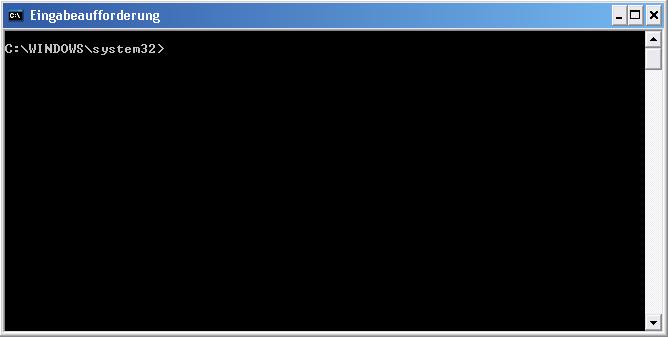
Batch-Lexikon

# Übersicht:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| %var% | Color | FC | Mplay32 | Sc | User/User32 |
| %var:~x,y% | Command.com | Find | Msg | Set | *Variabeln* |
| %var:a=b% | Comp | Findstr | Msgbox | Setlocal | Ver |
| %0 | Compact | For | Net | Shift | Wmic |
| >nul | Convert | Format | Netsh | Shutdown | Xcopy |
| >“Pfad“ | Copy | Ftp | Netstat | Sndrec32 |  |
| 2>&1 | Date | Ftype | Path | *Sonderzeichen* |  |
| *Argumente* | Defrag | Goto | Pause | Sort |  |
| *Ascii-Zeichen* | Del / Erase | Gpresult | Ping | *Springpunkte* |  |
| Assoc | Dir | Help | Ping localhost | Start |  |
| Attrib | Driverquery | Iexplore | *Programmpfade* | Systeminfo |  |
| Break | Echo | If | Prompt | Taskkill |  |
| Cacls | Edit | *Kurznamen* | Pushd / Popd | Tasklist |  |
| Call | Endlocal | Label | Rd / Rmdir | Time |  |
| Cd / Chdir | Erase / Del | Md / Mkdir | REG | Title |  |
| Chkdsk | *Errorlevel* | Mode | Rem | Tree |  |
| Cls | Exit | More | Ren | Tskill |  |
| Cmd | Explorer | Move | Runas | Type |  |

* **Fette**, *kursive*, unterstrichene Begriffe sind als Extra-Punkt vorhanden (z.B. ***Variabeln***).
* Auf den letzten Seiten ist eine mögliche Reihenfolge der Befehle für Neulinge.



# Erklärungen

* **%var%:**

Eine Zeichenfolge umschlossen von zwei **%-Zeichen** ist eine **Variable**, die vorher auf etwas

Bestimmtes festgelegt wurde. Dies geschieht durch den ***SET*-Befehl** (mehr dazu siehe SET).

Unter der Verwendung des Befehls „SETLOCAL ENABLEDELAYEDEXPANSION“ können Variablen auch innerhalb von Klammern gesetzt und aufgerufen werden, dann müssen sie allerdings von zwei **!-Zeichen** umschlossen sein. Mehr dazu siehe „Variablen“.

* **%var:~x,y% :**

Dies ist eine sehr komplizierte Art einer Variable. Normal würde dort stehen %var%. Der **Zusatz :~x,y** kann angeben, ab welchem Zeichen der Variable ein bestimmter Befehl (z.B. echo) ausgeführt werden soll. **X** ist das Zeichen, bei dem **begonnen** werden soll. **Y** ist das Zeichen, bei dem **aufgehört** werden soll. Ist y **im Minus**, wird von hinten angefangen zu zählen.

Beispiele:

echo %time:~0,2% **Die** momentane Zeit wird angegeben, jedoch nur die ersten

beiden Ziffern von 0 aus. Dies ist die Stundenzahl.

Echo %time:~3,2% **Die** momentane Zeit wird angegeben, jedoch nur die beiden

Ziffern der Minutenzahl. Die Zeit ist normalerweise

HH:MM:SS,MS. Die Doppelpunkte und das Komma sind auch

Zeichen. Nach dem 3. Zeichen (HH☺ wird geschrieben. Nach

zwei Zeichen wird aufgehört (MM).

Echo %time%:~0,-2% **Alle** Ziffern bis zu dem Vorletzten werden ausgegeben.

Das Ganze geht auch mit Buchstaben anstatt mit Zahlen:

echo %cd:~0,20% **Die** ersten 20 Buchstaben und Zeichen werden ausgegeben.

* **%var:a=b% :**

Dies ist eine andere Form einer Variable. Die normale Variable ist %var%. Der **Zusatz :a=b** bewirkt jedoch, dass **Zeichenfolge „a“**, die aus beliebig vielen Buchstaben, Zahlen und Zeichen bestehen kann (außer % oder =), **durch** die **Zeichenfolge „b“ ersetzt** wird. Dabei werden alle „a“ ersetzt.

Beispiele:

Set Stein=Dies wird eine Variable.

Echo %Stein%

Set Stein=%Stein:wird=ist%

Echo %Stein%

Ausgabe:

Dies wird eine Variable.

Dies ist eine Variable.

🡪 Das Wort „wird“ wurde durch „ist“ ersetzt. „wird“ ist Zeichenfolge a, „ist“ ist b:

%Stein:a=b% --- %Stein:wird=ist%

* **%0:**

%0 ist im Allgemeinen ein ***Argument***.

%0 ist die momentan geöffnete **Datei** unter ihrem **Pfad**. Z.B. gibt %0 von dieser Datei den aktuellen Pfad an.

Dies kann man dazu verwenden, um die Datei, die gerade geöffnet ist, schneller aufzurufen oder anzugeben.

Beispiel:

call %0

Damit öffnet man die momentan geöffnete Datei nochmal.

%0 verändert sich, wenn man einen ***Springpunkt*** über den ***call***-Befehl aufruft. In diesem Fall wird %0 zu dem Namen des Springpunktes.

* **>nul:**

Mit dem „**>nul**“-Anhängsel kann man bei jedem Befehl die **Anzeige des Verlaufes unterdrücken**, das bedeutet, die Ausgabe des Befehls wird unterdrückt. Fehler werden jedoch angezeigt. Um diese ebenfalls nicht anzeigen zu lassen, kann man „***2>&1***“ hinzufügen oder…

„**2>nul**“ als Anhängsel unterdrückt die Ausgabe von Fehlermeldungen, aber nicht die eigentlich Ausgabe.

„**>nul**“ und „**2>nul**“ können zusammen verwendet werden.

Beispiele:

Eingabe: pause

Ausgabe: Drücken Sie eine beliebige Taste. . .

Eingabe: pause >nul

Ausgabe:

Eingabe: ***find***

Ausgabe: FIND: Parameterformat falsch

Eingabe: find >nul

Ausgabe: FIND: Parameterformat falsch

Eingabe: find 2>nul

Ausgabe:

Eingabe: ***taskkill*** /IM taskmgr.exe >nul 2>nul

Ausgabe:

* In dem Fall wird die Ausgabe sowohl bei Erfolg als auch Misserfolg unterdrückt.
* **>“Pfad“:**

Dieses Anhängsel kann an die meisten Befehle gehangen werden. Die **Ausgabe des Befehls** wird dann **in die Datei hinter dem > gespeichert.**

Somit wird mit „***tasklist*** /SVC**>**“%userprofile%\Desktop\Prozessliste.txt““ die Liste aller laufenden Prozesse in eine Textdatei auf dem Desktop gespeichert.

Dieses Ergebnis kann man auch mit dem ***FOR***-Befehl erreichen, allerdings ist der komplizierter und dauert länger:

FOR /F “delims=” %%A IN (‘tasklist /SVC’) DO echo %%A**>>**”%userprofile%\Desktop\Prozessliste.txt”

Da jedoch der Befehl „wmic … get“ nicht in dem FOR-Befehl verwendet werden kann, muss man ihn mit dem Anhängsel **>“Pfad“** in eine Datei speichern. Diese kann man dann mit dem FOR-Befehl auswerten:

Wmic process get CommandLine,ProcessId**>**Prozesse.log

FOR /F “tokens=1-8 delims=\. “ %%A IN (Prozesse.log) DO if /I “%%D” == “svchost” echo %%A\%%B\%%C\%%D.%%E %%F %%G %%H

REM C:\Windows\system32\svchost.exe –k imgsvc *PID*

REM %A %B %C %D %E %F %G %H

* **2>&1:**

Dieses Anhängsel kann nur hinter „***>nul***“ stehen und führt dazu, dass auch **Fehler** beim Ausführen des Befehls **nicht angezeigt** werden.

Man kann stattdessen auch „**2>nul**“ verwenden, welches auch alleine stehen kann. Mehr dazu unter „***>nul***“.

Beispiele:

Eingabe : ***del*** „C:\Windows\NichtExistierendeDatei.txt“

Ausgabe: C:\Windows\NichtExistierendeDatei.txt konnte nicht gefunden werden.

Eingabe : del „C:\Windows\NichtExistierendeDatei.txt“ 2>&1

Ausgabe: C:\Windows\NichtExistierendeDatei.txt konnte nicht gefunden werden.

Eingabe : del „C:\Windows\NichtExistierendeDatei.txt“ >nul

Ausgabe: C:\Windows\NichtExistierendeDatei.txt konnte nicht gefunden werden.

Eingabe : del „C:\Windows\NichtExistierendeDatei.txt“ >nul 2>&1

Ausgabe:

# A

* **Argumente:**

Argumente sind Zeichenfolgen, die beim **Starten einer Datei** als Extra-Befehle verwendet

werden können. Argumente sind ***%0* bis %9**. Beispiel vom Starten einer Datei mit Argumenten:

***start*** batch-Lexikon-Helfer.bat „Dies wird in der Datei wiedergegeben, wenn in der Batch-Datei der Befehl ‚ echo %1 ‚ steht.“

Mehrere Begriffe mit Leerzeichen können zu einem Argument gefasst werden, wenn sie in

„ „ stehen wie in dem Beispiel angegeben. Hier das Beispiel ein wenig ausführlicher:

Inhalt der bat-Datei namens „batch-Lexikon-Helfer.bat“:

@echo off

echo %1 , %2 , %3 , %4

pause >nul

Befehl in CMD:

**call** “batch-Lexikon-Helfer.bat” “Argument Nr. 1” “Argument Nr. 2” usw. „wird wiedergegeben.“

Angezeigt werden würde:

Argument Nr. 1 , Argument Nr. 2 , usw. , wird wiedergegeben.

( %1 %2 %3 %4 )

Argumente können auch beim Aufrufen von ***Springpunkten*** verwendet werden:

@echo off

set Variable=viel machen.

Call **:Springpunkt3** Damit kann man „%Variable%“

pause >nul

:Springpunkt3

echo %1 %2 %3 %4

goto :PunktGibtsNicht

Angezeigt werden würde:

Damit kann man viel machen.

( %1 %2 %3 %4 )

Man kann Argumente erweitern. Die Erweiterungen können in der Rubrik „FOR“ eingesehen werden (z.B. %~nx1).

Beispiele:

***Set*** PfadDerDatei=%~s1

🡪 Dies würde die Variable zu einem Pfad mit ***Kurznamen*** machen.

***If*** /i „%~n1“ == „Virus“ echo Der Dateiname ist „Virus“!

* **Ascii-Zeichen**

#################################################################

# #

# Die unten genannten Ascii-Codes sind als Zahlen in Notepad #

# eingegeben und in CMD ausgeführt worden. Die Batch-Datei #

# musste als ANSI gespeichert werden, da Unicode nicht lesbar #

# ist. #

# #

#################################################################

Ascii-Codes werden wiefolgt geschrieben:

**Alt + *Zahl auf dem Ziffernblock***

Man hält Alt gedrückt und gibt folgende Nummern auf dem Ziffernblock ein:

|  |  |  |  |
| --- | --- | --- | --- |
| 140 | ¯ | 212 | ╚ |
| 143 | ┼ | 214 | ═ |
| 144 | ╔ | 215 | ╬ |
| 148 | ÷ | 222 | ╠ |
| 150 | ¹ | 223 | » |
| 151 | ¨ | 225 | ▀ |
| 154 | ▄ | 229 | groß ¹ |
| 163 | · | 231 | ■ |
| 166 | ¬ | 233 | ┌ |
| 167 | ║ | 234 | █ |
| 168 | ┐ | 235 | ┘ |
| 169 | « | 236 | ² |
| 172 | ╝ | 239 | ┤ |
| 175 | ╗ | 240 | ¡ |
| 181 | ┴ | 241 | ▒ |
| 182 | ┬ | 247 | © |
| 183 | └ | 248 | ░ |
| 184 | (R) | 249 | ¿ |
| 199 | ├ | 251 | ╣ |
| 210 | ╩ | 252 | │ |
| 211 | ╦ | 253 | ▓ |

Hier etwas geordneter:

|  |  |  |  |
| --- | --- | --- | --- |
| 144 | ╔ | 233 | ┌ |
| 212 | ╚ | 183 | └ |
| 175 | ╗ | 168 | ┐ |
| 172 | ╝ | 235 | ┘ |
| 214 | ═ | Minuszeichen | - |
| 215 | ╬ | 143 | ┼ |
| 222 | ╠ | 199 | ├ |
| 167 | ║ | 252 | │ |
| 251 | ╣ | 239 | ┤ |
| 210 | ╩ | 181 | ┴ |
| 211 | ╦ | 182 | ┬ |
| 148 | ÷ | 225 | ▀ |
| 163 | · | 231 | ■ |
| 151 | ¨ | 154 | ▄ |
| 150 | ¹ | 234 | █ |
| 229 | groß ¹ | 253 | ▓ |
| 236 | ² | 241 | ▒ |
| 223 | » | 248 | ░ |
| 169 | « | 166 | ¬ |
| 247 | © | 240 | ¡ |
| 184 | (R) | 249 | ¿ |

Beispiele:

1. Doppelte Linien.

144 – 214 – 214 - … – 214 – 214 – 175

167 167

167 Text Text Text Text Text Text 167

167 167

212 – 214 – 214 - … – 214 – 214 – 172

2. In der Mitte ausgefüllte Leerzeichen.

154 – 231 – 231 - ... – 231 – 231 – 154

234 234

234 Text Text Text Text Text Text 234

234 234

225 – 231 – 231 - … – 231 – 231 – 225

3. Am Rand volle LZ, oben obere LZ, unten untere Leerzeichen.

234 – 225 – 225 - ... – 225 – 225 – 234

234 234

234 Text Text Text Text Text Text 234

234 234

234 – 154 – 154 - … – 154 – 154 – 234

4. Einfache Linien.

233 - Ganz viele Minuszeichen - 168

252 252

252 Text Text Text Text Text Text 252

252 252

183 - Ganz viele Minuszeichen - 235

Außerdem kann man **an jeden Code eine Null** dranhängen. In dem Fall sind die **Zeichen eher geordnet**. Folgende Zeichen sind in CMD ausgegeben worden. Die Ascii-Codes wurden in Notepad eingegeben. Beispielsweise kommt bei 0134 in Notepad † (Jesuskreuz), bei CMD wird aber ein å (a mit Kreis drüber) gezeigt.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 01: ☺ | 045: - | 089: Y | 0133: à | 0177: ▒ | 0221: ¦ |
| 02: ☻ | 046: . | 090: Z | 0134: å | 0178: ▓ | 0222: Ì |
| 03: ♥ | 047: / | 091: [ | 0135: ç | 0179: │ | 0223: ▀ |
| 04: ♦ | 048: 0 | 092: \ | 0136: ê | 0180: ┤ | 0224: Ó |
| 05: ♣ | 049: 1 | 093: ] | 0137: ë | 0181: Á | 0225: ß |
| 06: ♠ | 050: 2 | 094: ^ | 0138: è | 0182: Â | 0226: Ô |
| 07: {BEL} (Alarmton) | 051: 3 | 095: \_ | 0139: ï | 0183: À | 0227: Ò |
| 08: {BS} (Rücktaste) | 052: 4 | 096: ` | 0140: î | 0184: © | 0228: õ |
| 09: (Tab) | 053: 5 | 097: a | 0141: ì | 0185: ╣ | 0229: Õ |
| 010: {LF} (Enter) | 054: 6 | 098: b | 0142: Ä | 0186: ║ | 0230: µ |
| 011: ♂ | 055: 7 | 099: c | 0143: Å | 0187: ╗ | 0231: þ |
| 012: ♀ | 056: 8 | 0100: d | 0144: É | 0188: ╝ | 0232: Þ |
| 013: {CR} (Enter ohne Wirkung) | 057: 9 | 0101: e | 0145: æ | 0189: ¢ | 0233: Ú |
| 014: ♫ | 058: : | 0102: f | 0146: Æ | 0190: ¥ | 0234: Û |
| 015: ☼ | 059: ; | 0103: g | 0147: ô | 0191: ┐ | 0235: Ù |
| 016: ► | 060: < | 0104: h | 0148: ö | 0192: └ | 0236: ý |
| 017: ◄ | 061: = | 0105: i | 0149: ò | 0193: ┴ | 0237: Ý |
| 018: ↕ | 062: > | 0106: j | 0150: û | 0194: ┬ | 0238: ¯ |
| 019: ‼ | 063: ? | 0107: k | 0151: ù | 0195: ├ | 0239: ´ |
| 020: ¶ | 064: @ | 0108: l | 0152: ÿ | 0196: ─ | 0240: ­ |
| 021: § | 065: A | 0109: m | 0153: Ö | 0197: ┼ | 0241: ± |
| 022: ▬ | 066: B | 0110: n | 0154: Ü | 0198: ã | 0242: ‗ |
| 023: ↨ | 067: C | 0111: o | 0155: ø | 0199: Ã | 0243: ¾ |
| 024: ↑ | 068: D | 0112: p | 0156: £ | 0200: ╚ | 0244: ¶ |
| 025: ↓ | 069: E | 0113: q | 0157: Ø | 0201: ╔ | 0245: § |
| 026: (Unterbefehl, etwa „(„ ) | 070: F | 0114: r | 0158: × | 0202: ╩ | 0246: ÷ |
| 027: ← | 071: G | 0115: s | 0159: ƒ | 0203: ╦ | 0247: ¸ |
| 028: ∟ | 072: H | 0116: t | 0160: á | 0204: ╠ | 0248: ° |
| 029: ↔ | 073: I | 0117: u | 0161: í | 0205: ═ | 0249: ¨ |
| 030: ▲ | 074: J | 0118: v | 0162: ó | 0206: ╬ | 0250: · |
| 031: ▼ | 075: K | 0119: w | 0163: ú | 0207: ¤ | 0251: ¹ |
| 032: (Leertaste) | 076: L | 0120: x | 0164: ñ | 0208: ð | 0252: ³ |
| 033: ! | 077: M | 0121: y | 0165: Ñ | 0209: Ð | 0253: ² |
| 034: „ | 078: N | 0122: z | 0166: ª | 0210: Ê | 0254: ■ |
| 035: # | 079: O | 0123: { | 0167: º | 0211: Ë | 0255: |
| 036: $ | 080: P | 0124: | | 0168: ¿ | 0212: È |  |
| 037: % | 081: Q | 0125: } | 0169: ® | 0213: ı |  |
| 038: & | 082: R | 0126: ~ | 0170: ¬ | 0214: Í |  |
| 039: ‚ | 083: S | 0127: ⌂ | 0171: ½ | 0215: Î |  |
| 040: ( | 084: T | 0128: Ç | 0172: ¼ | 0216: Ï |  |
| 041: ) | 085: U | 0129: ü | 0173: ¡ | 0217: ┘ |  |
| 042: \* | 086: V | 0130: é | 0174: « | 0218: ┌ |  |
| 043: + | 087: W | 0131: â | 0175: » | 0219: █ |  |
| 044: , | 088: X | 0132: ä | 0176: ░ | 0220: ▄ |  |

* **assoc:**

Mit diesem Befehl ändert man den **Dateityp** bei **einer** bestimmten **Endung**. Damit kann man bestimmte Dateien unbrauchbar machen.

Beispiel:

assoc .bat=ERROR

Damit sind alle bat-Dateien unbrauchbar, da der normale Dateityp „batfile“ **überschrieben** ist. Die meisten Dateien haben denselben Dateityp, wie die Endung, nur mit „file“ hintendran:

.bat=batfile

.txt=txtfile

! .jpg=jpegfile !

! .zip=CompressedFolder !

! .bmp=Paint.Picture !

! .doc=Word.Document.8 !

! .pdf=AcroExch.Document !

.exc=txtfile

Die **Liste der Dateiendungen** mitsamt Dateityp lässt sich mit „assoc“ ohne alles anzeigen.

* **attrib:**

Mit „attrib“ werden die **Attribute von Dateien oder Ordnern** geändert.

**+** = Setzt ein Attribut

**-**  = Löscht ein Attribut

**R** = Schreibgeschützte Datei

**A** = Zu archivierende Datei

**S** = Systemdatei

**H** = Versteckte Datei

**/D** = Macht alle Ordner mit dazu

**/S** = Macht alle möglichen Dateien mit dazu

Beispiele:

attrib C:\WINDOWS\system32\taskmgr.exe **+H**

attrib C:\WINDOWS\system32\taskmgr.exe **+H +R –S**

# B

* **Break:**

Dieser Befehl **bestimmt, ob** Strg+C aktiviert oder deaktiviert werden soll. Normalerweise wird der **Verlauf einer Befehls oder einer Batchdatei mit Strg+C unterbrochen.**

Um dies zu umgehen, kann man BREAK OFF verwenden. Möchte man die Tastenkombination wieder aktivieren, verwendet man BREAK ON.

Dieser Befehl hat, wenn die Befehlserweiterungen aktiviert sind, ab Windows XP Home keine Wirkung mehr.

# C

* **cacls:**

Hiermit ändert man die Einstellungen des möglichen **Zugriffs auf eine Datei** oder einen Ordner eines einzelnen Benutzers. Dazu braucht man den Namen des Benutzers und eine Verbindung zu ihm, um den Zugriff auf etwas zu verweigern oder zuzulassen. Die am einfachsten zu merkende Form ist:

/P Benutzername:Zugriffsoption (Zugriffsoption = N (Kein Zugriff)

= R (Nur lesen erlaubt)

= W (Nur schreiben erlaubt)

= C (Bearbeiten)

= F (Alles erlaubt))

Beispiele:

cacls „batch-Lexikon.txt“ /p Ducky:N

cacls „batch-Lexikon.txt“ /p Ducky:R

cacls “batch-Lexikon.txt” /p Ducky:F

Um die **Antwort des Nutzers** zu **überspringen**, muss man vor den Befehl „**echo J|**“ schreiben in der deutschen Sprache. In Englisch ist es demnach „**echo Y|**“.

**echo J|**cacls „batch-Lexikon.txt“ /p „Ducky:N

* **calc:**

Öffnet den Windows Taschenrechner.

* **call:**

Der call-Befehl wird dreierlei verwendet:

1. Aufruf einer beliebigen Datei

2. **Aufruf einer Batch-Datei**. Ist diese absolviert, fährt das Programm normal fort.

3. **Aufruf eines Springpunktes**.

Zu 1:

Der Befehl „call“ sollte nicht zum Aufrufen einer Datei verwendet werden, da der

**Verlauf der Batch-Datei stoppt**, bis das andere Programm geschlossen ist. Stattdessen

verwendet man „***pushd, start, popd***“.

Zu 2:

Dies ist nützlich, wenn man mit mehreren Dateien arbeitet und eine davon als **Kommando-**

**zentrale** verwendet, allerdings stoppt das Programm in diesem Fall auch.

Zu 3:

Hierfür ist der call-Befehl sehr nützlich, da das Programm sich nicht direkt beendet,

sollte der Springpunkt nicht existieren. Stattdessen wird eine Fehlermeldung ange-

zeigt. Der aufgerufene Punkt **stoppt mit seinen Befehlen bis** zu einem **„*goto*-„** oder

**„*exit* /b“**-Befehl. In diesem Fall geht die Batch-Datei **zurück zu dem call-Befehl** und

**fährt mit den Befehlen darunter fort**.

Zusätzlich können einem **Springpunkt Parameter** zugeordnet werden, die dann zu **%~1, %~2**

usw. werden:

call :Punkt1 firefox.exe „C:\Programme\Mozilla Firefox“

pause

:Punkt1

echo %~1 %~2

exit /b

Angezeigt werden würde:

firefox.exe C:\Programme\Mozilla Firefox

Drücken Sie eine beliebige Taste . . .

Durch die „ „ wird der Pfad als einzelnes gewertet, die „ „ jedoch entfernt.

* **cd / chdir:**

Dieser Befehl **wechselt das Verzeichnis**, in der die bat-Datei bei einem Befehl, der die Suche nach einer Datei auslöst, sucht. Dabei ist es egal, ob man den Pfad in Anführungszeichen oder ohne schreibt.

**/D = Wechselt auch das Laufwerk**

.. = Geht zum nächsten Überverzeichnis

Beispiele:

cd /D „C:\Programme“

…ist dasselbe wie…

cd /D C:\Programme

cd .. Ist man in C:\Programme, kommt man hiermit zu C:\

* **chkdsk:**

Dieser Befehl steht für „**ch**ec**k** **d**i**sk**“. Damit wird ein Datenträger analysiert und anschließend wird dessen freier und belegter Speicherplatz sehr detailliert angezeigt.

**/F** = Fehlerbehebung beschädigter Sektoren

**/R** = Anzeige der Fehlersektoren, stellt lesbare Dateien wieder her

Beispiele:

chkdsk F:

chkdsk F: /F /R

* **cls:**

Löscht alles bisher Geschriebene in der Batch-Datei. Dies betrifft nur das **Anzeigefeld** und nicht den Quelltext oder andere Dateien!

* **cmd:**

Öffnet cmd.exe, die Datei, in der alle Batch-Dateien ausgeführt werden.

* **color:**

Der Befehl dient zur **Farbmarkierung von Hintergrund und Schrift**. Gerechnet im Hexadezimalsystem steht der erste Buchstabe/die **erste** Zahl **für den Hintergrund**, der zweite Buchstabe/die **zweite** Zahl **für die Schriftfarbe**.

0 = Schwarz 8 = Dunkelgrau

1 = Dunkelblau 9 = Blau

2 = Dunkelgrün A = Hellgrün

3 = Dunkles Türkis B = Türkis

4 = Dunkelrot C = Rot

5 = Lila D = Pink

6 = Dunkelgelb E = Gelb

7 = Grau F = Weiß

Beispiele:

color 0e = Hintergrund schwarz, Schrift gelb.

Color 1c = Hintergrund dunkelblau, Schrift rot.

Color f4 = Hintergrund weiß, Schrift dunkelrot.

* **Command.com:**

Command.com ist die ältere Version von CMD.exe . Diese Datei kann **keine Namen** lesen, die

**länger als 8 Zeichen** sind (siehe ***Kurznamen***), wechselt Partitionen anders (z.B. F:\ statt

cd /D F:\) und arbeitet langsamer. Command.com erlaubt **kein Beenden durch Drücken des**

**X zum Schließen**, sondern muss durch EXIT beendet werden.

/P = Macht das Programm **nicht beendbar**, EXIT ist wirkungslos. Prozess muss beendet werden.

* **comp:**

„comp“ von „(to) **comp**are , vergleichen“ vergleicht zwei Dateien, ob sie identisch sind oder nicht.

Beispiel:

comp „Pfad1\Datei1.txt“ „Pfad1\Datei2.txt“

* **Compact:**

Dieser Befehl **komprimiert oder dekomprimiert Dateien** oder ganze Ordner. Er hat folgende Parameter:

/A = „Attributes“ = Zeigt versteckte und Systemdateien an.

/C = „Compress“ = Komprimiert die angegebenen Datei. Wird ein Verzeichnis

angegeben, werden für die Zukunft alle später hinzugefügte Dateien ebenfalls komprimiert.

/i = „Ignore“ = Ignoriert Fehler. Standardmäßig stoppt der Befehl bei Fehlern.

/F = „Force“ = Erzwingt das Komprimieren, auch wenn die Datei bereits

komprimiert wurde. Standardmäßig werden diese übersprungen.

/Q = „Quick“ = Zeigt nur die wichtigsten Informationen an.

/S = „Sub“ = Führt den Vorgang auf alle Dateien in allen Unterordner durch.

/U = „Uncompress“ = Dekomprimiert die angegebene Datei. Wird ein Verzeichnis

angegeben, werden für die Zukunft alle später hinzugefügte Dateien ebenfalls dekomprimiert.

Ohne Parameter wird die Komprimierung der Dateien des aktuellen Verzeichnisses angezeigt. Es können auch in einem einzigen Befehl mehrere Dateien angegeben werden oder Platzhalter verwendet werden.

Beispiel:

Compact /C /I /Q “C:\Windows\Granit.bmp”

* **Convert:**

Dieser Befehl **konvertiert** eine Partition von **FAT zu NTFS**. Syntax:

CONVERT Partition [/V] [/X]

/V = Wird gründlicher ausgeführt, dauert aber länger.

/X = Erzwingt, dass die Partition konvertiert werden kann. Alle Programme, die auf diese

Partition zugreifen, werden geblockt.

Beispiel:

Convert G: /X

* **copy:**

Mit diesem Befehl wird eine Datei zu einem anderen Ort kopiert. Die Pfade stehen in

Anführungszeichen. Die Datei kann bei der Ausführung umbenannt werden. Man kann auch mehrere Dateien zu einer zusammenfassen. Es gilt:

/b = Weist auf eine binäre Datei (Bild, Musik, exe) hin

/y = Keine Warnmeldung bei Überschreibung von Dateien.

/-y = Extra Warnmeldung bei Überschreibung von Dateien.

