Hoofdstuk 10

Geavanceerde batchbestanden

In dit hoofdstuk

> Het juiste gereedschap
> Geavanceerde technieken met omleiden en doorsluizen
> Vernieuwingen aan de omgeving
> Data en tijden met batchbestanden
> Een potpourri van batchbestanden
> Batchbestanden en Windows 3.1

Het juiste gereedschap

In DOS zitten meer dan 100 opdrachten, waarvan sommige met vijf schakelopties en meer. Het is dus niet te verwonderen dat er maar weinig gebruikers zijn die alle mogelijkheden benutten. Een batchbestand is het uitgelezen hulpmiddel om veel van de krachtige functies in DOS toe te passen.

AUTOEXEC.BAT, de moeder van alle batchbestanden, is een sprekend voorbeeld van hoe DOS te vereenvoudigen is. Stelt u zich eens voor dat u bij het starten van de computer alle opdrachten zelf zou moeten ingeven. De aardigheid zou er gauw van af zijn. Toch worden nog veel opdrachten door gebruikers uitgevoerd die net zo goed en veel beter door een batchbestand kunnen worden gedaan.

Niemand zal de procedure in een batchbestand vergelijken met een programmeertaal, zoals C of Pascal. Wat DOS biedt voor een batchbestand, stelt in feite niet veel voor. Toch kunt u met een beetje fantasie en handigheid van batchbestanden krachtige en zeer nuttige utility's maken, die veel werk uit handen kunnen nemen.

Een overzicht van de opdrachten in batchbestanden

In tabel 10-1 staat een alfabetische lijst met alle opdrachten in DOS 6 voor batchbestanden en een korte omschrijving van elke opdracht. Naast de standaard opdrachten worden er vaak enkele sleutelwoorden en -opdrachten gebruikt in batchbestanden (zie tabel 10-2).

Tabel 10-1 Overzicht opdrachten voor batchbestanden	
Opdracht	Beschrijving
@	Onderdrukt de weergave van de rest van de gegevens op de regel.
:LABEL	Geeft een lokatie in een batchbestand aan waar de verwerking naar kan worden geleid met behulp van de opdracht GOTO.
CALL	Roept een tweede batchbestand op. Wanneer het tweede bestand is uitgevoerd, wordt de rest van de andere bestand uitgevoerd.
CHOICE	Laat een prompt zien en wacht totdat er een toets wordt ingedrukt.
ECHO	Laat een reeks zien en wisselt tussen weergave en onderdrukken.
FOR	Voert een opdracht meerdere keren uit en voegt na elke uitvoering een nieuwe reeks of bestandsnaam toe.
GOTO	Leidt de opdrachtverwerking naar een bepaalde lokatie in een batchbestand.
IF	Zorgt voor een voorwaarde in de uitvoering van een batchbestand. Kan worden gebruikt voor het vergelijken van twee reeksen, het testen van de ERRORLEVEL die door een programma wordt geretourneerd, of het controleren of een bestand bestaat.
PAUSE	Stopt de uitvoering van een batchbestand en wacht tot op een toets wordt gedrukt.
REM	REM, de afkorting van remark, voegt commentaar toe aan een batchbestand (en aan CONFIG.SYS). Elke regel die begint met REM, wordt genegeerd.
SHIFT	Geeft toegang tot meer dan negen parameters door de eerste te laten vallen en alle andere parameters een plaats naar links op te schuiven.

Tabel 10-2 Andere hulpmiddelen bij batchbestanden	
Item	Omschrijving
FIND	Zoekt of gespecificeerde tekst aanwezig of afwezig is.

