DIETMAR **W**ALKER - **P**C-**B**LITZHELFER-**N**OTDIENST
Nationalgasse 14 • **7**2124 **P**liezhausen • ∅ **T**el. 07127 / 890 729 - **F**ax 89118
Internet: http://www.pc-blitzhelfer.de – **M**obil 0172-882 79 55

Anleitung Batch-Dateien erstellen – Befehle für Anfänger – und nützliche Expertentipps

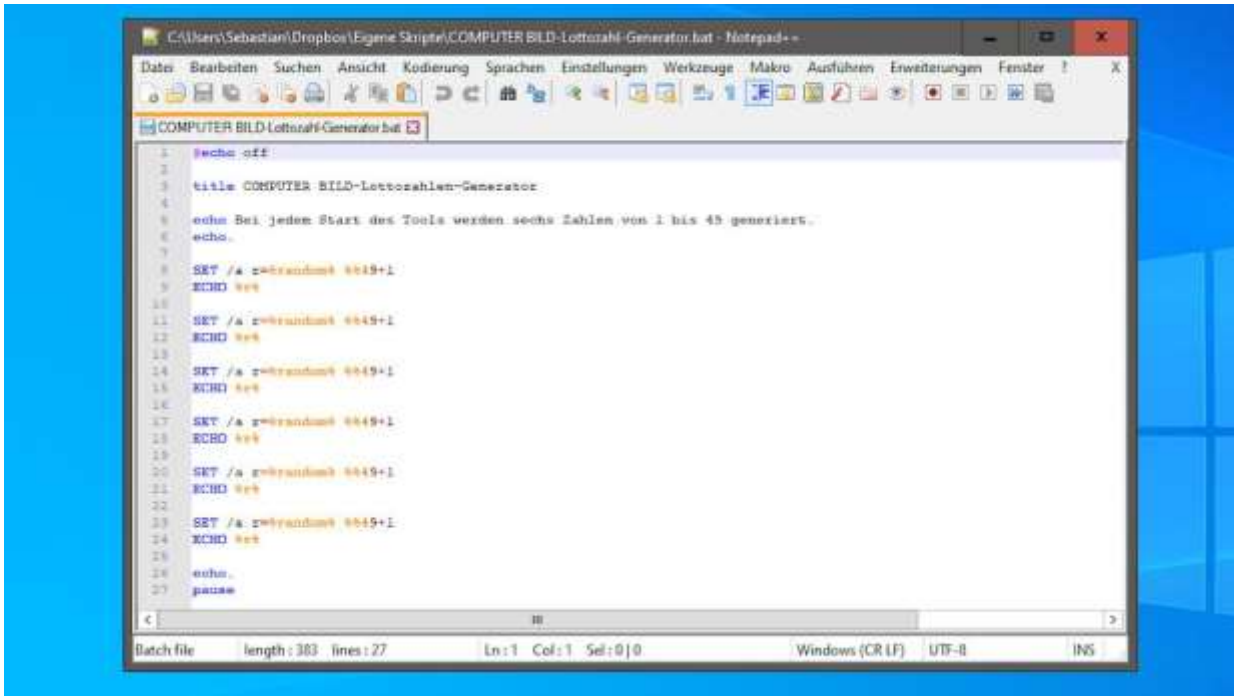
Befehle für Anfänger und nützliche Expertentipps

Wenn Sie erste Schritte im Bereich Programmierung wagen und möglichst schnell Erfolgserlebnisse wollen, sollten Sie sich mit der hierfür wohl einfachsten Sprache **"Batch"** beschäftigen. Batch-Dateien sind mit einem beliebigen Texteditor erstellbar. Hierfür eignet sich bereits der Windows-Editor, ist er für anfängliche Gehversuche eine gute Wahl. Batch eignet sich als (erste) Programmiersprache, wenn Sie zumindest etwas firm in Sachen Kommandozeilenbefehlen sind, denn auf denen basieren Batch-Programme, auch "Batch-Skripte" genannt, vollständig.
Batch-Dateien tragen die Endung ***.cmd*** **oder** ***.bat*** – das Dateiformat wählen Sie selbst. Letzteres Format ist identisch mit dem Aufrufbefehl der Windows-Kommandozeile: Diese starten Sie ebenfalls mit Windows-R und der Eingabe von *cmd*.
Ein Vorteil von Batch besteht darin, dass es relativ **einfach zu erlernen** ist. Wenn Sie viele Kommandozeilenbefehle kennen, kommen Sie damit binnen Minuten bis Stunden an kreative oder an zwar weniger erfinderische, dafür aber für Sie nützliche Programme. Kenntnisse über Kommandozeilenbefehle und deren Parameter vorausgesetzt, automatisieren Sie so zahlreiche Aufgaben. Am besten klappt das unter Windows 10, das leistungsfähigere Technik als Windows 7 und Windows 8.1 mitbringt: Hier sind etwa verschiedenfarbige Textdarstellungen parallel möglich; das ist ein Gimmick, allerdings für (Hobby-)Coder eventuell ein Upgrade-Grund. Die Tipps in diesem Artikel sind getestet mit Windows 7, Windows 8.1 und Windows 10 1909 (November 2019 Update).
Hinweis: Wenn Sie keinerlei Vorkenntnisse zu Batch mitbringen, empfiehlt es sich, den Artikel vom Anfang bis zum Ende zu lesen. Teilweise bauen die folgenden Absätze auf den vorherigen auf. Wem die im Folgenden erläuterten Batch-Grundlagen wiederum zu langweilig wirken, der kann direkt in die Fotostrecke einsteigen.
Der Artikel im Überblick:

Batch-Dateien erstellen mit Notepad++

Eine leistungsfähige Alternative zum Bordmittel Editor (notepad.exe) ist **Notepad++**. Das Tool ist ebenfalls schlank und mag anfangs zwar verwirrend wirken, unterstützt jedoch das Erstellen von Batch-Dateien sowie von Dateien in weiteren Sprachen. Außerdem bietet Notepad++ eine Syntax-Hervorhebung: Codeblöcke färbt es also farblich passend ein. Der Texteditor merkt sich beim Schließen die darin geöffnete Datei, die es beim nächsten Aufruf erneut öffnet. Ferner ermöglicht Notepad++ (anders als Windows) die Darstellung der Zeichen Ä, ä, Ü, ü, Ö, ö und ß über Batch-Dateien; die Kommandozeile zeigt hier bei der Erstellung mit notepad.exe nur unerwünschte Zeichen an. Hinzu kommen Tabs, mit denen Sie mehrere Projektdateien zugleich öffnen und so im Wechsel ansehen oder bearbeiten; selbst der Windows-10-1909-Editor kennt keine Tabs.

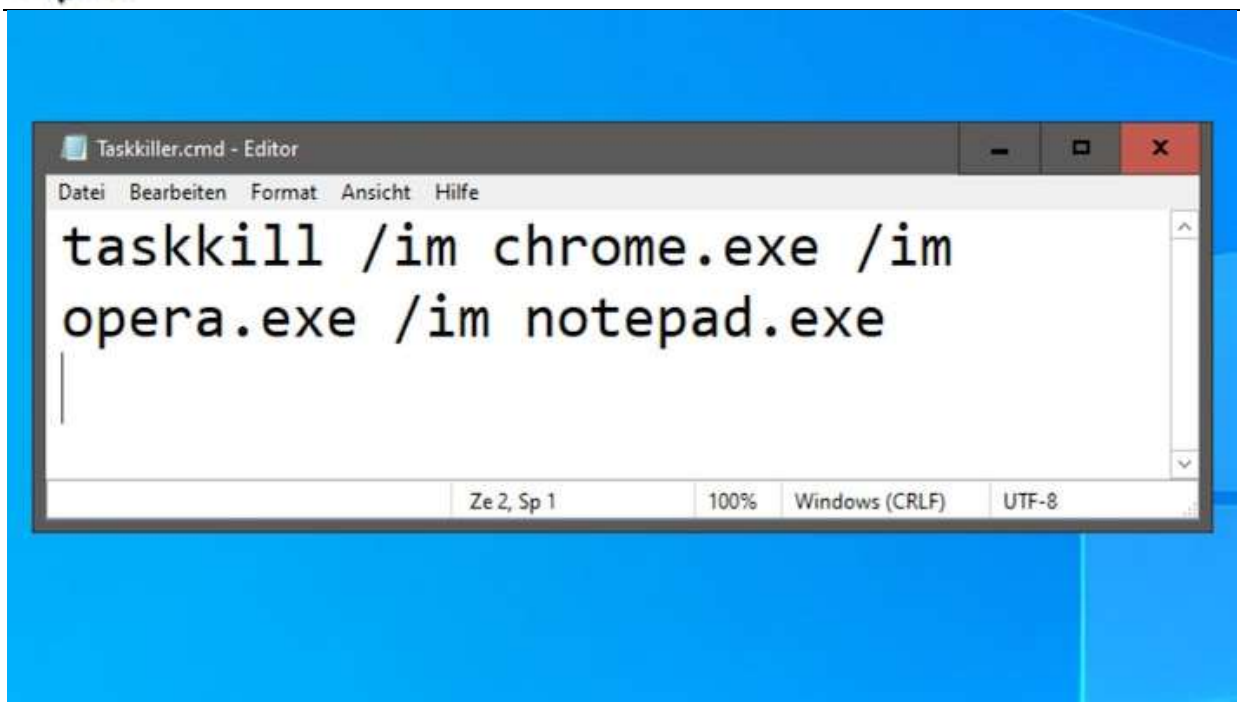
» Download: Notepad++ herunterladen
» Download: Notepad++ Portable herunterladen
» Download: NotePad++ (Windows-10-App) herunterladen



Notepad++ ist für das Batch-Coding empfehlenswert, da es Erleichterungen bietet und vor allem Umlaute in Batch-Programme bringt, die die Kommandozeile später sogar anzeigt.

