Batch-Programmierung: Batch-Befehle

1 Wichtiger Hinweis zu diesem Abschnitt

Diese Liste ist nicht vollständig, weitere Befehle werden in den anderen Kapiteln dieses Buches erklärt. Mit der Zeit sollen alle Befehle in diese Liste übertragen werden, um aus den anderen Kapitel referenziert werden zu können.

2 Vorsicht Leerzeichen!

Fehlplatzierte oder fehlende Leerzeichen können beim Programmieren einer Batch-Datei zu Fehlern führen. Bei den nachfolgenden Beispielen ist also auf die Platzierung von Leerzeichen und auf entsprechende Bemerkungen genau zu achten. Scheinbar grundlose Abbrüche beim Ausführen einer Batch-Datei können ebenfalls fehlplatzierten oder fehlenden Leerzeichen geschuldet sein.

3 @

Schaltet die Ausgabe der Befehlszeile auf dem Bildschirm nur für den aktuellen Befehl aus und ist selbst kein eigener Befehl.

Syntax

@befehl

Stapelanweisung: Ab MS-DOS bis Windows 7

Beispiel:

Inhalt

echo Diese Zeile wird mit Befehlszeile ausgefuehrt... @echo und diese ohne!

Ausgabe

C:\>echo Diese Zeile wird mit Befehlszeile ausgefuehrt... Diese Zeile wird mit Befehlszeile ausgefuehrt... und diese ohne!

In Batch Files verhindert "@echo off" zu Beginn des Skriptes die Ausgabe aller (!) Befehlszeilen auf dem Bildschirm bis die Stapelverarbeitung beendet wird, abbricht oder mittendrin ein "@echo on" Befehl erfolgt, um z. B. Befehlszeilen tatsächlich anzuzeigen und dann auszuführen. Kommentare (mit :: oder REM) werden natürlich auch nicht angezeigt. Ist aber nur ein Nebeneffekt. Bei REM sollte man jedoch unbedingt aufpassen, da dort ungewollt der "remove" Befehl gestartet werden kann, was zu massiven Datenverlust führt, weshalb ich die Benutzung von "REM" nicht empfehlen würde.

@echo off :: Verhindert, dass dieser Kommentar angezeigt wird.

Ohne @echo off:

C:\>:: Dieser Kommentar wird so angezeigt

4 ! (Ausrufezeichen)

Syntax:

- Windows XP

  !VARIABLENAME!

Web-Links:

- "EnableDelayedExpansion" auf SS64.com
- Beispiel zur fehlerhaften Verwendung von "!" inkl. Erkläuterung (engl.): "Batch - Adding Users to Multiple Groups Through For Loop" auf stackoverflow.com
- Allgemeine Info zur Allokation von Variablen

Erklärung:

Zur verzögerten Übersetzung von Variablen. Bewirkt, dass die Variable nicht zur Kompilierzeit sondern erst zur Laufzeit übersetzt wird (setzt die Verwendung von SETLOCAL zur Aktivierung von verzögerter Übersetzung voraus)

Verwendung:

Beispielsweise zur besser differenzierten Ausführung von For-Schleifen siehe LINK

Beispiele:

Inhalt der Batchdatei

Setlocal EnableDelayedExpansion Set _var=first Set _var=second& Echo %_var% !_var!

Ausgabe (unter XP)

first second

5 : (Doppelpunkt)

Erklärung:

Sprungmarke für ein Unterprogramm bzw. eine Kommentarzeile.
Sprungmarken werden benötigt, wenn mittels der Batchdatei eine Bedingung überprüft und erfüllt bzw nicht erfüllt wird und entsprechend weiter verfahren werden soll. Mit dem Batchbefehl goto wird die Sprungmarke angesprungen.

Anmerkung:

Der Doppelpunkt hat auch die Funktion der Manipulation von Variablen, wenn er direkt hinter einer Variablen steht. Siehe Kapitel “Variablen” in dieser Publikation.

Syntax

:NAMEDERSPRUNGMARKE

Sprungmarken können eine beliebige Länge haben, unter MS-DOS und älteren Windows-Versionen werden allerdings nur die ersten 8 Zeichen beachtet, der Rest wird ignoriert. Kommen in einer Batch also :Sprungmarke1 und :Sprungmarke2 vor, so wird unter Umständen nur die erste beim Aufruf einer der Beiden gefunden. Also besser :ziel1 oder :1st schreiben. Groß- und Kleinschreibung wird nicht unterschieden. Man kann jedoch mithilfe von Anführungszeichen dafür sorgen, dass der komplette Sprungname beachtet wird (Beispielsweise :"Sprungname”). Wenn man den kompletten Code mit Sprungmarken verbindet, kann man zur besseren Unterscheidung verschiedener Sprungmarken eine Leerzeile verwenden.

