms within a script project from the **File » New » Script Files** menu items on the Altium Designer menu.

The script editor supports the creation of script forms which provide a user interface mechanism for example a customized dialog. There are many different controls (components) that can appear on a script form.

The scripting panels in Altium Designer work together to help you write and debug your scripts.

### Scripting System Debugger

If your script has an error, an error dialog appears with a concise error message. You have the ability to preview variables in the watches window, trace code and control the execution of the script.

### Script Projects and scripts

Scripts live in a project and routines from any script in a project are accessible to a different script from the same project.

## A Tour of the Scripting System

Since there are different script languages available in Altium Designer, thus a script ends with an extension that denotes which language is being used; Delphiscript (\*.pas) EnableBasic (\*.bas), VisualBasic (\*.vbs), Javascript (js) and TCL (\*.tcl)

### Application Programming Interface is supported

The Altium Designer Object Models are exposed in scripts. This means you have the ability to massage a design document and its design objects in Altium Designer.

The objects of the Altium Designer are accessed through “interfaces” on a script. Client Interfaces, PCB Interfaces, Schematic Interfaces, Integrated Library Interfaces, Workspace Manager and Nexus Interfaces are available to use in scripts.

### Commands (parametric server processes) can be executed from scripts

Commands or parametric processes (server/client processes) can be used in scripts to execute high level commands in Altium Designer such as closing all documents in one go.

### Script examples

To learn more about the scripting system and executing scripts, the example script projects are located in the `\Program Files\Altium Designer 6\Examples\Scripts` folder or its equivalent.

There are also the **TU0121 Getting Started With Scripting** and **TU0125 Building Script Projects** tutorials.

## Projects and scripts

You can store a script in a project of any type in Altium Designer. All the scripts in a project are visible, that is, any procedure / function from any script in the same project are accessible to a different script from the same project. Therefore it is important to keep the variable and method names unique.

It is a good idea to move all the common procedures/functions and methods that are used in different scripts within the same project to a new script within the same project.

## Projects and Global Projects

To add a project in the **Installed Projects** List from the global *Preferences* dialog. Click on **DXP » Preferences** and choose Scripting System under the System folder on the left side tree of the dialog.

The real purpose of having global projects is that any script of the same type (DelphiScript, EnableBasic, VBScript, JScript) can call functions and procedures from the global projects.

For normal projects that live in a folder, scripts within the same project can access a routine from another script within this same project.

## Creating new scripts

To create a new script in your project, you can choose a script based on a particular scripting language. The available units and forms are:

- DelphiScript Unit
- DelphiScript Form
- VisualBasic Script Unit
- VisualBasic Script Form
- Javascript Unit

- Javascript Form
- TCL Script Unit
- Enable Basic Script Unit.

### New scripts into a Script Project (\*.PrjScr)

You add new scripts by default to a scripting project that ends with a PrjScr extension. To add a new script, with a project open in Altium Designer, right click on this project in the *Projects* panel, and a pop up menu appears, click on the **Add New to Project** item and you can add a Delphiscript unit, form, JavaScript unit, form, VisualBasic unit, form, Enable basic script unit or TCL script unit.

### Adding new scripts to a project other than a script project

You can add new scripts into the specified project in the *Projects* panel in Altium Designer. With a project open, click and drill down to the **File » New » Script Files** menu and choose one of the following script types. A new script appears in this project.

### Referencing scripts in a script project

You can have code in one script to call another procedure in another script in the same script project and access to any global variable in any script within the same project.

## Adding existing scripts to a project

### Existing scripts into a project

You can add existing scripts to a specified project in the *Projects* panel in Altium Designer. With a project open in Altium Designer, right click on this project in the *Projects* panel, and a pop up menu appears, click on the **Add Existing to Project** item.

A **Choose Documents To Add to Project** dialog appears. You can multi-select as many scripts you want to add into the specified project.

## Writing scripts

### Important Terms

Before we can start writing scripts successfully, we need to understand the following terms;

- Components are visual objects on the *Tool Palette* panel that you can manipulate on a script form at design time.
- A component consists of methods, properties, and events.
- Events are conditions a component on a script form can react to.
- Properties represent the data contained in the object.
- Methods are the actions an object can perform.
- Processes are command strings in Altium Designer that you can use to execute Altium Designer commands in your scripts.
- Object Interfaces are special object interfaces in Altium Designer that you can use to extract and modify data from Altium Designer design documents in your scripts.

### Different script languages

Since there are different script languages available in Altium Designer, thus each script ends with an extension that denotes which language is being used. Each script ends with an extension that denotes

## A Tour of the Scripting System

which language is being used; thus DelphiScript (\*.pas) EnableBasic (\*.bas), VisualBasic Script (\*.vbs), JavaScript (js) and TCL (\*.tcl).

The scripting engine is written in Borland Delphi, and the Tool Palette is based on the Borland Delphi's VCL (Visual Component Library), thus there are more DelphiScript examples. While DelphiScript is based on Object Pascal, there are differences between DelphiScript and Object Pascal.

Please refer to the Differences between DelphiScript and Object Pascal in [TR0120 DelphiScript Reference](#) document.

## Using Altium Designer Object Interfaces in a script

The biggest feature of the scripting system, is that the Interfaces of Altium Designer objects are available to use in scripts. For example you have the ability to message design objects on Schematic and PCB documents through the use of Schematic Interfaces and PCB interfaces respectively.