Batch: Grundlagen

In Batch-Dateien hinterlegen Sie Textanweisungen, die Sie wahlweise mit der Dateiendung *.bat oder im CMD-Format speichern. Die Formate sind gleichwertig, per Doppelklick auf so eine Datei öffnet sich die Kommandozeile (Eingabeaufforderung, Prompt, CMD) samt des Skriptes. Wenn es fehlerfrei formuliert ist, arbeitet die CMD den darin enthaltenen Code von oben nach unten ab. Grundsätzlich funktionieren alle Kommandozeilenbefehle auch in Batch-Dateien. In Ausnahmefällen sind Befehle leicht abzuwandeln, will man sie per Batch ausführen. Der Reiz an Batch liegt darin, dass Sie einen häufig gebrauchten oder gleich mehrere Befehle in CMD-/BAT-Dateien hinterlegen und die Kommandos fortan nicht mehr manuell in die CMD eingeben müssen; es genügt ein Doppelklick auf die Skripte oder sogar das Nichtstun (für Letzteres verschieben Sie eine Batch-Datei in den Windows-Autostart-Ordner, der sich etwa mit Windows-R und dem Befehl *shell:startup* öffnet). Ein anderer Nutzer oder der Batch-Programmierer selbst kann im ausgeführten Batch-Programm, wenn es passend entwickelt wurde, auch Einfluss auf die Aktionen nehmen: Beispielsweise startet er durch das Drücken einer beliebigen Taste die nächste Aktion oder er tippt etwas ein, was Teil des ausgeführten Befehls wird (ohne dass man den Befehl sehen muss, was selbst Laien ohne Interesse und Kenntnissen an Technik komplexe Windows-Eingriffe zugänglich macht; Stichwort Variablen). Dieser Artikel soll Ihnen eine Einführung in Batch mit einigen Praxis-Tipps liefern, ist aber bewusst nicht vollständig – denn im Internet finden Sie bei Interesse unerschöpfliche weiterführende Informationen. Der Artikel braucht das Rad nicht neu zu erfinden und wäre andernfalls noch länger als bereits jetzt schon.
Für fortgeschrittene und Profi-Batch-Nutzer finden sich in der folgenden Fotostrecke Tipps für Probleme mit dem Batch-Coding, die der Artikel-Autor hatte und lösen konnte. Diese Übersicht eignet sich als Nachschlagwerk. Das tun auch Batch-Dateien, die man selbst bereits erstellt hat: Sicherlich will und kann sich kaum jemand alle Befehl(sblöck)e merken; einige sind kaum auszusprechen und zudem kompliziert. Hier hilft es, per Windows-Editor (Kontextmenüpunkt "Bearbeiten" bei einer Batch-Datei auswählen) in den Quellcode einer Batch-Datei zu schauen, den benötigten Befehl dieses älteren Projekts für ein neues Programmierprojekt herauszukopieren und in das Editor-Fenster für das neue zu schaffende Werk einzufügen. Sinnvoll ist es also, fertig erstellte Batch-Programme aufzubewahren; das erleichtert das spätere Coden ungemein.



Batch-Dateien erstellen: Tipps für Fortgeschrittene und Profis

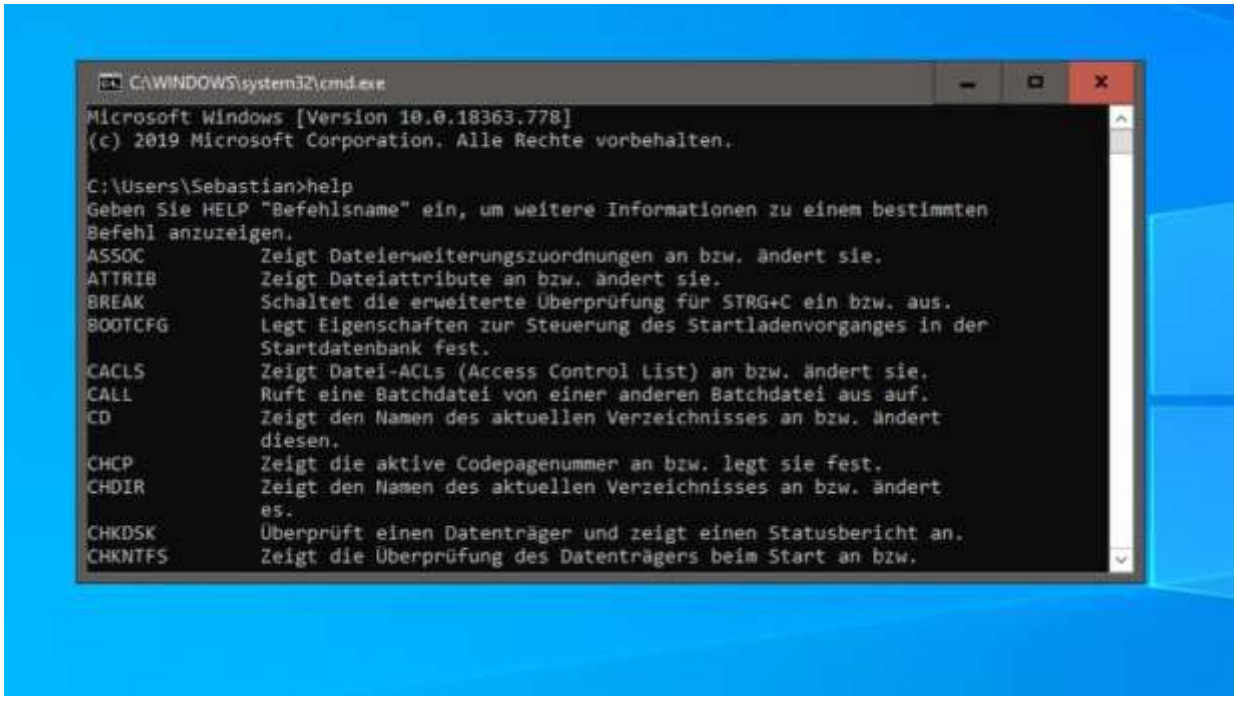
Batch-Datei erstellen

Die Frage, wie sich Batch-Dateien erstellen lassen, ist auf zwei Weisen zu beantworten. Die kurze Antwort beschränkt sich auf das Dateiformat: Öffnen Sie etwa den Windows-Editor und tippen Sie ein Codegerüst (oder gar nichts) ein. Drücken Sie Strg-S zum Speichern beziehungsweise klicken Sie im Falle einer geöffneten Batch-Datei, wenn Sie sie mit vorgenommenen nachträglichen Änderungen daran nicht überschreiben wollen, auf "Datei > Speichern unter". Vergewissern Sie sich, dass der Dateispeicherort Ihren Wünschen entspricht und ein Dateiname mit der Endung *.cmd* oder *.bat* eingetragen ist (etwa Skript.cmd), und bestätigen Sie per Klick auf "Speichern". Die ausführliche Antwort: Batch-Dateien erstellen Sie so wie eben beschrieben, doch machen Sie mehr, als bloß eine Datei im richtigen Format anzulegen – auch so erzeugen Sie Batch-Files, nur sind sie wenig nützlich. Machen Sie sich Gedanken um das, was Sie erreichen wollen, und fügen Sie die entsprechenden Befehle ein. Einige erste Schritte, die womöglich die Lust auf eine Recherche nach mehr Infos wecken, finden Sie in den folgenden Absätzen.

Batch: Erste Schritte

Mit am einfachsten ist das Batch-Szenario, **Programmstarts** zu realisieren und dies eventuell angereichert mit Parametern zu tun. Parameter sind Schalter, die nach einem Slash ("/") anzugeben sind und die eine Aufgabe präzisieren. So starten Sie etwa ein **Programm mit einer bestimmten Prozess-Priorität** (beschleunigt oder gebremst) oder **zeitverzögert** – oder **Sie fahren Windows nach dem Ablauf eines Counters herunter**. Zugegeben, so manches davon lässt sich ebenso per Verknüpfung (Rechtsklick auf den Desktop oder im Explorer, "Neu > Verknüpfung") erledigen, doch als erste Schritte für Batch sind die genannten Beispiele durchaus geeignet. Denn wenn Sie so etwas ausprobieren und sich danach an Weiteres heranwagen, setzen Sie zudem Komplexeres um. Solche Skripte sind den Verknüpfungen im Umfang ebenso wie in der Leistungsfähigkeit weit überlegen. Nützlich ist es etwa auch, **mehrere Programme in einem Rutsch zu starten**. Dabei haben Sie die Wahl, ob sich die Programme beim Aufrufen Ihrer Batch-Datei automatisch nacheinander öffnen oder ob das jeweils nächste Programm erst lädt, wenn Sie das vorige noch aufgerufene beendet haben.

Auch das bloße Ausgeben von Text in einem Kommandozeilenfenster via Batch gehört zu den simpelsten Gehversuchen. Gut zu wissen ist, dass die Groß- und Kleinschreibung in Batch in der Regel keine Rolle spielt; eine Ausnahme mag für Sie das Anzeigen von Text über den Befehl *echo* sein; wer Text mit korrekter Groß-/Kleinschreibung ausgibt, macht die Lesbarkeit angenehmer. Das Folgende sollte jeder Einsteiger beherrschen ...



Die Kommandozeile dient als Befehls-Interpreter und führt – ihrerseits optisch simpel gestrickt – Batch-Tools aus. Hier sehen Sie die Ausführung des Windows-eigenen *help*.

