APWIN BASIC

USER'S GUIDE AND LANGUAGE REFERENCE

Audio precision

APWIN BASIC
User's Guide and Language Reference

Version 2
August, 1999

APWIN Basic User's Guide and Language Reference

Version 2 August, 1999

Published by:

Audio precision®

Audio Precision, Inc.
PO Box 2209
Beaverton, Oregon 97075-2209
U.S. Toll Free: 1-800-231-7350
Tel: (503) 627-0832 Fax: (503) 641-8906
email: techsupport@audioprecision.com
Web: www.audioprecision.com

Printed in the United States of America
Audio Precision Part # 8211-0089

CONTENTS

Introduction **1**

APWIN Basic User's Guide and Extension Manuals 1-2

Manual Conventions 1-3

A Few Words About Terminology 1-4

Where to Find Sample Files and Examples 1-6

Getting Started In APWIN Basic 1-7

APWIN Basic Editor Overview 1-8

Editing Code with the APWIN Basic Editor 1-10

Where to Find More Information About Visual Basic 1-12

Information for Experienced Visual Basic Programmers 1-12

Fundamentals of APWIN Basic **2**

What is an APWIN Basic Program? 2-1

Using Procedures 2-2

Elements of a Procedure 2-3

Introduction to Objects, Methods, and Properties 2-10

Writing An APWIN Basic Program **3**

Converting S1.EXE Procedures to APWIN Basic 3-1

Using Learn Mode 3-6

Example APWIN Basic Program 3-8
Controlling Program Flow 3-20
Control Structures 3-21
Loop Structures 3-24

Testing and Debugging APWIN Basic Code **4**

Different Types of Programming Errors 4-1
Error Handling 4-9

Creating Custom User Interfaces **5**

Reports **6**

Language Reference **7**

Introduction 7-1
Groups 7-1
Operators 7-3
Data Types 7-5
Keywords 7-6
Language Commands 7-7
Abs 7-7
AppActivate 7-7
Array 7-8
Asc 7-8
Atn 7-9
Attribute 7-9
Beep 7-10
Begin Dialog 7-10
Call 7-11
CallByName 7-12
CallersLine 7-12
CancelButton Dialog Item 7-13
CBool 7-14
CByte 7-15
CCur 7-15
CDate 7-15
CDbl 7-16
ChDir 7-16
ChDrive 7-17

CheckBox 7-17
Choose 7-18
Chr$ 7-19
CInt 7-19
Class 7-20
Class_Initialize 7-21
Class_Terminate 7-21
Clipboard 7-22
CLng 7-22
Close 7-23
Code 7-23
ComboBox 7-23
Command$ 7-25
Const 7-25
Cos 7-26
CreateObject 7-26
CSng 7-27
CStr 7-27
CurDir$ 7-28
CVar 7-28
CVErr 7-29
Date 7-29
DateAdd 7-30
DateDiff 7-31
DatePart 7-31
DateSerial 7-32
DateValue 7-33
Day 7-33
dBToPowerRatio 7-34
dBToVoltageRatio 7-34
DDEExecute 7-35
DDEInitiate 7-35
DDEPoke 7-36
DDERequest$ 7-36
DDETerminate 7-37
DDETerminateAll 7-37
Debug 7-38
Declare 7-38
Def 7-39
DeleteSetting 7-41
Dialog 7-41
DialogFunc 7-42
Dim 7-44

Dir$ 7-45
DlgControlId 7-45
DlgCount 7-46
DlgEnable 7-47
DlgEnd 7-48
DlgFocus 7-49
DlgListBoxArray 7-50
DlgName 7-52
DlgNumber 7-53
DlgSetPicture 7-54
DlgText 7-55
DlgType 7-56
DlgValue 7-57
DlgVisible 7-59
Do 7-60
DoEvents 7-61
DropListBox 7-61
End 7-62
Enum 7-63
Environ 7-64
Eof 7-64
Erase 7-65
Err 7-65
Error 7-66
Exit 7-66
Exp 7-68
Exp10 7-68
FileAttr 7-69
FileCopy 7-69
FileDateTime 7-70
FileLen 7-70
Fix 7-71
For 7-71
For Each 7-72
Format$ 7-73
FreeFile 7-77
Function 7-78
Get 7-79
GetAllSettings 7-80
GetAttr 7-80
GetFilePath$ 7-81
GetObject 7-82
GetSetting 7-82