### Object Oriented scripting system

All the scripting languages supported by the Scripting system are object oriented. All objects in Altium Designer are represented by their interfaces which make it possible to have access and modify the attributes associated with these objects.

- Objects have **Properties**. A property is a characteristic attribute that you can modify. For example the width property of a track object.
- Objects have **Methods**. A method is something an object can do. Objects can be selected, objects can talk to other objects. For example the server object has an Open method.
- Objects also can have **Events**. Events are actions that user performs while using a dialog in Altium Designer. The Scripting system deals with events using event handlers. An action by the user on the dialog (script dialog) triggers an event handler in your script.

### Interfaces

In scripts, you don't create instances of objects, you just use an interface that points to an existing object in Altium Designer. From this interface, you can extract embedded or aggregate interface objects and as well, you can get or set values that denote some property for this interface.

Each object in Altium Designer that is supported by the DXP Object model has its own interface. All objects of the same type (for example PCB tracks) have the same interfaces; in this case, all track objects are represented by **IPCTrack** interfaces.

#### Unique interfaces

How do we know which interface is which? Each interface that represents an existing object has its own object address in memory.

In this case, all PCB objects' addresses are represented by the **TPCBObjectHandle** type. A PCB interface has the **Function I\_ObjectAddress : TPCBObjectHandle;** method.

All Schematic objects' addresses are represented by the **TSchObjectHandle** type. Thus a Schematic interface has the **Function I\_ObjectAddress : TSchObjectHandle;** method.

#### How do we have access to Altium Designer objects?

Basically you need to have the design document that has design objects on it open in Altium Designer first, before you run a script on it. The script needs to have a starting function that obtains the interface

to the design document object. From this design document object, you then have access to the sub objects which are the design objects.

For PCB objects, you obtain the PCB interface that points to the PCB editor object, the **PCBServer** function is invoked in your script which returns you the **IPCB\_ServerInterface** interface. This object interface obtains the PCB editor server object and then you can get the current PCB document and extract data from PCB objects and invoke PCB object's methods.

For Schematic objects, you obtain the Schematic interface that points to the Schematic editor object, invoke the **SchServer** function in your script which returns you the **ISch\_ServerInterface** interface. This object interface obtains the Schematic editor server object and then you can get the current schematic document and extract data from Schematic objects and invoke Schematic object's methods.

The **IWorkspace** interface is the main interface representing the WorkSpace Manager object in Altium Designer. The **IWorkspace** interface deals with projects, documents and objects on the open documents in Altium Designer. To use workspace interfaces, the project needs to be compiled first refreshing all the linkages and nets up to date.

### **Interfaces conventions and Examples**

Interface names as a convention have an I added in front of the name for example **IPCB\_Board** represents an interface for an existing PCB document in Altium Designer. An example of PCB interfaces in use is shown next.

#### **PCB Object Interfaces example**

```
Var
Board : IPCB_Board;
Via    : IPCB_Via;
Begin
// obtains the PCB server and then the currently open PCB document
Board := PCBServer.GetCurrentPCBBoard;
// if pcb document represented as a board doesnt exist, exit.
If Board = Nil Then Exit;
// Create a Via object using the PCBObjectFactory method.
Via := PCBServer.PCBObjectFactory(eViaObject, eNoDimension,
eCreate_Default);
// Setup properties for the via object and
// position it on the PCB document at 7500,7500
Via.X      := MilsToCoord(7500);
Via.Y      := MilsToCoord(7500);
Via.Size   := MilsToCoord(50);
Via.HoleSize := MilsToCoord(20);
// Via is a dual layer object.
Via.LowLayer := eTopLayer;
Via.HighLayer := eBottomLayer;
```

## A Tour of the Scripting System

```
// Put this via on the Board object (PCB document).  
Board.AddPCBObject(Via);  
End;
```

### DXP Object Model

The DXP Object Model is implemented in Altium Designer so that your scripts have the access to various Object Models in Altium Designer. You can use Work Space Manager Object Model for example to compile a PCB project and extract a Protel format netlist. You can use PCB Object Model to massage PCB objects on a PCB document, and so on.

You can use any scripting language to have access to the DXP Object Model.

#### See also

Check out the scripts in the `\Altium Designer 6\Examples\Scripts\` folder to see Altium Designer Object Interfaces and functions being used in scripts.

Refer to the **TU0121 Getting Started With Scripting** tutorial and the **TU0125 Building Script Projects** tutorial.

Refer to the **Altium Designer RTL Reference** for more information on design documents and design objects and the Application Programming Interfaces and its Object Models.

### Exploring Example Scripts

The examples that follow illustrate the basic features of scripting in Altium Designer. These scripts are organized into the subfolders of the main **\Examples\Scripts** folder.

Note that the DelphiScript folder contains far many more examples because the Altium Designer system as well as the Scripting engine and the Tool Palette are written in Borland Delphi. The DelphiScript language is a subset of Borland Delphi language and is the main language used in the Scripting System.

## Executing a script in Altium Designer

### Running a script in the Text Editor

You can configure the **Run** command when you are in the text editor to point to a script and execute it. Every time you click on the **Run** icon from the Text Editor menu or press **F5**, the scripting system executes the script pointed to by the **Set Project Startup Procedure** item.

You can change the start up procedure by clicking on the **Set Project Start Up Procedure** item in the **Run** menu which invokes the *Select Item to Run* dialog. You can then select which procedure of a script to be set. Note these scripts are of standalone nature, that is they do not rely on a design document to process results. If you need to run a script on a PCB document for example, this Text Editor method doesn't work.

