

DFÜ-Skript-Befehlssprache

Zur Unterstützung bei der Skripterstellung für das DFÜ-Netzwerk

Inhaltsverzeichnis

- 1.0 Übersicht
- 2.0 Grundstruktur eines Skripts
- 3.0 Variablen
- 3.1 Systemvariablen
- 4.0 Zeichenfolgenliterals
- 5.0 Ausdrücke
- 6.0 Kommentare
- 7.0 Schlüsselwörter
- 8.0 Befehle
- 9.0 Reservierte Wörter

1.0 Übersicht

Bei vielen Internet-Diensteanbietern und Online-Diensten müssen Sie Informationen wie Ihren Benutzernamen und Ihr Kennwort manuell eingeben, um eine Verbindung herzustellen. Mit der Unterstützung bei der Skripterstellung für das DFÜ-Netzwerk können Sie ein Skript schreiben, um diesen Vorgang zu automatisieren.

Ein Skript ist eine Textdatei mit einer Folge von Befehlen, Parametern und Ausdrücken, die ein Internet-Diensteanbieter oder Online-Dienst benötigt, um die Verbindung herzustellen und den Dienst zu nutzen. Sie können eine Skriptdatei mit jedem beliebigen Texteditor, z.B. dem Microsoft Editor, erstellen. Nachdem Sie die Skriptdatei erstellt haben, können Sie sie einer bestimmten DFÜ-Netzwerk-Verbindung zuweisen, indem Sie den DFÜ-Skript-Editor ausführen.

2.0 Grundstruktur eines Skripts

Ein Befehl ist die grundlegende Anweisung in einer Skriptdatei. Einige Befehle erfordern Parameter, die genauer definieren, welche Aufgabe der Befehl ausführen soll. Ein Ausdruck ist eine Kombination aus Operatoren und Argumenten, die ein Ergebnis erstellen. Ausdrücke können als Werte in jedem beliebigen Befehl verwendet werden. Beispiele für Ausdrücke sind arithmetische Vergleiche, Vergleichsoperatoren und Zeichenfolgenverkettungen.

Die Grundform eines Skripts für das DFÜ-Netzwerk sieht wie folgt aus:

```
;
; Ein Kommentar beginnt mit einem Semikolon und reicht bis
; zum Ende der Zeile.
;

proc main
    ; Ein Skript kann eine beliebige Anzahl Variablen
    ; und Befehle enthalten.
```

Kadir Mekic <kmekic@comonline.net>

Variablendeklarationen

Befehlsblock

endproc

Ein Skript muß eine Hauptprozedur enthalten, die durch das Schlüsselwort **proc** eingeleitet wird und mit dem Schlüsselwort **endproc** endet.

Sie müssen Variablen deklarieren, bevor Sie Befehle hinzufügen. Der erste Befehl in der Hauptprozedur wird ausgeführt. Anschließend werden alle nachfolgenden Befehle in der Reihenfolge ausgeführt, in der sie im Skript aufgeführt sind. Das Skript endet, wenn das Ende der Hauptprozedur erreicht wird.

3.0 Variablen

Skripts können Variablen enthalten. Variablenamen müssen mit einem Buchstaben oder einem Unterstrich ('_') anfangen und können eine beliebige Folge von Groß- oder Kleinbuchstaben, Ziffern und Unterstrichen enthalten. Ein reserviertes Wort kann nicht als Variablenname verwendet werden. Weitere Informationen finden Sie in der Liste der reservierten Wörter am Ende dieses Dokuments.

Sie müssen Variablen deklarieren, bevor Sie sie verwenden. Wenn Sie eine Variable deklarieren, müssen Sie auch ihren Typ definieren. Eine Variable eines bestimmten Typs kann nur Werte desselben Typs enthalten. Die folgenden drei Variablentypen werden unterstützt:

<u>Typ</u>	<u>Beschreibung</u>
Integer (Ganzzahl)	Eine negative oder positive Zahl, wie z.B. 7, -12 oder 5698.
String (Zeichenfolge)	Eine Folge von Zeichen in Anführungszeichen; z.B. "Hallo Welt!" oder "Geben Sie ein Kennwort ein:".
Boolean (boolesch)	Ein logischer boolescher Wert von TRUE (Wahr) oder FALSE (Falsch).

Variablen werden Werten mit der folgenden Anweisung zugewiesen:

Variable = Ausdruck

Die Variable erhält den ausgewerteten Ausdruck.