Beispiel:

Inhalt

if exist C:\blabla.txt goto EDITBLA goto END :: Kommentarzeile, sofern es erforderlich ist, einen Kommentar zu schreiben :: Zur Unterscheidung von Sprungmarken verwende ich zwei "::" :EDITBLA edit c:\blabla.txt :END

Sofern die Datei C:\blabla.txt existiert, wird sie mit edit geöffnet, sonst wird das Unterprogramm übersprungen und die Batchdatei bei der Marke :END fortgesetzt, also beendet.

Seit die Befehlserweiterungen aktiviert sind, steht in Batchdateien die Sprungmarke :EOF zur Verfügung, welche sich unsichtbar am Ende der Batch-Datei befindet.

6 :: (doppelter Doppelpunkt)

Syntax

- Windows 2000, XP

  ::KOMMENTAR

Erklärung:

Der :: **kann** zur Einleitung eines “Kommentars” verwendet werden, ist aber eigentlich ein Spezialfall der Verwendung vom Doppelpunkt (Label). Eine ausführliche Erklärung zu Kommentaren findet sich beim Befehl REM

Beispiel:

Inhalt der Batchdatei

::echo Ich bin ein Kommentar echo Ich bin ein Befehl

Ausgabe der Batchdatei

Microsoft Windows XP [Version 5.1.2600] (C) Copyright 1985-2001 Microsoft Corp. C:\>echo Ich bin ein Befehl Ich bin ein Befehl

Die Zeile “Ich bin ein Kommentar” bleibt bei der Ausführung “unberücksichtigt”.

7 CALL

Mit call kann man eine andere Batch-Datei aufrufen. Sobald diese beendet wurde, wird die ursprüngliche Batchdatei weiter ausgeführt.

Beispiel:

REM Call.cmd @echo off echo Diese Batchdatei ruft eine andere auf. call anderedatei.bat pause REM anderedatei.bat @echo off echo Die andere Datei

Die Ausgabe wenn man die Datei Call.cmd startet:

Diese Batchdatei ruft eine andere auf. Die andere Datei {Pause}

Wenn die Befehlserweiterungen aktiviert sind (Standard ab Windows 2000) kann man auch Sprungmarken aufrufen (und auch Parameter übergeben):

Beispiel:

@echo off REM Diese Batchdatei ruft eine eigene Sprungmarke auf echo Vor dem Sprung call :sprungmarke echo Nach dem Sprung pause goto end :sprungmarke echo Sprungmarke aufgerufen! goto :eof :: ":EOF” führt nicht zum unsichtbaren Ende der Batch-Datei, wie unten beschrieben, :: sondern führt die Batch-Datei nach dem Aufruf der Sprungmarke fort :end exit

Die Ausgabe:

Vor dem Sprung Sprungmarke aufgerufen! Nach dem Sprung

Beispiel mit Parameter:

@echo off REM Diese Batchdatei ruft eine eigene Sprungmarke auf echo Vor dem Sprung call :sprungmarke meinParameter echo Nach dem Sprung pause goto end :sprungmarke echo Sprungmarke aufgerufen und Parameter %1 uebergeben! goto :eof :end exit

Die Ausgabe:

Vor dem Sprung Sprungmarke aufgerufen und Parameter meinParameter uebergeben! Nach dem Sprung

goto :eof

Dieser Befehl springt automatisch zum Ende der Batchdatei (beendet die aktuelle Prozedur)

8 CHCP

Anzeige der aktuellen CodePage oder Setzen einer neuen CodePage (**ch**ange **c**ode**p**age)

chcp [nnn]

Beispiele für nnn:

- 437 – Die ursprüngliche Zeichensatztabelle des IBM-PC
- 720 – Arabisches Alphabet
- 737 – Griechisches Alphabet
- 850 – westeuropäische Sprachen (DOS-Latin-1)
- 857 – Türkisches Alphabet
- 866 – Kyrillisches Alphabet

9 CLS

Mit cls (clear screen) wird der Bildschirm gelöscht.

