AutoIt 3 Quick Reference

for v3.3.6.1

目錄

內容主要從 AutoIt 3 說明檔做重點式節錄，為方便查閱，採用說明檔的標題。

Language Reference

Datatypes

AutoIt 只有一種資料型態：Variant，它可存放數字或字串資料，並視使用情形決定該如何用

Numbers

Numbers can be standard decimal numbers like 2, 4.566, and -7.

Strings

字串可用單引號或雙引號： "here is a ""double-quote"" - ok?"

You can also use single-quotes like 'this' and 'here is a ' 'single-quote' ' - ok?'

Booleans

只二種：true 與 false

Datatypes and Ranges

The following table shows the internal variant datatypes and their ranges.

Data Sub-type	Range and Notes
Int32	A 32bit signed integer number.
Int64	A 64bit signed integer number
Double	A double-precision floating point number.
String	Can contain strings of up to 2147483647 characters.
Binary	Binary data, can contain up to 2147483647 bytes.
Pointer	A memory address pointer. 32bit or 64bit depending on the version of AutoIt used.

Variables

Each variable has a name (again, similar to a mailbox) and must start with the $ character and may only contain letters, numbers and the underscore _ character. Here are some example names:

$var1
$my_variable

Each variable is stored as a variant.

Declaring Variables

Variables are declared and created with the Dim, Local and Global keywords:

```
Dim $var1
```

Or you can declare multiple variables at once:

```
Dim $var1, $myvariable
```

You can also assign a variable without declaring it first, but many prefer explicit declarations.

```
$var1 = "create and assign"
```

Declaring Constants

Constants are declared and created using Const keyword like:

```
Const $const1 = 1, $const2=12
```

Constants can be declared and initialized using Enum keyword like:

```
Enum $const1 = 1, $const2, $const3 ; 1, 2, 3
Enum STEP 2 $incr0, $incr2, $incr4 ; 0, 2, 4
Enum STEP *2 $mult1, $mult2, $mult4 ; 1, 2, 4
```

Arrays

An Array is a variable containing series of data elements of the same type and size. Each element in this variable can be accessed by an index number.

Data types in Arrays

It was said that an Array contains only one datatype of the same type. But technically speaking, a Variant in AutoIt can contain anything from a number to a boolean value. So an AutoIt-Array could also contain different types, even other Arrays

This has not been strictly forbidden in AutoIt. However, it is NOT ADVISABLE to mix different datatypes in an Array.

Macros

AutoIt has an number of Macros that are special read-only variables used by AutoIt. Macros start with the @ character

Operators

Operator	Description
	Assignment operators
=	Assignment. e.g. **$var = 5** (**assigns the number 5 to $var**)
+=	Addition assignment. e.g. **$var += 1** (adds 1 **to $var**)
-=	Subtraction assignment.
*=	Multiplication assignment.
/=	Division assignment.
&=	Concatenation assignment. e.g. **$var = "one"**, and then **$var &= 10** (**$var now equals "one10"**)
	Mathematical operators
+	Adds two numbers. e.g. **10 + 20** (**equals 30**)
-	Subtracts two numbers. e.g. **20 - 10** (**equals 10**)
*	Multiplies two numbers. e.g. **20 * 10** (**equals 200**)
/	Divides two numbers. e.g. **20 / 10** (**equals 2**)
&	Concatenates/joins two strings. e.g. **"one" & 10** (**equals "one10"**)

^	Raises a number to the power. e.g. **2 ^ 4** (**equals 16**)
	Comparison operators (case insensitive if used with strings except for ==)
=	Tests if two values are equal. e.g. **If $var= 5 Then** (**true if $var equals 5**). **Case insensitive** when used with strings.
==	Tests if two **strings** are equal. **Case sensitive.** The left and right values are converted to strings if they are not strings already. This operator should only be used for string comparisons that need to be case sensitive.
<>	Tests if two values are not equal. **Case insensitive** when used with strings. To do a case sensitive not equal comparison use **Not ("string1" == "string2")**
>	Tests if the first value is greater than the second. Strings are compared lexicographically even if the contents of the string happen to be numeric.
>=	Tests if the first value is greater than or equal to the second. Strings are compared lexicographically even if the contents of the string happen to be numeric.
<	Tests if the first value is less than the second. Strings are compared lexicographically even if the contents of the string happen to be numeric.
<=	Tests if the first value is less than or equal to the second. Strings are compared lexicographically even if the contents of the string happen to be numeric.
	Logical operators
AND	Logical AND operation. e.g. **If $var = 5 AND $var2 > 6 Then** (**True if $var equals 5 and $var2 is greater than 6**)
OR	Logical OR operation. e.g. **If $var = 5 OR $var2 > 6 Then** (**True if $var equals 5 or $var2 is greater than 6**)
NOT	Logical NOT operation. e.g. **NOT 1** (**False**)

operator precedence. The precedence used in AutoIt is given below. Where two operators have the same precedence the expression is evaluated left to right.