Beispiele:

```
integer count = 5
integer timeout = (4 * 3)
integer i

boolean bDone = FALSE

string szIP = "getip 2"

set ipaddr szIP
```

3.1 Systemvariablen

Systemvariablen werden durch Befehle für die Skripterstellung festgelegt oder durch die Informationen bestimmt, die Sie beim Einrichten einer DFÜ-Netzwerk-Verbindung eingeben. Systemvariablen sind schreibgeschützt und können deshalb im Skript nicht geändert werden. Es gibt folgende Systemvariablen:

Kadir Mekic <kmekic@comonline.net>

<u>Name</u>	<u>Typ</u>	<u>Beschreibung</u>
\$USERID	String	Die Benutzererkennung für die aktuelle Verbindung. Diese Variable ist der Wert des im Dialogfeld Verbinden mit des DFÜ-Netzwerks angegebenen Benutzernamens.
\$PASSWORD	String	Das Kennwort für die aktuelle Verbindung. Diese Variable ist der Wert des im Dialogfeld Verbinden mit des DFÜ-Netzwerks angegebenen Benutzernamens.
\$SUCCESS	Boolean	Diese Variable wird durch bestimmte Befehle festgelegt und soll anzeigen, ob der Befehl erfolgreich ausgeführt wurde oder nicht. Ein Skript kann anhand des Wertes dieser Variablen Entscheidungen treffen.
\$FAILURE	Boolean	Diese Variable wird durch bestimmte Befehle festgelegt und soll anzeigen, ob bei der Ausführung des Befehls ein Fehler auftrat oder nicht. Ein Skript kann anhand des Wertes dieser Variablen Entscheidungen treffen.

Diese Variablen können an allen Stellen eingesetzt werden, an denen ein Ausdruck eines ähnlichen Typs verwendet wird. So ist z.B.

```
transmit $USERID
```

ein gültiger Befehl, weil \$USERID eine Variable vom Typ "String" ist.

4.0 Zeichenfolgenliterals

Die Skripterstellung für das DFÜ-Netzwerk unterstützt Escape-Folgen und Übersetzungen des Caret-Zeichens entsprechend der folgenden Beschreibung.

<u>Zeichenfolgenliteral</u>	<u>Beschreibung</u>
<code>^Zchn</code>	Übersetzung des Caret-Zeichens Ist <i>Zchn</i> ein Wert zwischen '@' und '_', wird die Zeichenfolge in einen Ein-Byte-Wert zwischen 0 und 31 übersetzt. Zum Beispiel wird ^M in einen Wagenrücklauf umgewandelt. Ist <i>Zchn</i> ein Wert zwischen a und z, wird die Zeichenfolge in einen Ein-Byte-Wert zwischen 1 und 26 übersetzt. Ist <i>Zchn</i> irgendein anderer Wert, wird die Zeichenfolge nicht besonders behandelt.
<code><cr></code>	Wagenrücklauf
<code><lf></code>	Zeilenvorschub
<code>\"</code>	Anführungszeichen
<code>\^</code>	Einfaches Caret-Zeichen
<code>\<</code>	Einfaches '<'
<code>\\</code>	Umgekehrter Schrägstrich

Beispiele:

Kadir Mekic <kmekic@comonline.net>

```
transmit "^M"  
transmit "Hans^M"  
transmit "<cr><lf>"  
waitfor "<cr><lf>"
```

5.0 Ausdrücke

Ein Ausdruck ist eine Kombination aus Operatoren und Argumenten, die ein Ergebnis auswerten. Ausdrücke können als Werte in einem Befehl verwendet werden.

Ein Ausdruck kann jede Variable oder aber Integer-, String- oder Boolean-Werte mit jedem der unären und binären Operatoren in den folgenden Tabellen kombinieren. Alle unären Operatoren haben höchste Priorität. Die Priorität von binären Operatoren wird durch ihre Position in der Tabelle angegeben.

Unär sind die folgenden Operatoren:

<u>Operator</u>	<u>Art der Operation</u>
-	Unäres Minus
!	Einerkomplement

Die binären Operatoren sind in der folgenden Tabelle in der Reihenfolge ihrer Priorität aufgeführt. Operatoren mit höherer Priorität sind zuerst aufgeführt:

<u>Operatoren</u>	<u>Art der Operation</u>	<u>Einschränkungen beim Typ</u>
* /	Multiplikation	Integer
+ -	Addition	Integer, String (nur +)
< > <= >=	Relational	Integer
== !=	Gleichheit	Integer, String, Boolean
and	Logisches UND	Boolean
or	Logisches ODER	Boolean

Beispiele:

```
count = 3 + 5 * 40  
transmit "Hallo" + " Ihr"  
delay 24 / (7 - 1)
```

6.0 Kommentare

Der gesamte Text in einer Zeile hinter einem Semikolon wird ignoriert.