Syntax

cls

Interner Befehl: Ab MS-DOS bis Windows 8

Beispiel:

Inhalt

@echo off echo Hier schreibe ich jetzt ganz viel Text. echo Hier kann ich z.B. hinschreiben, dass ich jemanden mag. echo. echo Aber den Text sieht man gleich sowieso nicht mehr … Hihi! cls echo War irgendetwas? pause>nul

Ausgabe

War irgendetwas?

10 CMD

Syntax:

- Windows 2000, XP, Vista, 7, und 8

  cmd /a|/u|/q|/d|/e (ON|OFF)|/f (ON|OFF)|/v (ON|OFF)|/c *befehl*|/s|/k|/y

- DOS, Windows 95, 98, ME

  entspricht dem Batch-Befehl COMMAND (CMD ist unter DOS nicht vorhanden)

Web-Links:

- Windows XP Professional Product Documentation

Erklärung:

Der Befehl CMD startet eine neue Instanz des Kommandozeileninterpreters (cmd.exe).

Verwendung:

In Kombination mit dem Befehl START öffnet sich die neue Instanz auch in einem neuen Kommandozeilenfester (ohne START öffnet sie sich im gleichen Fenster).

Beispiele:

Inhalt der Batchdatei

cmd

Ausgabe (unter XP)

C:\>cmd Microsoft Windows XP [Version 5.1.2600] (C) Copyright 1985-2001 Microsoft Corp. C:\>

Ausgabe (unter Vista)

C:\>cmd Microsoft Windows [Version 6.0.6001] Copyright (c) 2006 Microsoft Corporation. Alle Rechte vorbehalten. C:\>

Ausgabe (unter Win 7)

C:\>cmd Microsoft Windows [Version 6.1.7600] Copyright (c) 2009 Microsoft Corporation. Alle Rechte vorbehalten. C:\>

Ausgabe (unter Win 8 Beta)

C:\>cmd Microsoft Windows [Version 6.2.8250] Copyright (c) 2012 Microsoft Corporation. Alle Rechte vorbehalten. C:\>

11 COLOR

Mit dem Befehl COLOR kann man die Vorder- und Hintergrundfarbe verändern. Die COLOR Werte bestehen aus zwei HEX-Werten, wobei der erste HEX-Wert für die Hintergrundfarbe und der zweite HEX-Wert für die Vordergrundfarbe steht. Jede Ziffer kann einen der folgenden Werte annehmen:

Der folgende Befehl ergibt z.B. einen dunkelgrünen Hintergrund mit weißer Schrift.

COLOR 2F

Zum Zurücksetzen der Vorder- und Hintergrundfarbe wird COLOR einfach ohne Argumente aufgerufen.

COLOR

12 COMMAND

Syntax:

- DOS, Windows 95, 98, 98 SE, ME

  command *Laufwerk*:*Pfad Gerät* /e /l /u /P /MSG /LOW (/Y (/c|/k) *Befehl*)

Web-Links:

Erklärung/Verwendung:

Startet einen neuen Kommandointerpreter, dieser kann mit exit wieder beendet werden.

Beispiel:

Inhalt

command

Ausgabe (unter Windows 95)

C:\WINDOWS>command Microsoft(R) Windows 95 (C)Copyright Microsoft Corp 1981-1996. C:\WINDOWS>

13 DATE

Gibt das aktuelle Datum aus und ermöglicht dem Benutzer die Änderung des Datums. Wird der Befehl mit dem Parameter /t aufgerufen, so wird nur das aktuelle Datum ausgegeben.

DATE kann auch als Variable benutzt werden, so kann man zum Beispiel mit %date:~6,4% auf das Jahr zugreifen.

Beispiele:

z:\>date /T 12.06.2013 z:\>echo Heute ist der %date%. Heute ist der 12.06.2013. z:\>echo %date:~6,4% 2013

Hier werden vom Datum die ersten 6 Zeichen weggelassen und dann vier Stellen angezeigt. Bei time funktioniert das analog.

Bei einigen Betriebssystemversionen erfordert das Ändern des Systemdatums administrative Rechte.
XP: Die Ausgabe des Datumsformates ist abhängig von den Einstellungen in den Regions- und Sprachoptionen (Systemsteuerung)

Hinweis: die Uhrzeit laesst sich mit dem Befehl “TIME” ermitteln

14 ECHO

Gibt einen Text aus oder schaltet die Befehlszeilen an/aus. Wenn ein Text ausgegeben wird, können dort auch Variablen angezeigt werden, wie z. B. die Variable **%ver%** (in Windows XP **%os%**).