Goto 7-83
GroupBox Dialog Item 7-83
Hex$ 7-84
Hour 7-85
If 7-85
IIf 7-86
Input 7-86
Input$ 7-87
InputBox$ 7-88
InStr 7-88
InStrRev 7-89
Int 7-89
Is 7-90
IsArray 7-90
IsDate 7-91
IsEmpty 7-91
IsError 7-92
IsMissing 7-93
IsNull 7-94
IsNumeric 7-94
IsObject 7-95
Kill 7-96
LBound 7-96
LCase$ 7-97
Left$ 7-97
Len 7-98
Let 7-98
Like 7-98
Line Input 7-99
ListBox Dialog Item 7-99
Loc 7-100
Lock 7-101
LOF 7-102
Log 7-102
Log10 7-103
LSet 7-103
LTrim$ 7-104
MacroDir$ 7-104
MacroRun 7-105
MacroRunThis 7-105
Main 7-106
Me 7-106
Mid$ 7-107

Minute 7-108
MkDir 7-108
Month 7-109
MonthName 7-109
MsgBox 7-110
Name 7-111
Now 7-111
Oct$ 7-111
Object 7-112
Object_Initialize Sub 7-113
Object_Terminate Sub 7-113
Oct$ 7-114
OKButton Dialog Item 7-114
On Error 7-115
Open 7-116
Option 7-117
OptionButton Dialog Item 7-117
OptionGroup 7-118
Pow 7-119
Picture Dialog Item 7-119
PowerRatioTodB 7-121
Print 7-121
Private 7-122
Private 7-122
Property 7-122
Public 7-124
Public 7-124
PushButton Dialog Item 7-124
Put 7-125
QBColor 7-126
Randomize 7-128
ReDim 7-128
Reference 7-129
Rem 7-129
Replace 7-130
Reset 7-130
Resume 7-131
RGB 7-132
Right$ 7-132
RmDir 7-133
Rnd 7-133
Round 7-134
RSet 7-134

RTrim$ 7-135
SaveSetting 7-135
Second 7-136
Seek 7-136
Seek 7-137
Select Case 7-137
SendKeys 7-138
Set 7-140
SetAttr 7-141
Sgn 7-141
Shell 7-142
Sin 7-142
Space$ 7-143
Sqr 7-143
Static 7-144
Stop 7-144
Str$ 7-145
StrComp$ 7-145
StrConv$ 7-146
StrReverse$ 7-147
String$ 7-147
Sub 7-148
Tan 7-149
Text Dialog Item 7-150
TextBox Dialog Item 7-151
Time 7-152
Timer 7-152
TimeSerial 7-152
TimeValue 7-153
Trim$ 7-153
Type 7-154
TypeName 7-155
UBound 7-156
UCase$ 7-157
Unlock 7-157
Uses 7-158
Val 7-159
VarType 7-159
VoltageRatioTodB 7-161
Wait 7-161
WaitAndDoEvents 7-161
Weekday 7-162
WeekdayName 7-163

While 7-163
With 7-164
WithEvents 7-164
Write 7-165
Year 7-165

Appendix A Terms . . . . . A

Appendix B Error List . . . . . B

Introduction

Welcome to the APWIN BASIC User's Manual and Programmers Reference, your guide to creating custom test programs for Audio Precision's System One, System Two, and System Two Cascade platforms (referred to collectively as "System" throughout this Guide). APWIN Basic is a powerful and easy-to-use programming language compatible with Microsoft's Visual Basic for Applications. In this book, you'll learn how to create APWIN Basic programs (i.e. macros) that can load and run tests, automate repetitive tasks, and add custom features and functions to APWIN to suit your measurement needs.