Beispiele:

```
; dies ist ein Kommentar  
  
transmit "Hallo" ; Zeichenfolge "Hallo" übertragen
```

7.0 Schlüsselwörter

Schlüsselwörter legen die Struktur des Skripts fest. Im Gegensatz zu Befehlen führen Schlüsselwörter keine Aktion aus. Die Schlüsselwörter werden im folgenden aufgelistet.

proc *Name*

Kadir Mekic <kmekic@comonline.net>

Bezeichnet den Anfang einer Prozedur. Alle Skripts müssen eine Hauptprozedur (**proc main**) enthalten. Die Skriptausführung beginnt bei der Hauptprozedur und endet am Ende der Hauptprozedur.

endproc

Bezeichnet das Ende einer Prozedur. Wenn das Skript bis zur Anweisung **endproc** für die Hauptprozedur ausgeführt worden ist, startet das DFÜ-Netzwerk PPP oder SLIP.

integer *Name* [= *Wert*]

Deklariert eine Variable vom Typ "Integer". Sie können die Variable mit jedem numerischen Ausdruck oder jeder Variablen initialisieren.

string *Name* [= *Wert*]

Deklariert eine Variable vom Typ "String". Sie können die Variable mit jedem Zeichenfolgenliteral oder jeder Variablen initialisieren.

boolean *Name* [= *Wert*]

Deklariert eine Variable vom Typ "Boolean". Sie können die Variable mit jedem booleschen Ausdruck oder jeder Variablen initialisieren.

8.0 Befehle

Alle Befehle sind reservierte Wörter. Deshalb können Sie keine Variablen deklarieren, die dieselben Namen wie die Befehle haben. Die Befehle werden im folgenden aufgelistet.

delay *nSekunden*

Hält während der durch *nSekunden* angegebenen Anzahl Sekunden an, bevor der nächste Befehl im Skript ausgeführt wird.

Beispiele:

```
delay 2      ; hält 2 Sekunden lang an
delay x * 3  ; hält x * 3 Sekunden lang an
```

getip *Wert*

Wartet darauf, eine IP-Adresse vom Remote-Computer zu erhalten. Wenn der Internet-Diensteanbieter mehrere IP-Adressen in einer Zeichenfolge zurückgibt, legen Sie mit dem Parameter *Wert* fest, welche IP-Adresse das Skript verwenden soll.

Beispiele:

```
; zweite IP-Adresse erhalten
set ipaddr getip 2

; erste erhaltene IP-Adresse einer Variablen zuweisen
szAddress = getip
```

goto *Marke*

Kadir Mekic <kmekic@comonline.net>

Springt zu der im Skript durch *Marke* angegebenen Position und fährt mit der Ausführung der auf die Marke folgenden Befehle fort.

Beispiel:

```
waitfor "Prompt>" until 10
if !$SUCCESS then
    goto Übergabe; springt zu "Übergabe" und führt die
                                ; sich daran anschließenden Befehle aus
endif

transmit "bbs^M"
goto End

Übergabe:
transmit "^M"
```

halt

Beendet das Skript. Dieser Befehl entfernt nicht das Terminal-Dialogfeldfenster. Sie müssen auf **Weiter** klicken, um die Verbindung herzustellen. Das Skript kann nicht erneut gestartet werden.

if *Bedingung* then
 Befehle
endif

Führt die Folge von *Befehlen* aus, wenn *Bedingung* TRUE ist.

Beispiel:

```
if $USERID == "Hans" then
    transmit "Hansi^M"
endif
```

Marke :

Gibt die Stelle im Skript an, zu der der Sprung erfolgen soll. Eine Marke muß ein eindeutiger Name entsprechend den Namenskonventionen für Variablen sein.

set port databits 5 | 6 | 7 | 8

Ändert die Anzahl der Bits in den während der Sitzung übertragenen und empfangenen Bytes. Die Bitanzahl kann zwischen 5 und 8 betragen. Wenn Sie diesen Befehl nicht einbeziehen, verwendet das DFÜ-Netzwerk die für die Verbindung festgelegten Eigenschafteneinstellungen.