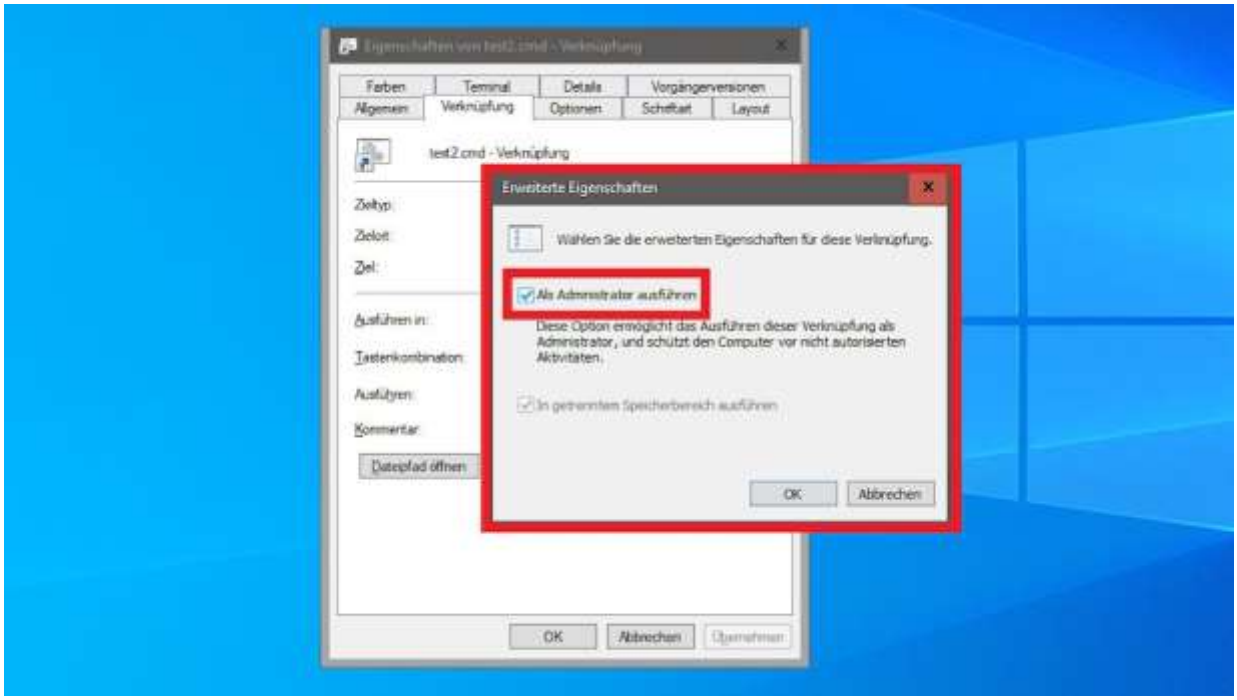
Programme per Batch starten

Programm oder Datei per Batch starten: Hinterlegen Sie in Ihrer Batch-Datei den Pfad zum aufzurufenden Element. Liegt die Batch-Datei im selben Ordner wie das Element, können Sie den Pfad auch weglassen und nur den Dateinamen angeben. Ein Beispiel:
F:\LCISOCreator.exe
Befindet sich im Pfad der aufzurufenden Datei ein Leerzeichen, so ist der Pfad innerhalb des Befehls in Anführungszeichen zu setzen:
"F:\Neuer Ordner\LCISOCreator.exe"
Es ist üblich, Anführungszeichen am Anfang und Ende des Pfades zu setzen, es genügt jedoch ein "-Zeichen am Anfang – entscheiden Sie sich für eine Variante:
"F:\Neuer Ordner\LCISOCreator.exe
Programm mit Prozess-Priorität starten: Hier arbeiten Sie mit Parametern. Soll etwa das portable Tool LCISOCreator mit einer bestimmten Prozess-Priorität starten, ist je nach Wunsch-Priorität ein anderer Parameter anzugeben. Der Pfad zur EXE-Datei ist in Ihrem Fall sicherlich abzuändern, abhängig davon, was Sie aufrufen wollen:
start /Low "crome" F:\LCISOCreator.exe (für die Priorität "Niedrig")
start /BelowNormal "crome" F:\LCISOCreator.exe ("Niedriger als normal")
start /AboveNormal "crome" F:\LCISOCreator.exe ("Höher als normal")
start /High "crome" F:\LCISOCreator (für die Priorität "Hoch")
start /Realtime "crome" F:\LCISOCreator (theoretisch für die Priorität "Echtzeit“)
Ferner gibt es noch den Parameter ***/normal*** (also ***start /Normal "crome" <Pfad>***), doch ist er in der Regel witzlos, weil Programme in den meisten Fällen ohnehin mit der Priorität "Normal"

starten. Die Priorität einer (per Batch aufgerufenen) Software kontrollieren Sie im Task-Manager im Kontextmenü des entsprechenden Prozesses.
Übrigens hat *crome* nichts mit dem Browser Google Chrome zu tun: Unabhängig davon, ob das Programm installiert ist, funktioniert der Batch-Ausdruck *crome*.

Batch-Datei als Administrator ausführen

Problem: Bei der **Echtzeit-Priorität** respektive bei Realtime führt Windows entsprechende Software nicht mit der "Echtzeit"-Priorität aus, **stattdessen weist der Task-Manager die um eine Stufe schlechtere Priorität "Hoch" aus.** Um in Echtzeit Software zu laden, sodass sie die maximalen Prozessor-Ressourcen erhält, klicken Sie Ihre Batch-Datei mit der rechten Maustaste an und wählen **"Als Administrator ausführen"**. Administrator-Rechte sind ein generelles Problem einiger Batch-Skripte: Benötigen sie diese erweiterten Rechte, reicht ein Doppelklick auf so eine Batch-Datei nicht aus; entweder sie verhält sich abweichend (wie hier mit der Hoch- statt zugewiesener Echtzeit-Priorität) oder es erscheint eine Fehlermeldung und das Skript-Programm funktioniert gar nicht erst. **Es nervt häufig, den Kontextmenüpunkt "Als Administrator ausführen" auszuwählen;** zumal andere Nutzer darauf ohne eine Dokumentation nicht immer kommen und dies die Sache verkompliziert.
Abhilfe schaffen Sie auf zwei Weisen: Erstens erstellen Sie eine **Verknüpfung zu Ihrer Batch-Datei**, indem Sie sie rechtsklicken und im Kontextmenü den Befehl "Verknüpfung erstellen" auswählen; die Verknüpfung bearbeiten Sie nun so, dass sie ohne das Kontextmenü als Administrator die Batch-Datei lädt. Hierfür gehen Sie per Doppelklick bei gedrückter Alt-Taste auf Ihre Verknüpfung in deren **Eigenschaften** und wählen **"Erweitert"**, setzen einen Haken vor **"Als Administrator ausführen"** und bestätigen mit "OK > Übernehmen > OK". Es genügt nun ein Doppelklick auf die Verknüpfung, um einen Administrator-Start auszulösen; wie bei der Rechtsklick-Methode erscheint ein Warnfenster der Benutzerkonten-Steuerung, worin mit "Ja" zu bestätigen ist.
Die zweite Variante für Administrator-Starts ist eleganter, aber technisch aufwendiger: **Hinterlegen Sie in Ihrer Batch-Datei Code, der den Aufruf mit Administrator-Rechten bewirkt.** Da er den Rahmen diese Artikels sprengen würde, finden Sie dieses komplexe Code-Ungetüm in der Fotostrecke mit den 41 Profi-Batch-Tipps.



Haken rein und Administrator sein: So führen Sie bestimmte Batch-Dateien korrekt aus.

Batch: Programm verzögert starten, Herunterfahren

Programme verzögert starten: Der Aufruf von Programmen mit einer Verzögerung ist nützlich, wenn Sie wissen, ein Programm erst in X Minuten zu benötigen – oder wenn eine bestimmte CMD-Routine, etwa das Einblenden von Text als Erfolgsmeldung zu einem bestimmten abgearbeiteten Kommandozeilenbefehl, zur angenehmeren Benutzerführung nicht abrupt sofort erscheinen soll. Erforderlich ist hierfür *timeout*:

timeout /t 60
F:\LCISOCreator.exe

startet im Beispiel den LCISOCreator erst nach 60 Sekunden; ein Counter zählt die verbleibende Wartezeit herunter. Für die sofortige Programmausführung ist eine beliebige Taste zu drücken, worauf die Kommandozeile zur Laufzeit hinweist. Eine Besonderheit bei Windows 10: Seit dem Anniversary Update (1607) führen Sie Batch-hinterlegte Programme mit einem Timeout sofort aus, indem Sie das Fenster der Kommandozeile per Mauszeiger in der Breite vergrößern oder verkleinern. Das Skalieren in der Höhe über die Fensterränder wirkt hingegen nicht; Windows 7/8.1 unterstützen den Trick nicht einmal teilweise, hier drückt man wie empfohlen eine beliebige Taste.

Windows automatisch herunterfahren: Zum Herunterfahren von Windows dient das Startmenü, wer das Ganze wiederum befehlsbasiert abwickeln will, braucht die systemeigene Datei shutdown.exe. Als Parameter dient der EXE *-s -t <Zahl>* zum Herunterfahren nach der definierten Minutenzahl; ***-r -t <Zahl>*** dagegen (s = shutdown = Herunterfahren; r = reboot = Neustarten) startet das System neu. Beispielsweise verabschiedet sich Windows folgendermaßen nach zwei Minuten in die Nachtruhe:

shutdown -s -t 120

Batch: Befehlsausgabe unterdrücken

Befehle unterdrücken: Womöglich wollen Sie keinerlei Kommandozeilenbefehle im CMD-Fenster bei einer Batch-Datei-Ausführung sehen, denn was intern passiert, sieht nicht gerade ästhetisch aus. Mit einem *@echo off* am Anfang Ihres Dokuments unterdrücken Sie die Befehlsanzeige im Klartext (zur Fehlersuche beim Experimentieren mit Batch-Dateien und zum schrittweisen Anpassen ist das oft ungeeignet, für ein fertiges Batch-Exemplar hingegen stellt das echo-Unterdrücken in der Regel eine gute Lösung dar). *@echo off* gehört zu den Befehlen, die sich jeder Anfänger merken sollte. Es ist einmalig in einer Batch-Datei einzufügen und wirkt sich auf die folgenden Inhalte aus.

Counter unterdrücken: Kommandozeilen-seitige Textausgaben wie den timeout-Counter werden Sie mit *@echo off* nicht los. Mit NUL, einzufügen als *" > NUL"*, klappt es im Beispiel vom LCISOCreator-Aufruf allerdings dennoch:

@echo off
timeout /t 60 > NUL
F:\LCISOCreator.exe

Einen Batch-Datei-Titel definieren

Batch-Datei einen Titel geben: Etwas fürs Auge: Wollen Sie Ihrer Batch-Datei in der Titelleiste einen bestimmen Namen geben, definieren Sie ihn per *title*. Etwa so:

title Selbstbau-Programm

Batch: Mehrere Programme nacheinander starten

Mehrere Programme starten: Mehrere Programme starten Sie so recht plump:

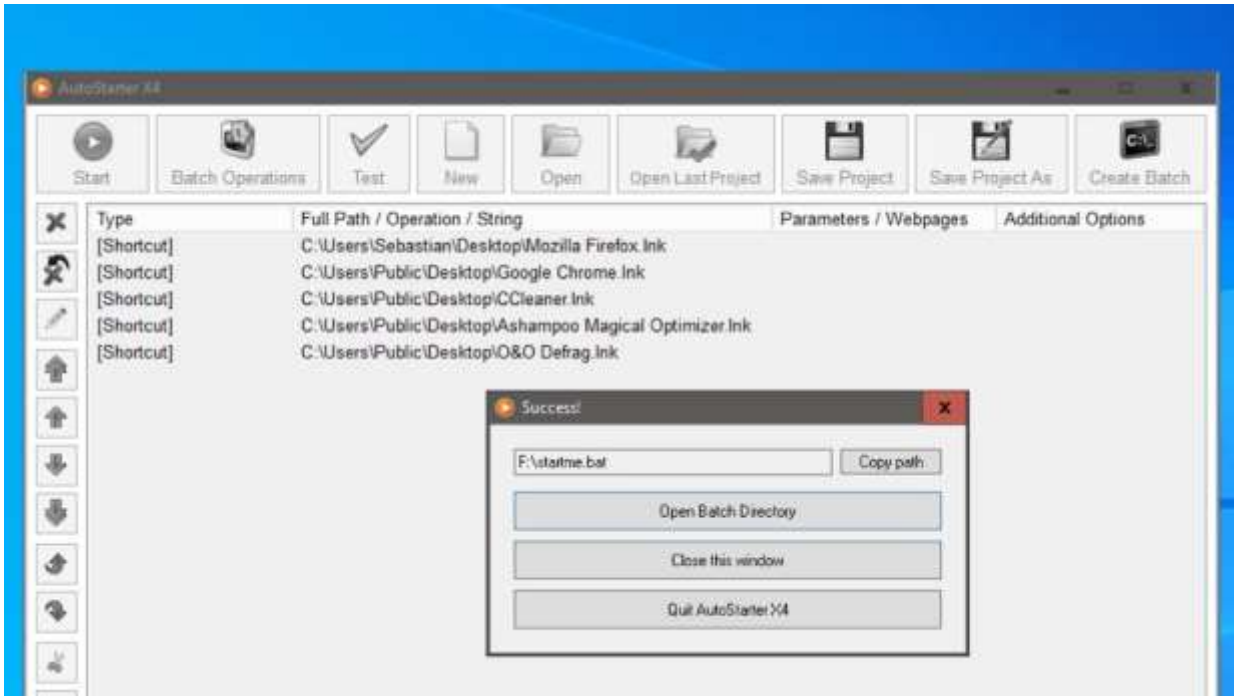
F:\LCISOCreator.exe
C:\Windows\System32\winver.exe