With APWIN Macros are lists of commands that tell APWIN Basic what to do. Included with APWIN Basic are many extension commands you can use in your programs to automate control of Audio Precision's System hardware. You do not need to develop any special commands to control APWIN and its attached hardware; all of these commands are available when you begin using APWIN Basic.

One of the most exciting features in APWIN Basic is its support of OLE automation. OLE stands for Object Linking and Embedding and is a standard used in Microsoft Windows to allow OLE compliant applications to share information. Using the OLE automation features in APWIN Basic it is possible, for example, to take the results from a System measurement, move the data into any Excel spreadsheet where it can be further manipulated, then take these results into Microsoft Word where they can be inserted into a standard report form. All of this can be automated and run entirely within APWIN Basic. The results of your Word document can even be printed from inside APWIN Basic.

All of this power and functionality might lead you to think APWIN Basic is a difficult and complex programming language. In fact, APWIN Basic is one of the easiest development environments to use. Even if you have never programmed before, you will be surprised how quickly you will begin developing interesting and powerful programs.

APWIN Basic User's Guide and Extension Manuals

The following are descriptions of the *APWIN Basic User's Guide and Language Reference*, and the Basic Extension Reference manuals for each hardware platform.

APWIN Basic User's Guide and Language Reference

This book provides an introduction to programming in APWIN Basic. It is intended as a tutorial to help beginning users understand what APWIN Basic is and how to use it to develop programs. Depending on your experience programming with Visual Basic, you may want to read some or all of these introductory chapters. Section One is covered in chapters 1-6.

Chapter 7 is organized as a Language Reference and lists the generic commands available in APWIN Basic. These are the same commands you will find available in any Visual Basic compatible application.

APWIN Basic Extensions

Extensions to the commands found in chapter 7 of this User's Guide are located in specific *Extension Reference* manuals. Extension commands are used to control the operation of APWIN and specific Audio Precision System hardware. These Extension Reference manuals are:

APWIN Basic Extensions Reference for System One

APWIN Basic Extensions Reference for System Two

APWIN Basic Extensions Reference for System Two Cascade

Chapter Overviews

- Chapter 1 provides an overview of APWIN to help the first time user get started quickly. The first time user should review this chapter before continuing.

- Chapter 2 provides an introduction to the fundamentals of APWIN Basic. Several of the key concepts in Visual Basic are introduced, including objects, methods and properties, and the use of macros.
- Chapter 3 moves beyond the concepts of Visual Basic and jumps into the basics of writing a program. Working from a simple example, each of the key elements of a program is introduced and discussed. Some of the key topics discussed in this chapter include the structure of a program, syntax, and an introduction to commonly used commands.
- Chapter 4 describes how to test and debug a program. APWIN Basic provides a number of tools to assist in verifying correct operation of a program. Additional topics include tips for simplifying the debugging process, common programming mistakes to avoid, and error handling.
- Chapter 5 provides an introduction to the APWIN Basic Dialog Editor. The Dialog Editor provides an easy way of creating a user interface consisting of menus, and other dialogs that an operator can interact with to control your program.
- Chapter 6 provides an introduction and an example of how to interact with other applications using OLE to produce custom reports.
- Chapter 7 is a listing of generic APWIN Basic commands available to you regardless of which hardware is being used and are used by all applications which utilize Visual Basic-compatible commands.

Manual Conventions

This manual uses the following typographic conventions.

Example	Description
event, var, arg	For the syntax part of each command, italicized words indicate placeholders where the user must enter additional information.
FILENAME.TXT	Words in all CAPITOL letters indicate file names.

`Sub Main`
`  AP.Gen.Amp = 1.0`
`End Sub`
	This font is used in all example macros and code modules.

[*expressionlist***]**	In syntax, items inside square brackets are optional.
{*While* **|** *Until***}**	In syntax, braces and a vertical bar indicate a choice between two or more items.
Command	For the syntax part of each command, the bold characters identify the part of the command that must be entered.