### Running a script repeatedly in the text editor

To run a script repeatedly in the text editor in Altium Designer, assign the script to the **Set Project Startup Procedure** item from the **Run** menu of the Text editor server.

You can then click on the **Run** button or press **F5** to execute this script. To run a different script, you will need to re-invoke the **Set Project Startup Procedure** from the **Run** menu and assign a new script to it.

## Executing a script on a design document in Altium Designer

To execute a script in Altium Designer, there are two methods and there are two different Altium Designer dialogs for each method. These methods are necessary if you wish to run a script on a server specific document such as PCB or Schematic documents.

### 1. Using the Select Item To Run dialog to execute a script

Click on the **Run Script** from the system menu and the *Select Item to Run* dialog appears with a list of procedures (those parameter-less procedures/functions only appear) within each script in a opened project in Altium Designer.

Note that you can also click on a script unit filename within this *Select Item to Run* dialog and the functionless/procedureless **Begin End.** block within the script gets executed. See code example below;

#### DelphiScript unit

```
Var
Begin
//script here with no function/procedure
A := 50;
A := A + 1;
ShowMessage(IntToStr(A));
End.
```

Now, only parameter-less functions and procedures for each script of an opened project only appear on the *Select Item to Run* dialog. It is a good idea for script writers to write the functions in scripts so that they will appear in this dialog and the other functions with parameters not to appear in this same dialog.

When you are working in a different editor such as PCB editor, you can assign the script to a process launcher and use it to run a specified script easily. See the Assigning a script to a process launcher.

### 2. Using the Run Process dialog to execute a script

Invoke the *Run Process* dialog from Altium Designer's System menu and execute the **ScriptingSystem:RunScript** process in the Process: field and specify the script parameters, the **ProjectName** parameter which is the path to the project name and the **ProcName** parameter to execute the specified procedure from a specified script in the Parameters: field.

You need the following parameters for the **ScriptingSystem:RunScript** process to execute a specified script.

#### Process

ScriptingSystem:RunScript

#### Parameters

ProjectName (string)

## A Tour of the Scripting System

ProcName (string)

### Example

Process: ScriptingSystem:RunScript

Parameters : ProjectName = C:\Program Files\Altium Designer

6\Examples\Scripts\General\HelloWorld.PrjScr | ProcName = HelloWorld>HelloWorld.

### 3. Running a script in the Filter panel

You can execute scripts with specially defined functions in the Expression (Query) window of the **Filter** panel for Schematic and PCB documents. These scripts need to go in a global project that is installed in the **Installed Projects** list. These scripts need to have functions that return a boolean value, and have query statements such as **IsComponent** for PCB, or **IsBus** for Schematic.

```
Function HighlightPadsWithZeroHoleSize : Boolean;
```

```
Begin
```

```
Result := IsComponentPad and (Holesize = 0);
```

```
End;
```

Save the script in a project that is in the **Installed List**. Type the function name in the **Filter** panel, and then click the **Apply** button to execute the function.

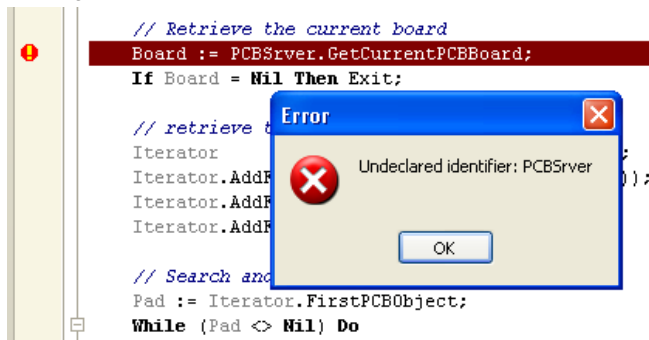
### Debugging a script

The Script Debugger provides you with full featured debugging environment. The script that compiles correctly doesn't always work as planned. This usually boils down to problems such as errors in logic, invalid assumptions, misplaced grouping operators and typographical errors.

### Script Errors

The Delphi Script's debugger permits you to intercept exceptions, trigger break points, evaluate expressions, and more. Script Errors can be one of the following:

- Design Time Errors
- Compile Time Errors
- Run Time Errors
- Logical errors.



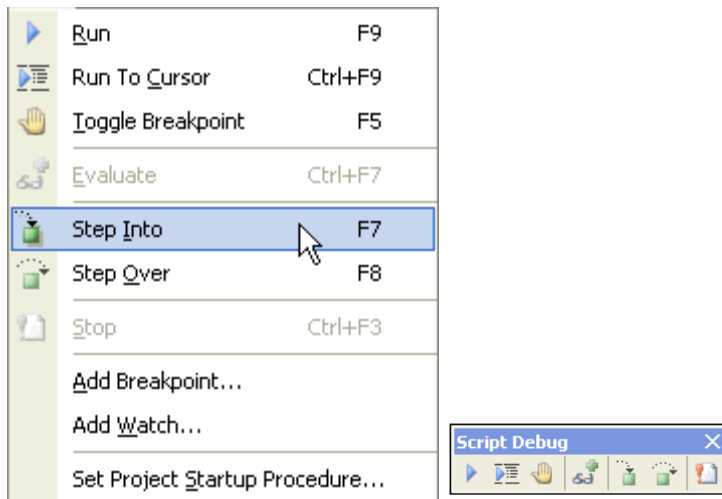
## Debugging scripts

When you encounter errors in the script, you will debug the script to resolve the errors and the Script Debugger provides some ways to find and resolve errors (bugs). To break out of the debugging mode, you can either finish executing the script or stop the script and edit the script before re-running this script.