Beispiele:

copy /B „C:\WINDOWS\Granit.bmp“ „C:\Programme\Granit.bmp“

copy „C:\Windows\Tetdoku1.txt“ + „C:\Windows\Textdoku2.txt“ „Textdokumente.log“

Es ist auch möglich, über den COPY-Befehl Eingaben vom Benutzer zu bedingen. Der Befehl dazu lautet:

Copy con „Dateiname.Endung“

Bei diesem Befehl kann der Benutzer so lange Codes eingeben, bis er Strg+Z drückt und bestätigt. Der eingegebene Code wird in die angegebene Datei kopiert. Bevor man den Befehl in einer Batchdatei verwendet, sollte man den Benutzer darauf hinweisen, Strg+Z am Ende zu betätigen.

# D

* **date:**

Das Datum wird angezeigt und kann bearbeitet werden.

Beispiel:

date 01.01.2000

* **defrag:**

Mit diesem Befehl defragmentiert man eine Partition. Erweiterungen sind möglich:

**-a** = Partition **nur überprüfen**, nicht defragmentieren

**-f** = Defragmentieren wird auch bei Fehlern fortgesetzt

**-v** = Anzeige von **mehr Informationen** über Defragmentierung

Beispiele:

defrag C: -a

* **del / erase:**

Diese beiden Befehle dienen zum (unwiderruflichen) **Löschen von Dateien**, welche angegeben werden müssen, zusammen mit dem Pfad, der zu ihnen führt. Erweiterungen verfeinern das Verfahren:

**/p** = Löschen muss bestätigt werden bei jeder Datei vom Benutzer

**/f** = Schreibgeschützte Dateien werden auch gelöscht

**/s** = Alle Dateien in angegebenem Pfad und die in Unterordnern werden gelöscht

**/q** = Platzhalter wie \* werden erlaubt

Beispiele:

erase “C:\WINDOWS\Granit.bmp”

del “C:\WINDOWS\Granit.bmp”

erase /f /q “C:\WINDOWS\Granit.\*”

* **dir:**

**Listet** sämtliche **Dateien auf**. Ein Pfad ist zu raten der Übersicht zuliebe. Sind in dem Pfad Leerzeichen, muss den Pfad mit Anführungszeichen umschlossen werden.

**/A** = Attribute der Dateien: **D** = Verzeichnisse ; **R** = schreibgeschützt

**H** = Versteckt ; **S** = Systemdatei

**- vorangestellt kehrt die Bedeutung um.**

**/B** = Nur Dateipfade werden angezeigt, keine weiteren Informationen.

**/C** = Zeigt das Tausendertrennzeichen bei Dateigrößen an (= 1.000 statt 1000). Dies

ist standardmäßig aktiviert. /-C deaktiviert sie.

**/L** = Pfade der Dateien werden kleingeschrieben.

**/O:Folge** = Sortiert die anzuzeigenden Dateien nach angegebener Folge: **N** = Name

**E** = Erweiterung

**G** = Verzeichnisse

**D** = Datum/Zeit

**S** = Dateigröße

**/P** = Pause nach jeder vollen Seite.

**/Q** = Gibt den Besitzer der Datei aus.

**/S** = Zeigt auch Dateien aller Unterverzeichnisse an.

**/W** = Breitformat (Dateinamen auch nebeneinander).

**/X** = Zeigt auch ***Kurznamen*** an.

Beispiele:

dir „C:\Dokumente und Einstellungen“

dir C:\

dir /S

dir /O:E /P C:\Programme

* **driverquery:**

Listet installierte **Treiber** auf. /V erhöht die Menge gegebener Informationen.

# E

* **echo. :**

Mit dem Befehl „echo.“ Macht man eine **Leerzeile** in die Anzeige des Programmes, damit ein Text übersichtlicher wird.

Beispiel:

@echo off

echo Ein schöner Tag heute.

Echo.

Echo Und die Erkenntnis der einfachen Leerzeilen ist noch besser :D

pause

EXIT

* **echo:**

1. Dieser Befehl wird zur **Wiedergabe von Text** verwendet. Die meist lästige Anzeige der

Verzeichnisse davor wird mit @echo off deaktiviert, was der Übersicht dient.

Beispiele:

@echo off

echo Diese Nachricht wird angezeigt.

Pause

**2.** Mit dem Befehl „echo“ lassen sich auch **Dateien neu auf dem Computer speichern**, falls sie mit Notepad/Notizbuch/Editor geschrieben werden können.

Beispiele:

echo Ich überschreibe eine Datei. > „C:\WINDOWS\Granit.bmp“

echo Ich erstelle eine Datei. > „C:\WINDOWS\Neue Datei.txt“

echo Ich füge Text zu einer Datei hinzu. >> „C:\WINDOWS\kb900685.log“

* **edit:**

Mit diesem Befehl öffnet man ein **Text**e abrufendes und schreibendes **Programm**, das die kompliziertesten Texte innerhalb von ein paar Sekunden abrufen kann. Dieses Programm hat eine **Zeilen- und Zeichenangabe** zum leichteren Lesen und dasselbe Design wie Qbasic.

Es ist eine ganz andere Form als das eigentliche bat-Fenster, folgt aber daraus. Farben von allem sind einstellbar, sogar die Maus ist abstellbar.

**/H** = Großes blaues Fenster, ähnlich wie ein Blue Screen.

**/B** = Kleines schwarzes Fenster, QBASIC-Format.

**/R** = Datei im schreibgeschützten Modus anzeigen.

Datei = Dateipfad, bei Leerzeichen „ setzen!

Beispiele:

edit /H

Damit öffnet man einfach nur das Fenster.

Edit /H /R “C:\WINDOWS\system32\shell32.dll”

* **endlocal:**

Dieser Befehl beendet den Raum von dem Befehl „***setlocal***“.

* **erase / del:**

Diese beiden Befehle dienen zum (unwiderruflichen) **Löschen von Dateien**, welche angegeben werden müssen, zusammen mit dem Pfad, der zu ihnen führt. Erweiterungen verfeinern das Verfahren:

**/p** = Löschen muss bestätigt werden bei jeder Datei vom Benutzer

**/f** = Schreibgeschützte Dateien werden auch gelöscht

**/s** = Alle Dateien in angegebenem Pfad und die in Unterordnern werden gelöscht

**/q** = Platzhalter wie \* werden erlaubt

Beispiele:

erase “C:\WINDOWS\Granit.bmp”

del “C:\WINDOWS\Granit.bmp”

erase /f “C:\WINDOWS\Granit.\*”

* **Errorlevel:**

Ein „Errorlevel“ wird **nach jedem Befehl neu festgelegt**, solange dieser nicht mit dem Errorlevel arbeitet. Er wird meist in ***IF***-Befehlen verwendet, kann aber auch andere Funktionen erfüllen.

Der Errorlevel ist eine Zahl. Die Bedeutung des Errorlevels ist oft ein Rätsel. Einige sind aber sicher:

Errorlevel 0 = Befehl wurde erfolgreich und ohne Fehler ausgeführt.

Errorlevel 1 = Befehl wurde ausgeführt, wies aber Fehler auf.

Errorlevel 9009= Befehl oder Datei konnte nicht gefunden werden.

* **exit:**

Damit schließt man die aktuelle CMD.EXE.

**/B** = Schließt nur die Batch-Datei, nicht das CMD-Fenster

* **explorer:**

Öffnet den Windows Explorer. Über den Befehl „***start***“ lassen sich unter Windows XP mit Hilfe des Explorers Warnmeldungen erscheinen. Dazu gibt man ein:

***start*** explorer „Pfad\Datei“

Im Normalfall kommt die Meldung, dass der Herausgeber der Datei nicht erkannt werden konnte und dass die Ausführung der Datei gefährlich sei, was jedoch selten der Fall ist.

# F

* **FC:**

Vergleicht zwei Dateien oder zwei Dateiensätze und zeigt die Unterschiede zwischen ihnen an.

Syntax für Normalmodus:

FC [/C] [/L] [/N] [/T] [/U] [/W] „Dateipfad1“ „Dateipfad2“

Syntax für binären Modus:

FC /B „Dateipfad1“ „Dateipfad2“

/B = Vergleicht auf binärer Basis (z.B. Bilder, Office Word-Dokumente, exe-Dateien usw.)

/C = Ignoriert Groß- und Kleinschreibung

/L = Vergleicht Dateien auf Ascii-Basis (z.B. Dateien von Notepad mit Codec ANSI)

/N = Zeigt die Zeilennummern unterschiedlicher Zeilen an

/T = Ändert Tabulatorzeichen nicht zu Leerzeichen.

/U = Vergleicht Dateien auf Unicode-Basis (z.B. Dateien von Notepad mit Codec Unicode)

/W = Komprimiert Tabulator- und Leerzeichen.

Beispiele:

FC /B „C:\Programme\Mozilla Firefox\firefox.exe“ „%appdata%\Mozilla Firefox\firefox.exe“

FC /L /W „%userprofile%\Desktop\Neu Textdokument.txt“ „C:\Programme\Batch\Test1.bat”

* **find:**

Dieser Befehl **sucht nach einer Zeichenfolge** in einer **einzelnen Datei**. Anschließend werden

alle Zeilen mit der Zeichenfolge ausgegeben. Als Überschrift wählt der Befehl den

Dateinamen.

**/V** = Zeigt alle Zeilen an, die die Zeichenfolge NICHT enthalten.

**/C** = Zeigt nur die Anzahl der die Zeichenfolge enthaltenden Zeilen an.

**/N** = Zeigt die Zeilen mit ihren Zeilennummern an.

**/I** = Ignoriert Groß-/Kleinschreibung bei der Suche.

Beispiel:

find /N „1“ „Neu Textdokument.txt“

* **findstr:**

„findstr“ ist der genauere Befehl zu „***find***“. Der Befehl findet eine Zeichenfolge in einer

oder mehreren Dateien.

**/C:“Zeichenfolge“** = Sucht nach Zeichenfolge buchstabengetreu.

**/D:Verzeichnis** = Durchsucht eine mit **Semikolons** getrennte Verzeichnisliste

**/I** = Ignoriert Groß-/Kleinschreibung.

**/M** = **Gibt nur die Namen der Dateien aus**, die die Zeichenfolge enthalten.

**/N** = Gibt die Zeilennummer vor jeder Trefferzeile an.

**/P** = Überspringt Dateien mit Zeichen, die nicht gedruckt werden können.

**/S** = Sucht nach entsprechenden Dateien im aktuellen Verzeichnis

**und allen Unterverzeichnissen**.

**/V** = Gibt Zeilen aus, die die Zeichenfolge nicht enthalten.

**/X** = Gibt Zeilen aus, die vollkommen übereinstimmen.

Beispiel:

findstr /C:”This program cannot be run in DOS mode.” /M /S “C:\\*”

Der Nachteil dieses Befehls: Kann eine Datei nicht geöffnet werden, kommt eine Fehler-

meldung. Dasselbe gilt für Zeilen, die zu lang sind.

* **FOR:**

Der FOR-Befehl ist oft der **wichtigste und vielseitigste Befehl**. Meist verwendet wird

der Parameter /F, um für alles in Klammern angegebenen Dingen einen oder mehrere

Befehle auszuführen. Dazu zuerst die Syntax:

**FOR /F [„Optionen“] %%Variable IN (Dateiensatz) DO Befehl [Parameter]**

**FOR /F [„Optionen“] %%Variable IN („Zeichenkette“) DO Befehl [Parameter]**

**FOR /F [„Optionen“] %%Variable IN (‚Befehl‘) DO Befehl [Parameter]**

oder, unter Verwendung der Option „usebackq“:

**FOR /F [„Optionen“] %%Variable IN (Dateiensatz) DO Befehl [Parameter]**

**FOR /F [„Optionen“] %%Variable IN (‚Zeichenkette‘) DO Befehl [Parameter]**

**FOR /F [„Optionen“] %%Variable IN (`Befehl`) DO Befehl [Parameter]**

%%Variable ist immer nur **ein Buchstabe**! Dabei ist zu beachten, dass **a nicht dasselbe wie A** ist!

Die „Optionen“ sind hierbei wie folgt:

**eol=**c - Gibt das Anfangszeichen für Zeilen an, die ignoriert werden sollen

(nur eins).

**Skip=n** - Gibt die Anzahl der Zeilen an, die am Anfang einer

Datei übersprungen werden.

**Delims=**xxx - Gibt einen Satz von Trennzeichen an in %Variable. Diese ersetzen

die Standardtrennzeichen TAB und Leerzeichen.

**Tokens=**a,b-d \* - Gibt an, welche Token von jeder Zeile an die

FOR-Schleife weitergegeben werden.

Das führt dazu, dass zusätzliche Variablen erzeugt

werden. Mit der Form b-d wird dabei ein Bereich vom

b-ten bis zum d-ten Token angegeben. Wenn das letzte

Zeichen ein Sternchen ist, wird eine zusätzliche

Variable deklariert, die den verbleibenden Text

dieser Zeile enthält. Für jede Zahl von b-d gibt es ein Token.

**Usebackq** - Ändert die Syntax in den Klammern. Normalerweise nutzlos.

Die **Variabeln** können immer nur ein Zeichen sein, allerdings sind diese **nach dem „DO“-**

**Befehl veränderbar:**

%%~L - Expandiert %%L und entfernt alle umschließenden

Anführungszeichen („).

%%~fL - Expandiert %%L zu einem **vollständigen Dateipfad**.

%%~dL - Erzeugt nur den Laufwerkbuchstaben von %%L.

%%~pL - Erzeugt nur den Pfad von %%L.

%%~nL - Erzeugt **nur den Dateinamen** von %%L.

%%~xL - Erzeugt nur die Dateierweiterung von %%L.

%%~sL - Erzeugter Pfad enthält **nur kurze Dateinamen**.

%%~aL - Erzeugt die **Dateiattribute** von %%L.

%%~tL - Erzeugt **Datum und Zeit** von %%L.

%%~zL - Erzeugt die **Dateigröße** von %%L.

%%~$PATH:L - Durchsucht die in der ***PATH***-Umgebungsvariablen

angegebenen Verzeichnisse und expandiert die erste

gefundene Datei %%L zu dem vollständigen Dateinamen.

Wenn der Name der Umgebungsvariablen nicht definiert

ist oder diese Datei bei der Suche nicht gefunden wurde,

wird dieser Parameter zu einer leeren Zeichenkette

expandiert.

Diese Parameter können auch miteinander kombiniert werden:

%%~dpL - Erzeugt den Laufwerkbuchstaben und Pfad von %%L.

%%~nxL - Erzeugt den **Dateinamen und die Dateierweiterung** von %%L.

%%~fsL - Expandiert %%L zu einem vollständigen Namen, der nur

kurze Dateinamen enthält.

%%~dp$PATH:L – Durchsucht die in der ***PATH***-Umgebungsvariablen

angegebenen Verzeichnisse nach %%L und erzeugt den

Laufwerkbuchstaben und Pfad der ersten gefundenen Datei.