`AP.Prompt. _`
`  Text "This _`
`  is just an _`
`  example."`
	The line continue character (_) is used to indicate that the code from one line to the next should be typed on one line.

A Few Words About Terminology

Audio Precision has used the term Procedure since the first product to identify a facility that will automatically run a sequence of tests. This term has in fact been used for many years in the industry to generally describe the process of performing one or more tests and/or measurements. A Test Procedure has described what to measure, how to measure it, what equipment to use and other details that a technician would need to know to carry out the task in a consistent fashion that meets the objectives of the test.

Programmers have also used the word Procedure for several years to identify specific programs or parts of programs. In particular, Visual Basic and Applications Basic uses the term Procedure to mean a specific part of a program. Unfortunately, this use of the word is at odds with the testing industry use of the term Procedure as described

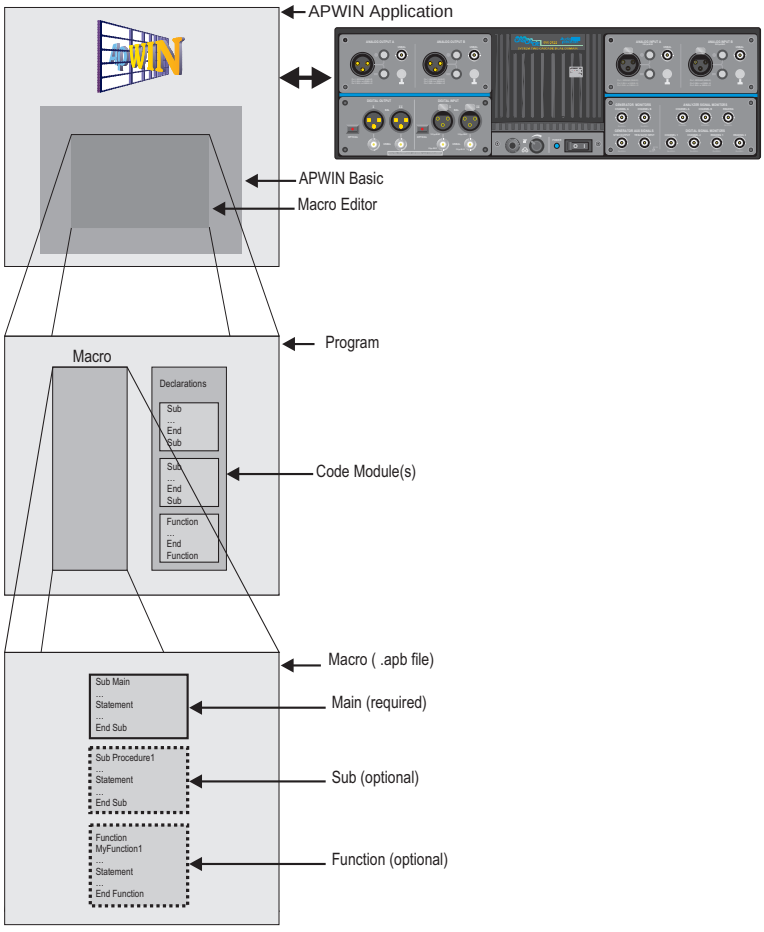

Figure 1-1

above. The programming world uses the term Macro to describe what test engineers call a Procedure.

In reference to APWIN BASIC and for purposes of this manual the term Procedure refers to Sub and Function Procedure parts of a macro or code module as shown in figure 1-1.

Where to Find Sample Files and Examples

Included with your APWIN software are sample programs for System One, System Two, and System Two Cascade hardware. If you choose to include the samples as part of your APWIN installation, they can be found under the APWIN subdirectory. Examples for each System are found under their own subdirectory. The directory structure for the sample files is shown in figure 1-2.

These examples are excellent learning tools and are representative of the type of programs your are likely to develop. You can load these macros into the APWIN Basic editor where you can edit them or even use them in whole or in part within your own program.