SORT	Sorteert ASCII-gegevens en zendt de gesorteerde uitvoer naar een apparaat of bestand.
%0..%9	Variabelen die op de opdrachtregel worden ingevoerd als het batchbestand wordt uitgevoerd. %0 is de naam van een batchbestand, %1 is de eerste parameter, %2 de tweede parameter, enzovoorts.
%ENVVAR%	Een variabele waarvan de waarde gelijk is aan de inhoud van de omgevingsvariabele met de naam *ENVVAR*. In een batchbestand heeft %PATH% de waarde van een volledig pad en %COMSPEC% is het pad en de bestandsnaam van COMMAND.COM.
ERRORLEVEL	Een waarde tussen 0 en 255 die door een programma wordt geretourneerd en die goed of fout aangeeft. U moet met IF de ERRORLEVELs in omgekeerde volgorde testen.

Eindelijk een keuzemogelijkheid

Ik zal niet stil blijven staan bij de gewone toepassing van batchopdrachten; deze worden uitgebreid beschreven in deel 4. Er is echter een nieuwe opdracht die wel enige aandacht verdient.

We hebben tien jaar moeten wachten op een opdracht in een batchbestand waarbij om invoer door de gebruiker wordt gevraagd. Het is een externe opdracht, CHOICE genaamd, die niet groter is dan 2000 bytes. Het is een raadsel waarom wij hier zolang op hebben moeten wachten, temeer omdat de helft van dit bestand uit helptekst bestaat.

De belangrijkste mogelijkheden van CHOICE zijn:

- CHOICE laat een optioneel bericht zien en wacht tot de gebruiker op een toets drukt.
- Standaard wacht CHOICE tot er op Y of N is gedrukt, maar u kunt ook een serie geldige toetsen definiëren.
- U kunt onderscheid instellen tussen hoofdletters en kleine letters, maar standaard zijn beide toegestaan.
- Een ERRORLEVEL wordt ingesteld om aan te geven welke toets er is ingedrukt. Normaal volgen er op een CHOICE één of meerdere IF's.

- U kunt een wachttijd opgeven van 0 tot 99 seconden en een standaardtoets. Nadat de wachttijd is verlopen, stopt CHOICE en stelt de ERRORLEVEL overeenkomstig de standaardtoets in. In tegenstelling tot de meeste andere opdrachten stelt CHOICE een ERRORLEVEL van 0 in om een fout aan te geven (bijvoorbeeld een syntaxisfout in de opdracht of een Ctrl-Break).

Ondanks dat CHOICE een zeer nuttige opdracht is, zijn er ook enkele minpunten aan te ontdekken:

- Niet alle toetsen worden door CHOICE ondersteund. Terwijl wel toetsen als ;, [, en * kunnen worden gebruikt, worden de spatiebalk, functietoetsen en toetscombinaties zoals Alt-X niet ondersteund.

- CHOICE kan niet makkelijk worden ingesteld, zodat alle toetsen worden geaccepteerd.

- CHOICE ondersteunt alleen enkele toetsen. Een woord of een tekstreeks kan niet worden ingevoerd.

Het volgende bestand illustreert de toepassing van CHOICE waardoor de gebruiker om invoer wordt gevraagd, in dit geval een keuze te maken uit een aanwijsapparaat:

```
1 @echo off
2 choice /c:mtn /t:n,10 Wilt u de
Muis, het Tableau of Niets laden
3 if errorlevel 3 goto end
4 if errorlevel 2 goto Tablet
5 mouse /y
6 goto end
7 :Tablet
8 tabk /c:345
9 :end
```

In dit hoofdstuk worden de regels in een batchbestand genummerd. Deze nummers horen niet bij het batchbestand; zij worden als referentie in de tekst gebruikt. Voeg geen regelnummers toe als u batchbestanden maakt.

Door de schakeloptie /C:MTN op regel 2 accepteert CHOICE één van de drie letters M, T of N. De schakeloptie /S is niet gebruikt, zodat er geen onderscheid wordt gemaakt tussen hoofdletters en kleine letters.

De ERRORLEVELs die van CHOICE worden geretourneerd, worden in omgekeerde volgorde getest om te bepalen welke toets er is ingedrukt. Door de schakeloptie /C komen de ERRORLEVELs overeen met de volgorde waarin de toetsen zijn opgegeven. In dit geval wordt er een 3 geretourneerd als de gebruiker N indrukt, een 2 als de gebruiker T indrukt en een 1 als de gebruiker M indrukt.