C:\Windows\System32\appwiz.cpl

Als Nachteil erweist es sich beim obigen Dreizeiler, dass das nächste Programm erst lädt, wenn Sie das vorige per Klick beendet haben. Wenn Sie Ihren Batch-Code aber wie folgt gestalten, laden der LCISOCreator, winver und appwiz.cpl parallel:
start "" F:\LCISOCreator.exe
start "" C:\Windows\System32\winver.exe
start "" C:\Windows\System32\appwiz.cpl

Sammelaufrufe ohne einzutippende Befehle

Ein Software-Tipp: **AutoStarter X4** ermöglicht es, Batch-Dateien für Multi-Programmstarts ohne eigenes Coding zu erstellen. Es genügt das Hinzufügen der gewünschten Programme in das Fenster. Anschließend klicken Sie oben rechts auf den Button "Create Batch", wonach für die zu erzeugende BAT-Datei optional mit "Browse" ein anderer Speicherort festlegbar ist. Das Programm stellt den Code für Ihre Batch-Datei zusammen und arbeitet dabei mit mehreren Zeilen *start "" <EXE-/LNK-Pfad>*.
» Download: AutoStarter herunterladen



Der AutoStarter X4 erlaubt es Ihnen, ohne Vorkenntnisse beziehungsweise bei geringem Aufwand eigene Batch-Dateien zu erstellen. Per Drag & Drop befördern Sie Programme für einen Sammelaufruf in das Tool, per Klick auf "Create Batch" erhalten Sie ein Skript startme.bat.

Batch: Text anzeigen

Per Batch Text ausgeben: *echo* sorgt für das Anzeigen von Text innerhalb der Kommandozeile. Das sollte jeder Batch-Anfänger einmal ausprobiert haben. Gut kombinierbar ist das Ganze auch mit *@echo off* und " >NUL": Beide unterdrücken in Batch-Programmen bestimmten Textinhalt, den der Nutzer (Laie) nicht sehen soll. Was Sie ihm jedoch zeigen könnten, ist der Einfachheit halber ein erklärender Text (per *echo*). Letzterer ist technisch zwar nicht unbedingt nötig, doch dient derlei dem Verständnis und erläutert etwa, worum es gerade geht oder was der Anwender tun soll. Auch Spiele oder Scherzprogramme lassen sich per

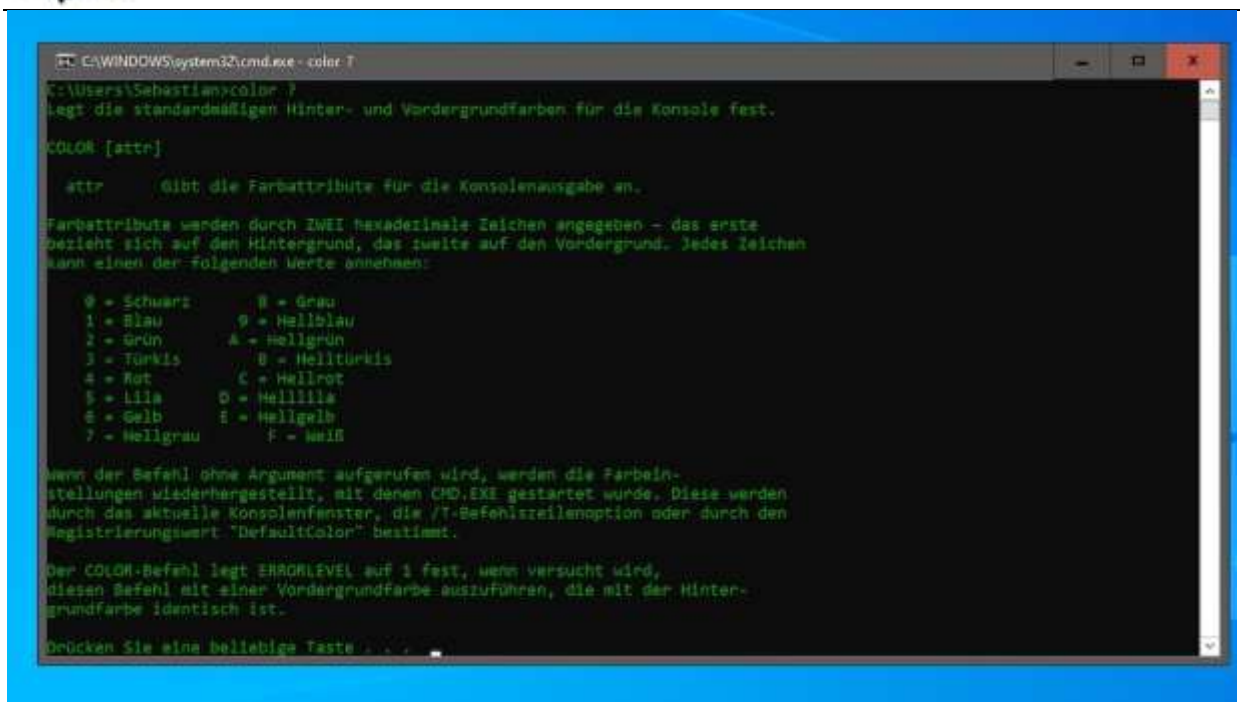
echo realisieren, jedenfalls wenn man es mit Weiterem kombiniert. ***echo Ich mag Schafe***
pause
etwa zeigt an: **Ich mag Schafe**
Beachten Sie, dass zu Beginn Ihres CMD-Dokuments ein *@echo off* stehen sollte, andernfalls gibt die Kommandozeile korrekt, aber entgegen Ihres Interesses aus:
echo Ich mag Schafe
Ich mag Schafe

Umlaute in Batch-Dateien

Es stört gleichermaßen unter Windows 7, Windows 8.1 und Windows 10, dass die Kommandozeile Umlaute verschluckt. Hinterlegen Sie im Batch-Code etwa Folgendes, zeigt die CMD unerwünschte Zeichen an:
@echo off
echo Ich mag die Buchstaben Ä ä Ü ü Ö ö ß
pause
Die Kommandozeile verschluckt hier die letzten sieben Zeichen – es erscheinen alternative Pendants, die unschön aussehen. Das überrascht womöglich, da beim Eintippen des Batch-Datei-Inhalts im Windows-Editor alles noch korrekt aussah. Einen Ausweg bietet Notepad++. Installieren und starten Sie das Tool. Danach vergewissern Sie sich in der Menüleiste, dass zwei Einstellungen aktiviert sind: "Kodierung > UTF-8" und "Sprachen > M > MS-DOS-Style". In die Anwendung tippen Sie zum Beispiel ein:
@echo off
echo Ä ä Ü ü Ö ö ß
pause
Anschließend speichern Sie mit Strg-S und tippen im Speichern-unter-Dialog eine passende Dateinamenserweiterung ein, geben der zu erzeugenden Batch-Datei also die Endung *.bat* oder *.cmd*. Wenn Sie nun doppelt auf die Datei klicken, erfolgt in der sich öffnenden Kommandozeile eine durchweg korrekte Zeichenausgabe.

Batch: Schriftfarbe ändern

In der (Batch-)Kommandozeilen lesen Sie Inhalte standardmäßig Weiß auf Schwarz. Sowohl die Hintergrund- als auch die Textfarbe ändern Sie. So heben sich Ihre Skripte von anderen visuell ab. Das erfolgt mittels *color* und einem Parameter:
color <Zahl oder Buchstabe> definiert, welche Farbe Ihr Text hat.
Möglich sind hexadezimalen Zeichen: also Zahlen von 0 bis 9 und Buchstaben von A bis F.
Wollen Sie die Text- und die Hintergrundfarbe beeinflussen, deklarieren Sie beides mit
color <Zahl oder Buchstabe><Zahl oder Buchstabe>
Hierbei legen Sie, anders als beim ersten Befehl, mit dem ersten Parameter die Hintergrundfarbe fest; der zweite Parameter beschreibt die Textfarbe.



Die Kommandozeile beherrscht verschiedene Farben, in der Folge realisieren Sie sie per Batch.

Batch: Mehrere Farben in einer Zeile/zugleich

Wollen Sie mehrere Schriftfarben zugleich nutzen, brauchen Sie Windows 10; Windows 7 und Windows 8.1 unterstützen das Folgende noch nicht. Zunächst hinterlegen Sie in Ihrem Batch-File eine Zuordnung von Variablen mit Farb-Informationen:

set farbenwahl=<*Hier wird noch ein spezielles Zeichen benötigt, das in diesem Artikel leider nicht darstellbar ist; siehe unten*)*>
set green=%farbenwahl%[32m
set rot=%farbenwahl%[31m
set gelb=%farbenwahl%[33m
set blau=%farbenwahl%[36m
set Weiss=%farbenwahl%[37m
Tippen Sie als eigentlichen Inhalt zudem ***%IndividuelleFarbbezeichnung****, etwa das obere* ***green%*** ein sowie dahinter ein Wort, färbt Windows es gemäß der oben anhand von "[<Zahl>m" definierten Farbnummer. Ein Beispiel für einen bunten Farbmix:
@echo off
set farbenwahl=<*Hier wird noch ein spezielles Zeichen benötigt, das in diesem Artikel leider nicht darstellbar ist; siehe unten*)*>
set green=%farbenwahl%[32m
set rot=%farbenwahl%[31m
set gelb=%farbenwahl%[33m
set blau=%farbenwahl%[36m
set Weiss=%farbenwahl%[37m
echo %green% Dieser Text ist gruen %rot% Rot %gelb% Angenehmes Gelb %blau% Das hier ist blau %Weiss% Weisse Farbe
pause
* Es wird ein Zeichen benötigt, das in diesem Artikel nicht darstellbar ist, das Sie aber im Batch-Tool WLAN-Booster finden. So kopieren Sie es heraus: Laden Sie das Tool herunter