Samples contained in the DEMO directories are designed to be run without System hardware attached.

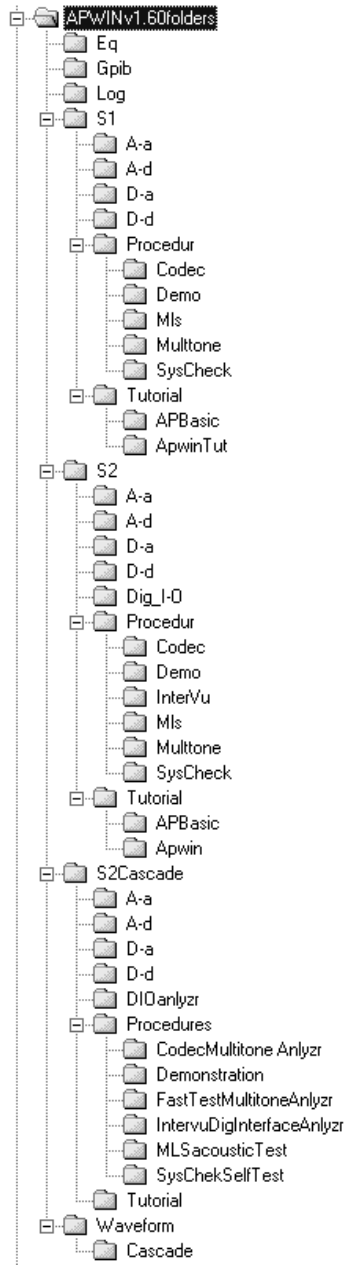

Figure 1-2 Default APWIN directory

Language Reference

Introduction

Groups

Declaration	#Reference, #Uses, Attribute, Class Module, Code Module, Const, Declare, Deftype, Dim, Enum...End Enum, Function...End Function, Object Module, Option, Private, Property...End Property, Public, ReDim, Static, Sub...End Sub, Type...End Type, WithEvents.
Assignment	Erase, Let, LSet, RSet, Set.
Flow Control	Call, CallByName, Do...Loop, End, Exit, For...Next, For Each...Next, GoTo, If...ElseIf...Else...EndIf, MacroDir, MacroRun, MacroRunThis, Select Case...End Case, Stop, While...Wend,
Error Handling	Err, Error, On Error, Resume.
Conversion	Array, CBool, CByte, CCur, CDate, CDbl, CInt, CLng, CSng, CStr, CVar, CVDate, CVErr, Val.
Variable Info	IsArray, IsDate, IsEmpty, IsError, IsMissing, IsNull, IsNumeric, IsObject, LBound, TypeName, UBound, VarType.
Math	Abs, Atn, Cos, dBToPowerRatio, dBToVoltageRatio, Exp, Exp10, Fix, Int, Log, Log10, Pow, PowerRatioTodB, Randomize, Rnd, Round, Sgn, Sin, Sqr, Tan, VoltageRatioTodB.
String	Asc, AscB, AscW, Chr, ChrB, ChrW, Format, Hex, InStr, InStrB, InStrRev, LCase, Left, LeftB, Len, LenB, LTrim, Mid, MidB, Oct, Replace, Right, RightB, RTrim, Space, String, Str, StrComp, StrReverse, StrConv, Trim, UCase.
Object	CreateObject, GetObject, Me, With...End With.
Time/Date	Date, DateAdd, DateDiff, DatePart, DateSerial, DateValue, Day, Hour, Minute, Month, MonthName, Now, Second, Time, Timer, TimeSerial, TimeValue, Weekday, WeekdayName, Year.