%%~ftzaL - Expandiert %%L zu einer Zeile, die der Ausgabe des DIR-

Befehls entspricht.

Hier eine Erläuterung der Optionen:

„**usebackq**“ verändert die **Zeichen innerhalb der Klammer** zwischen IN und DO (siehe oben).

„**tokens**“ und „**delims**“ sind an einem einfachen Beispiel zu erkennen:

FOR /F “tokens=1-4 delims=, “ %%A IN (“Dieser Text wird angezeigt.”) DO echo %%A %%B %%C %%D

* „tokens“ gibt es von 1 bis 4. „delims“ sind die **Trennzeichen**, also **Komma und Leerzeichen**.
* In den Klammern ist die Zeichenfolge „Dieser Text wird angezeigt“ zwischen jedem Wort ein Leerzeichen und somit auch ein Trennzeichen, das durch „delims“ festgelegt wurde. Da es 4 Wörter sind und da es 1-4, also insgesamt 4 „tokens“ gibt, lassen sich alle wiedergeben.
* Das erste Token ist „Dieser“, da darauf ein Leerzeichen folgt, was durch „delims“ zu einem Trennzeichen gemacht wurde.

FOR /F „tokens=1,\* delims==“ %%J IN („Nun ist das = ein Trennzeichen, kein Komma mehr oder Leerzeichen. “) DO echo %%J %%K %%L %%M

* Ausgegeben werden würde „Nun ist das ein Trennzeichen, kein Komma mehr oder Leerzeichen. %L %M”
* **Token 1** ist „Nun ist das „, **Token 2** ist „ ein Trennzeichen, kein…oder Leerzeichen. „, Token 3 ist nicht gegeben und Token 4 auch nicht, deswegen kommt %L und %M.
* Das **Sternchen bei „tokens“** steht dafür, dass das letzte Token **alle** **folgenden Tokens** sein wird. Als Trennzeichen wurde „=“ angegeben, deswegen ist Token 1 „Nun ist das „.

FOR /F „tokens=1,4 delims= „ %%f IN („Ein weiterer Text wird auseinander genommen. “) DO echo %%f %%g **%%h**

* Ausgegeben werden würde: „Ein wird %h“. Token 1 ist „Ein“, Token 2 ist „weiterer“ etc, da aber **Token 2,3,5… nicht beachtet** werden, sondern nur 1 und 4, werden nur diese ausgegeben. Dabei wird %%g nicht zu Token 2, sondern zu Token 4.
* **%%h existiert nicht** als weiteres Token, da nur 1 und 4 da sind. Deshalb kommt „%h“ noch in die Ausgabe.

Hier einige Beispiele, was diese Befehle tun können:

FOR /F “delims=” %%A IN (‘***dir*** /A /B /S “%homedrive%\”’) DO (if “%%~**nx**A” == “firefox.exe” echo %%A)

=> Durchsucht alle Pfade in dem DIR-Befehl nach „firefox.exe“ als Dateiname und gibt dessen Pfad wieder.

FOR /F „tokens=1\* delims=: „ %%A IN (‚***systeminfo***‘) DO (if „%%A“ == „Betriebssystemname“ echo Sie verwenden %%B als Betriebssystem.)

=> Durchsucht die Ausgabe von „***systeminfo***“ (wird nicht angezeigt) nach „Betriebssystemname“ und gibt dessen Wert nach dem Doppelpunkt wieder.

FOR /F %%A IN (‘***tasklist*** /SVC’) DO if /I “%%A” == “notepad.exe” ***taskkill*** /F /IM notepad.exe

=> Erstellt eine Liste einiger Prozesse und deren Dienste in einer Datei auf dem Desktop.

FOR /F “delims=” %%A IN (‘***dir*** /A /B /-C /S’) DO set /a Size=%Size% + %%~**z**A

echo Der aktuelle Ordner ist %Size% KB gross.

=> Addiert die Dateigrößen aller Dateien im aktuellen Ordner und allen

Unterverzeichnissen und gibt sie am Ende wieder.

Man kann mit dem „FOR /F“-Befehl noch viel mehr anstellen, aber es gibt auch noch **/R** und **/L**:

**FOR /R [[Laufwerk:]Pfad] %%Variable IN (Satz) DO Befehl [Parameter]**

Mit diesem Befehl führt das Programm den Befehl nach „DO“ für alle Dateien aus, die in „(Satz)“ angegeben wurden in dem Pfad, der nach /R eingegeben wurde.

Dadurch lassen sich Dateien z.B. nach ihrem Quellcode durchsuchen:

FOR /R “C:\” %%A IN (\*.txt) DO (

findstr /I /M /C:”This program cannot be run in DOS mode.” „%%A“ >nul

if „%ERRORLEVEL%“ == „0“ echo %%~**f**A kann nicht in DOS ausgeführt werden.

)

=> Damit wird jede Datei mit der Endung .txt nach einer Zeile „This … in DOS mode.“ Gesucht, wobei dir Groß- und Kleinschreibung ignoriert wird.

**FOR /L %%Variable IN (Start,Schritt,Ende) DO Befehl [Parameter]**

Dieser Befehl führt den Befehl nach DO so lange durch, wie der Benutzer es bestimmt. (Start,Schritt,Ende) ist dabei entscheidend:

Das Programm geht in **„Schritt“ Schritten von „Start“ zu „Ende“** und führt den Befehl so oft aus, bis das Ende erreicht wurde:

FOR /L %%A IN **(1,1,9)** DO (if “%Platz1%” == “%%A” set /a Platz2=%%A +1)

=> Damit wird der Befehl 9mal ausgeführt.

Der FOR-Befehl erlaubt durchaus Klammern zur Einschränkung von internen Befehlen und zur Übersicht, dazu das letzte Beispiel:

FOR /L %%A IN (1,1,9) DO (

if “%Platz1%” == “%%A” (

set /a Platz2=%%A +1

))

**Man darf auf keinen Fall vergessen**, die **Klammern** zu **schließen**, ansonsten **beendet sich das Programm selbst**!

**Übersicht Syntax:**

FOR /F [„Optionen“] %%Variable IN (Dateiensatz) DO Befehl [Parameter]

FOR /F [„Optionen“] %%Variable IN („Zeichenkette“) DO Befehl [Parameter]

FOR /F [“Optionen”] %%Variable IN (‘Befehl’) DO Befehl [Parameter]

oder, unter Verwendung der Option „usebackq“:

FOR /F [„Optionen“] %%Variable IN (Dateiensatz) DO Befehl [Parameter]

FOR /F [„Optionen“] %%Variable IN (‚Zeichenkette‘) DO Befehl [Parameter]

FOR /F [„Optionen“] %%Variable IN (`Befehl`) DO Befehl [Parameter]

FOR /R [[Laufwerk:]Pfad] %%Variable IN (Satz) DO Befehl [Parameter]

FOR /L %%Variable IN (Start,Schritt,Ende) DO Befehl [Parameter]

* **format:**

Formatiert eine Partition.

/Q = Schnellformatierung (normalerweise schneller, aber trotzdem fehlerfrei)

* **FTP:**

„**F**ile **t**ransfer **p**rotocol“ zum **Senden und Empfangen von Dateien über das Internet**. Der Befehl ist recht einfach zu verstehen, allerdings muss man einen **FTP-fähigen Server** mit Logindaten haben, um diesen Befehl für dort gespeicherte Dateien verwenden zu können.

Der FTP-Befehl kann entweder automatisiert verwendet werden oder Schritt für Schritt eingegeben werden.

Parameter für automatisierte Version:

Syntax: **FTP** [-v] [-d] [-i] [-n] [-g] [-s:Dateiname] [-A] [Host]

-A Meldet den Benutzer als „Anonym“ an.

-d Aktiviert debugging.

-g Deaktiviert Platzhalter.

-i Lädt mehrere Dateien ohne Nachfrage hoch.

-n Keine automatische Anmeldung beim Öffnen der Seite.

-s:*Datei* Gibt eine Datei an, aus der FTP-Befehle gelesen werden sollen.

-v Keine Informationen anzeigen.

Host Website oder IP.

Parameter für manuelle Version:

? Zeigt alle Befehle an.

Ascii Setzt den Datentransfertyp auf Ascii (.bat, .txt…).

Bell (De-)Aktiviert eine akustische Meldung bei erfolgreichem Senden.

Binary Setzt den Datentransfertyp auf Binär (.exe, .zip…).

Bye / Quit Schließt FTP.

Cd *Pfad* Wechselt das Verzeichnis auf dem Server.

Close / Disconnect Ausloggen

Delete *Datei* Datei auf dem Server löschen.

Dir Verzeichnis des Servers anzeigen.

Get *Datei* Datei runterladen.

Glob (De-)Aktiviert Platzhalterzeichen zum Runter- oder Hochladen.

Help *Befehl* Zeigt Hilfe zu einem Befehl an.

Lcd *Pfad* Wechselt das Verzeichnis auf dem Computer.

Mdelete *Dateien* Löscht mehrere Dateien.

Mget *Dateien* Lädt mehrere Dateien runter.

Mkdir *Ordner* Erstellt einen Ordner auf dem Server.

Mput *Dateien* Lädt mehrere Dateien hoch.

O [www.*website*.de](http://www.website.de) O für „Open“. Die Website wird nie mit http geschrieben.

Prompt (De-)Aktiviert den automatischen Sendemodus.

Put *Datei* Lädt eine Datei hoch.

Pwd *Passwort* Wird nach Benutzereingabe direkt verlangt.

Ren „*Datei“* „*Datei2“* Benennt eine Datei um.

Rmdir *Ordner* Löscht einen Ordner.

User *Benutzername* Wird beim öffnen einer Seite direkt verlangt.

Verbose Zeigt mehr/weniger Informationen an beim Senden oder Empfangen.

Bei der automatisierten Version stehen **Befehle der manuellen Version in der Datei hinter -s**:.

Beispiel:

ftp

o [www.batchlog.pytalhost.com](http://www.batchlog.pytalhost.com)

Ascii

Prompt

Lcd „%userprofile%\Desktop“

Mput „Neu Textdokument.txt“ „Neu Skripttextdokument.npp“

Binary

Put Batch.html

Bye

* **ftype:**

Dieser Befehl legt fest, **welches Programm einen Dateityp** (z.B. txtfile) **öffnet**. Er wird

oft benutzt in Verwendung mit „***assoc***“, wodurch sich eine neue Endung erstellen lässt.

Dazu jedoch muss der Computer dazu neustarten.

Syntax:

ftype Dateityp=Befehl

Beispiel:

ftype txtfile=“C:\Programme\Windows NT\Zubehör\wordpad.exe“ %1

# G

* **goto:**

Mit diesem Befehl geht man zu einem festgelegten (Spring-)Punkt (z.B. :Springpunktname).

Beispiele:

:1

echo Text, der pausenlos wiederholt wird.

Goto :1

EXIT

goto :end

echo Diese Nachricht wird nicht angezeigt, da sie per goto übersprungen wird.

:end

EXIT

* **gpresult:**

Listet einige Informationen über die Computereinstellungen auf. **/V** oder **/Z** zeigen ausführlichere Infos an.

# H

* **Help:**

Dieser Befehl zeigt einige Befehle in Batch an. Kann auch in manchen Fällen Hilfe zu einem Befehl anzeigen: Help *Befehl*

# I

* **iexplore:**

Öffnet den Microsoft Internet Explorer. Es kann direkt eine Seite angegeben werden:

iexplore [www.google.de](http://www.google.de)

* **if:**

Der IF-Befehl **stellt Bedingungen** dar, die erfüllt werden müssen, damit bestimmte Befehle, die nach der IF-Bedingung folgen, ausgeführt werden. Die Konstruktur erlaubt Klammern.

Es gibt verschiedene Typen von IF-Befehlen, hier ein kurzer Überblick:

Typ 1: IF [/I] [not] „%***Variable***%“ Vergleichsoperator „Zeichenfolge“ *Befehl*

Typ 2: IF [not] exist “Pfad” *Befehl*

Typ 3: IF [not] defined ***Variable*** *Befehl*

Typ 4: IF [not] ***errorlevel*** Nummer *Befehl*

Vergleichsoperatoren:

**EQU** = **Equ**al = Gleich

**NEQ** = **N**ot **Eq**ual = Nicht gleich

**GTR** = **G**rea**t**e**r** Than = Größer als

**GEQ** = **G**reater and **Eq**aul = Größer gleich

**LEQ** = **L**ess and **Eq**ual = Kleiner gleich

**LSS** = **L**e**ss** than = Kleiner als

**Vergleichsoperatoren** werden vor allem bei der **Arbeit mit Zahlen** verwendet. Ist dies der Fall, sollte in der Zeile zuvor oder danach nicht dieselbe ***Variable*** mit „==“ mit etwas verglichen werden.

**Typ 1** der IF-Befehle kann anstelle von „Variable==Zeichenfolge“ auch „Variable==Variable“ oder „Zeichenfolge==Zeichenfolge“ sein.

Die anschließenden Befehle können **all ihre Erweiterungen verwenden** ohne Einschränkung. Somit können auch **mehrere Bedingungen** gleichzeitig gesetzt werden.

If „%a%“ == „1“ (

If “%a%” == “2” (

If “%a%” == “3” ***echo*** a ist 3.

))

Man darf jedoch nicht vergessen, die **Klammern zu schließen**!

Bei der Arbeit mit Zahlen werden die Vergleichsoperatoren verwendet:

***Set*** /p Zahl=Geben Sie hier eine Zahl ein:

IF "%Zahl%" GTR "0" echo Ihre Zahl war größer als 0.

IF "%Zahl%" LSS "0" echo Ihre Zahl war kleiner als 0.

IF "%Zahl%" EQU "0" echo Ihre Zahl war genau 0.

**Achtung:** Wenn eine der beiden Zahlen **mehr oder weniger Stellen** als die andere hat, verläuft der Befehl **fehlerhaft**!

Beispiel: if "500" LSS "51" echo Der Befehl hat was falsch gemacht!

Um diesen Fehler zu umgehen, kann man folgende Befehlsreihe verwenden:

***Set*** /p Zahl1=Zahl1:

***Set*** /p Zahl2=Zahl2:

***Set*** /a Differenz=%Zahl1% - %Zahl2%

IF "%Differenz:~0,1%" == "-" (***echo*** Zahl1 ist größer als Zahl2.) ELSE (***echo*** Zahl1 ist kleiner als Zahl2.)

Was die Befehlsreihe tut: Sie schaut, ob bei der Differenz der beiden Zahlen das erste Zeichen ein Minus ist. Wenn nicht, muss Zahl2 kleiner sein als Zahl1.

Die beiden ***ECHO***-Befehle können durch beliebige andere ersetzt werden.

Das Problem an dieser Funktion ist, dass keine der beiden Zahlen größer sein darf als „4.294.967.295“ (~ 232 oder 655362).

**Typ 2** der IF-Befehle ist wohl der einfachste Teil. Man stellt die Bedingung, dass wenn eine Datei [nicht] existiert, der Befehl danach ausgeführt wird.