Syntax:

echo *text*|ON|OFF oder alternativ echo.[*text*]

Interner Befehl: Ab MS-DOS bis Windows NT 5.1 (XP)

Beispiel:

Inhalt

@echo off echo Die aktuelle Datei heißt %0. echo Die aktuelle Version Ihrer Befehls-Konsole oder -OS heißt %ver%

Ausgabe

Die aktuelle Datei heißt beispiel.bat. Die aktuelle Version Ihrer Befehls-Konsole oder -OS heißt Windows NT

Mit echo. können Sie zudem leere Zeilen ausgeben. **Beispiel:**

Inhalt

@echo off echo Jetzt gibt es 3 Leere Zeilen zu sehen! echo. echo. echo. echo So! Da waren sie.

Ausgabe

Jetzt gibt es 3 Leere Zeilen zu sehen! So! Da waren sie.

14.1 Benutzereingaben mit ECHO

Mit Hilfe des echo-Befehls können Sie auch in einem Skript Benutzereingaben simulieren, indem Sie den Pipe-Operator | verwenden.
Beispiel: Uhrzeit anzeigen ohne Nachfrage
Das normale Verhalten des time-Befehls ist, die aktuelle Zeit der verwendeten Systemuhr anzuzeigen und in der nächsten Zeile die Eingabe einer neuen Uhrzeit zu erwarten. Drückt man auf Enter, bleibt die Systemzeit unverändert. Will man die Zeit nur anzeigen lassen (z. B. in einer Batchdatei vor und nach einer Befehlsfolge, um zu messen, wie lange der PC dafür braucht), lässt sich die Betätigung der Enter-Taste durch einen entsprechenden echo-Befehl ersetzen.

echo.|time

Über den Pipe-Mechanismus lässt sich darüber hinaus die Zeile „Geben Sie die neue Uhrzeit ein:“ unterdrücken:

echo.|time|find /v “neue”

Dabei ist die Groß-/Kleinschreibung von “neue” zu beachten oder der Schalter /I zu verwenden, denn find ist case-sensitiv! Diese Beispiele dienen allerdings nur zur Demonstration, denn die Zeitausgabe wäre auch ohne Pipes (aber erst ab Windows 2000) möglich mit:

time /t

oder einfach:

echo %time%

Sehr nützlich ist das echo-Piping auch zur Übergabe von Benutzereingaben, welche von einzelnen Befehlen abgefragt werden.
Beispiel: Überprüfung einer Festplatte

chkdsk c: /f /r

kann (da es sich beim Laufwerk C um das Systemlaufwerk handelt) erst nach einem Systemstart ausgeführt werden. Normalerweise müsste der Benutzer deswegen den Systemstart durch Eingabe von “Y” bestätigen. Diese Aktion kann man in einem Batch so abbilden:

echo y | chkdsk c: /f /r

.

15 ERRORLEVEL

Syntax:

- Windows XP, Windows Vista, Windows 7

  %errorlevel%

Verwendung:

Zeigt an, ob der letzte Befehl erfolgreich war.

Beispiel

del %homepath%\Desktop\Ordner echo %errorlevel%

- **del %homepath%\Desktop\Ordner** = Im Homepath (C:\Users\Username) wird im Ordner “Desktop” der Ordner “Ordner” gelöscht.
- **echo %Errorlevel%** = Errorlevel wird angezeigt. 1 bedeutet “Befehl fehlgeschlagen”, 0 bedeutet “Befehl ausgeführt”. 2 bedeutet “Unbekannt”.

16 ENDLOCAL / (SETLOCAL)

Syntax:

- Windows XP

  ENDLOCAL

Links:

- http://ss64.com/nt/endlocal.html

Verwendung:

Schliesst den Befehl SETLOCAL ab. Siehe SETLOCAL

17 EXIT

Der Befehl exit beendet die Abarbeitung der Batchdatei bzw. die Kommandozeilenfenster.

18 FOR

Ermöglicht die Schleifenbearbeitung.