File	ChDir, ChDrive, Close, CurDir, Dir, EOF, FileAttr, FileCopy, FileDateTime, FileLen, FreeFile, Get, GetAttr, Input, Input, Kill, Line Input, Loc, Lock, LOF, MkDir, Name, Open, Print, Put, Reset, RmDir, Seek, Seek, SetAttr, Unlock, Write.
User Input	Dialog, GetFilePath, InputBox, MsgBox.
User Dialog	Begin Dialog...End Dialog, CancelButton, CheckBox, ComboBox, DropListBox, GroupBox, ListBox, OKButton, OptionButton, OptionGroup, Picture, PushButton, Text, TextBox.
Dialog Function	Dialog Func, DlgControlId, DlgCount, DlgEnable, DlgEnd, DlgFocus, DlgListBoxArray, DlgName, DlgNumber, DlgSetPicture, DlgText, DlgType, DlgValue, DlgVisible.
DDE	DDEExecute, DDEInitiate, DDEPoke, DDERequest, DDETerminate, DDETerminateAll.
Settings:	DeleteSetting, GetAllSettings, GetSetting, SaveSetting
Miscellaneous	AppActivate, Attribute, Beep, CallersLine, Choose, Clipboard, Command, Debug.Print, DoEvents, Environ, IIf, MacroDir, QBColor, Rem, RGB, SendKeys, Shell, Wait, WaitAndDoEvents.
Operator	Operators: +, -, ^, *, /, \, Mod, +, -, &, =, <>, <, >, <=, >=, Like, Not, And, Or, Xor, Eqv, Imp, Is.

Operators

Syntax

```
^ Not * / \ Mod + - & < <= > >= = <> Is And Or Xor
Eqv Imp
```

Description

These operators are available for numbers n1 and n2 or strings s1 and s2. If any value in an expression is *Null* then the expressions value is *Null*. The order of operator evaluation is controlled by operator *precedence*.

Operator	Description
`-n1`	Negate *n1*.
`n1 ^ n2`	Raise *n1* to the power of *n2*.
`n1 * n2`	Multiply *n1* by *n2*.
`n1 / n2`	Divide *n1* by *n2*.
`n1 \ n2`	Divide the integer value of *n1* by the integer value of *n2*.
`n1 Mod n2`	Remainder of the integer value of *n1* after dividing by the integer value of *n2*.
`n1 + n2`	Add *n1* to *n2*.
`s1 + s2`	Concatenate s1 with *s2*.
`n1 - n2`	Difference of *n1* and *n2*.
`s1 & s2`	Concatenate s1 with *s2*.
`n1 < n2`	Return *True* if *n1* is less than *n2*.
`n1 <= n2`	Return *True* if *n1* is less than or equal to *n2*.
`n1 > n2`	Return *True* if *n1* is greater than *n2*.
`n1 >= n2`	Return *True* if *n1* is greater than or equal to *n2*.
`n1 = n2`	Return *True* if *n1* is equal to *n2*.
`n1 <> n2`	Return *True* if *n1* is not equal to *n2*.
`s1 < s2`	Return *True* if s1 is less than *s2*.
`s1 <= s2`	Return *True* if s1 is less than or equal to *s2*.
`s1 > s2`	Return *True* if s1 is greater than *s2*. \
`s1 >= s2`	Return *True* if s1 is greater than or equal to *s2*.
`s1 = s2`	Return *True* if s1 is equal to *s2*.
`s1 <> s2`	Return *True* if s1 is not equal to *s2*.
`Not n1`	Bitwise invert the integer value of *n1*. Only Not *True* is *False*.
`n1 And n2`	Bitwise and the integer value of *n1* with the integer value *n2*.
`n1 Or n2`	Bitwise or the integer value of *n1* with the integer value *n2*.

n1 Xor n2	Bitwise exclusive-or the integer value of *n1* with the integer value *n2*.
n1 Eqv n2	Bitwise equivalence the integer value of *n1* with the integer value *n2* (same as Not (n1 Xor n2)).
n1 Imp n2	Bitwise implicate the integer value of *n1* with the integer value *n2* (same as (Not n1) Or n2).