Beispiel:

```
set port databits 7
```

set port parity none | odd | even | mark | space

Ändert das Paritätsschema für den Anschluß während der Sitzung. Wenn Sie diesen Befehl nicht einbeziehen, verwendet das DFÜ-Netzwerk die für die Verbindung festgelegten Eigenschafteneinstellungen.

Beispiel:

Kadir Mekic <kmekic@comonline.net>

```
set port parity even
```

set port stopbits 1 | 2

Ändert die Anzahl der Stopbits für den Anschluß während der Sitzung. Diese Zahl kann 1 oder 2 sein. Wenn Sie diesen Befehl nicht einbeziehen, verwendet das DFÜ-Netzwerk die für die Verbindung festgelegten Eigenschafteneinstellungen.

Beispiel:

```
set port stopbits 2
```

set screen keyboard on | off

Aktiviert oder deaktiviert die Tastatureingabe im Terminalfenster für die Skripterstellung.

Beispiel:

```
set screen keyboard on
```

set ipaddr *Zeichenfolge*

Gibt die IP-Adresse der Arbeitsstation für die Sitzung an. *Zeichenfolge* muß in Form einer IP-Adresse eingegeben werden.

Beispiele:

```
szIPAddress = "11.543.23.13"  
set ipaddr szIPAddress  
  
set ipaddr "11.543.23.13"  
  
set ipaddr getip
```

transmit *Zeichenfolge* [, raw]

Sendet die durch *Zeichenfolge* angegebenen Zeichen an den Remote-Computer.

Der Remote-Computer erkennt Escape-Folgen und Übersetzungen des Caret-Zeichens, außer wenn Sie den Parameter **raw** in den Befehl einbeziehen. Der Parameter **raw** ist zweckmäßig bei der Übertragung der Systemvariablen \$USERID und \$PASSWORD, wenn der Benutzername oder das Kennwort Zeichenfolgen enthalten, die ohne diesen Parameter als Caret-Zeichen oder Escape-Folgen interpretiert würden.

Beispiele:

```
transmit "slip" + "^M"  
transmit $USERID, raw
```

waitfor *Zeichenfolge* [, matchcase] [then *Marke* { , *Zeichenfolge* [, matchcase] then *Marke* }] [until *Zeit*]

Kadir Mekic <kmekic@comonline.net>

Wartet, bis der Computer eine oder mehrere der angegebenen Zeichenfolgen vom Remote-Computer empfängt. Der Parameter *Zeichenfolge* ist unabhängig von Groß-/Kleinschreibung, außer wenn Sie den Parameter **matchcase** einbeziehen.

Wenn eine entsprechende Zeichenfolge empfangen und der Parameter **then Marke** verwendet wird, springt dieser Befehl zu der Stelle in der Skriptdatei, die durch *Marke* bezeichnet wird.

Der optionale Parameter **until Zeit** definiert die maximale Anzahl Sekunden, während der der Computer auf den Empfang der Zeichenfolge wartet, bevor er den nächsten Befehl ausführt. Ohne diesen Parameter wartet der Computer endlos lang.

Wenn der Computer eine der angegebenen Zeichenfolgen empfängt, wird die Systemvariable \$SUCCESS auf TRUE eingestellt. Andernfalls wird sie auf FALSE eingestellt, wenn die durch *Zeit* festgelegten Sekunden vor Empfang der Zeichenfolge ablaufen.

Beispiele:

```
waitfor "Anmelden:"

waitfor "Kennwort?", matchcase

waitfor "Prompt>" until 10

waitfor
    "Anmelden:" then DoLogin,
    "Kennwort:" then DoPassword,
    "BBS:"      then DoBBS,
    "Andere:"   then DoOther
until 10
```

while *Bedingung* **do**
 Befehle
endwhile

Führt die Folge von *Befehlen* aus, bis *Bedingung* den Wert FALSE zurückgibt.

Beispiel:

```
integer count = 0

while count < 4 do
    transmit "^M"
    waitfor "Anmelden:" until 10
    if $SUCCESS then
        goto DoLogin
    endif
    count = count + 1
endwhile
...
```

9.0 Reservierte Wörter

Die folgenden Wörter sind reserviert und können nicht als Variablennamen verwendet werden.

and	boolean	databits	delay
do	endif	endproc	endwhile
even	FALSE	getip	goto
halt	if	integer	ipaddr
keyboard	mark	matchcase	none

Kadir Mekic <kmekic@comonline.net>

odd	off	on	or
parity	port	proc	raw
screen	set	space	stopbits
string	then	transmitTRUE	
until	waitfor	while	