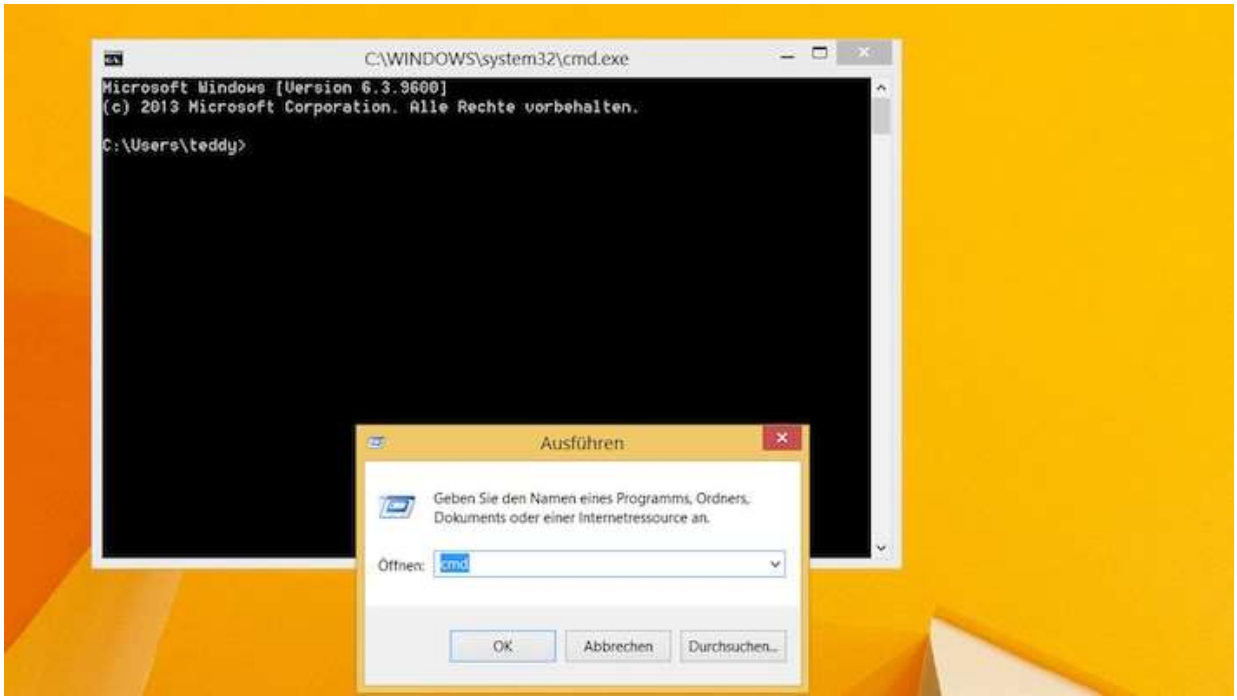
und entpacken Sie es. Klicken Sie die Datei "COMPUTER BILD-WLAN-Booster.cmd" mit der rechten Maustaste an; per Klick auf "Bearbeiten" laden Sie sie zur Source-Code-Einsicht im Editor. Drücken Sie Strg-F, fügen Sie in das so geöffnete Suchfenster *set farbenwahl=* ein und drücken Sie die Eingabetaste. Notepad markiert nun den Suchtreffer blau; markieren und kopieren Sie das Zeichen rechts davon mit Strg-C.
» Download: COMPUTER BILD-WLAN-Booster herunterladen

Nur Windows 10 führt so eine Batch-Datei ansehnlich aus, Windows 7 und Windows 8.1 zeigen stattdessen eine (weiße) Einheitsfarbe. Enthält ein Skript derlei Code, ist eine Abfrage des installierten OS und ein dementsprechendes *echo* abhängig vom ausführenden OS nicht verkehrt. Wie eine Systemabfrage umsetzbar ist, erfahren Sie in der Tipps-Fotostrecke oben.

Batch: Textausgabe in Datei umleiten

Ob ein per *echo* angezeigter Text oder das Ergebnis eines abgefragten Systemwerts per CMD-Befehl: Das, was die Kommandozeile hier anzeigt, leiten Sie in eine neu zu erstellende Datei um. Das funktioniert per > <*Dateipfad*>, etwa so:
echo Hallo Welt > F:\Gesichterter-Text.txt
Hier landet der Text "Hallo Welt" in der auf Laufwerk F:\ neu erzeugten Datei "Gesichteter Text.txt". Ein weiteres Beispiel zum Ausleiten von CMD-Inhalt in ein File (zum Ermitteln der beim Nutzer installierten Windows-Version):
ver > F:\Mein-eigenes-Windows.txt
Etwas trickreich: Führen Sie per CMD-Skript Folgendes aus:
echo 123 > F:\Kauderwelsch.txt
echo Dies und das > F:\Kauderwelsch.txt
... landet in der Datei Kauderwelsch.txt nur das zuletzt Eingefügte ("Dies und das"). Um beide Zeilen ins File zu schreiben, verwenden Sie ab der zweiten Zeile das Zeichen > doppelt. Optional nutzen Sie bereits bei der ersten Zeile das >>", unbedingt nötig ist es aber nur in der zweiten Befehlszeile (und in etwaigen weiteren Zeilen):
echo 123 >> F:\Kauderwelsch.txt
echo Dies und das >> F:\Kauderwelsch.txt

Übrigens bearbeiten Sie so auch die Windows-Hosts-Datei: Darin vermerken Sie unliebsame Websites, um sie Browser-weit zu sperren. Eine Anleitung finden Sie im Artikel "Windows 7/8/10: Unerwünschte Webseiten per Hosts-Datei sperren".



Die Windows-Kommandozeile bedienen – und konfigurieren

Batch: Bestätigung anfordern

pause einfügen: Bei einigen Kommandozeilenbefehlen ist der Befehl ***pause*** wichtig. Bei *echo* gilt dies etwa und ebenfalls, wenn Sie dem Nutzer Zeit lassen möchten, zu entscheiden, ob er fortfahren will. Mit eingefügtem pause-Befehl fordert die Kommandozeile dazu auf, eine beliebige Taste zu drücken. Erst damit geht es weiter. Im Fall von *echo* (und hier der Sympathie-Bekundung der Schafe) ist *pause* wichtig, damit sich die Kommandozeile nicht gleich wieder schließt. Der Befehlssatz lautet also etwa:

```
@echo off
echo Ich mag Schafe
pause
```

Wollen Sie zwei Zeilen Text anzeigen, verwenden Sie entsprechend zwei echo-Zeilen:

```
@echo off
echo Ich mag Schafe
echo Ich mag sie wirklich sehr
pause
```

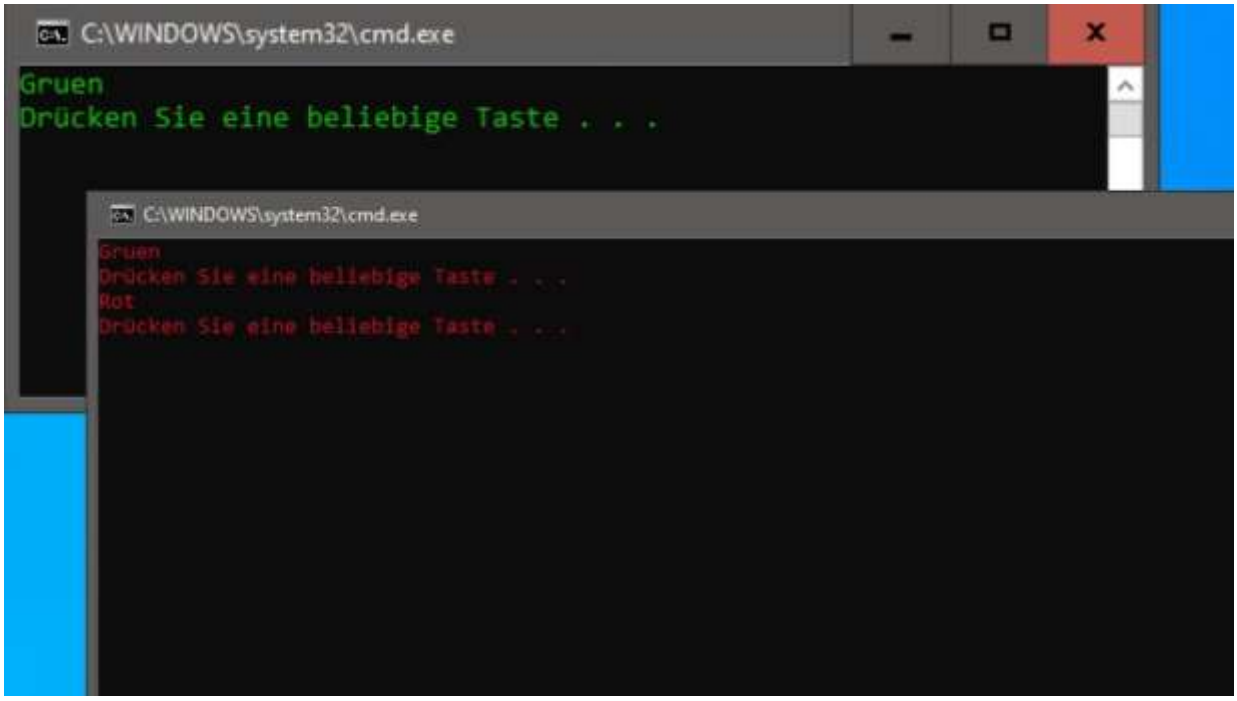
Wünschen Sie eine Leerzeile, etwa um die Übersicht zu erhöhen, realisieren Sie diese nicht etwa mit *echo* und danach einer leergelassenen Zeile, sondern per *echo.:*

```
@echo off
echo Ich mag Schafe
echo.
echo Ich mag sie wirklich sehr
pause
```

echo. ist nötig, um Leerzeilen darzustellen, zumindest in der Kommandozeilen-Batch-Ausführung. In CMD-Skripten im Editor wiederum können Sie munter Leerzeilen setzen; dies erhöht für Sie womöglich die Übersichtlichkeit im Skript. Die Kommandozeile ignoriert die Leerzeilen. Umbrüche zur Skript-Laufzeit setzen Sie so allerdings nicht um; übertreiben Sie es mit reinen Übersichts-Leerzeilen besser nicht, da sonst Codeblöcke durch zu großen Abstand voneinander kaum noch zusammenhängend wirken.
Neben dem Befehl *pause* gibt es noch das ***exit*** – die Kommandozeile beendet sich abhängig Ihrer Befehle von selbst, soll sie das unbedingt tun, erzwingen Sie das Schließen des CMD-Fensters mit *exit* am Ende Ihrer Befehlszeilen. Wichtig ist *exit* außerdem für Sprungmarken, wozu Sie weiter unten im Artikel mehr erfahren.
So erstellen Sie interaktives Programm, bei dem der Nutzer aufgefordert wird, eine beliebige Taste zu drücken (durch *pause*). Kommt er dem nach, führt die Kommandozeile den Code nach dem *pause* aus. Da *color* eine weitere Farbe definiert, wechselt sie von Grün (A definiert dieses) zu Rot (4 gehört zu dieser Farbe):

```
@echo off
color A
echo Gruen
pause
color 4
echo Rot
pause
```

Etwas unschön ist, dass der grüne Text "Gruen", wenn Sie per Tastendruck *color 4* ausführen, rot wird. Ein rot geschriebenes Wort "Gruen" wirkt merkwürdig. Auf Wunsch bauen Sie den Befehl *cls* nach dem ersten *pause* oder nach *color 4* ein; dadurch erscheint beim Betätigen der Taste kein Wort "Gruen" mehr, sondern alleine "Rot" in roter Kolorierung. Mehr zu CLS erfahren Sie über den nächsten Artikel-Absatz.