Example

```
Sub Main
    N1 = 10
    N2 = 3
    S1$ = "asdfg"
    S2$ = "hjkl"
    Debug.Print -N1         '-10
    Debug.Print N1 ^ N2     ' 1000
    Debug.Print Not N1      '-11
    Debug.Print N1 * N2     ' 30
    Debug.Print N1 / N2     ' 3.3333333333333
    Debug.Print N1 \ N2     ' 3
    Debug.Print N1 Mod N2   ' 1
    Debug.Print N1 + N2     ' 13
    Debug.Print S1$ + S2$   '"asdfghjkl"
    Debug.Print N1 - N2     ' 7
    Debug.Print N1 & N2     '"103"
    Debug.Print N1 < N2     'False
    Debug.Print N1 <= N2    'False
    Debug.Print N1 > N2     'True
    Debug.Print N1 >= N2    'True
    Debug.Print N1 = N2     'False
    Debug.Print N1 <> N2    'True
    Debug.Print S1$ < S2$   'True
    Debug.Print S1$ <= S2$  'True
    Debug.Print S1$ > S2$   'False
    Debug.Print S1$ >= S2$  'False
    Debug.Print S1$ = S2$   'False
    Debug.Print S1$ <> S2$  'True
    Debug.Print N1 And N2   ' 2
    Debug.Print N1 Or N2    ' 11
    Debug.Print N1 Xor N2   ' 9
    Debug.Print N1 Eqv N2   ' -10
    Debug.Print N1 Imp N2   ' -9
End Sub
```

Data Types

Any, Boolean, Byte, Currency, Date, Double, Integer, Long, Object, Single, String, String*n, Variant, user type.

Type	Description
`Any`	Any variable expression (**Declare** only).
`Boolean`	A *True* or *False* value.
`Byte`	An 8 bit unsigned integer value.
`Cdec`	Convert a number or string value to a 96 bit scaled real.
`Currency`	A 64 bit fixed point real. (A twos complement binary value scaled by 10000.)
`Date`	A 64 bit real value. The whole part represents the date, while the fractional part is the time of day. (December 30, 1899 = 0.) Use #date# as a literal date value in a macro.
`Double`	A 64 bit real value.
`Integer`	A 16 bit integer value.
`Long`	A 32 bit integer value.
`Object`	An object reference value. (see Objects)
`PortInt`	A portable integer value. For Win16: A 16 bit integer value. For Win32: A 32 bit integer value.
`Single`	A 32 bit real value.
`String`	An arbitrary length string value.
`String*n`	A fixed length (n) string value.
`UserDialog`	A *usertype* defined by **Begin Dialog** UserDialog.
`Variant`	An empty, numeric, currency, date, string, object, error code, null or array value.

Keywords

Empty, False, Nothing, Null, True. Win16, Win32.

Word	Description
Empty	A *variantvar* that does not have any value.
False	A *condexpr* is false when its value is zero. A function that returns False returns the value 0.
Nothing	An *objexpr* that does not refer to any object.
Null	An *variant expression* that is null. A null value propagates through an expression causing the entire expression to be Null. Attempting to use a Null value as a string or numeric argument causes a run-time error. A Null value prints as #NULL#.

Example

```
Sub Main
   X = Null
   Debug.Print X = Null '(even this expression is Null)
   Debug.Print IsNull(X) '(use IsNull to test for a _
      Null value)
End Sub
```

Example Output

```
Null
True
```

True	A *conditional expression* is true when its value is non-zero. A function that returns *True* returns the value -1.
Win16	**True** if running in 16 bits. **False** if running in 32 bits.
Win32	**True** if running in 32 bits. **False** if running in 16 bits.

Language Commands

Abs

Function

Syntax **Abs(***num***)**

Parameters

Name	Description
num	Return the absolute value of this number value.

Description Return the absolute value.

Example

```
Sub Main
   Debug.Print Abs(9)
   Debug.Print Abs(0)
   Debug.Print Abs(-9)
End Sub
```

Example Output

```
 9
 0
 9
```

AppActivate

Instruction

Syntax **AppActivate** *title$*

-or-

AppActivate *TaskID*

Parameters

Name	Description
title$	The name shown in the title bar of the window.
TaskID	This numeric value is the task identifier.