The process of executing script one line at a time is referred to tracing or script stepping. The Script debugger provides options related to tracing: stepping into and stepping over. When you are debugging a script, you can trace the script and watch certain expressions.

Choose any of the debugging commands from the Scripting menu. Few debugging panels are available, including **Breakpoints** List, **Call Stack**, **Watches** list, Modules. There is also the **Evaluate** dialog which displays the properties for the focussed object.

Display them by clicking on the **Script** button on the bottom right of the Altium Designer status bar.



### Assigning a script to the menu, toolbar or key

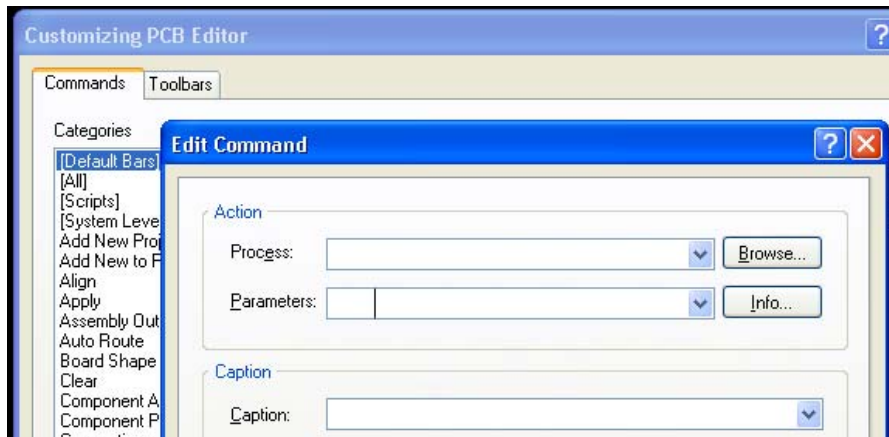
You have the ability to assign a script to a server menu, toolbar or hot key in Altium Designer which makes it possible for you to run the script over a current PCB document for example. You will need to specify the full path to a project where the script resides in and specify which unit and procedure to execute the script.

There are two parameters in this case: the **ProjectName** and the **ProcName**. For the **ProcName** parameter, you need to specify the script filename and the procedure. So the format is as follows: **ProcName = ScriptFileName>ProcedureName**. Note the Greater Than (>) symbol used between the script file name and the procedure name.

#### Assigning to a process launcher example

To illustrate this ability to assign a script to a resource, we will open a PCB document in Altium Designer and use the HelloWorld script example from the **\Scripts\General\** folder.

1. Double click on the PCB menu and the *Customizing PCB Editor* dialog appears.
2. Click on the **New** button from the *Customizing PCB Editor* dialog.
3. Choose **ScriptingSystem:RunScript** process in the **Process:** field of the *Customizing PCB Editor* dialog.
4. Enter **ProjectName = C:\Program Files\Altium Designer 6\Examples\Scripts\DelphiScriptGeneral\HelloWorld.PrjScr | ProcName = HelloWorldDialog>RunHelloWorld** text in the **Parameters:** field for example.



5. You will need to give a name to this new command and assign a new icon if you wish. In this case, the name is **PCBScript** in the **Caption:** field of this dialog. The new commands appear in the **[Custom]** category of the **Categories** list. Click on the **[Custom]** entry from the **Categories** list. The **PCBScript** command appears in the **Commands** list of this dialog.
6. You then need to drag the new **PCBScript** command onto the PCB menu from the *Customizing PCB Editor* dialog. The command appears on the menu. You can then click on this new command and the HelloWorldDialog form appears.

## Assigning a script to a pop up menu

You can assign a script to a pop up menu in one of the editors in Altium Designer. For example you can assign a script to a pop up menu in the PCB editor. You would need to invoke the customizing dialog by double clicking on the menu bar of the editor. In this **Customizing [ServerName]** dialog, click on the **New** button.

The **Edit Command** dialog shows up then you need to add the **ScriptingSystem:RunScriptText** process in the **Process:** field and then specify the parameters for this process.

Enter the caption value for the **Caption:** field. You can assign a bitmap to this command as well. Click ok and this command appears in the **[Commands]** item within the **Categories** list of the **Customizing [ServerName] Editor** dialog, and then you can drag the new command anywhere.

### Example

As an example, to execute a windows **Notepad** from a pop up menu in the PCB editor:

1. double click on the PCB menu bar.
2. Then with the *Customizing PCB Editor* dialog, click on the **New** button to invoke the **Edit Command** dialog. Insert **ScriptingSystem:RunScriptText** in the **Process:** field, and insert **Text=Begin RunApplication('notepad.exe'); End;** in the **Parameters:** field.
3. Enter **Run Notepad** for the **Caption:** field, which in this case is **Run Notepad**.
4. Click **OK** and this command appears in the **[Commands]** item within the **Categories** list of the *Customizing PCB Editor* dialog.
5. Drag this new command to the **Help** menu on the PCB menu (don't release the left mouse button), and once the **Help** menu appears, drag further to the **Popups** submenu and again drag further to the **Right Mouse Click Free Space** submenu. Drop the command somewhere in this submenu.
6. Alternatively drop the command in the **Right Mouse Click Free Space** entry within the **Categories** section of the *Customizing PCB Editor* dialog.