Oben sehen Sie die Kommandozeile nach dem Start, unten die Darstellung nach dem Drücken einer Taste. Dass "Grün" hier in der Form "Gruen" geschrieben ist, liegt daran, dass Buchstaben wie Ü und ü in der CMD ohne Klimmzüge falsch dargestellt werden.

Dass die Kommandozeile bei hinterlegtem Befehl *pause* stets "Drücken Sie eine beliebige Taste . . ." anzeigt, machen Sie sich zunutze. Stellen Sie per *echo* dem *pause* etwas Kreatives voran – hier nimmt *echo* Bezug auf pauses Tasten-Drück-Aufforderung:

@echo off
echo 123
echo Um fortzufahren, ...
pause
cls
echo 234
pause

Hier sieht der Nutzer Folgendes:

123
Um fortzufahren, ...
Drücken Sie eine beliebige Taste . . .

und wenn er eine Taste betätigt, erscheint

234
Drücken Sie eine beliebige Taste . . .

Batch-Variablen, Benutzereingaben, cls

Variablen sind praktisch, da Sie dank ihnen Teile langer Befehle nur einmal in Ihrem Batch-File unterzubringen brauchen – und künftig eine verkürzte Version davon nutzen. Ebenso lassen Sie damit den Nutzer Einfluss darauf nehmen, wie Ihr Batch-Skript arbeitet. Zum Beispiel definieren Sie eine Variable auf Basis der Phrase ABC123; diesen Namen können Sie variieren, wichtig ist, dass ABC123 später in %%-Zeichen erneut zum Einsatz kommt, so nimmt das Ganze jene Phrase an, die damit zusammengelegt wurde:

@echo off
set /p ABC123="Was geht ab: "
echo %ABC123% ist toll.
pause

fragt den Nutzer, was bei ihm abgeht. Er tippt es ein; daraufhin zeigt die Kommandozeile per echo an, was der Nutzer soeben definiert hat; das Ganze ist sogar noch in einen Satz eingekleidet ("[...] ist toll"). Soll bei der echo-Ausführung die Zeile darüber mit der Variablen-Konfiguration nicht angezeigt werden, realisieren Sie das über *cls*:

@echo off
set /p ABC123="Was geht ab: "
cls
echo %ABC123% ist toll.
pause

cls leert die Anzeige in der Kommandozeile. Es ist wie *@echo off* und *> NUL* kosmetischer Natur. Mehr zum CMD-Tool CLS erfahren Sie im Artikel "Windows 7/8/10: CLS-Befehl 'löscht den Bildschirminhalt– Was bedeutet das?".

Wollen Sie nicht vom Nutzer den Wert einer Variable erfragen, sondern diesen selbst definieren, geht das zum Beispiel wie folgt:

set Testvariable=BeliebigesWort
md %Testvariable%

md (**m**ake **d**irectory) erstellt im Beispiel in jenem Ordner, in dem das Batch-Skript gespeichert ist, ein Unterverzeichnis. Letzterer Ordner ist so benannt, wie der Wert der Variable. Es lassen sich übrigens auch Inhalte aus der Registry als Variable definieren.

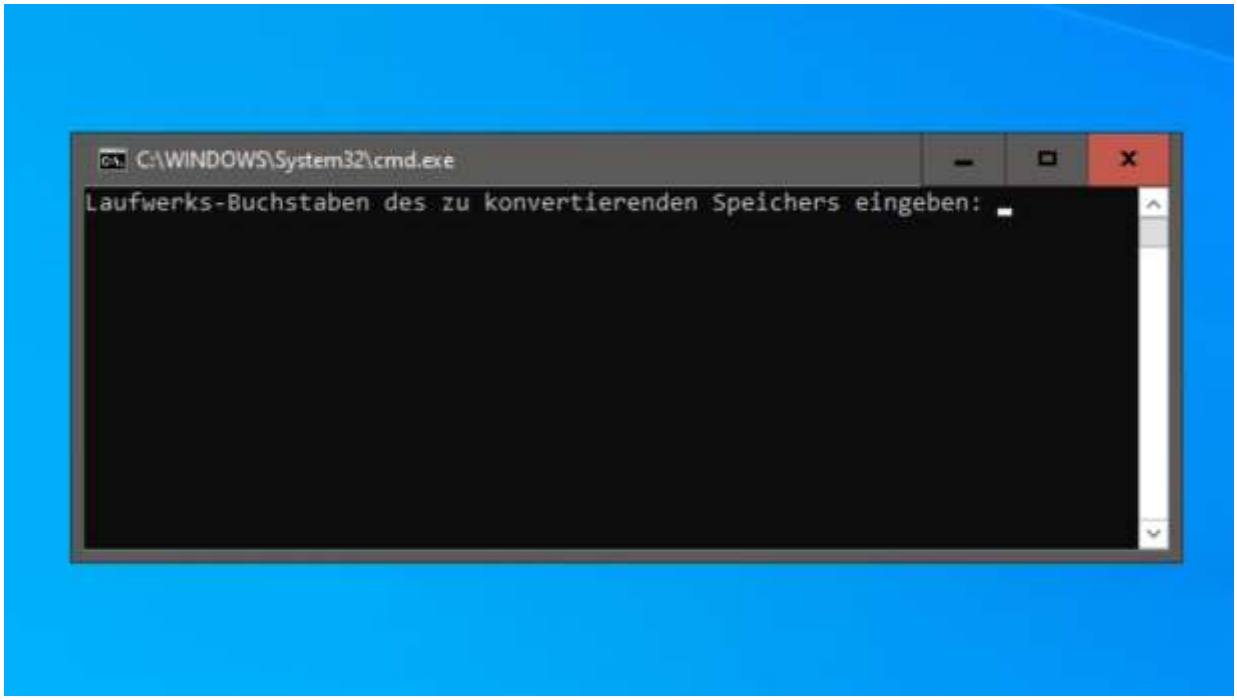
Ein Beispiel für etwas im Windows-Alltag durchaus Nützliches ist der Bau eines Dateisystem-Konverters. Windows bringt hierfür das nötige Befehlszeilenprogramm convert bereits mit. Es wandelt FAT32 in NTFS um, umgekehrt funktioniert es nicht. Ihr eigener Konverter weiß nicht, bei welcher Partition er FAT32 zu NTFS machen soll, also soll der Nutzer das festlegen. Seine Buchstaben-Angabe vervollständigt den intern (unsichtbar) abgearbeiteten Kommandozeilenbefehl. Der beim Nutzer abgefragte und von ihm getippte Laufwerksbuchstabe wird zum Wert der Variable, die in der letzten Zeile bei der Ausführung etwa durch F (wenn der User F eintippt) ersetzt wird:

@echo off
set /p eingabe="Laufwerks-Buchstaben des zu konvertierenden Speichers eingeben: "
convert %eingabe%: /fs:ntfs

Die Batch-Datei ist mit Administrator-Rechten auszuführen, da convert sie benötigt. Der Nutzer muss den kompletten Umwandlungsbefehl für FAT32 nach NTFS (convert <Laufwerksbuchstabe> /fs:ntfs) nicht kennen und sieht nur folgenden Text:

Laufwerks-Buchstaben des zu konvertierenden Speichers eingeben:

Normalerweise ist im convert-Befehl nach einem Laufwerksbuchstaben ein Doppelpunkt anzugeben; Nutzern bleibt das hier erspart, da der Doppelpunkt bereits in dem Befehl in der letzten Zeile vorkommt. Der vom User angegebene Laufwerksbuchstabe ist mit [Eingabe] zu bestätigen, er wird als Variablen-Inhalt gesetzt und landet via %eingabe% im Befehl in der dritten Zeile. Der Variablen-Name könnte auch ein anderer sein, es ist aber sinnvoll, ihn zum eigenen späteren Verständnis selbsterklärend zu benennen.



Skripte wie dieser FAT32-zu-NTFS-Konverter sind in etwa einer Minute gecodet.

Batch: Auskommentieren mit REM

Apropos "eigenes Verständnis": Wollen Sie erklärende Hinweise in Batch-Dateien vermerken, um sie später besser nachvollziehen zu können, fügen Sie ein *REM* ein. Windows ignoriert *REM <Irgendwas an Text>* und führt es nicht aus. Ein Beispiel:

@echo off
echo Ich mag Schafe
REM Die folgende Zeile ist für den Zeilenumbruch da

echo.
echo Schafe sind super.
REM Folgende Zeile unterdrückt den Hinweis "Drücken Sie eine beliebige Taste . . ."
pause > NUL
... zeigt lediglich (also ohne die REM-Erinnerung) an:
Ich mag Schafe
[Diese Zeile ist in der Kommandozeile frei, das echo. ist nicht zu sehen]
Schafe sind super

Batch-Variablen: Speicherplatz ermitteln

Hier ein weitaus komplexeres Beispiel für Variablen: Wmic ist ein leistungsfähiges Windows-Bordmittel und vielfältig einsetzbar. Im folgenden Beispiel ermittelt es, wie viele Byte an Speicherplatz auf C:\ bereitstehen. Kombiniert mit einer Variablen können Sie sich anschließend den Inhalt mit *echo* anzeigen lassen:

@echo off
for /F "skip=2 tokens=2 delims=," %%A in ('wmic LogicalDisk where "DeviceID='C:' " Get Size /FORMAT:csv') do (set "bytes=%%A")
echo %bytes%
pause

Die wichtigsten Batch-Befehle

Einige Anregungen für stundenlange Experimente: Neben Wmic relevant sind unter anderem noch ***reg*** (zum Hinzufügen und Löschen von Registry-Schlüsseln und -Einträgen; bei Eingriffen in HKEY_CURRENT_USER braucht es keine Administrator-Rechte, wohl aber bei welchen in HKEY_LOCAL_MACHINE), ***tasklist*** (zum Auflisten gestarteter Programme; fortgeschrittene Coder lösen etwa abhängig von der RAM-Präsenz eines Prozesses eine Aktion aus), ***taskkill*** (zum Beenden von Programmen über die EXE-Namen von deren Prozessen als Parameter), ***powercfg*** (für Arbeiten rund um Windows-Energiesparpläne), ***schtasks*** (um etwa geplante Aufgaben anzulegen und so Programmstarts respektive Registry-Tweaks ohne UAC-Abfragefenster vorzunehmen), ***md*** (für das Anlegen von Ordnern im Dateisystem) und ***del*** (um Dateien zu löschen; ***reg delete*** ist ein Äquivalent für Registry-Inhalte). ***move*** verschiebt Dateien, ***copy*** kopiert sie. Sinnvoll sind all diese OS-eigenen Programme nur mit Parametern.