Description Form 1: Activate the application top-level window titled Title$. If no window by that title exists then the first window with at title that starts with Title$ is activated. If no window matches then an error occurs.

Form 2: Activate the application top-level window for task TaskID. If no window for that task exists then an error occurs.

See Also DateSerial, DateValue, TimeValue.

Example

```
Sub Main
   Debug.Print TimeSerial(13,30,0)
End Sub
```

Example Output

```
1:30:00 PM
```

TimeValue — Function

Syntax **TimeValue(***date$***)**

Parameters

Name	Description
date$	Convert this string value to the time part of date it represents.

Description Return the time part of date encoded as a string value.

See Also DateSerial, DateValue, TimeSerial.

Example

```
Sub Main
    Debug.Print TimeValue("1/1/2000 12:00:01 AM")
End Sub
```

Example Output

```
12:00:01 AM
```

Trim$ — Function

Syntax **Trim**[$]**(***string$***)**

Parameters

Name	Description
string$	Copy this string without the leading or trailing spaces.

Description Return the string with S$s leading and trailing spaces removed.

See Also LTrim$( ), RTrim$( ).

Example

```
Sub Main
    Debug.Print ".";Trim$("  x  ");"."
End Sub
```

Example Output

```
.x.
```

Type

Definition

Syntax

```
[Private|Public] Type name
  elem [(Dim[, …])] As type[...]
End Type
```

Description

Define a new *usertype*. Each *elem* defines an element of the type for storing data. As *type* defines the type of data that can be stored. A *User-defined type variable* has a value for each *elem*. Use *.elem* to access individual element values.

Public is assumed if neither *Private* or *Public* is specified.

Example

```
Type Employee
    Name As String
    Title As String
    Salary As Double
End Type

Sub Main
    Dim e As Employee
    e.Name = "John Doe"
    e.Title = "President"
    e.Salary = 100000
    Debug.Print e.Name   '"John Doe"
    Debug.Print e.Title  '"President"
    Debug.Print e.Salary ' 100000
End Sub
```

Example Output

```
John Doe
President
100000
```

TypeName

Function

Syntax `TypeName[$](var)`

Parameters

Name	Description
var	Return a string indicating the type of value stored in this variable.

Result

Value	Description
Empty	*Variant* variable is empty. It has never been assigned a value.
Null	*Variant* variable is null.
Integer	Variable contains an *integer* value.
Long	Variable contains a *long* value.
Single	Variable contains a *single* value.
Double	Variable contains a *double* value.
Currency	Variable contains a *currency* value.
Date	Variable contains a *date* value.
String	Variable contains a *string* value.
Object	Variable contains a *object* reference that is not Nothing. (An object may return a type name specific to that type of object.)
Nothing	Variable contains a *object* reference that is Nothing.
Error	Variable contains a error code value.
Boolean	Variable contains a *boolean* value.
Variant	Variable contains a variant value. (Only used for arrays of variants.)
Unknown	Variable contains a non-OLE Automation object reference.
Byte	Variable contains a byte value.
( )	Variable contains an array value. The TypeName of the element followed by ( ).

Description Return a string indicating the type of value stored in *var*.

See Also VarType.

Example

```
Sub Main
    Dim X As Variant
    Debug.Print TypeName(X)
    X = 1
    Debug.Print TypeName(X)
```

Audio precision®

Audio Precision, Inc.
PO Box 2209
Beaverton, Oregon 97075-2209
U.S. Toll Free: 1-800-231-7350
Tel: (503) 627-0832 Fax: (503) 641-8906
email: techsupport@audioprecision.com
Web: www.audioprecision.com

Audio Precision
PO Box 2209
Beaverton Oregon 97075-2209
Tel 503 627-0832 Fax 503 641-8906
US Toll Free 1-800-231-7350
email techsupport audioprecision.com
Web www.audioprecision.com