**Typ 3** der IF-Befehle bezieht sich auf Variablen, die durch ***SET*** festgelegt wurden. Ist die Variable [nicht] festgelegt, wird der Befehl danach ausgeführt:

If defined Dateipfad echo Es wurde bereits ein Dateipfad angegeben!

**Typ 4** der IF-Befehle stellt die Bedingung, dass wenn der ***Errorlevel*** [nicht] gleich einer bestimmten Zahl ist, der Befehl danach ausgeführt wird.

If errorlevel 1 echo Fehler! Befehl konnte nicht korrekt ausgeführt werden!

# K

* **Kurznamen:**

Kurznamen sind **verkürzte Namen von Dateien und Ordner**, die länger als **8 Zeichen** sind. Sie

wurden früher großteils in DOS verwendet, da es nicht mehr als 8 lesen konnte. Außerdem

enthalten Kurznamen niemals Leerzeichen.

Kurznamen sind einfach herauszufinden: Man nimmt die ersten 6 Zeichen und hängt ein **~1** dran.

Wenn nötig wird NACH dem **~1** die Ordner- oder **Dateiendung** angehängt.

Nach dem Prinzip wird aus „batch-Lexikon.txt“ „BATCH-**~1**.**TXT**“.

Existiert ein Ordner oder eine **Datei mit seinen ersten 6 Zeichen doppelt**, wird **nach**

**Alphabet** gerechnet. Der zweite Ordner / Datei ist anstatt **~1** dann **~2**, der dritte **~3** etc.

Nun mal als Darstellungen dieser Pfad umgewandelt:

C:\Dokumente und Einstellungen\Benutzername\Lokale Einstellungen\Temp\batch-Lexikon.txt

C:\DOKUME~1\BENUTZ~1\LOKALE~1\TEMP\BATCH-~1.TXT

Kurznamen sind vor allem wichtig in ***Command.com*** .

# L

* **Label:**

Der Befehl „Label“ erstellt, löscht oder benennt eine Partition um. Syntax:

Label Partition:[Name]

Label [/MP] [Partition: | Partitionsname]

Partition: = Beispiel: „F:“

Partitionsname = Beispiel: „Daten“

/MP = Gibt an, dass die Partition als Bereitstellungspunkt behandelt werden soll.

# M

* **Md / mkdir:**

Erstellt einen Ordner. (md=mkdir) Was man davon benutzt, ist ohne Belang.

Beispiel:

mkdir "C:\Programme\Neuer Ordnername"

* **mode:**

Mit diesem Befehl lassen sich **Geräte beeinflussen** wie z.B. das **Fenster von CMD**, in dem Fall

lautet die Syntax wie folgt:

mode con CP SELECT=yyy

=> Ändert Codepage (~Sprache)

mode con CP [/STATUS]

=> Zeigt Codepage an

mode con [COLS=c] [LINES=n]

=> Ändert Fenstergröße: COLS=**Col**umn**s** (Spalten / Breite) ; LINES=Zeilen / Tiefe

=> Bei 1280x1024 ist **Vollbild COLS=157 LINES=300**

mode con [RATE=r DELAY=d]

=> Ändert die Geschwindigkeit, wenn man eine Taste gedrückt hält (aaaaaaaaa…)

=> RATE=Geschwindigkeit der Wiederholungen

=> DELAY=Dauer, bis die Taste ständig wiederholt wird

* **more:**

Zeigt den Quellcode einer Datei seitenweise an.

**/E** = Aktiviert folgende Erweiterungen:

**/C** = Löscht den Bildschirminhalt vor dem Anzeigen der nächsten Seite

**/S** = Fasst mehrere Leerzeilen zu einer Leerzeile zusammen

**/Tn** = Fasst Tabulatorzeichen zu n Anzahl Leerzeichen zusammen (Standard = 8)

**+n** = Beginnt mit der Anzeige des Quellcodes ab dem Zeichen n.

Beispiel:

more /E /C /S "batch-Lexikon.txt"

Mit der Eingabe Q bei –Fortsetzung–wird der Befehl geschlossen (Q von Quit)

* **move:**

**Verschiebt Dateien**. Datei und Ziel müssen in Anführungszeichen stehen. Zwischen Datei und Ziel ist ein Leerzeichen.

**/y** = Keine Warnmeldung bei Überschreibung von Dateien.

**/-y** = Extra Warnmeldung bei Überschreibung von Dateien.

Beispiele:

move "C:\WINDOWS\Granit.bmp" "C:\Programme\Granit.bmp"

* **mplay32:**

Dies ist der Media Player im Verzeichnis C:\Windows\system32. Er kann nur wenige Formate

abspielen und wird im Normalfall nie gebraucht.

Beispiel:

***start*** mplay32 "C:\Dokumente und Einstellungen\Benutzername\Eigene Dateien\Eigene Videos\Video.XXX"

* **msg:**

Dieser Befehl wird bis spätestens „Windows XP Professional SP3“ verwendet. Er **sendet eine Nachricht** an alle auf diesem Computer angemeldeten Benutzer. Es kommt ein kleines Fenster – meinst in der Mitte des Bildschirms – welches eine Nachricht anzeigt.

Die Syntax lautet:

Msg \* Nachricht-Text

Dabei muss der Stern bestehen bleiben.

* **msgbox:**

Dieser Befehl wird ab „Windows Vista“ anstelle von „***Msg***“ verwendet. Ein kleines Fenster erscheint – meist in der Mitte des Bildschirms – mit der eingegebenen Nachricht. Es wird dieselbe Syntax **mit Stern** verwendet:

Msgbox \* Nachricht

# N

* **net:**

Arbeitet „net“ mit **Diensten**, bezieht sich der Befehl auf den **Anzeigenamen des Dienstes**.

**1.** net start „Dienst“ : Startet einen installierten Dienst.

**2.** net stop „Dienst“ : Stoppt einen installierten Dienst.

**3.** net pause „Dienst“ : Pausiert einen installierten, gestarteten Dienst.

**4.** net continue „Dienst“: Fährt einen pausierten Dienst fort.

**5.** net helpmsg Zahl : Zeigt die Hilfenachricht mit der Nummer „Zahl“ an.

**6.** net send Nachricht : Sendet eine Nachricht an alle mit dem aktuellen Computer

verbundenen PCs.

**7.** net user: Erstellt oder löscht einen Benutzer oder ändert das Passwort.

Net user Benutzername /ADD

net user Benutzername /DELETE

net user Benutzername Kennwort|\*

* **netsh firewall add/delete allowedprogram:**

Dieser Befehl fügt der **Windows-Firewall** eine Datei zu oder löscht eine, ob diese **auf das Internet zugreifen darf**. Dabei gilt:

„netsh firewall add allowedprogram Pfad\Dateiname **Dateiname** Modus“

oder

„netsh firewall delete allowedprogram Pfad\Dateiname“

Beispiele:

netsh firewall add allowedprogram C:\WINDOWS\system32\cmd.exe **cmd.exe** ENABLE

netsh firewall delete allowedprogram C:\WINDOWS\system32\cmd.exe

* **netsh firewall set opmode:**

Dieser Befehl ändert die **Konfigurationseinstellungen der Windows-Firewall**, solange **keine andere Firewall installiert** ist. Parameter sind ENABLE und DISABLE. Sollte keine andere Firewall installiert sein, kommt bei DISABLE eine Warnmeldung, der Computer könne gefährdet sein.

Beispiel:

netsh firewall set opmode disable

* **netsh firewall show opmode:**

Dieser Befehl zeigt die Aktivität der Firewall in allen Bereichen an.

* **netstat:**

Mit diesem Befehl kann man die Verbindungen zwischen dem Computer mit anderen anzeigen lassen. Hierzu gibt es Erweiterungen:

**-a** = Zeigt alle Verbindungen und abhörenden Ports an.

**-n** = Zeigt alle Verbindungen in Zahlen bzw. Ips an.

# P

* **path:**

Legt die **Standardpfade** fest, **in denen CMD nach einem Befehl sucht**. Die vorherigen Pfade werden überschrieben. Die einzelnen Pfade werden durch ein Semikolon getrennt.

* **pause:**

Pausiert den Verlauf der Batch-Datei und fordert zu einem Tastendruck des Benutzers auf.

Mit „***>nul***“ wird die Anzeige „Drücken Sie eine beliebige Taste . . .“ unterdrückt.

* **ping:**

Dieser Befehl pingt eine IP oder eine Internetseite an und zeigt die Dauer der Antwort in

Millisekunden an.

Beispiel:

ping 65.99.250.115

* **ping localhost –n Zahl >nul:**

Pausiert den Verlauf der Batch-Datei für **n** Pings (**1 Sekunde = 2 Pings** ; Allgemein:

**n+1 Ping = n Sekunden**). Pingt im Allgemeinen den eigenen Computer an, allerdings kann

auch statt „localhost“ eine IP geschrieben werden.

* **Programmpfade:**

%SystemRoot% = „Partition“:\Windows

%WinDir% = „Partition“:\Windows

%UserProfile% = „Partition“:\Dokumente und Einstellungen\„Benutzer“

„Partition“:\Users\User

%AllUsersProfile% = „Partition“:\Dokumente und Einstellungen\All Users

%AppData% = %UserProfile%\Anwendungsdaten

%Temp% = %UserProfile%\Lokale Einstellungen\Temp

%SystemRoot%\Temp

**Folgende Befehle öffnen Fenster:**

|  |  |
| --- | --- |
| Calc | Taschenrechner |
| Certmgr.msc | Zertifikate-Manager |
| Charmap | Zeichentabelle |
| Cleanmgr | Datenträgerbereinigungsassistent |
| Cmd | Cmd.exe |
| Compmgmt.msc | Computerverwaltung |
| Devmgmt.msc | Geräte-Manager |
| Dfrg.msc | Defragmentierungs-Manager |
| Diskmgmt.msc | Datenträgerverwaltung |
| Drwtsn32 | Doktor Watson, Windows Debugger |
| Edit | DOS Texteditor |
| Eudcedit | Zeichenerstellungsprogramm |
| Eventvwr | Ereignisprotokollierung |
| Start excel | Microsoft Office Excel |
| Explorer | Windows Explorer |
| Start firefox.exe | Firefox (Internetbrowser) |
| Freecell | Kartenspiel |
| Fsmgmt.msc | Ordnersicherheit / Freigabe von Ordnern |
| Gpedit.msc | Gruppenrichtlinien |
| Iexplore | Internet Explorer (Internetbrowser) |
| Lusrmgr.msc | Lokale Benutzer und Gruppen-Manager |
| Magnify | Bildschirmlupe |
| Start moviemk | Movie Maker |
| Mplay32 | Media Player |
| Start mplayer2 | Media Player 2 |
| Start msconfig | Microsoft Konfigurationsprogramm zu Autostart u. Tools |
| Mshearts | Kartenspiel |
| Start msinfo32 | Systeminformationen |
| Mspaint | Zeichenprogramm |
| Narrator | Sprachassistent |
| Notepad | Texteditor |
| Ntbackup | Sicherungskopien |
| Ntmsmgr.msc | Wechselmedien-Manager |
| Ntmsoprq.msc | Wechselmedien-Manager |
| Start OneNote | Microsoft Office OneNote |
| start Opera | Opera (Internetbrowser) |
| Osk | On-Screen-Keyboard, Bildschirmtastatur |
| Perfmon.msc | Systemmonitor über Leistung |
| Start pinball | Pinball |
| Start powerpnt | Microsoft Office PowerPoint |
| Regedit | Registry-Editor |
| Regedt32 | Registry-Editor |
| Rsop.msc | Gruppenrichtlinienauswertung im Internet |
| Secpol.msc | Lokale Sicherheitsrichtlinien |
| Services.msc | Dienste |
| Sndrec32 | Soundrecorder |
| Sndvol32 | Volumen, Ton |
| Sol | Kartenspiel |
| Spider | Kartenspiel |
| Taskmgr | Task-Manager |
| Tourstart | Windows-Tour |
| Utilman | Programm-Manager |
| Verifier | Treiber-Überprüfungsassistent |
| Winhelp | Windows Hilfe |
| Winmine | Spiel |
| Winver | Windows Version |
| Start winword | Microsoft Office Word |
| Start wmplayer | Windows Media Player |
| Start wordpad | Texteditor |
| Write | Texteditor |

* **Prompt:**

Dieser Befehl ändert die Anzeige bei "echo on" oder im aktiven CMD-Fenster. Das bedeutet, anstelle von "C:\Dokumente und Einstellungen\Benutzer>" bzw. "C:\Users\Benutzer" steht das da, was man hinter prompt schreibt:

[

C:\Dokumente und Einstellungen\Benutzer>prompt Hallo alle!

Hallo alle! echo Hi

Hi

Hallo alle! …

]

Der Prompt-Befehl erlaubt, Sonderzeichen anders zu schreiben, da beispielsweise "&" in CMD eine besondere Bedeutung hat. Möchte man dieses Zeichen trotzdem verwenden, kann man sich an die folgende Liste halten:

$A & (Und)

$B | (Verkettungszeichen)

$C ( (Klammer auf)

$D Aktuelles Datum

$E Escapezeichen (ASCII-Code 27)

$F ) (Klammer zu)

$G > (Größer-als-Zeichen)

$H Rückschritt (löscht vorangehendes Zeichen)

$L < (Kleiner-als-Zeichen)

$N Aktuelles Laufwerk

$P Aktuelles Laufwerk und Pfad

$Q = (Gleichheitszeichen)

$S (Leerzeichen)

$T Aktuelle Zeit

$V Windows 2000-Versionsnummer

$\_ Wagenrücklauf und Zeilenvorschub

$$ $ (Dollarzeichen)

* **pushd ; popd:**

PUSHD speichert das aktuelle Verzeichnis und wechselt dann zum angegebenen Verzeichnis.

POPD wechselt zu dem Verzeichnis, das durch PUSHD festgelegt wurde.

Beispiel:

echo %cd% Zeigt das **aktuelle Verzeichnis** an

PUSHD "C:\Programme\Mozilla Firefox" **Speichert dieses** und **wechselt zu Firefox**

start firefox.exe Startet Firefox aus "Mozilla Firefox"

popd **Wechselt zu dem Verzeichnis vorher**

# R

* **Rd/Rmdir:**

Löscht ein Verzeichnis.

**/S** = Löscht alle Dateien und Unterverzeichnisse mit.

**/Q** = Keine Nachfrage bei /S.

* **REG:**

Der Befehl „reg“ zeigt Werte der Registry an, ändert oder löscht sie. Viele Parameter haben folgende weitere Parameter:

Schlüssel = Registrypfad = Zu bearbeitender Pfad

/v Wert = „value“ = Zu bearbeitender Wert im Pfad

/ve = „empty value“ = Standardwert (Standard) wird bearbeitet im Pfad

/t Typ = „type“ = Typ des zu bearbeitenden Wertes

/d Daten = „data“ = Daten des zu bearbeitenden Wertes

/f = „force“ = Erzwingt, den Pfad/Wert zu überschreiben

Um den REG-Befehl richtig nutzen zu können, muss man einige Tricks in der Registry anwenden können. Einige Tricks sind am Ende der Erklärung des REG-Befehls zu lesen.