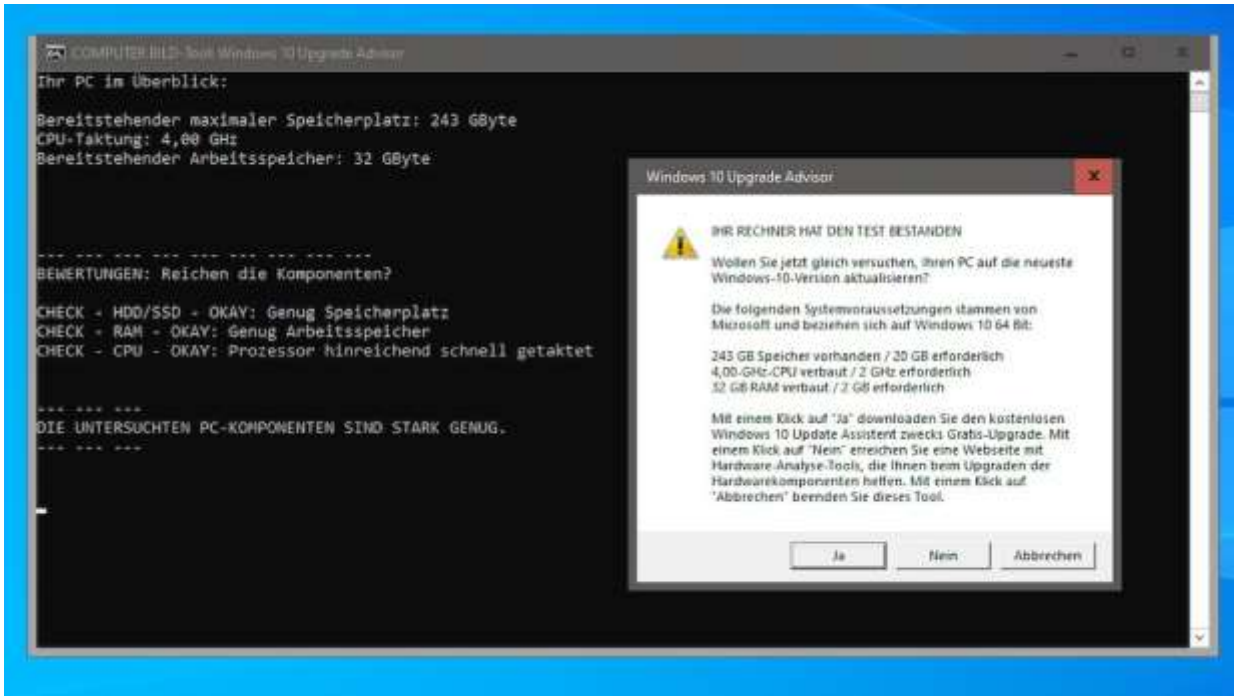
Mit *copy D:\LCISOCreator.exe F:* etwa kopieren Sie die genannte EXE-Datei auf Laufwerk F. Der Dateiname bleibt dabei erhalten. Soll das neue Element hingegen einen anderen Namen tragen, ergänzen Sie das F: zu einem Pfad samt Dateinamen, etwa so:

copy D:\LCISOCreator.exe F:\LCISO.exe

Batch mit GUI

Interessieren Sie sich dafür, wie man **Kommandozeilen-Programme mit GUI** (**G**raphical **U**ser **I**nterface, also grafischer Benutzeroberfläche) erstellt, finden Sie mehrere Tipps hierzu in der **obigen Profi-Tipps-Fotostrecke**. Ferner erfahren Sie darin, wie sich CMD-Skripte minimiert aufrufen lassen, wie man Text aus Textdateien extrahiert und wie sich per Batch Verknüpfungen erstellen lassen. Letzteres ist etwa für Installationsroutinen wichtig: Man könnte in einem Batch-Skript Funktionen unterbringen, die ihrerseits einige neue (versteckte) Batch-Skripte einrichten sowie etwa auf dem Desktop Verknüpfungen zu diesen; da Batch-Dateien keine alternativen Symbole zuweisbar sind, Verknüpfungen (LNKs) jedoch schon, bieten sich letztere an, damit das Auge Futter bekommt. GUI-Umsetzungen tragen gleichfalls zu einer geschmeidigen Optik bei und verschaffen den üblicherweise visuell nüchternen

Batch-Programmen mehr Professionalität; wer möchte nicht gern als Batch-Entwickler einmal ein schickes Pop-up-Fenster öffnen?
Je nachdem, welchen der **GUI-Buttons** der Nutzer anklickt, löst er eine bestimmte definierte Aktion aus (über das sogenannte %errorlevel% geregelt, mehr Infos hierzu siehe Tipps-Fotostrecke). Deinstallations-Routinen mithilfe von Batch sind ebenfalls möglich; anhand von Zustands-Abfragen, ob bestimmte Dateien oder Registry-Einträge angelegt sind oder nicht, erfährt ein Batch-Tool (das Installer und Uninstaller zugleich sein kann), ob es dem Nutzer eine Installation oder eine Deinstallation anbieten soll.
Eine eher unbefriedigende Antwort auf die Frage "Batch mit GUI?" sind Empfehlungen für Programme, die eine GUI besitzen und mit denen Sie Kommandozeilen-Skripte zusammenklicken (AutoStarter X4) oder sie (erleichtert) coden (Notepad++). Häufig kennen Nutzer solche Software bereits und wenn sie etwa nach **"GUI" und "Batch"** per Suchmaschine suchen, wollen sie **per Skript-Sprache grafische Fenster umsetzen**. Das ist per Kommandozeile mit nervigen Klimmzügen verbunden, aber nach Einarbeitung halbwegs eingängig möglich (**Details in der Fotostrecke ganz oben**).



Mithilfe des Windows Script Host erzeugt die Kommandozeile Pop-up-Fenster. Die Gestaltung Letzterer ist nicht frei änderbar, aber in einem gewissen Rahmen trotzdem wählbar.

Batch-Umgebungsvariablen

Umgebungsvariablen sind Alternativen oder Ergänzungen zu Pfaden: Soll eine Batch-Datei am eigenen PC zum Einsatz kommen, ist es okay, als Befehl zum Beispiel *move C:\Users\Sebastian\Desktop\Text.txt F:* zu verwenden. Auf einem anderen Gerät würde der Befehl nicht funktionieren, da das Benutzerkonto und in der Folge der Benutzerordner ziemlich sicher nicht "Sebastian" heißt; obgleich auch hier eine Datei namens Text.txt auf dem Desktop gespeichert sein könnte, die im Beispiel auf Laufwerk F:\ (beim Autor eine RAM-Disk) verschoben wird. Es sind also häufig universelle Pfade erforderlich, die unabhängig der individuellen Windows-Konfiguration funktionieren. Unter anderem sind die folgenden Umgebungsvariablen wichtig, die Sie in Pfade einbauen und außerdem alleinstehend verwenden können:
%userprofile% führt zu C:\Users\<Benutzername>
%username% zeigt den Benutzernamen an

%windir% zeigt auf C:\Windows
%systemroot% zeigt auf C:\Windows (%systemroot%\System32 wäre demnach C:\Windows\System32)
%temp% zeigt auf C:\Users\<Benutzername>\AppData\Local\Temp
Bei *%userprofile%* etwa könnten Sie ein *explorer* voranstellen, sodass sich der Benutzerkonten-Ordner im Windows Explorer öffnet:
explorer %userprofile%
Wollen Sie etwa den Benutzernamen ohne Fehlermeldung anzeigen, reicht dafür
@echo off
%username%
pause
als Dreizeiler allein nicht aus. Nötig ist es, hier *echo* voranzustellen:
@echo off
echo %username%
pause
Dank des *echo* interpretiert Windows den Inhalt der Umgebungsvariable nicht als Befehl, der nicht ausführbar wäre (daher rührt die Fehlermeldung); es kommt dank *echo* zur reinen Textausgabe. Wer mag, kleidet das in einen Satz ein:
@echo off
echo Guten Tag, ich weiss, dass du %username% heisst.
pause