De schakeloptie /T:N,10 op regel 2 laat CHOICE tien seconden wachten op invoer. Als er geen toets wordt ingedrukt, gaat CHOICE verder alsof de gebruiker N zou hebben ingedrukt.

CHOICE geeft de geldige toetsen automatisch in de opmaak [M,T,N]? weer. U kunt de prompt met de schakeloptie /N onderdrukken.

De prompt die de geldige tekens weergeeft, begint altijd direct na de tekst van het bericht. Als u tussen het bericht en de tekens een spatie wilt, moet u een spatie aan het einde van de CHOICE-opdrachtregel toevoegen.

Als u een slashteken (/) in de tekst wilt opnemen, moet u de tekst tussen aanhalingstekens plaatsen. Zonder aanhalingstekens zal het teken na het slashteken worden opgevat als een schakeloptie. Wanneer u een / gebruikt zonder aanhalingstekens, verschijnt één van de volgende berichten (afhankelijk van het teken achter de /):

```
Invalid switch on command line
```

of

```
Only one prompt string allowed
```

Gebruik geen doorsluisteken of filtertekens (|, <, >, >> en <<) in de CHOICE-tekst, ook niet als de tekst tussen aanhalingstekens staat.

Tips voor EDIT

De meeste batchbestanden zijn maar één of twee pagina's lang. EDIT, de tekstverwerker van DOS 6, is een handig programma voor het maken en bewerken van batchbestanden. Alhoewel in EDIT de normale conventies worden ondersteund voor het bewerken van bestanden, kunnen de nu volgende tips zeer waardevol voor u zijn.

Speciale ASCII-tekens typen. Voor een niet-standaard ASCII-teken, zoals een lijnteken voor het maken van kaders, moet u de ASCII-waarde invoeren van het teken. In appendix C van de DOS-handleiding staat een complete lijst met alle ASCII-tekens. U moet de volgende punten in het oog houden als u de ASCII-code van drie cijfers wilt invoeren:

1. Houd de Alt-toets ingedrukt.

2. Voer de drie cijfers op het numerieke gedeelte in.

3. Laat de Alt-toets los.

Het speciale ASCII-teken zal op de plaats van de cursor worden ingevoerd.

De tekstverwerker van QBASIC. Wanneer u de opdracht EDIT start, wordt eigenlijk ook de opdracht QBASIC met de schakeloptie /EDITOR gestart. Door de /EDITOR schakelt QBASIC over naar de bewerkmodus, waarin veel menu-opties worden verborgen die met QBASIC hebben te maken. Het is jammer dat daarbij ook enkele waardevolle functies worden uitgeschakeld. Werk met QBASIC in plaats van met EDIT om de twee volgende redenen:

- Met View/Split kunt u het scherm in tweeën splitsen. Zo kunt u het ene gedeelte van een bestand bekijken, terwijl u een ander deel aan het bewerken bent. Met F6 wisselt u tussen de twee vensters. Wanneer u View/Split voor een tweede keer selecteert, keert het programma weer terug naar een enkel scherm.

- Met F4 kunt u overschakelen naar het DOS-scherm als u QBASIC hebt gestart. Dit is erg handig als u een bug in een batchbestand probeert te herstellen en u wilt zien wat er fout gaat als u het batchbestand start.

Een heldere weergave. Op een kleurenmonitor kunt u de saaie kleuren veranderen. Selecteer Options/Display om de vensterkleuren aan te passen. Een prettige kleurstelling is gele tekst op een blauwe achtergrond. Tegen nieuwsgierige ogen helpt zwart op zwart; er is niets meer te zien.

