

Variablen Notation und Datentypen

\$aFruitsList	Array (\$a)
\$bIsVisible	Bool (\$b)
\$dBinData	Binary (\$d)
\$fPrice	Float (\$f)
\$hGui	Handle (\$h)
\$iNumber	Integer (\$i)
\$idButtonOk	GUI control id (\$i)
\$mPairs	Map (\$m)
\$oExcel	Object (\$o)
\$pRect	Pointer (\$p)
\$sText	String (\$s)
\$tSTRUCT	Struct (\$t)
\$vData	Variant (\$v)

☰ [Komplette Liste hier](#)

Variablen Geltungsbereich

Global	Geltungsbereich über das gesamte Programm
Local	Geltungsbereich bezogen auf den Codeblock
Dim	Hybrid-ähnlicher Geltungsbereich, lokal, falls nicht bereits global vorhanden

❗ Tipp: Vermeide Dim, verwende explizit Local oder Global

❗ Tipp: Verwende Global nicht innerhalb von Funktionen

Zuweisungsoperatoren

=	Zuweisung	\$sText = 'Hi'
&	Verkettet/verbindet zwei Strings	\$sText & ' Max'
&=	Verkettungs-zuweisung	\$sText &= ' Max' gleichbedeutend mit \$sText = \$sText & ' Max'
+=	Additionszuweisung	\$iNum += 1 gleichbedeutend mit \$iNum = \$iNum + 1

Zuweisungsoperatoren (cont)

-=	Subtraktionszuweisung	\$iNum -= 1 gleichbedeutend mit \$iNum = \$iNum - 1
*=	Multiplikative Zuweisung	\$iNum *= 2 gleichbedeutend mit \$iNum = \$iNum * 2
/=	Divisive Zuweisung	\$iNum /= 2 gleichbedeutend mit \$iNum = \$iNum / 2

Arithmetische Operatoren

+	Addition	10 + 20
-	Subtraktion	20 - 10
*	Multiplikation	20 * 10
/	Division	20 / 10
^	Potenzieren	2 ^ 4
Mod	Modulo	Mod(val1, val2)

Vergleichsoperatoren

=	Gleich (GKu)	If \$sText = 'Hi' Then ..
==	Streng gleich (GK)	If \$sText == 'Bye' Then ..
<>	Ungleich (GKu)	If \$sText <> 'hallo' Then ..
>	Größer als (LV)	
>=	Größer als oder gleich (LV)	
<	Kleiner als (LV)	
<=	Kleiner als oder gleich (LV)	

❗ Legende: GK: Groß- und Kleinschreibung unbeachtet, GK: Groß- und Kleinschreibung beachtet, LV: lexikografisch verglichen



By **SOLVE-SMART** (SOLVE-SMART)
cheatography.com/solve-smart/

github.com/Sven-Seyfert

Published 23rd February, 2023.
 Last updated 23rd February, 2023.
 Page 1 of 4.

Sponsored by **CrosswordCheats.com**
 Learn to solve cryptic crosswords!
<http://crosswordcheats.com>

Konditionaler Operator

? :	Ternary	Siehe Bereich "Verzweigung"
-----	---------	-----------------------------

Logische Operatoren

And	Logisch Und	If \$iNum1 = 5 And \$iNum2 > 6 Then ..
-----	-------------	---

Or	Logisch Oder	If \$iNum1 = 5 Or \$iNum2 > 6 Then ..
----	--------------	--

Not	Logisch Nicht (Negation)	If Not \$iNum = 5 Then ..
-----	-----------------------------	---------------------------

Kommentare

;	Einzeiliger Kommentar
---	-----------------------

```
#cs
    Dies ist ein
    mehrzeiliger Kommentar
#ce
```

Zeichenketten und Zeichen-Maskierung

"Erster string"	Zeichenketten werden in doppelte Anführungszeichen eingeschlossen
-----------------	---

'Zweiter string'	Zeichenketten können auch in einfache Anführungszeichen eingeschlossen werden
------------------	---

"Er ist ""Max"""	Doppeltes Anführungszeichen als Maskierungszeichen
------------------	--

'Er ist "Max"'	Oder stattdessen einfache Anführungszeichen
----------------	---

'Max auf"m Fahrrad'	Einfaches Anführungszeichen als Maskierungszeichen
---------------------	--

"Max auf'm Fahrrad"	Oder stattdessen doppelte Anführungszeichen
---------------------	---

☰ [Komplette Liste hier](#)

Funktionen

Ohne Parameter	Func _PrintHello() ConsoleWrite('Hello' & @CRLF) EndFunc
----------------	--

Funktionen (cont)

Mit Parameter	Func _PrintText(\$sText) ConsoleWrite(\$sText & @CRLF) EndFunc
---------------	--

Mit optionalen Parametern	Func _SendWithDelay(\$sKeys, \$iDelay = 150) Send(\$sKeys) Sleep(\$iDelay) EndFunc
---------------------------	---

Mit Default-Keyword Parameter	Func _SendWithDelay(\$sKeys, \$iDelay = Default) \$iDelay = (\$iDelay == Default) ? 150 : \$iDelay Send(\$sKeys) Sleep(\$iDelay) EndFunc
-------------------------------	--