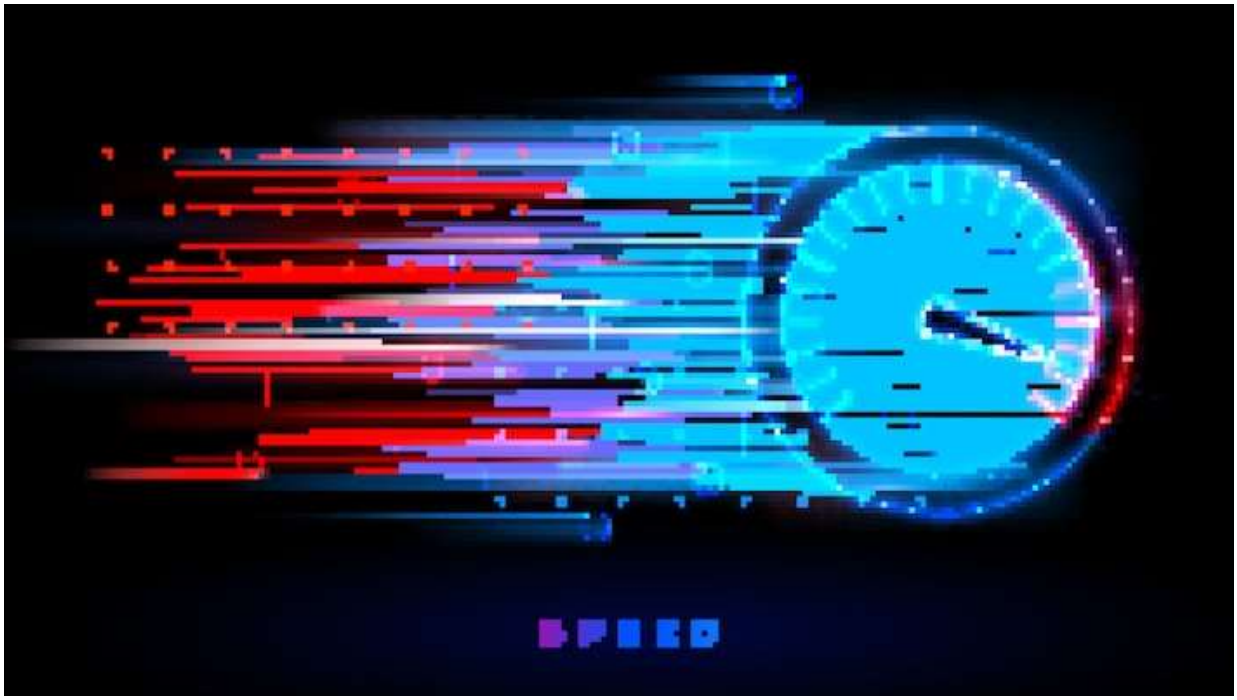
Batch-Sprungmarken

Ein Batch-Prögrämmchen führt nicht immer dieselben Schritte aus, sondern durchaus auch abhängig von einer bestimmten Gegebenheit. Beispielsweise fragen Sie vom Nutzer Ihres CMD-Programms eine Ziffer ab. Per *echo* könnten Sie zunächst erläutern, welche Einstelländerungen an Windows Ihr CMD-Skript offeriert. Per Nummerneingabe wählt der Anwender eine Variante davon aus. Für erste Tests bieten sich simple echo-Textnachrichten an, die an Windows nichts kaputtmachen. Ein Beispiel, bei dem es abhängig von der Eingabe einer 1 oder 2 mit dem Bereich :1 oder :2 weitergeht:

```
@echo off
set /p eingabe= Gib eine Zahl ein:
IF %eingabe% EQU 1 GOTO diesdas
IF %eingabe% EQU 2 GOTO ananas
:diesdas
echo Du hast eine 1 eingegeben, hier kann ein Text per echo erscheinen, aber auch irgendein anderer Befehl zum Einsatz kommen.
pause
exit
:ananas
echo Die Sonne lacht.
pause
```

Das *pause* bewirkt auch hier, dass der echo-seitige Text nicht verschwindet; der Nutzer ist angehalten, eine beliebige Taste zu drücken, damit der nächste Befehl (*exit*) in Kraft tritt. Beim oberen Sprungmarken-Block :1 ist übrigens das *exit* am Ende wichtig: So beendet sich die Kommandozeile nach dem Drücken einer beliebigen Taste. Würde hier das *exit* fehlen, würde die Kommandozeile bei der Nutzereingabe [1] erst Block :1 abarbeiten und danach (unerwünscht) zum Block :2 übergehen und schönes Wetter attestieren. Da es sich um ein Entweder-oder-Programm handeln soll, wäre dieses Verhalten blöd. Ein *exit* beim Block :2 ist

wiederum überflüssig, da nach dem *pause* ohnehin keine (Sprungmarken-)Befehle mehr folgen und sich die Kommandozeile auch so beendet. Vor Sprungmarken ist generell ein Doppelpunkt zu setzen.



Programme in Endlosschleife starten mit Sprungmarken

Eher als Aprilscherz als ernsthaft einsetzbar, sind Programmaufrufe, die wieder und wieder und wieder ... geschehen. Es ist keine Nutzereingabe erforderlich, die Kommandozeile springt ohne timeout-Verzögerung immer wieder zum Anfang zurück (und ruft hier den Internet Explorer iexplore.exe vielfach auf):

:IrgendeinWortMitPunktVorangestellt
start iexplore.exe
goto IrgendeinWortMitPunktVorangestellt

Ein weiteres Beispiel:

:wasWeissDennIch
start winver.exe
goto wasWeissDennIch

Das *goto* bewirkt, dass die Kommandozeile zurück zum zugehörigen Befehl oben geht, dem ein Doppelpunkt vorauseilt; es handelt sich dabei um eine Sprungmarke. In der Folge des Sprungs arbeitet Windows mit seiner Kommandozeile in einer Endlosschleife.

Klicken Sie im Kommandozeilenfenster, das die vielen Programmstarts initiiert, um zu erreichen, dass Windows keine weiteren Instanzen mehr öffnet; die schon geladenen bleiben im RAM. Zur Speicherbereinigung beenden Sie die Instanzen eventuell: Drücken Sie Windows-R und geben Sie etwa *taskkill /f /im iexplore.exe* ein, um den Internet Explorer zu terminieren. Zum Abbrechen der zahlreichen CMD-seitigen Aufrufe drücken Sie notfalls den PC-Ein-/Ausschaltknopf – um den Computer abstürzen zu lassen (danach löschen Sie das heijkle Batch-Skript auf Wunsch).

Batch startet Programme nicht nur endlos oft erneut, sondern ebenso mit einem bestimmten Datum. So sehen Sie zum Beispiel den VLC Media Player stets mit einer Weihnachtsmann-Mütze. Er zeigt eine solche vom 20. bis 31. Dezember an. Wer das Windows-Datum nicht manuell per Einstelldialog (GUI) oder administrativer CMD/Shell (etwa über *date 20.12.2020*)

verstellen möchte, erledigt die Schummelei per Befehl respektive einem Doppelklick auf eine entsprechende Batch-Datei. Im illegalen Kontext ließen sich so einige Zeit-Beschränkungen von Testversionen überwinden; man würde durch den Datum-Trick einer bald ablaufenden Testversion vorgaukeln, es sei noch mehr Testzeit vorhanden als aktuell tatsächlich besteht. Details zum VLC-Easter-Egg finden Sie im Artikel "VLC Media Player: Weihnachtsmütze das ganze Jahr sehen".

Batch: Größer als, kleiner als, gleich

Die Kommandozeile bewertet auf Wunsch, wie groß eine Zahl ist. Hier ein Skript-Code-Beispiel für ein Tool, das abfragt, wie alt der Nutzer ist. Je nachdem, ob die Antwort **mindestens** 18 lautet, sorgt **GEQ** dafür, dass "Volljaehrig" ausgegeben wird. Bei einem Wert bis einschließlich 17 sieht der Nutzer die Meldung "Noch ein Kind":

```
@echo off
echo Bitte eigenes Alter angeben:
set /p eingabe=
if %eingabe% GEQ 18 (echo Volljaehrig) else (echo Noch ein Kind)
pause
```

So etwa funktioniert eine **"kleiner als"-Abfrage per LSS**:

```
@echo off
echo Bitte eigenes Alter angeben:
set /p eingabe=
if %eingabe% LSS 18 (echo Alter niedriger als 18) else (echo Mindestens 18)
pause
```

Etwa wie folgt gibt die Kommandozeile aus, ob der eingegebene **Wert exakt dem definierten** (hier 18) **entspricht**, und zwar per **EQU**:

```
@echo off
echo Bitte eigenes Alter angeben:
set /p eingabe=
if %eingabe% EQU 18 (echo Genau 18) else (echo Irgendwas, aber keine 18)
pause
```

Ungleichheit prüft **NEQ**:

```
@echo off
echo Bitte eigenes Alter angeben:
set /p eingabe=
if %eingabe% NEQ 18 (echo Ungleich zu 18) else (echo Es wurde genau 18 eingegeben)
pause
```

Programmier-Ideen für Batch

Sprungmarken sind auch im Datums-Kontext möglich: Wenn ein bestimmtes Datum ist, soll die Kommandozeile zu diesem Abschnitt springen und etwa einen Spruch anzeigen. Wer mag, programmiert auf diese Weise einen Adventskalender in Batch (siehe Artikel "Windows 7/8/10: Adventskalender programmieren – in Batch"). Im verlinkten Artikel finden Sie auch den nötigen Code, um per Batch das aktuelle Datum und die Uhrzeit auszulesen und anzuzeigen. Ein Lese-Tipp für Windows-Enthusiasten ist der Ratgeber "Windows 7/8/10: Wo ist die Registry gespeichert? Wie groß ist sie?"; mit dem dort veröffentlichten Code bauen Sie einen Registry-Hives-Analysierer. Registry-Hives sind einige Dateien im Dateisystem, die ranghohen Registry-Schlüsseln entsprechen, worin Windows seine Einstellungen speichert; mit dem

Batch-Code ermittelt Windows die Größe seiner verstreuten Hive-Dateien in Byte, rechnet sie in Megabyte (MB) um und zeigt die beiden Werte (Byte, MB) Copy-&-Paste-fertig in einer Kommandozeile an.
Cool ist es, eigene Tuning-Tools zu schreiben: Wie das funktionieren kann, steht im Ratgeber "Registry-Cleaner programmieren: RunMRU-Cleaner in wenigen Minuten bauen". Das resultierende Batch-Tool ist recht simpel gestrickt, aber es ist schön, dass so etwas geht. Es bereinigt durch das Löschen eines Registry-Schlüssels die Chronik des Ausführen-/Win-R-Dialogs von Windows und verbessert so den Datenschutz.

Batch in EXE umwandeln

Abschließend noch ein Hinweis zum Programm Bat to Exe Converter: Das Tool ist interessant, weil es aus BAT- oder CMD-Dateien ausführbare EXE-Dateien macht. Hierzu kompiliert es sie. Die Nutzung des Konverters hat den Vorteil, dass Sie den Quellcode Ihrer Batch-Dateien schützen; normalerweise lassen sie sich im BAT-/CMD-Format ja recht unkompliziert per Windows-Editor einsehen. Das Programm erschwert es durch seine Umwandlungen aber auch, dass ein Batch-Entwickler (später) in seinen eigenen Quellcode Einsicht nimmt; ferner ändern Sie ihn aufgrund des EXE-Formats nicht mehr einfach so, weshalb Sie die originalen Dateien stets beibehalten sollten. Ein weiterer Nachteil: Zwar wirkt das EXE-Dateiformat professioneller als eine BAT-/CMD-Datei, da jeder Nutzer in letztere hineinsehen kann. Doch oft beschweren sich bei den resultierenden EXE-Dateien zahlreiche Virenscanner: Laden Sie eine vom BAT to EXE Converter erstellte Datei bei VirusTotal hoch, hagelt es (offensichtlich) Fehlalarme. Beim Artikel-Autor blockierte das installierte Schutzprogramm "F-Secure SAFE" mit seinem DeepGuard zwei erzeugte EXE-Files. Fazit: Bat to Exe Converer ist ein nettes Spielzeug, eignet sich aber kaum für EXE-Dateien außerhalb der eigenen vier Wände. Batch-Dateien sollen ja gerade andere Nutzer unterstützen; sind das aber unbedarfte Personen, die die Hilfe eines Skriptes für systemnahe Eingriffe nötig haben, sind sie von einer Viruswarnung eher verunsichert. Meiden Sie daher den Bat to Exe Converter.

Quelle: https://www.computerbild.de/artikel/cb-Tipps-Software-Batch-Dateien-erstellen-22564713.html