Syntax:

for *Variable* in *Satz* do *Befehl* [*Parameter*]

Interner Befehl: Ab MS-DOS bis Windows NT 6.1 (Windows 7)

ACHTUNG:

Die Variable darf nur aus einem Buchstaben bestehen! "%t” ist erlaubt, "%test” nicht! Bei der Verwendung mehrerer Befehle muss zwischen “DO” und der Klammer "(" ein Leerzeichen sein.

Falsch

for *Variable* in *Satz* do(

RICHTIG

for *Variable* in *Satz* do (*Befehl1 Befehl2*)

Beispiel:

Zeigt alle Dateien im Verzeichnis %temp% an. Es werden nur Dateien, keine Verzeichnisse angezeigt. Um Verzeichnisse anzuzeigen siehe Liste der FOR-Optionen unten. Der Parameter /R bewirkt, dass alle Unterverzeichnisse mit einbezogen werden (Rekursive Schleife).

Inhalt

@echo off for /R %temp% %%f in (*.*) do (echo %%f) REM Den Befehl könnte man auch einzeilig schreiben. pause

Ausgabe

(Alle Temp-Dateien) Bitte beliebige Taste drücken...

Zählschleifen

Mit solchen Schleifen kann man Aktionen eine bestimmte Anzahl oft ausführen. Dazu muss man den Parameter /L angeben.

Syntax: for /L {Variable} IN (Startzahl, Schrittweite, Endzahl) DO (Aktion)

REM Schreibe Text 5 Mal for /L %%N IN (1, 1, 5) DO echo Nummer %%N

Ausgabe:

C:\>for /L %%N IN (1, 1, 5) DO echo Nummer %%N C:\>echo Nummer 1 Nummer 1 C:\>echo Nummer 2 Nummer 2 C:\>echo Nummer 3 Nummer 3 C:\>echo Nummer 4 Nummer 4 C:\>echo Nummer 5 Nummer 5

verschachtelte Zählschleife:

for /L %%N IN (1, 1, 5) DO for /L %%N IN (1, 1, %N)DO echo Nummer %N

Ausgabe:

C:\>for /L %%N IN (1 1 5) DO (for /L %N IN (1 1 %N) DO echo Nummer %N) C:\>(for /L %%N IN (1 1 1) DO echo Nummer %N) C:\>echo Nummer 1 Nummer 1 C:\>(for /L %%N IN (1 1 2) DO echo Nummer %N) C:\>echo Nummer 1 Nummer 1 C:\>echo Nummer 2 Nummer 2 C:\>(for /L %%N IN (1 1 3) DO echo Nummer %N) C:\>echo Nummer 1 Nummer 1 C:\>echo Nummer 2 Nummer 2 C:\>echo Nummer 3 Nummer 3 C:\>(for /L %%N IN (1 1 4) DO echo Nummer %N) C:\>echo Nummer 1 Nummer 1 C:\>echo Nummer 2 Nummer 2 C:\>echo Nummer 3 Nummer 3 C:\>echo Nummer 4 Nummer 4 C:\>(for /L %%N IN (1 1 5) DO echo Nummer %N) C:\>echo Nummer 1 Nummer 1 C:\>echo Nummer 2 Nummer 2 C:\>echo Nummer 3 Nummer 3 C:\>echo Nummer 4 Nummer 4 C:\>echo Nummer 5 Nummer 5

Weitere Möglichkeiten der FOR-Schleife:

syntax-FOR-Files

FOR %%parameter IN (set) DO command

syntax-FOR-Files-Rooted at Path

FOR /R [[drive:]path] %%parameter IN (set) DO command

syntax-FOR-Folders

FOR /D %%parameter IN (folder_set) DO command

syntax-FOR-List of numbers

FOR /L %%parameter IN (start,step,end) DO command

syntax-FOR-File contents

FOR /F ["options"] %%parameter IN (filenameset) DO command FOR /F ["options"] %%parameter IN (“Text string to process”) DO command

syntax-FOR-Command Results

FOR /F ["options"] %%parameter IN ('command to process’) DO command

Beispiel: Sucht im Ordner C:\Windows\Temp rekursiv nach Dateien mit dem Namen //temp.dat// und gibt die Liste aus. Die Option “token=*" ist notwendig, damit die Ausgabe zeilenweise gelesen wird und auch Pfade mit enthaltenem Leerzeichen ausgegeben werden können.

for /F “tokens=*" %%f in ('dir /S /b C:\Windows\Temp\temp.dat') do (echo "%%f”.)