## Scripting System Shortcut Keys

### Display Scripting Panels

<b>Ctrl Alt B</b>	Display the Breakpoints panel
<b>Ctrl Alt S</b>	Display Call Stack panel
<b>Ctrl Alt W</b>	Display Watch List panel
<b>Ctrl Alt E</b>	Display Code Explorer panel
<b>Ctrl Alt P</b>	Show Tool Palette panel
<b>Ctrl Alt I</b>	Show Object Inspector panel

### When editing scripts

When editing scripts in the Text Editor; the shortcut keys are:

<b>F3</b>	Find Next dialog appears
<b>F5</b>	Toggle the existing breakpoint on the current script
<b>F9</b>	Run the current script
<b>Ctrl + F</b>	Find Text dialog appears
<b>Ctrl + H</b>	Replace Text dialog appears
<b>Ctrl + F4</b>	Close active document
<b>Alt + F4</b>	Close Altium Designer
<b>Ctrl + TAB</b>	Cycle through open documents.

### Bookmarks

#### Ctrl-Shift 0-9

Hitting Ctrl-Shift and one of the number keys (0..9) toggles the indexed bookmark on the gutter of the current line of the script (where the cursor is at).

#### Ctrl 0 -9

Hitting Ctrl and one of the number keys (0..9) jumps to book mark (0..9) of the script.

### When debugging a script

When debugging scripts in the Debugger, the short cut keys are

- F5** Hitting the F5 key or clicking on the Hand button from the toolbar toggles the breakpoint on the current line of the script (where the cursor is located).
- F7** The Step Into process executes script one statement at a time. If the statement being executed calls another procedure, stepping into that statement simply transfers control to the first line in the called procedure. The blue line on a script indicates the current line where the script is up to.
- F8** The Step Over process is similar to Step Into, except that if the current statement is a call to another procedure, the entire called procedure is executed without stopping rather than tracing through the called procedure. So for example, stepping over the statement, ProcB; in the previous procedure executes procedure ProcB and stops at the next line ProcC; in Procedure ProcA. This is useful if we

are certain that the Procedure ProcB is not the cause of our problem and we don't want to trace through that procedure line by line.

**F9** When the F9 key / Arrow button is pressed, the script is executed depending on where the cursor is at, for example at the beginning of the script or continuing from the point where it was interrupted due to a breakpoint.

**Ctrl F3** When this Stop key/button is pressed, the debugging of the script is halted in Altium Designer.

**Ctrl F7** When the key/button or the Evaluate toolbar button is pressed, the *Evaluate* dialog appears. You can type in an expression that represents a variable in the script. For example, DisplayForm.Label1.Caption in the Expression field reveals a text message in the Result field if the DisplayForm object representing a Delphi form exists in the script.

**Ctrl F9** When this Run To Cursor key/button is pressed, whenever the cursor is on the script, the script is executed and then it is paused at where the cursor is located.



## Scripting panels

The scripting panels can be displayed by clicking **Script** button on the bottom of the Altium Designer application. Various scripting panels include, Tool Palette, Code Explorer, Object Inspector, Watch List, Breakpoint List and Call Stack panels.

- The *Tool Palette*, *Object Inspector* and the *Form Designer* all work closely to help you build a user interface for your script.
- The *Code Explorer*, *Watch List*, *Breakpoint List* and *Call Stack* panels all work closely to help you debug your script.

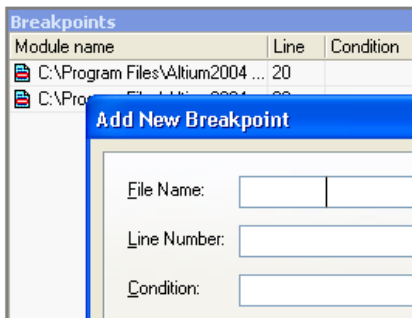
### Breakpoint List panel

By default, script execution pauses each time the breakpoint is encountered. A breakpoint defined on a line in a script is marked by a red line.

A way to get ordered feedback on the errors that may be occurring in your script is to control the script execution. Controlling scripts with breakpoints allow you to execute single lines of code, an entire function or a specified script block.

Break-points are stored in the project that the script is associated with.

When you create a breakpoint, it must be on a statement line of script. Any breakpoints set on blank lines, commented lines or other non statements will be ignored and treated as invalid. Breakpoints can be accessed using the *Breakpoints List* panel so you can easily locate all of your set breakpoints without needing to go through your script to find them.



Use the **Ctrl-Alt-B** short cut keys to show up the Breakpoint List panel.

### Call Stack panel

The *Call Stack* panel permits you to inspect the chain of procedure / function invocations that resulted in the execution of the current line of script in the debugger.



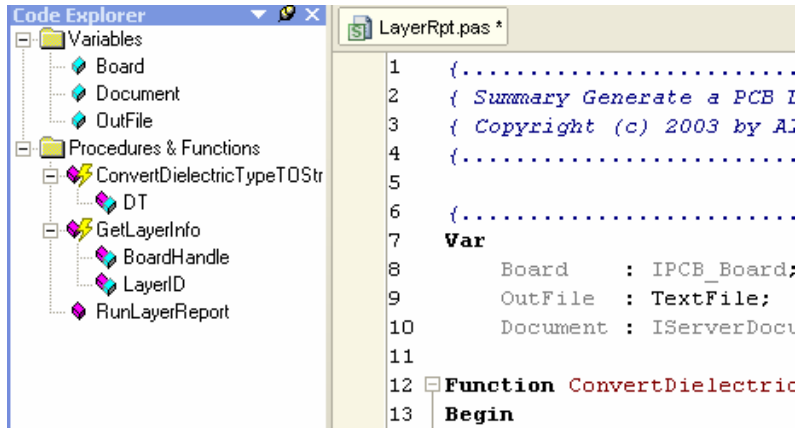
## A Tour of the Scripting System

Use the **Ctrl-Alt-S** short cut keys to show up the Call Stack panel.