Geen *.TXT als standaard meer. Het standaard bestandsmasker *.TXT in het dialoogvenster File Open is soms knap vervelend. Deze standaardreeks,

die staat in QBASIC.EXE, kunt u makkelijk met de opdracht DEBUG veranderen in *.BAT. Zie de paragraaf *Het standaard bestandsmasker van EDIT aanpassen* in hoofdstuk 11.

Controle over de cursor. Vergeet vooral de minder bekende sneltoetsen niet waarmee u snel door een batchbestand kunt gaan:

Toets	*Handeling*
Ctrl-Home	Naar begin bestand
Ctrl-End	Naar einde bestand
Ctrl-PgUp	80 tekens naar rechts
Ctrl-PgDn	80 tekens naar links
Ctrl-Pijl omhoog	Eén regel omhoog
Ctrl-Pijl omlaag	Eén regel omlaag
Ctrl-Pijl links	Eén woord naar links
Ctrl-Pijl rechts	Eén woord naar rechts
Ctrl-Enter	Naar begin volgende regel (zonder harde return)
Ctrl-Q,E	Naar bovenkant venster
Ctrl-Q,X	Naar onderkant venster

Technieken met omleiding en doorsluizen

De DOS-tekens voor het omleiden en doorsluizen spelen een belangrijke rol in geavanceerde batchbestanden. U kunt bijvoorbeeld tekens doorgeven aan opdrachten, tijdelijke bestanden maken en de uitvoer van opdrachten opzoeken of onderdrukken.

In tabel 10-3 staan de tekens voor omleiden en doorsluizen en waarvoor zij dienen.

DOS maakt achter de schermen een tijdelijk bestand wanneer er gegevens van de ene naar de andere opdracht worden doorgesluisd. De uitvoer van een opdracht wordt omgeleid naar een tijdelijk bestand, dit tijdelijke bestand wordt vervolgens

Tabel 10-3 Omleidings- en doorsluistekens van DOS 6	
Teken	Doel
>	Leidt de opdrachtuitvoer om naar een andere opdracht, ander apparaat of bestand. Wanneer de uitvoer wordt omgeleid naar een bestaand bestand, worden de oorspronkelijke gegevens gewist en wordt de nieuwe uitvoer aan het begin van het bestand geplaatst.
>>	Leidt de opdrachtuitvoer om naar een bestand maar plaatst de uitvoer aan het einde van het bestand zonder dat de oude gegevens worden verwijderd.
<	Leidt de inhoud van een bestand om als invoer naar een opdracht.
|	Sluist de uitvoer van de ene opdracht als uitvoer door naar een andere opdracht.

omgeleid als invoer naar een tweede opdracht waarna het tijdelijke bestand wordt verwijderd. DOS maakt het tijdelijke bestand in de directory die wordt gespecificeerd met de omgevingsvariabele TEMP. Als een dergelijke variabele niet bestaat, wordt het bestand in de huidige directory gemaakt.

In de volgende paragrafen worden enkele praktische toepassingen bekeken voor de omleidings- en doorsluisopdrachten.

Gewoon ja

Bij sommige DOS-opdrachten wordt de gebruiker om toestemming gevraagd voordat de opdracht wordt uitgevoerd. Wanneer u bijvoorbeeld de opdracht

```
del *.*
```

invoert, verschijnt het bericht:

```
Are you sure (Y/N)?
```

Dit bericht is een veiligheidsklep bij opdrachten die vanaf de opdrachtregel worden uitgevoerd. Het kan echter vervelend zijn als een batchbestand halverwege stopt en op antwoord wacht.

Door de uitvoer van ECHO door te sluizen naar een opdracht kunt u de opdracht laten geloven, dat er een toets is ingedrukt. In het volgende batchbestand wordt een voorbeeld van deze

techniek gegeven door middel van de opdracht DEL:

```
1 @echo off
2 echo y | del *.*
```

Wanneer u dit batchbestand maakt, zorg er dan voor dat u het vanuit een tijdelijke directory start. Vergeet niet dat elk bestand in de huidige directory wordt verwijderd, zonder dat u daarvoor wordt gewaarschuwd. U kunt de opdracht `echo y` ook als `echo n` ingeven.