Zählvariablen in Zeichenketten einbetten

Um die Zählvariable %%f (%f auf der Kommandozeile) in einer Zeichenfolge zu verwenden, wird einfach die Variable in dem String eingebettet

FOR %%f IN (A B C D E) DO (echo mitten%%fdrinnen)

19 GOTO

Mit dem Batchbefehl goto wird eine Sprungmarke : (s.o.) angesprungen.

Syntax

goto NAMEDERSPRUNGMARKE

Beispiel

Siehe unter : (Doppelpunkt).

20 IF

Der IF-Befehl ermöglicht eine einfache Verzweigung und wird oft zusammen mit dem GOTO-Befehl eingesetzt. IF ermöglicht hierbei sowohl die Prüfung auf eine Gleichheit als auch auf das Vorhandensein von Dateien.

Beispiel 1:

@echo off IF exist c:\temp\my.log echo.>c:\temp\my.log echo.Log Datei erstellt>>c:\temp\my.log

Beispiel 1 prüft, ob eine Logdatei vorhanden ist und erstellt ggf. eine Neue.

Beispiel 2:

@echo off IF "%COMPUTERNAME%" == “Bastie” GOTO WAHR REM hier landet man wenn der if-Ausdruck falsch ist GOTO WEITER :WAHR REM hier landet man wenn der if-Ausdruck wahr ist echo Willkommen Zuhause REM Jetzt wird der if Zweig verlassen GOTO WEITER :WEITER echo.Have a nice Day!

Beispiel 3:

IF "%COMPUTERNAME%" == “Bastie” (echo Willkommen zu Hause!) ELSE (echo Du bist auf Computer: %COMPUTERNAME%) echo. Schönen Tag noch!

Beachten Sie, bei der Prüfung von Umgebungsvariablen niemals

IF %Umgebungsvariable% == Prüfwert ...

zu schreiben, wenn die Umgebungsvariable nicht gesetzt ist; Sie erhalten sonst einen Syntaxfehler. Der Parameter /i unterbindet eine Differenzierung der Groß-/Kleinbuchstaben.

ACHTUNG:

Bei der Verwendung mehrerer Befehle muss zwischen *Bedingung* und der Klammer "(" ein Leerzeichen sein.

Falsch

IF *Bedingung*(

Richtig

IF *Bedingung* (*Befehl1 Befehl2*)

20.1 Syntaxvergleiche

IF **<NOT>** Variable1**==**Variable2

IF **%Variable% EQU %Variable2% (Befehl)** An die Stelle von EQU kann jede der Optionen gesetzt werden.

NOT Der Befehl wird nur ausgeführt, wenn die Bedingung NICHT Wahr ist. Optional.

== ist gleich

EQU ist gleich

NEQ nicht gleich

LSS kleiner als

LEQ kleiner als oder gleich

GTR größer als

GEQ größer als oder gleich

20.2 Hinweis zu UND- bzw. ODER-Verknüpfung

Eine UND- bzw. ODER-Verknüpfung von zwei Bedingungen scheint nicht direkt möglich zu sein. Beim Vergleichen von Strings hilft es aber eventuell, wenn man die beiden Strings miteinander verkettet.

Beispiel

set A=true set B=false if "%A%;%B%"=="true;true" (echo A und B sind beide TRUE) else (echo mindestens eines von beiden - A oder B - ist ungleich TRUE)

Als Workaround können mehrere aufeinanderfolgende IFs zu einer UND- bzw. ODER-Verknüpfung kombiniert werden. Bei einer ODER-Verknüpfung wird der Code ausgeführt, sobald eine der Bedingungen wahr ist. Wenn alle Bedingungen geprüft wurden und keine erfolgreich war, werden die Befehle im ELSE-Zweig ausgeführt.

set A=true set B=false if "%A%"=="true" goto :WAHR // Diese Zeile ist doch erfüllt, also sollte der in :WAHR springen if "%B%"=="true" goto :WAHR REM keine der Bedingungen ist zu :WAHR gesprungen, wir sind also im ELSE-Zweig REM hier waere :FALSCH echo Weder A noch B ist TRUE goto :eof :WAHR echo A oder B ist TRUE

Für ein UND wird in den ELSE-Zweig gesprungen (:FALSCH) sobald eine der Bedingungen **nicht** zutrifft. Nur wenn alle Bedingunen zutreffen wird der Code ausgeführt.