### Code Explorer panel

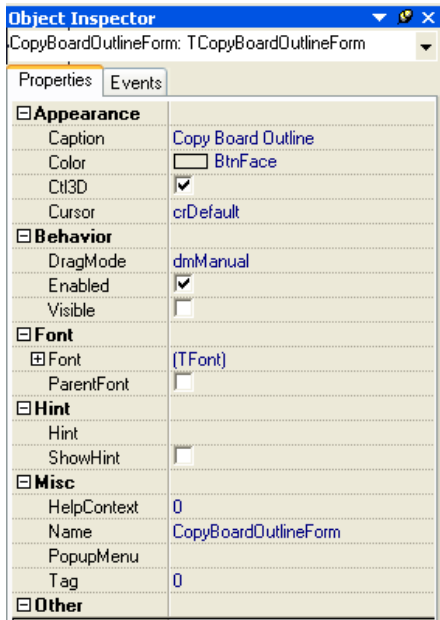
The *Code Explorer* panel displays the variables, procedures and functions (along with their parameters) for a currently focussed script. This panel serves as a quick reference to what variables and procedures/functions are used in the script.

The variables used in a current script are marked as aqua boxes under the Variables folder. The functions used are marked with pink boxes along with yellow flash and these procedures used are marked just as pink boxes under the Procedures & Functions folder in the Code Explorer panel.



### Object Inspector panel

The *Object Inspector* panel is a snap shot of the attributes (properties and events) of the current component on the form or the form itself. You have the ability to change object's properties and define new event handlers for this object (windowed controls only).

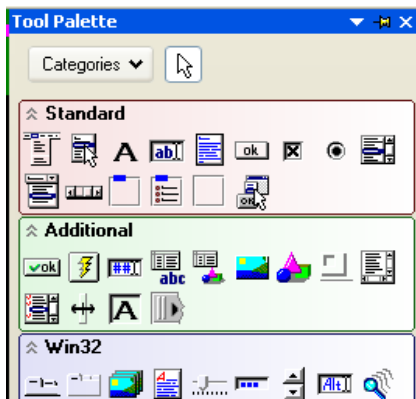


Since components have three items: **Properties**, **Events** and **Methods**. The **Properties** and **Events** are displayed for the currently focussed component such as a button or the script form on the pages of the tabs of the *Object Inspector* panel. The methods are “invisible” but available for use in the scripts.

For example if you wanted to make a button invisible, you invoke the Button’s **Hide** method in your script.

## Tool Palette panel

The range of controls is available on the *Tool Palette* panel with different tabs. You can place components on a script form by double clicking the component on the *Tool Palette* panel or click the component once and then click on the form where you want the component to appear. To get out of the component placement mode, click on the Arrow button next to the **Categories** button.



## A Tour of the Scripting System

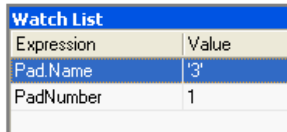
The *Tool Palette* panel is equivalent to the Borland Delphi's Component palette and in fact is a subset of the Borland Delphi's Component palette.

Tool Palette Tab name	Description
Standard	Standard controls and menus
Additional	Specialized controls
Win32	common Window controls such as RichEdit, TrackBar and Coolbar components
System	components and controls for system level access, including timers, multimedia and DDE
Dialogs	Commonly used dialogs such as Open dialog, Font dialog and Print dialog components.
Win 3.1	Old style win3.1 components.
Altium XP Standard	Standard controls and menus but with the Windows XP look and feel.

## Watch List Panel

### Watch List

Use this handy panel to track the values of values or expressions as you step over or trace into your script code. As you step through your script the value of the watch expression will change if your script updates any of the variables contained in the watch expression.



Expression	Value
PadName	'3'
PadNumber	1

Use the Ctrl-Alt-W short cut keys to show up the *Watch List* panel.

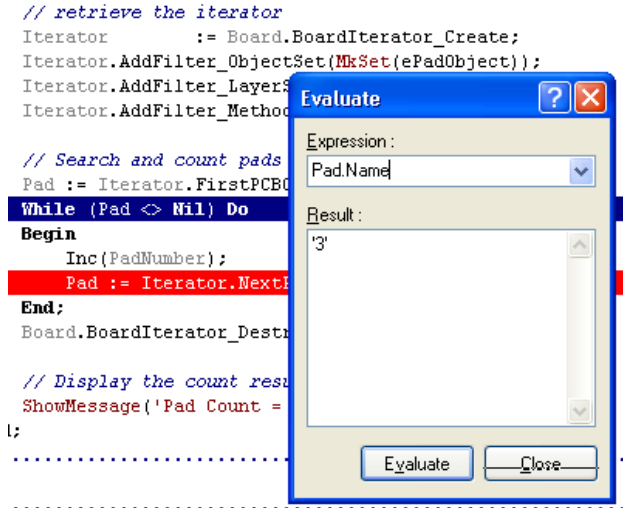
## Using tools for your scripts

---

The scripting system provides various tools to help you write scripts. You can interactively step line by line through scripts, inspect variables and objects and set breakpoints on one or more lines in the script.