Es gibt folgende Typen (/t Typ) in der Registry:

**REG\_SZ** = Zeichenfolge

**REG\_DWORD** = Hexadezimale Folge

**REG\_BINARY** = Binäre Folge

REG\_MULTI\_SZ = ???

REG\_EXPAND\_SZ = ???

REG\_DWORD\_BIG\_ENDIAN = ???

REG\_DWORD\_LITTLE\_ENDIAN = ???

REG\_NONE = ???

Die ersten 3 Typen sind die meist verwendeten.

Fortlaufend nun die möglichen Hauptparameter des REG-Befehls:

- REG **ADD** Schlüssel [/v Wert | /ve] [/t Typ] [/d Daten] [/f]

Fügt der Registry einen Pfad oder einen Wert hinzu.

Beispiel:

Reg add "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\Virus.exe"

Reg add "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\Virus.exe" /v Debugger /t REG\_SZ /d calc

**->** Erstellt einen neuen Schlüssel “Virus.exe”, der bewirkt, dass beim Starten dieses Prozesses der Taschenrechner stattdessen geöffnet wird (calc).

- REG **COPY** Schlüssel1 Schlüssel2 [/s] [/f]

Kopiert alle Werte (und unter Verwendung von /s auch Unterschlüssel) von Schlüssel1 in Schlüssel2.

Beispiel:

Reg copy "HKCR\.bat" "HKCR\.command" /s

**->** Kopiert alle Schlüssel und Werte der Dateiendung .bat zu .command.

- REG **DELETE** Schlüssel [/v Wert | /ve | /va] [/f]

Löscht einen Schlüssel, einen Wert oder mehrere Werte.

/va = „all values“ = Bearbeitet alle Werte im Pfad

Beispiel:

Reg delete "HKCR\.command" /F

**->** Löscht die Dateiendung .command, wenn vorhanden.

- REG **COMPARE** Schlüssel1 Schlüssel2 [/v Wert | /ve] [/oa | /od | /os | /on] [/s]

Vergleicht zwei Schlüssel oder Werte.

/oa = Gibt Unterschiede und Gleichheiten an [a = all]

/od = Gibt nur Unterschiede an (standardmäßig verwendet) [d = differences]

/os = Gibt nur Gleichheiten an [s = same]

/on = Gibt keins von beidem an, schaut nur, ob sie gleich sind [n = none]

Beispiel:

Reg compare "HKCR\.bat" "HKCR\.cmd"

- REG **EXPORT** Schlüssel Datei

Exportiert einen kompletten Schlüssel in eine .REG-Datei, die bei Doppelklick den Schlüssel komplett wiederherstellt. Wird verwendet für Backups oder manuelle Systemwiederherstellungen.

Beispiel:

Reg export "HKCR\batfile" "%userprofile%\Desktop\batfile.reg"

**->** Macht ein Backup des Schlüssels "batfile".

- REG **IMPORT** Datei

Importiert eine .REG-Datei, die einen Schlüssel wiederherstellen kann. Die Datei muss vorher durch REG EXPORT gespeichert worden sein.

Beispiel:

Reg import "%userprofile%\Desktop\batfile.reg"

**->** Stellt die Schlüssel und Werte aus "batfile.reg" wiederher.

- REG **QUERY** Schlüssel [/v Wert | /ve] [/s]:

Durchsucht die Registry nach einem angegebenen Pfad oder Wert und gibt diesen

aus.

Beispiel:

Reg query "HKLM\Software\Microsoft\Windows\CurrentVersion\Run" /S

**->** Zeigt den Autostartordner der Registry aller Benutzer an.

- REG **RESTORE** Schlüssel Datei

Stellt einen Schlüssel aus der angegebenen Datei wieder her. Die angegebene Datei muss vorher durch REG SAVE gespeichert worden sein.

Beispiel:

Reg restore "HKCR\cmdfile" "%userprofile%\Desktop\cmdfile.reg"

**->** Stellt den Schlüssel "HKCR\cmdfile" anhand der Datei "cmdfile.reg" wiederher.

- REG **SAVE** Schlüssel Datei

Speichert den angegeben Schlüssel in die angegebene Datei, um eine Wiederherstellung durch REG RESTORE zu ermöglichen. Die Datei ist in einem nicht-lesbaren Format gespeichert. Somit kann Datenschutz besser gewährt werden als durch REG EXPORT.

Beispiel:

Reg save "HKCR\cmdfile" "%userprofile%\Desktop\cmdfile.reg"

**->** Macht ein Backup des Schlüssel "HKCRcmdfile" in die Datei "cmdfile.reg".

Tipps und Tricks, die man in der Registry anwenden kann:

1. Gruppenrichtlinien über Batch bearbeiten:

"HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced"

/v Hidden /t REG\_DWORD /d 1 **=** Versteckte Dateien anzeigen

/v HideFileExt /t REG\_DWORD /d 0 **=** Dateiendungen anzeigen

/v HideIcons /t REG\_DWORD /d 0 **=** Icons anzeigen

/v ShowSuperHidden /t REG\_DWORD /d 1 **=** Systemdateien anzeigen

/v DisableThumbnailCache /t REG\_DWORD /d 1 **=** Thumbs.db blocken

/v SuperHidden /t REG\_DWORD /d 1

**=** Systemdateien besser verstecken als normal versteckte Dateien

/v StartMenuRun /t REG\_DWORD /d 0

**=** "Ausführen" vom Startmenü entfernen

/v Start\_ShowRecentDocs /t REG\_DWORD /d 0

**=** "Zuletzt verwendete Dokumente" aus dem Startmenü entfernen

"HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer"

/v SearchHidden /t REG\_DWORD /d 1

**=** Versteckte Dateien bei Suche mit einbeziehen

/v SearchSystemDirs /t REG\_DWORD /d 1

**=** Systemdateien bei Suche mit einbeziehen

"HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer"

/v NoSimpleStartMenu /t REG\_DWORD /d 1 **=** Einfaches Startmenü erzwingen

/v NoRun /t REG\_DWORD /d 1 **=** "Ausführen" verbieten

/v NoChangeStartMenu /t REG\_DWORD /d 1

**=** Ändern der Einstellungen des Startmenüs verbieten

/v NoChangeTaskBar /t REG\_DWORD /d 1

**=** Ändern der Einstellungen der Taskleiste verbieten

/v NoClose /t REG\_DWORD /d 1

**=** "Herunterfahren" vom Startmenü entfernen

/v NoStartMenuMorePrograms /t REG\_DWORD /d 1

**=** "Alle Programme" vom Startmenü entfernen

/v NoStartMenuMFUprogramsList /t REG\_DWORD /d 1

**=** Liste der häufig verwendeten Programme vom Startmenü entfernen

/v NoFind /t REG\_DWORD /d 1

**=** "Suchen" aus dem Startmenü entfernen

/v StartMenuLogOff /t REG\_DWORD /d 1

**=** "Abmelden" aus dem Startmenü entfernen

/v LockTaskbar /t REG\_DWORD /d 1

**=** Taskleiste fixieren erzwingen

/v HideClock /t REG\_DWORD /d 1

**=** Windows-Uhr aus der Taskleiste entfernen

/v NoTrayContextMenu /t REG\_DWORD /d 1

**=** Kontextmenü der Taskleiste verbieten (Rechtsklick)

/v NoFolderOptions /t REG\_DWORD /d 1

**=** Ändern der Ordneroptionen über die Systemsteuerungen verbieten

/v NoFileMenu /t REG\_DWORD /d 1

**=** "Datei"-Kontextmenü aus dem Explorer entfernen (oben links)

/v NoDesktop /t REG\_DWORD /d 1

**=** Desktopsymbole ausblenden (auffindbar im Ordner)

/v NoPropertiesMyComputer /t REG\_DWORD /d 1

**=** Eigenschaften des Arbeitsplatzes ausblenden

/v DisableRegistryTools /t REG\_DWORD /d 1

**=** Bearbeiten der Registry durch regedit u.Ä. verhindern

/v ForceClassicControlPanel /t REG\_DWORD /d 1

**=** Klassische Systemsteuerungen erzwingen

"HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\System"

/v DisableTaskMgr /t REG\_DWORD /d 1

**=** Windows Task-Manager verbieten

/v NoDispSettingsPage /t REG\_DWORD /d 1

**=** Ändern der Bildschirmeinstellungen verbieten

/v NoDispScrSavPage /t REG\_DWORD /d 1

**=** Ändern der Bildschirmschonereinstellungen verbieten

/v NoDispAppearancePage /t REG\_DWORD /d 1

**=** Ändern der Farben verhindern

/v NoDispBackgroundPage /t REG\_DWORD /d 1

**=** Ändern des Bildschirmhintergrundes verbieten

/v NoDispCPL /t REG\_DWORD /d 1

**=** Systemsteuerungen verstecken

"HKLM\Software\Microsoft\Windows\CurrentVersion\policies\system"

/v HideStartupScripts /t REG\_DWORD /d 0

**=** Startskripts von Windows anzeigen lassen

/v HideShutdownScripts /t REG\_DWORD /d 0

**=** Runterfahrskripts von Windows anzeigen lassen

1. Anderes:

"HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options" ist ein wichtiger Registryschlüssel, der oft von Viren befallen wird.

Man kann in ihm **verhindern, dass ein bestimmter Prozess gestartet wird** oder selbigen auf etwas anderes umlenken. Das geht wie folgt:

1. Man fügt einen **neuen Schlüssel** obrigem Pfad hinzu. Der Schlüssel trägt den **Namen des zu stoppenden Prozesses** (z.B. setup\_wm.exe).
2. Man fügt diesem Schlüssel einen **neuen Wert** vom Typ Zeichenfolge hinzu (REG\_SZ). Dieser Wert heißt "**Debugger**".
3. Man fügt dem Wert "Debugger" den Prozessnamen hinzu, der dabei starten soll. **Gibt man nichts an, passiert beim Starten des Prozesses garnichts.**

Autostartordner:

* "HKCU\Software\Microsoft\Windows\CurrentVersion\Run"
* "HKCU\Software\Microsoft\Windows\CurrentVersion\RunOnce"
* "HKLM\Software\Microsoft\Windows\CurrentVersion\Run"
* "HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnce"
* **REM:**

Dieser Befehl wird verwendet, um demjenigen, der den **Quellcode liest** oder einem selbst

**Informationen zukommen zu lassen** z.B. über die Syntax der Befehle (für Neulinge). Dieser

Befehl wird **beim Ausführen der Batchdatei nicht beachtet**.

REM Es wird nicht in dem Verlauf der Batch-Datei angezeigt, aber es ist vorhanden.

* **Ren:**

Benennt eine Datei um: REN "Dateipfad1" "Dateipfad2"

* **Runas:**

**Ersetzt „Ausführen als…“.** Mit diesem Befehl kann man bestimmen, **unter welchem Benutzernamen** ein **Programm ausgeführt** wird. Syntax:

RUNAS /user:*Benutzername* "*Programmname oder Pfad zu einer Datei*"

„*Benutzername*“ muss durch den gewollten Benutzer ersetzt werden.

„*Programmname oder Pfad zu einer Datei*“ muss durch das gewollte Programm ersetzt werden. Hier muss ein **kompletter Pfad** angegeben werden, wenn sich das Programm nicht direkt in system32/system64 befindet! **Anderenfalls** könnte der Befehl **Fehler** verursachen!

**Wenn** der Befehl **ausgeführt** worden ist, **fragt das Programm nach** einem **Kennwort**. Dieses Kennwort wird in CMD **nicht angezeigt** und kann somit weder gelesen, noch in eine Datei kopiert werden und ist **somit geschützt**.

Sind Kennwort oder Benutzername falsch, kommt eine Fehlermeldung. Diese **Fehlermeldung** kann **nicht mit** "***2>nul***" **unterdrückt** werden, sondern nur mit "***>nul***". Wird jedoch "***>nul***" verwendet, wird auch kein Dialog für das Kennwort angezeigt. Um dies zu umgehen, kann man "***echo*** Bitte geben Sie ihr Passwort ein: " verwenden.

# S

* **sc (~net):**

Dieser Befehl, ähnlich wie der ***NET***-Befehl, beschränkt sich auf die Arbeit mit **Dienste**n.

Er arbeitet mit den **wahren Dienstnamen** im Gegensatz zu "***NET***". Allgemeine Syntax:

**sc [Befehl] [Dienstname] [Parameter]**

**query** Zeigt den **Status** eines Dienstes an.

**queryex** Zeigt den **Status** eines Dienstes ausführlich an.

**start** Startet einen Dienst.

**pause** Pausiert einen Dienst.

**continue** Fährt einen pausierten Dienst fort.

**stop** Stoppt einen Dienst.

**config** Konfiguriert einen Dienst:

**start**= <boot|system|auto|demand|disabled>

--> Ändert die **Startkonfiguration**.

--> boot = VOR Windows XP.

--> system = Während Windows XP-Ladebildschirm.

--> auto = Startet mit der Anmeldung.

--> demand = Muss manuell gestartet werden.

--> disabled = Kann nicht gestartet werden!

**error**= <normal|severe|critical|ignore>

--> Ändert die **Konfiguration bei Fehlern**.

--> normal = Fehler wird protokolliert, Benutzer wird gewarnt.

--> severe = **Computer startet neu** mit der zuletzt funktionierenden,

bekannten Konfiguration.

--> critical = **Computer startet neu** mit der zuletzt funktionierenden,

bekannten Konfiguration. Klappt das nicht, erscheint ein

**BSOD**.

--> ignore = Fehler wird ignoriert.

**depend**= Dienstname(n)

--> Legt fest, welche Dienste vor besagtem Dienst gestartet werden müssen.

**Dienstnamen werden mit "/" getrennt.**

**DisplayName**= "Display-Name"

--> Konfiguriert den **Anzeigenamen des Dienstes**.

**--> Das Leerzeichen nach dem "=" muss stehen bleiben!**

**description** **Ändert** die Beschreibung eines Dienstes.

**failure** Ändert den Befehl bei einem Absturz des Dienstes.

--> **reset** = Länge der Dauer ohne Fehler bis zum Absturz in Sekunden

--> **reboot** = Anzuzeigende Nachricht vor dem Neustart

--> **command** = Befehlszeile, die beim Absturz des Dienstes ausgeführt

werden soll

**qc** Zeigt **Konfigurationsinformationen** zu einem Dienst an.

**qdescription** **Zeigt** die Beschreibung eines Dienstes **an**.

**qfailure** Zeigt die Aktion, die bei einem Fehler ausgeführt wird, an.

**delete** Löscht einen Dienst aus der Registry.

**create** Erstellt einen Dienst (auch in der Registry).

**GetDisplayName** Zeigt den **Anzeigenamen eines Dienstes** an.

**GetKeyName** Zeigt den **wahren Namen** eines Dienstes an (**arbeitet mit Anzeigenamen**).

**EnumDepend** Listet die Dienste auf, die ohne den angegebenen Dienst nicht gestartet

werden können.