From highest precedence to lowest:
NOT
^
* /
+ -
&
< > <= >= = <> ==
AND OR

Conditional Statements

If...Then...Else	**Select...Case**

If <expression> Then
    statements
    ...
[ElseIf expression-n Then
    [elseif statements ...]]
    ...
	Select
	    Case <expression>
	        statement1
	        ...
	    [Case
	        statement2

[Else	...]
[else statements]	[Case Else
...	statementN
EndIf	...]
	EndSelect

Switch...Case

```
Switch <expression>
    Case <value> [To <value>] [,<value> [To <value>] ...]
        statement1
        ...
    [Case <value> [To <value>] [,<value> [To <value>] ...]
        statement2
        ...]
    [Case Else
        statementN
        ...]
EndSwitch
```

Loop Statements

For...Next

```
For <variable> = <start> To <stop> [Step <stepval>]
    statements
    ...
Next
```

While...WEnd

```
While <expression>
    statements
    ...
WEnd
```

Do...Until

```
Do
    statements
    ...
Until <expression>
```

For...In...Next

Enumerates elements in an Object collection or an array

```
For <$Variable> In <expression>
    statements
    ...
Next
```

Obj Statements

An obj is how you refer to an object. You might want to enumerate elements in an Object collection

The following obj statements are available in AutoIt:

With...Endwith

```
With <expression>
    statements
    ...
EndWith
```

For...In...Next

User Functions

A function is a section of code that can be called from the script to perform a certain "function". There are two sorts of functions in AutoIt, built-in functions and user functions.

User functions are declared using the Func...EndFunc statements.

Comments

Although only one statement per line is allowed, a long statement can span multiple lines if an underscore " _" preceded by a blank is placed at the end of a "broken" line. String definition cannot be split in several lines, concatenation need to be used.

The semicolon (;) is the comment character.

It is also possible to comment of large blocks of script by using the #comments-start and #comments-end directives.

Keyword Reference

Keyword	Description
False / True	Boolean values for use in logical expressions.
#comments-start	Specify that an entire section of script should be commented out.
ContinueCase	Abort the current case and continue a case into the next case in a Select or Switch block.
ContinueLoop	Continue a While/Do/For loop.
Default	Keyword value use in function call.
Dim / Global / Local / Const	Declare a variable, a constant, or create an array.
Do...Until	Loop based on an expression.
Enum	Enumerates constants.
Exit	Terminates the script.
ExitLoop	Terminate a While/Do/For loop.
For...To...Step...Next	Loop based on an expression.
For...In...Next	Enumerates elements in an Object collection or an array
Func...Return...EndFunc	Defines a user-defined function that takes zero or more arguments and optionally returns a result.
If...Then	Conditionally run a single statement.

If...ElseIf...Else...EndIf	Conditionally run statements.
#include-once	Specifies that the current file may only be included once.
#include	Includes a file in the current script.
#NoAutoIt3Execute	Specifies that the current compiled script cannot run with /AutoIt3ExecuteLine or /AutoIt3ExecuteScript switch.
#NoTrayIcon	Indicates that the AutoIt tray icon will not be shown when the script starts.
#OnAutoItStartRegister	Registers a function to be called when AutoIt starts.
ReDim	Resize an existing array
#RequireAdmin	Specifies that the current script requires full administrator rights to run.
Select...Case...EndSelect	Conditionally run statements.
Static	Declare a static variable or create a static array.
Switch...Case...EndSwitch	Conditionally run statements.
While...WEnd	Loop based on an expression.
With...EndWith	Used to reduce long references to object type variables

Macro Reference

有以下幾類，用時查閱即可

AutoIt Related	Directory	System Info	Time And Date

Function Reference

必要時查 Help 中的分類

Environment	Message Boxes and Dialogs	Registry
File, Directory and Disk	Misc. functions	String
Graphic and Sound	Mouse	Timer and Delay
GUI	Network	Tray
Keyboard	Obj/COM	Variables and Conversions
Math	Process	Window

GUI Reference

要產生圖形界時，可查這裡

COM Extensions

用 COM 物件時查這裡

Appendix

一些列表供查詢

Send Key list

Windows Message Codes

User Defined Function Reference

AutoIt 3 有很多現成 UDF，#include 就可以用

歷史版號記錄：v.0-2012-4-8