### Script Evaluate feature

When your script is stopped in Altium Designer due to a breakpoint or an error, you can copy an expression from your script and invoke the **Evaluate** dialog and to paste this expression. Click the **Evaluate** button to see the resultant value for this expression.



Press **Ctrl - F7** keys to invoke the evaluate expression.

## Script Completion feature

The Code completion feature is that, when you type an object name and a dot after the object name, the list of properties and methods for the object are displayed.

For example just after the object name in your script, type in '.' and the Code Completion pop-up window appears with a list of properties and methods.



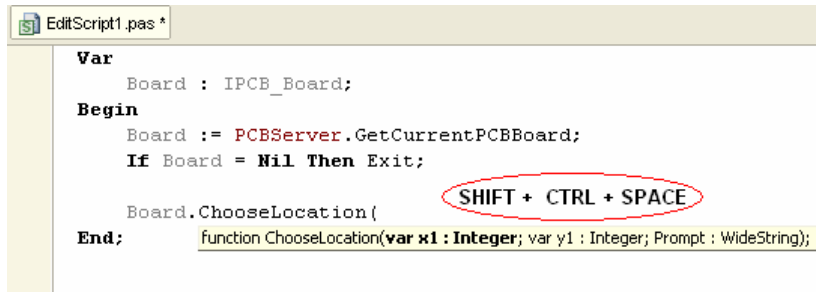
You can press **CTRL + Space** to see available properties or methods for the interface object.

## Script Parameters feature

The Script Parameters feature is that, when you type a method name and an open parenthesis, the syntax for the method's arguments are displayed.

For example just after the method in your script, type in '(' and the Code parameters pop-up window appears with a list of parameters for this method.

## A Tour of the Scripting System



```
Var
    Board : IPCB_Board;
Begin
    Board := PCBServer.GetCurrentPCBBoard;
    If Board = Nil Then Exit;

    Board.ChooseLocation(
End;    function ChooseLocation(var x1 : Integer; var y1 : Integer; Prompt : WideString);
```

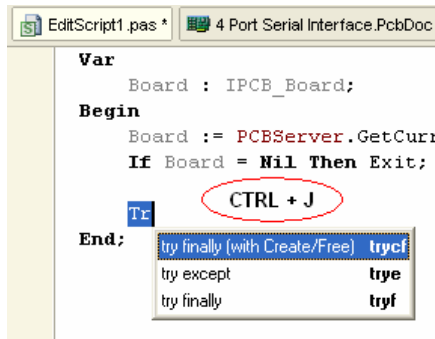
The text **SHIFT + CTRL + SPACE** is circled in red in the original image, pointing to the `ChooseLocation` function call.

You can also press **SHIFT + CTRL + SPACE** keys to display the Script parameters tooltip.

## Script Templates feature

Invoke **Tools » Show Code Templates** and a pop up list of available code templates appears depending on which scripting language the script is using.

You can select a code template from the list by clicking on a code template from the pop-up Code Templates window..



```
Var
    Board : IPCB_Board;
Begin
    Board := PCBServer.GetCurr
    If Board = Nil Then Exit;

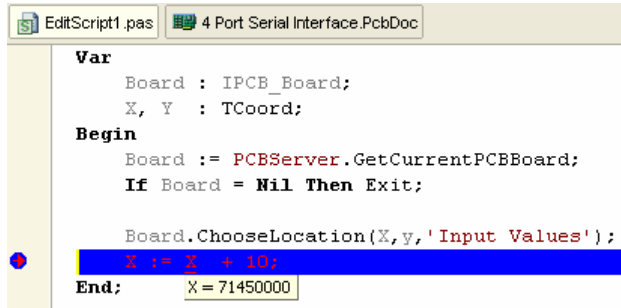
    Tr
End;
```

The text **CTRL + J** is circled in red in the original image, pointing to the `Tr` statement. A tooltip is visible below the cursor, listing options: `try finally (with Create/Free) trycf`, `try except trye`, and `try finally tryf`.

Note, you can press **Ctrl + J** to see a list of common scripting statements that you can insert into your script.

## Script Expression Tooltip

The Script Expression Tooltip is a great debugging tool in your scripts. You can view the value of a variable by hovering the mouse cursor over the variable while the script is stopped in Altium Designer.



```
Var
    Board : IPCB_Board;
    X, Y : TCoord;
Begin
    Board := PCBServer.GetCurrentPCBBoard;
    If Board = Nil Then Exit;

    Board.ChooseLocation(X, y, 'Input Values');
    X := X + 10;
End;
```

The text `X := X + 10;` is highlighted in blue in the original image. A tooltip is visible below the cursor, showing the value `X = 71450000`.

## Script Symbol Insight tooltip

This feature displays declaration information for any identifier by passing the mouse over it in the script editor. You can also use Ctrl to underline the variable and hover the mouse on the underlined variable.

The screenshot shows a script editor window titled 'EditScript1.pas' and '4 Port Serial Interface.PcbDoc'. The script content is as follows:

```

Var
  Board : IPCB_Board;
  X, Y : TCoord;
Begin
  Board := PCBServer.GetCurrentPCBBoard;
  If Board function PCBServer : IPCB_ServerInterface;

```

A tooltip is visible over the underlined variable 'function PCBServer : IPCB\_ServerInterface;', displaying its declaration.

## Using bookmarks

You can use the book marks in your scripts to jump to a statement quickly. in your script. You can define up to 10 bookmarks in your script. Each book mark is marked by a dark green box with a number in it.