Diese Befehle brauchen keinen Dienstnamen:

**boot** (ok|bad) Legt fest, ob die letzte Konfiguration beim Booten gut oder schlecht war.

Lock Sperrt die Dienst-Datenbank

QueryLock Zeigt den "Lock-Status" an

Beispiele:

Sc config Schedule start= demand

Sc start Schedule

AT 20:00 ***msg*** \* hi

***REM*** Schedule = Taskplaner

* Stellt den Startmodus auf manuell, startet den Dienst und führt einen Befehl aus.

----

Sc stop Themes

Sc config Themes start= disabled

***REM*** Themes = Designs

* Stoppt den Dienst und stellt den Startmodus auf deaktiviert, sodass er nicht neu gestartet werden kann. Damit wird das Windows 98-Design erzwungen.
* **set:**

Dieser Befehl dient dazu, ***Variabeln*** festzulegen. Er erlaubt mathematische Rechnungen oder

Eingaben des Benutzers:

**/a** = Variable arbeitet mit Zahlen

**/p** = Wartet auf eine **Eingabe des Benutzer** (User Prompt)

Beispiele:

set fire=C:\Programme\Mozilla Firefox\firefox.exe

set /a Rechnung=5 + 5

set /p Passwort=Bitte geben Sie hier das Passwort ein:

set /p Zahl=Geben Sie eine dieser Zahlen ein: 1,2,3

set /a Zahl=%Zahl%

---> Gibt der Benutzer einen Buchstaben ein, **wird aus "Zahl"** durch den "SET /A"-Befehl **0.**

"set" kann auch verwendet werden, um **aus einer** bereits festgelegten **Variable einen** **bestimmten Teil** zu **ersetzen**, wenn vorhanden. Das bedeutet, dass aus einem Text ein Buchstabe, eine Zahl oder ein Wort entfernt wird und durch etwas anderes ersetzt wird. Die Syntax hierzu lautet:

Set Variable2=%Variable1:TextZuErsetzen=TextEinzufügen%

Es ist **möglich, Variable1 direkt zu überschreiben** anstatt eine zweite festzulegen.

Beispiel:

Set steine=Dies ist ein Beispiel, um Teile eines Textes zu ersetzen.

Echo %steine%

Set steine=%steine:Teile=Dinge%

Echo %steine%

* Zuerst wird "Dies ist ein Beispiel, um Teile eines Textes zu ersetzen." angezeigt,

dann aber "Dies ist ein Beispiel, um Dinge eines Textes zu ersetzen.".

Das Wort "Teile" wurde durch "Dinge" ersetzt.

* **setlocal:**

"setlocal" alleine stehend erstellt einen **Zwischenraum innerhalb einer Batchdatei**, in dem

***Variabeln***, die vor dem Befehl festgelegt wurden, nicht beachtet werden. Das lässt sich so

vorstellen:

[ BATCH-DATEI ] -> Der Raum der Batchdatei wird begrenzt durch [ ].

[BATCH- **(Setlocal)** DATEI] -> Setlocal erstellt einen Zwischenraum **( )** in [ ].

Der Befehl "setlocal" wird beendet durch "***endlocal***".

Erweiterungen:

setlocal enabledelayedexpansion / disabledelayedexpansion:

"enable" erlaubt, dass Variabeln auch mit **!var!** aufgerufen werden können. Anders als mit

***%var%*** können diese innerhalb von Klammern ( (Befehl1 Befehl2 Befehl3...) ) verwendet werden.

Beispiel:

setlocal enabledelayedexpansion

set Nummer=0 REM Die Variable "Nummer" wird auf 0 gesetzt.

if "%Nummer%" == "0" ( REM Durch "IF" wird eine Klammer geöffnet.

set Nummer=%Nummer% +1 REM "Nummer" wird +1 gesetzt.

echo !Nummer! ist 1. REM Die Nummer INNERHALB der Klammer wird gegeben.

echo %Nummer% ist 0. REM Die Nummer AUßERHALB der Klammer wird

) REM gegeben, weil CMD nach der Variable allgemein

REM sucht.

setlocal enableextensions / disableextensions:

Dieser Befehl aktiviert oder deaktiviert die Befehlserweiterungen von CMD. Damit kann man

bestimmen, ob ungewollte Befehle verwendet werden oder man viel Spielraum hat. Standardmäßig sind Befehlserweiterungen aktiviert.

Wird der "setlocal"-Befehl außerhalb einer Batchdatei verwendet, hat er keine Wirkung.

* **shift:**

Dieser Befehl **ändert die Position der** ***Argumente*** einer Datei (%1, %2... %9). Damit lassen

sich **mehr als 10 Argumente** für eine Datei verwenden. Möchte man die ersten Argumente

definitiv beibehalten, kann man **/N** eingeben, wobei **N die Anzahl der Argumente** ist, die

übersprungen werden.

shift **/2** = %0 und %1 werden übersprungen, %3 geht auf %2, %8 auf %7 usw.

* **shutdown:**

Der shutdown-Befehl wird zum **Herunterfahren**, Neustarten und Abmelden von Windows verwendet:

**/a** = Shutdown **abbrechen**

**/c** = **Kommentar** hinzufügen (gefolgt von "Kommentar" mit " ")

**/f** = Benutzer abmelden und Windows herunterfahren, stoppt bei der Benutzerauswahl

**/l** = Benutzer **abmelden**

**/r** = **Neustart** [Restart]

**/s** = **Herunterfahren** [Shutdown]

**/t** = **Dauer** in Sekunden **bis Shutdown** [Time] (gefolgt von einer Zahl)

Beispiel:

Shutdown –s –t 120 –c "Windows fährt in 120 Sekunden runter. "

* **sndrec32:**

Öffnet den Windows Soundrecorder. Er kann dazu benutzt werden, .WAV-Dateien automatisch abzuspielen:

Start sndrec32 "Pfad zu der Datei.WAV" /play

Möchte man, dass er sich danach wieder beendet, fügt man /close hinzu:

Start sndrec32 "Pfad zu der Datei.WAV" /play /close

* **Sonderzeichen (& / || / ² / ...):**

Sonderzeichen sind oft ein Problem in der Wiedergabe. Hier ein paar spezielle:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Erklärung: \*

\* -Eine Datei ins Autostartmenü bei deutschem Windows erstellen: \*

\* \*

\* Die Datei, die eine weitere Datei erstellen soll, muss mit dem aktiven CMD (Start- \*

\* Ausführen-cmd) geschrieben werden, sodass das Ü von Startmenü erkannt wird. In dem \*

\* Quelltext wird das Ü als unerkanntes Viereck dargestellt, was nicht verändert \*

\* werden darf. \*

\* \*

\* Der Quellcode mitsamt Viereck kann anschließend kopiert und an einer anderen \*

\* Stelle eingefügt werden. \*

\* Hier sind einige Zeichen, die von CMD nicht erkannt werden können: \*

\* \*

\* Das hier ist ein ü (klein)...:  \*

\* Das hier ist ein Ü (groß)....: š \*

\* Das hier ist ein ß...........: á \*

\* Das hier ist ein ä (klein)...: „ \*

\* Das hier ist ein Ä (groß)....: Ž \*

\* Das hier ist ein ö (klein)...: ” \*

\* Das hier ist ein Ö (groß)....: ™ \*

\* Das hier ist ein Alarmzeichen: \*

\* Das hier ist ein volles Leerzeichen: Û \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Dies sind jedoch nicht die einzigen, die umgangen werden müssen:

& = Befehlstrennung. Trennt Befehl1 und Befehl2.

&& = Befehlstrennung. Bedingt, dass Befehl2 nur ausgeführt wird, wenn Befehl1 geglückt ist.

| = Befehlstrennung. Trennt Befehl1 und Befehl2 wie &, kann aber auch anders verwendet

werden. Mehr dazu siehe unten.

|| = Befehlstrennung. Bedingt, dass Befehl2 nur ausgeführt wird, wenn Befehl1 misslungen

ist.

% = Wird verwendet für ***Variabeln***.

( = Eröffnet eine Unterbefehlskette.

) = Schließt eine Unterbefehlskette.

Wie man das umgehen kann:

Bei nahezu allen Sonderzeichen kann man einfach ein ^ davorsetzen. Damit kann man es in echo z.B. anzeigen lassen. %-Zeichen müssen mit einem Doppelten % ersetzt werden: %%.

Außerdem wird ² als voll ausgefülltes Leerzeichen dargestellt, allerdings schwächer als Û.

Verwendung von |:

Schreibt man die allgemeine Syntax

"**echo Eingabe|Befehl**"

wird in die Nachfrage von "Befehl" die Eingabe "Eingabe" eingegeben. Dabei ist entscheidend, **ob** zwischen "Eingabe" und | **ein Leerzeichen** ist. Hier ein Beispiel:

**echo J|**cacls "C:\Programme\Mozilla Firefox" /p %username%:N

--> Da bei dem Befehl "cacls ..." gefragt wird "**Sind Sie sicher? (J / N)**" und da direkt **"J"**

**durch den echo-Befehl eingefügt** wird, kann man so das Zugriffsrecht einfach ändern.

Dasselbe gilt für andere Befehle z.B. "**help|sort**". In diesem Befehl wird die Ausgabe von dem Befehl „help“ nach dem Alphabet sortiert.

**Für mehr Sonderzeichen siehe das Thema *Ascii***.

* **sort:**

Mit diesem Befehl kann man Dateien in lesbarem Textformat sortieren lassen.

* **Springpunkte**

Springpunkte sind Punkte innerhalb einer Batch-Datei, die über den ***goto***-Befehl oder den ***call***-Befehl aufgerufen werden können. Sie bestehen aus einem **Doppelpunkt und** **einem** **Wort** und/oder einer Zahl. Auf Springpunkte können ***Argumente*** verwendet werden.

Hier ein Beispiel für einen Springpunkt:

:Name\_Des\_Springpunktes

Springpunkte haben **niemals Leerzeichen**. Ansonsten ist nur das erste Wort der Sprinpunktname.

* **start:**

Der start-Befehl ist ein wichtiger Befehl, um Programme zu starten, **ohne den Batchverlauf**

**zu unterbrechen**. Die zu startende Datei **MUSS OHNE " " stehen**, da sich ansonsten ein CMD-Fenster mit dem Titel " " öffnet.

Um dies zu umgehen, kann man den Befehl "***pushd***" oder "***cd/chdir***" verwenden. Ist selbst in dem Dateinamen ein Leerzeichen enthalten, muss man den Kurznamen der Datei verwenden (siehe "***Kurznamen***").

Außerdem kann man die Option **/D** (= **/D**irectory) nach dem Befehl hinzufügen, um den **Pfad**, in dem die zu startende Datei liegt, **angeben** zu können.

Der Befehl erlaubt, die Priorität des zu öffnenden Programmes zu bestimmen:

/LOW

/BELOWNORMAL

/NORMAL

/ABOVENORMAL

/HIGH

/REALTIME

Aber auch die Anzeige:

/MIN

/MAX

Anders als der ***CALL***-Befehl wird eine zu öffnende Batchdatei nicht in die momentane geladen, sondern **startet in einem neuen CMD-Fenster**, sodass das momentane erhalten bleibt.

Beispiele:

start /MIN ***sndrec32*** /play /close "%SystemRoot%\Media\ding.wav"

start /LOW ***iexplore*** <http://www.google.com>

start **/D** "%userprofile%\Desktop" *Dateiname*

* **systeminfo:**

Zeigt viele Systeminformationen an.

# T

* **taskkill:**

Dieser Befehl ist der erweiterte Befehl von "***tskill***" und funktioniert anstatt unter XP Home

auch unter Vista. Er hat viele wichtige Erweiterungen:

**/F** = **Erzwingt das Beenden** des Prozesses.

**/IM** = **Name** des Prozesses mit Endung.

**/PID** = **Prozess-ID** des Prozesses.

**/T** = Beenden der gesamten **Prozessstruktur** (=auch untergeordnete Prozesse).

Es kann mehrmals in einem Befehl /IM oder /PID angegeben werden. Beispiele:

taskkill /IM explorer.exe

taskkill /F /IM notepad.exe /PID 3056 **REM notepad.exe ist nicht unbedingt PID 3056!**

* **tasklist:**

Dieser Befehl listet sämtliche **laufende Prozesse** auf mit folgenden Informationen:

Prozessname, PID, Sitzungsname (=Console), Sitzungsnummer (=0), Speichernutzung (Memory).

**/FI "Filter"** = Zu verwendender Filter.

**/FO Format** = Gültige Formate sind "table", "list" und "csv".

**/M Modul** = Zeigt alle Tasks mit geladenem Modul "Modul" an (DLL-Dateien).

**/NH** = Lässt die **Überschrift** bei den Formaten table und csv **weg**.

**/V** = Zeigt **ausführliche Infos** an:

Status, Benutzername, Laufzeit bisher (klappt nicht!), Fenstertitel

**Filter:**

Filtername Gültige Operatoren Gültige Werte

--------------- ------------------------- ------------------

STATUS eq, ne RUNNING | NOT RESPONDING

IMAGENAME eq, ne Abbildname

PID eq, ne, gt, lt, ge, le PID-Wert

SESSION eq, ne, gt, lt, ge, le Sitzungsnummer

SESSIONNAME eq, ne Sitzungsname

CPUTIME eq, ne, gt, lt, ge, le CPU-Zeit im Format:

hh:mm:ss.

hh - Stunden,

mm - Minuten, ss - Sekunden

MEMUSAGE eq, ne, gt, lt, ge, le Speichernutzung in KB

USERNAME eq, ne Benutzername im Format:

Benutzer

SERVICES eq, ne Dienstname

WINDOWTITLE eq, ne Fenstertitel

MODULES eq, ne DLL-Name

eq = **Eq**ual = gleich

ne = **N**ot **E**qual = nicht gleich

gt = **G**reater **T**han = größer als

ge = **G**reater and **E**qual = größer gleich

le = **L**ess and **E**qual = kleiner gleich

lt = **L**ess **T**han = kleiner als

/FI "STATUS eq NOT RESPONDING"

* **time:**

Gibt alleine angegeben die **aktuelle Zeit** an und verlangt nach einer Änderung.

**/T** gibt nur die Zeit an.

"time *Zeit*" ändert die Zeit zu "Zeit", z.B. time 20:00 .

* **title:**

Setzt den Titel des CMD-Fenster fest: title "Titel"

* **tree:**

Zeigt den **Verzeichnisbaum** des aktuellen Ordners oder des angegebenen Ordners an.

**/A** = Verwendet ***Ascii-Zeichen*** statt erweiterten Zeichen.

**/F** = Zeigt auch die **Dateinamen** jedes Ordners an.

"Pfad" gibt den Ordner an, dessen Baum angezeigt werden soll.