set A=true set B=false if "%A%"NEQ"true" goto :FALSCH if "%B%"NEQ"true" goto :FALSCH REM wird sind durch die IFs gekommen, also hat keine der Bedingungen angeschlagen. REM hier waere die :WAHR-Sprungmarke echo A und B sind beide TRUE goto :eof :FALSCH echo A oder B (oder beide) sind FALSE

Eine weitere gut lesbare UND-Variante ist, zwei oder mehrere IF-Bedingungen nacheinander zu schreiben.

set P1=Frankreich set P2=Spanien if %P1%==Frankreich if %P2%==Spanien echo Die Nachbarländer von Andorra sind %P1% UND %P2%

Diese beiden Beispiele lassen sich einfach durch Kopieren der “IF..."-Zeile um beliebig viele Bedingungen erweitern. Ein Mischen von UND- und ODER-Verknüpfungen ist leider nicht ohne weiteres möglich.

21 KEYB

Lädt Tastaturtreiber. keyb gr,,c:\dos\keyboard.sys lädt den deutschen Tastaturtreiber (keyboard.sys muss sich im Verzeichis c:\dos\ befinden)

22 PAUSE

Unterbricht die Abarbeitung der Batchdatei und wartet auf einen Tastendruck.

Syntax

pause

Interner Befehl: Ab MS-DOS bis Windows NT 5.1 (XP)

Beispiel 1:

Inhalt

@echo off echo Willkommen zur Batchdatei %0 !!! echo. echo Die Batchdatei wird auf einer neuen Seite fortgeführt. pause cls echo Hier fängt meine Batchdatei an... pause

Ausgabe

Willkommen zur Batchdatei beispiel.bat !!! Die Batchdatei wird auf einer neuen Seite fortgeführt. Bitte beliebige Taste drücken...

(Neue Seite)

Hier fängt meine Batchdatei an... Bitte beliebige Taste drücken...

Beispiel 2:

Inhalt

@echo off echo Diese Zeile wird angezeigt anstelle von “Bitte beliebige Taste drücken...” pause > NUL

Ausgabe

Diese Zeile wird angezeigt anstelle von “Bitte beliebige Taste drücken...”

Um eine Pause für eine bestimmte Zeitdauer vorzugeben, kann der Befehl sleep verwendet werden.

23 PROMPT

- **Syntax:**

prompt [text]

- **Erklärung:**

  Legt das Aussehen der Eingabezeile mit Hilfe unterschiedlicher Parameter fest.

- **Hinweis:**

  Der Prompt kann auch permanent über eine Umgebungsvariable geändert werden:
  set PROMPT=pg

  Sollte der Befehl alleine und ohne Funktion eingegeben werden, so wird die Wirkung aufgehoben.

24 PUSHD / POPD

pushd wechselt zum angegebenen Pfad und speichert den aktuellen Pfad bis zum Aufruf von popd.

popd wechselt zum gespeicherten Pfad.

Die Befehle können geschachtelt werden.

Syntax

pushd pfad

popd

Beispiel:

C:\WINDOWS>pushd c:\temp C:\temp>pushd c:\ C:\>popd C:\temp>popd C:\WINDOWS>

25 REM

Syntax:

rem [comment]

Erklärung:

REM leitet einen Kommentar ein. Die Zeile wird ignoriert, beachten Sie jedoch, dass REM von einem Leerzeichen / Tabulator gefolgt werden muss, sonst wird die Ausführung der Batchdatei unmittelbar beendet. Alternativ können zwei Doppelpunkte hintereinandergeschrieben werden :: um einen Kommentar anzuführen (hier ist ein Leerzeichen nicht notwendig). Alternativ können auch Sprungmarken : oder der Echobefehl echo verwendet werden.

Beispiele:

REM kill iexplore.exe ::kill iexplore.exe :kill iexplore.exe echo kill iexplore.exe

Web-Links:

- Windows XP Professional Product Documentation (Syntax-Link)
- Unterschiede alternativer Verwendungen
- Unterschiede alternativer Verwendungen
- Unterschiede alternativer Verwendungen
- Zu den Unterschieden von REM oder "::"

26 RENAME

Mit dem rename-Befehl kann man Dateien umbenennen.