The screenshot shows a script editor with the following code:

```

55  Str
56  Filename
57  OutFile
58  Layer
59  ReportDocum
60  Begin
61  // This mai
62  BoardHandle

```

Dark green boxes with numbers 55, 58, and 61 are placed in the left margin next to the corresponding lines of code, indicating the positions of bookmarks.

Note, use Ctrl Shift 0..9 to toggle a bookmark, and Ctrl 0..9 to jump to an existing bookmark in the script.

## Script Editing options

Invoke the **Preferences** dialog from the **Tools » Editor Preferences** menu and from the *Preferences* dialog, double click on the **Text Editors** folder on the left panel of this dialog. Then you can set up the script editor, set up the color of the elements within the script editor, for example black color for the background, white for the comment keyword and so on.

## Using Installed Projects list

### Using the Installed Script Projects

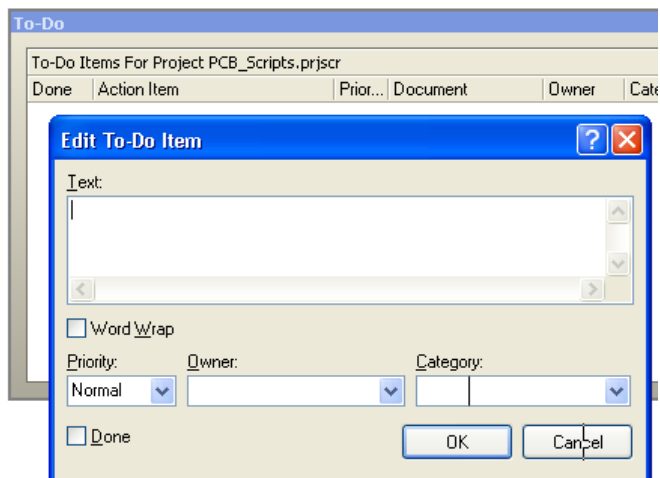
You can add a list of installed script projects so that, every time you invoke the *Select item to Run* dialog, the installed script projects (even if they are not loaded in Altium Designer) will appear along with other script projects currently open in the **Projects** panel.

Invoke **Preferences** item from **DXP » Preferences** menu and the *Preferences* dialog appears. Click on the **Scripting System** item in the left pane of the dialog and the *Installed Projects* list box appears where you can add or delete script projects.

## Using To-Do Items panel

You can use the **To\_Do Items** panel to store items that need to be completed for a project in Altium Designer. You can add Project To Do Item and you have the ability to define the information, priority, owner and the category for this item.

## A Tour of the Scripting System



## Index

<b>A</b>	
A Tour of the Scripting System.....	1
Adding scripts to a project .....	3
Assigning a script to a menu, toolbar or key .....	10
Assigning a script to a pop up menu .....	11
<b>B</b>	
Bookmarks in scripts .....	21
Breakpoint List panel.....	15
<b>C</b>	
Call Stack panel.....	15
Code completion feature .....	19
Code Explorer Panel .....	16
Creating new scripts.....	2
<b>E</b>	
Executing a script in Altium Designer.....	6
Exploring Example Scripts.....	6
Exploring the Scripting System.....	1
<b>O</b>	
Object Inspector panel.....	16
<b>P</b>	
Project and scripts .....	2
<b>S</b>	
Script Debugger.....	8
Script Editing options.....	21
Script Evaluate feature .....	18
Script Expression Tooltip.....	20
Script parameters feature.....	19
Script Symbol Insight tooltip .....	21
Script templates feature.....	20
Scripting	
Adding scripts to a project .....	3
Assigning a script to a menu, toolbar or key .....	10
Assigning a script to a pop up menu .....	11
Bookmarks .....	21
Creating new scripts .....	2
Debugging a script.....	8
Executing a script in Altium Designer .....	6
Exploring Example Scripts.....	6
Exploring the Scripting System.....	1
Project and scripts .....	2
Script Completion feature .....	19
Script Editing options .....	21
Script Evaluate feature .....	18
Script Expression Tooltip.....	20
Script parameters feature .....	19
Script Symbol Insight Tooltip .....	21
Script templates feature.....	20
Scripting panels .....	15
Scripting System Short Cut Keys.....	12
Using Altium Designer Object Interfaces in a script .....	4
Using Installed Projects list.....	21
Using To-Do Items panel.....	21
Using tools for your scripts .....	18
Writing scripts .....	3
Scripting panels .....	15
Breakpoint List panel .....	15
Call Stack panel.....	15
Code Explorer panel .....	16
Object Inspector panel.....	16
Tool Palette panel.....	17
Watch List Panel.....	18
Scripting System Short Cut Keys.....	12
<b>T</b>	
Tool Palette panel .....	17
<b>U</b>	
Using Altium Designer Object Interfaces in scripts..	4
Using Installed Projects list.....	21

## ***A Tour of the Scripting System***

Using To-Do Items panel .....21

Using tools for your scripts ..... 18

## **W**

Watch List Panel.....18

Writing scripts .....3

## Revision History

---

Date	Version No.	Revision
01-Dec-2004	1.0	New product release
26-Apr-2005	1.1	Updated for Altium Designer
08-Jul-2005	1.2	Revised
03-Oct-2005	1.3	Changed from technical reference to a guide.
09-Dec-2005	1.4	Updated for Altium Designer 6
07-May-2007	1.5	Revisions made to the text.

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.