Beispiel:

tree /F "C:\"

* **tskill:**

Dies ist ein recht alter Befehl, der ab XP SP-3 zu "***taskkill***" mit erweiterten Funktionen

geändert wurde. Hier muss der **Prozessname ohne Endung** angegeben werden, z.B.

tskill notepad

tskill 3056

* **type:**

Zeigt den **Inhalt von** textlich geschriebenen **Dateien**: ' type "Dateipfad" '

# U

* **user&user32:**

Diese Befehle sind **DLL**-Dateien, die per "**rundll32.exe**" verwendet werden können. Sie erlauben **System- und Geräteeinstellungen** von z.B. Maus oder Tastatur zu **ändern**:

rundll32 user32.dll,SwapMouseButton

rundll32 user.dll mouse, enable

rundll32 user.dll keyboard, enable

# V

* **Variabeln:**

Variabeln sind Werte, die vorher durch den ***SET***-Befehl festgelegt wurden. Normalerweise werden sie aufgerufen, indem man die Variable umschlossen von **% %** schreibt.

Unter der Verwendung von "***SETLOCAL*** ENABLEDELAYEDEXPANSION" lassen sich Variabeln auch **innerhalb von Klammern** durch **!Variable!** wiedergeben (siehe ***setlocal***).

set Variable1=echo Das ist einer

set Stein=der wichtigsten Teile von CMD.

%Variable1% %Stein%

Ausgegeben werden würde "Das ist einer der wichtigsten Teile von CMD." über den echo-Befehl.

Variabeln sind etwas einfacher als ***Argumente***, jedoch nach einem ähnlichen Prinzip.

Folgende Variabeln sind meist bereits festgelegt:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| %ALLUSERSPROFILE% | | | C:\Dokumente und Einstellungen\All Users | |
| %APPDATA% | | C:\Dokumente und Einstellungen\!Benutzer!\Anwendungsdaten | | |
| %CD% | | | !Momentaner Pfad / Current Directory! | |
| %CmdCmdLine% | | | "C:\WINDOWS\system32\cmd.exe" | |
| %CommonProgramFiles% | | | C:\Programme\Gemeinsame Dateien | |
| %COMPUTERNAME% | | | !COMPUTERNAME! | |
| %ComSpec% | | | C:\WINDOWS\system32\cmd.exe | |
| %date% | | | !MOMENTANES DATUM! | |
| %errorlevel% | | | !Errorlevel des vorigen Befehls! | |
| %FP\_NO\_HOST\_CHECK% | | | NO | |
| %HOMEDRIVE% | | | C: | |
| %HOMEPATH% | | | \Dokumente und Einstellungen\!BENUTZER! | |
| %LOGONSERVER% | | | \\!COMPUTERNAME! | |
| %NUMBER\_OF\_PROCESSORS% | | | !ANZAHL PROZESSOREN! | |
| %OS% | | | Windows\_NT | |
| %Path% | C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem | | | |
| %PATHEXT% | | | .COM;.EXE;.BAT;.CMD | |
| %PROCESSOR\_ARCHITECTURE% | | | !PROZESSOR-ARCHITEKTUR! | |
| %PROCESSOR\_IDENTIFIER% | | | !PROZESSOR-INFOS! | |
| %PROCESSOR\_LEVEL% | | | !PROZESSOR-LEVEL! | |
| %PROCESSOR\_REVISION% | | | !PROZESSOR-???! | |
| %ProgramFiles% | | | C:\Programme | |
| %PROMPT% | | | $P$G | |
| %random% | | | Zufällige Zahl zwischen 1 und 65536 | |
| %SESSIONNAME% | | | Console | |
| %SystemDrive% | | | C: | |
| %SystemRoot% | | | C:\WINDOWS | |
| %TEMP% | | | | C:\DOKUME~1\!BENUTZERNAME!\LOKALE~1\Temp |
| %time% | | | | !MOMENTANE ZEIT! |
| %TMP% | | | | C:\DOKUME~1\!BENUTZERNAME!\LOKALE~1\Temp |
| %tvdumpflags% | | | | ? |
| %USERDOMAIN% | | | | !COMPUTERNAME! |
| %USERNAME% | | | | !BENUTZERNAME! |
| %USERPROFILE% | | | | C:\Dokumente und Einstellungen\!BENUTZERNAME! |
| %windir% | | | | C:\WINDOWS |

* **ver:**

Zeigt die momentane Windows-Version an.

# W

* **wmic:**

Dieser komplizierte Befehl muss oft **extra installiert** werden, um verwendet werden zu können. Dafür jedoch kann man mit ihm sehr viel machen. Er erlaubt, folgendes zu verwalten:

ALIAS - Zugriff auf die auf dem lokalen System verfügbaren Aliase

BASEBOARD - Hauptplatinenverwaltung (auch Motherboard oder Systemkarte

genannt).

BIOS - BIOS-Verwaltung.

BOOTCONFIG - Startkonfigurationsverwaltung.

CDROM - CD-Verwaltung.

COMPUTERSYSTEM - Computersystemverwaltung.

CPU - CPU-Verwaltung.

CSPRODUCT - Computersystem-Produktinformationen von SMBIOS.

DATAFILE - Datendatei-Verwaltung.

DCOMAPP - DCOM-Anwendungsverwaltung.

DESKTOP - Benutzerdesktopverwaltung.

DESKTOPMONITOR - Desktopbildschirmverwaltung.

DEVICEMEMORYADDRESS - Gerätespeicheradress-Verwaltung.

DISKDRIVE - Verwaltung des physikalischen Laufwerks.

DISKQUOTA - Datenträgerspeicherplatz-Nutzung für NTFS-Volumes.

DMACHANNEL - DMA-Kanalverwaltung.

ENVIRONMENT - Systemumgebungs-Einstellungsverwaltung.

FSDIR - Dateisystem-Verzeichniseintragsverwaltung.

GROUP - Gruppenkontenverwaltung.

IDECONTROLLER - IDE-Controllerverwaltung.

IRQ - IRQ-Verwaltung.

JOB - Bietet Zugriff auf mit dem Zeitplandienst geplante Aufträge.

LOADORDER - Verwaltung von Systemdiensten, die

Ausführungsabhängigkeiten definieren.

LOGICALDISK - Verwaltung des lokalen Speichergeräts.

LOGON - Anmeldesitzungen.

MEMCACHE - Cacheverwaltung.

MEMLOGICAL - Systemspeicherverwaltung (Konfigurationslayout und

Speicherverfügbarkeit).

MEMPHYSICAL - Verwaltung des physikalischen Speichers des Computers.

NETCLIENT - Netzwerkclientverwaltung.

NETLOGIN - Verwaltung der Netzwerkanmeldeinformationen (eines

bestimmten Benutzers).

NETPROTOCOL - Protokollverwaltung (und die Verwaltung der

Protokollnetzwerkeigenschaften).

NETUSE - Verwaltung der aktiven Netzwerkverbindung.

NIC - NIC-Verwaltung.

NICCONFIG - Netzwerkadapterverwaltung.

NTDOMAIN - NT-Domänenverwaltung.

NTEVENT - Einträge im NT-Ereignisprotokoll.

NTEVENTLOG - NT-Ereignisprotokoll-Dateiverwaltung.

ONBOARDDEVICE - Verwaltung der in der Hauptplatine eingebauten

Standardadapter (Systemkarte).

OS - Verwaltung der installierten Betriebssysteme.

PAGEFILE - Verwaltung des virtuellen Speichers für den Dateiaustausch.

PAGEFILESET - Auslagerungsdatei-Einstellungsverwaltung.

PARTITION - Verwaltung von partitionierten Bereichen des physikalischen

Datenträgers.

PORT - E/A-Anschlussverwaltung.

PORTCONNECTOR - Verwaltung von physikalischen Verbindungsanschlüssen.

PRINTER - Druckerverwaltung.

PRINTERCONFIG - Druckerkonfigurationsverwaltung.

PRINTJOB - Druckauftragsverwaltung.

PROCESS - Prozessverwaltung.

PRODUCT - Installationspaket-Aufgabenverwaltung.

QFE - QFE (Quick Fix Engineering).

QUOTASETTING - Setzt Informationen für Datenträgerkontingente auf einem

Volume.

RECOVEROS - Informationen, die aus dem Speicher beim Fehlschlagen des

Betriebssystems zusammengestellt werden.

REGISTRY - Computersystemregistrierungsverwaltung.

SCSICONTROLLER - SCSI-Controllerverwaltung.

SERVER - Serverinformationsverwaltung.

SERVICE - Dienstanwendungsverwaltung.

SHARE - Verwaltung von gemeinsam genutzten Ressourcen.

SOFTWAREELEMENT - Verwaltung der auf dem System installierten

Softwareproduktelemente.

SOFTWAREFEATURE - Verwaltung von Softwareprodukten, die untergeordnete

Sätze von "SoftwareElement".

SOUNDDEV - Audiogeräteverwaltung.

STARTUP - Verwaltung von Befehlen, die automatisch bei der

Benutzeranmeldung am System ausgeführt werden.

SYSACCOUNT - Systemkontenverwaltung.

SYSDRIVER - Verwaltung von einem Systemtreiber für einen Basisdienst.

SYSTEMENCLOSURE - Systemgehäuseverwaltung.

SYSTEMSLOT - Verwaltung von physikalischen Verbindungspunkten,

einschließlich Anschlüssen, Steckplätzen und

Peripheriegeräten, und proprietären Verbindungspunkten.

TAPEDRIVE - Bandlaufwerkverwaltung.

TEMPERATURE - Datenverwaltung eines Temperatursensors (elektronisches

Thermometer).

TIMEZONE - Zeitzonen-Datenverwaltung.

UPS - Verwaltung der unterbrechungsfreien Stromversorgung

(USV).

USERACCOUNT - Benutzerkontenverwaltung.

VOLTAGE - Spannungssensor-Datenverwaltung.

VOLUMEQUOTASETTING - Assoziiert die Datenträgerkontingentseinstellung auf ein

bestimmtes Volume.

WMISET - Verwaltung für die optionalen WMI-Dienstparameter.

Für nahezu jeden dieser Befehle stehen diese Parameter zu Verfügung:

ASSOC

CALL

CREATE

DELETE

GET

LIST

SET

Nützliche Befehle sind jedoch folgende:

Wmic **startup** get Description,Command

* Zeigt die Programme an, die beim Windowsstart geladen werden.

Wmic **process** get ExecutablePath,ProcessId

* Zeigt die momentan laufenden Prozesse mit Pfad und PID an.

Wmic **service** get DisplayName,Name,PathName,ProcessID

* Zeigt alle Dienste mit Anzeigenamen, wahrem Namen, Prozesspfad und PID des laufenden Prozesses, falls vorhanden, an.

# X

* **xcopy:**

Dieser Befehl ist dasselbe wie ***copy***, nur werden hierbei noch die Ordner mitkopiert und es

gibt zahlreiche Befehlserweiterungen:

**/C** = Setzt das Kopieren fort, **auch wenn Fehler auftreten**.

**/E** = Kopiert **alle leeren Ordner mit**, jedoch keine nicht-leeren Ordner!

**/F** = **Zeigt alle Namen** der Quell- und Zieldateien beim Kopieren **an**.

**/G** = Ermöglicht das Kopieren von verschlüsselten Dateien auf ein

Ziel, wo Verschlüsselung nicht unterstützt wird.

**/H** = Kopiert auch **Dateien mit den Attributen 'Versteckt'** und **'System'**.

**/K** = Kopiert die **Attribute mit**. Normal wird 'Schreibgeschützt' gelöscht.

**/L** = Listet alle momentan kopierten Dateien auf.

**/O** = Kopiert **Informationen über den Besitzer und ACLs** (Zugriffserlaubnis).

**/P** = Fordert vor dem Erstellen jeder kopierten Datei zu einer **Bestätigung** auf.

**/Q** = Zeigt beim Kopieren keine Dateinamen an.

**/R** = Überschreibt schreibgeschützte Dateien.

**/S** = Kopiert **alle Ordner mit, die nicht leer sind**.

**/X** = Kopiert **Dateiüberwachungseinstellungen** (geht nicht ohne /O).

**/Y** = Keine Nachfrage, wenn eine Datei überschrieben wird.

**/-Y** = Extra Nachfrage, wenn eine Datei überschrieben wird.

**/Z** = Kopiert Dateien in einem Modus, der einen **Neustart ermöglicht**.

Unterschied:

copy "C:\Programme\Mozilla Firefox" "C:\Software\Mozilla Firefox"

kopiert alle Dateien, aber keine Ordner.

xcopy /s "C:\Programme\Mozilla Firefox" "C:\Software\Mozilla Firefox“

kopiert auch die Ordner mit.

Andere Beispiele:

xcopy "C:\WINDOWS\" "F:\WINDOWS\" /S /E /L /G /H /R /K /O /X /-Y

# Befehlsreihenfolge für Einsteiger

|  |  |
| --- | --- |
| 1. Echo | 40. Path |
| 1. Pause | 41. Type |
| 1. Cls | 42. Sort |
| 1. Color | 43. More |
| 1. Exit | 44. Find |
| 1. Title | 45. Findstr |
| 1. Ping localhost | 46. Xcopy |
| 1. >nul | 47. *Errorlevel* |
| 1. Ping | 48. Netsh |
| 1. Shutdown | 49. Netstat |
| 1. Cd / chdir | 50. *Argumente* |
| 1. Chkdsk | 51. %0 |
| 1. Comp | 52. Shift |
| 1. Copy | 53. Defrag |
| 1. Move | 54. Format |
| 1. Date | 55. Attrib |
| 1. Time | 56. Assoc |
| 1. Del / Erase | 57. Ftype |
| 1. Ver | 58. Reg |
| 1. Edit | 59. *Sonderzeichen* |
| 1. Start | 60. Cacls |
| 1. *Programmpfade* | 61. User / User32 |
| 1. Call | 62. Systeminfo |
| 1. Goto | 63. Gpresult |
| 1. *Springpunkte* | 64. Driverquery |
| 1. Rem | 65. Tasklist |
| 1. Ren | 66. Tskill |
| 1. Dir | 67. Taskkill |
| 1. Tree | 68. Pushd / Popd |
| 1. Cmd | 69. Mode |
| 1. Explorer | 70. Setlocal |
| 1. Command.com | 71. Endlocal |
| 1. Md /mkdir | 72. Net |
| 1. Rd / rmdir | 73. Sc |
| 1. Set | 74. Wmic |
| 1. *Variabeln* | 75. *Kurznamen* |
| 1. %var% | 76. For |
| 1. %var:~x,y% | 77. *Ascii-Zeichen* |
| 1. If |  |

Spielen Sie mit den Befehlen, wenn möglich, bei jedem neuen dazugekommenen erst etwas herum, bevor Sie mit den nächsten lesen. Viel Spaß beim Batchen!

Bei jeglichen Beschwerden und Fragen richten Sie sich an 414-702-600 in ICQ oder GrellesLicht28 in YouTube.