Syntax:

- MS-DOS und Windows: REN
- seit Windows 98 auch RENAME

RENAME [Laufwerk:][Pfad]Dateiname1 Dateiname2
REN [Laufwerk:][Pfad]Dateiname1 Dateiname2

27 SET

Syntax:

- Windows XP

  set [[/a [expression]] [/p [variable=]] string]

Links:

- http://www.scriptcode.com/batchfilecommands/set.html

Verwendung:

Wird hauptsächlich verwendet, um einer Variablen einen Wert zuzuweisen. Siehe Batchbefehle (Variablen und Set-Befehl)

Ohne Parameter gibt set eine Liste aller Umbegebungsvariablen aus.

Der Parameter /p kann dazu verwendet werden der Variable eine Benutzereingabe zuzuweisen.

Der Parameter /a kann dazu verwendet werden, um mit den Variablen Rechenoperation durchzuführen.

Beispiel zur Wertezuweisung:

Inhalt Batchdatei:

set VARIABLENNAME=test echo %VARIABLENNAME% set VARIABLENNAME=test2 echo %VARIABLENNAME%

Ausgabe (gekürzt):

c:\>set VARIABLENNAME=test c:\>echo %VARIABLENNAME% test c:\>set VARIABLENNAME=test2 c:\>echo %VARIABLENNAME% test2 C:\>set /p test=Dies ist ein Test:

28 SETLOCAL / (ENDLOCAL)

Syntax:

- Windows XP

  SETLOCAL

  SETLOCAL EnableDelayedExpansion | DisableDelayedExpansion

  SETLOCAL EnableExtensions | DisableExtensions

Links:

- http://ss64.com/nt/setlocal.html

Siehe auch Batchbefehle (Variablen und Set-Befehl)

29 TIME

Gibt die aktuelle Zeit aus und ermöglicht dem Benutzer die Änderung der Uhrzeit. Wird der Befehl mit dem Parameter /t aufgerufen, so wird nur die aktuelle Zeit ausgegeben. time kann auch als Variable benutzt werden, so kann man zum Beispiel mit %time:~0,5% die ersten 5 Zeichen übernehmen.

Beispiele:

C:\>time Aktuelle Zeit: 10:03:04,63 Geben Sie die neue Zeit ein: c:\>echo %time% 10:03:04,63 c:\>echo %time:~0,5% 10:03

Bei einigen Betriebssystemversionen erfordert das Ändern der Systemzeit administrative Rechte.

30 TITLE

Dieser Befehl ändert die Fensterüberschrift des Programmfensters.

title Beispiel

31 ECHO >, >>

Erklärung

Dieser Befehl überschreibt, beziehungsweise hängt etwas an eine vorhandene Datei an.

Beispiel >:

Inhalt:

@echo off echo Die Datei wird nun überschrieben. echo Die Datei wurde überschrieben! > %homepath%\Desktop\Beispiel.bat

Ausgabe:

Die Datei wird nun überschrieben.

In Beispiel.bat:

Die Datei wurde überschrieben!

Beispiel >>:

Inhalt:

@echo off echo Nun wird etwas an die Datei angehängt. echo Dies wurde an die Datei angehängt! >> %homepath%\Desktop\Beispiel.bat

Ausgabe:

Nun wird etwas an die Datei angehängt.

In Beispiel.bat:

Die Datei wurde überschrieben! Dies wurde an die Datei angehängt!

32 Text- und Bildquellen, Autoren und Lizenzen

32.1 Text

- **Batch-Programmierung: Batch-Befehle** *Quelle:* https://de.wikibooks.org/wiki/Batch-Programmierung%3A_Batch-Befehle?oldid=775549 *Autoren:* Daniel B, Trixium, Timon.Freitag, Bastie, E^(nix), ThePacker, WeißNix, Klaus Eifert, Worker, MF-Warburg, Reseka, CorneliusWasmund~dewikibooks, MichaelFrey, Klartext, Michfrm, Prog, Dirk Huenniger, Spectrums~dewikibooks, Codejunkie, Heuler06, Markus Bärlocher, Autor, Stefan wichmann, Pc-world, Se4598, Stephan Kulla, Prince Kassad, Azaël, Schneijo, Batchscripter93, Juetho, Saerdnaer, Haddock, Der Leo, Emes2k, Fehlerkorrektur, Komikaa, Theres no global warming, Albin~dewikibooks, Johanna31~dewikibooks, LilTV, Langläufer, TBotV63, Dirk Schmitzkamp, Silver-tm-, Wikitimme, S536870912 und Anonyme: 169

32.2 Bilder

32.3 Inhaltslizenz