📌 Funktionsaufrufe:

```
_PrintHello(){
_PrintText('Dies ist ein Text')
_SendWithDelay('{ENTER}')
_SendWithDelay('{ENTER}', 300)
_SendWithDelay('{ENTER}', Default)
```

Schleifen

For..Next	For \$i = 1 To 10 Step 1 ConsoleWrite(\$i & @CRLF) Next
-----------	---

For..In..Next	For \$sFruit In \$aFruitsList ConsoleWrite(\$sFruit & @CRLF) Next
---------------	---

While..WEnd	While \$iNum < 25 Sleep(200) \$iNum += 1 WEnd
-------------	--

Do..Until	Do Sleep(200) \$iNum += 1 Until \$iNum = 25
-----------	--



Verzweigungen

If..Elseif..Else	<pre>If \$iNum = 1 Then ; tue etwas Elseif \$iNum = 2 Then ; tue etwas Else ; tue etwas EndIf</pre>
Select	<pre>Select Case \$iNum = 42 ; tue etwas Case \$sText = 'Test' ; tue etwas Case Else ; tue etwas EndSelect</pre>
Switch	<pre>Switch @HOUR Case 6 To 11 \$sMessage = 'Guten Morgen' Case 12 To 17 \$sMessage = 'Hallo' Case Else \$sMessage = 'Es ist nach 17 Uhr' EndSwitch</pre>
Ternary	<pre>\$sVariable = (Bedingung) ? 'dannAusdruck' : 'sonstAusdruck' \$bIsAnswerToLife = (\$iNum = 42) ? True : False</pre>

Kommandozeile

\$CmdLine[0]	Gesamtanzahl der angegebenen Kommandozeilenparameter
\$CmdLine[1]	Erster Parameter (Wert) des angegebenen Kommandozeilenaufrufs

i Kommandozeilenaufruf (bspw. als GUI Parameter):
C:\Development\Autolt>MeinProgramm.exe "800" "600"
\$GuiWidth = \$CmdLine[1]
\$GuiHeight = \$CmdLine[2]

Makros (spezielle schreibgeschützte Variablen)

@Compiled	Gibt an, ob das Skript kompiliert ist oder nicht
@CRLF	Wagenrücklauf Zeilenvorschub
@error	Status des Fehlerflags
@extended	Erweiterte Funktionsrückgabe
@HOUR	Stundenwert
@MIN	Minutenwert
@SEC	Sekundenwert
@ScriptDir	Verzeichnis des ausgeführten Skripts

☰ [Komplette Liste hier](#)

Direktiven (Richtlinien)

#include	Fügt eine Datei in das aktuelle Skript ein #include <Array.au3> oder #include "Pfad\Dateiname.au3"
#include-once	Aktuelle Datei darf nur einmal eingebunden werden
#NoTrayIcon	Das Autolt-Taskleistensymbol wird nicht angezeigt
#OnAutoltStartRegister	Registriert eine Funktion, die beim Start von Autolt aufgerufen werden soll
#pragma	Ändert, wie das Skript kompiliert wird
#RequireAdmin	Für die Ausführung des aktuellen Skripts sind vollständige Administratorrechte erforderlich

Au3Check* allgemeine Parameter

-d	Wie Opt("MustDeclareVars", 1)
-w 1	Bereits eingebundene Datei (an)
-w 2	Fehlende #Kommentare-Ende (an)
-w 3	Bereits deklarierte Variablen (aus)
-w 4	Lokale Variable im globalen Bereich verwendet (aus)



By **SOLVE-SMART** (SOLVE-SMART)
cheatography.com/solve-smart/

github.com/Sven-Seyfert

Published 23rd February, 2023.
Last updated 23rd February, 2023.
Page 3 of 4.

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
<http://crosswordcheats.com>

Au3Check* allgemeine Parameter (cont)

-w 5	Lokale Variable deklariert, aber nicht verwendet (aus)
-w 6	Warnen bei Verwendung von Dim (aus)
-w 7	Warnung bei Übergabe von Const oder Ausdruck auf ByRef-Parameter(n) (ein)

i Au3Check* überprüft das Skript auf Syntaxfehler

≡ Komplette Liste [hier](#)

Au3Stripper* allgemeine Parameter

/sf	Alle ungenutzten Funktionen entfernen
/sv	Nicht verwendete globale und lokale Variablen-deklarationen entfernen
/mo	Fügt die Include-Dateien in das Skript ein und entfernt die Kommentare. Dies ist ähnlich wie aut2exe und hilft beim Auffinden der Fehlerzeile.
/rm	Benennt Variablen- und Funktionsnamen zu kürzeren Namen um
/rsln	Ersetzt @ScriptLineNumber durch die tatsächliche Zeilennummer

i Au3Stripper* bereinigt das Skript durch verschiedene Optionen

≡ Komplette Liste [hier](#)



By **SOLVE-SMART** (SOLVE-SMART)
cheatography.com/solve-smart/

github.com/Sven-Seyfert

Published 23rd February, 2023.
Last updated 23rd February, 2023.
Page 4 of 4.

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
<http://crosswordcheats.com>