Stilte graag

Met `@echo off` kan alle 'rommel' worden verborgen die normaal gesproken te zien is tijdens het uitvoeren van een batchbestand. De foutberichten worden hiermee niet onderdrukt; deze verschijnen indien dit nodig is. Als u bijvoorbeeld een bestand wilt herbenoemen en er gaat iets fout, ziet u het volgende bericht:

```
Invalid filename or file not found
```

Ook deze berichten zijn te verbergen. Met de volgende syntaxis sluist u de uitvoer van de opdracht door naar NUL:

```
opdracht > nul
```

De volgende opdrachten staan in het batchbestand RN.BAT waarmee bestanden kunnen worden herbenoemd:

```
...
5 if not exist %1 goto nosource
6 if exist %2 goto oldtarget
7 rename %1 %2 > nul
8 if not exist %2 goto failed
9 echo Bestand hernoemd!...
```

Berichten die op regel 7 worden gegenereerd, worden niet weergegeven, omdat zij worden omgeleid naar NUL. Zie ook de opdracht IF in deel 4 voor het volledige bestand RN.BAT.

Een batchbestand kan ook pauzeren zonder dat het standaardbericht `Press any key to continue` verschijnt. Met Echo genereert u een eigen bericht en de uitvoer van Pause leidt u om naar NUL:

```
1 @echo off
2 Echo Druk op een toets voor het
echte werk
3 pause > nul
```

Een andere manier om berichten te onderdrukken is de opdracht CTTY waarmee het toetsenbord en het scherm kunnen worden uitgezet. Eerst voert u de opdracht

```
ctty nul
```

uit, waarmee alle I/O (invoer/uitvoer) naar NUL wordt omgeleid. Met de opdracht

```
ctty con
```

worden de normale bewerkingen weer hervat. Het volgende (uitgeklede) batchbestand illustreert het principe van CTTY:

```
@echo off
ctty nul
.....
.....
ctty con
```

Op deze manier hoeft u niet elke opdracht om te leiden naar NUL. De opdracht CTTY schakelt het toetsenbord uit, hetgeen impliceert dat u er zeker van moet zijn dat het batchbestand werkt en alle fouten kunnen worden verwerkt. Dit batchbestand negeert Ctrl-Break en elke andere toets als CTTY is ingesteld op NUL.

De uitvoer van opdrachten selectief weergeven

In het voorafgaande gedeelte is besproken hoe de uitvoer van alle opdrachten kan worden onderdrukt door deze uitvoer naar NUL om te leiden. Niet altijd is het nodig alle uitvoer te onderdrukken. U kunt de uitvoer naar FIND doorsluizen en de uitvoer beperken naar een regel waarop een bepaalde tekst staat.

Deze techniek kan worden toegepast met de opdracht DIR om het aantal bestanden te laten zien en de ruimte die door een directory wordt ingenomen. Wanneer u DIR zonder schakelopties uitvoert, ziet u gegevens over de vaste schijf gevolgd door een lijst met alle bestanden en een samenvatting. Het wordt file(s) staat altijd op de regel waarop de totaalgegevens staan. U kunt alleen deze regel weergeven door met FIND te zoeken

naar "file(s)". In het batchbestand DIRSIZE.BAT wordt van deze benadering uitgegaan:

```
@echo off
cd
echo Directory-statistiek
dir | find "file(s)"
```

Het batchbestand CHKFRG.BAT gaat uit van hetzelfde principe om de uitvoer van de volgende opdracht te filteren, dat de fragmentatie per bestand vermeldt:

```
chkdsk bestandsmasker
```

Nadat de integriteit van de schijf is gecontroleerd, vermeldt CHKDSK de fragmentatie van elk afzonderlijk bestand of deelt mee dat er geen bestand is gefragmenteerd. In beide berichten is sprake van het woord `contiguous`, zodat dit kan worden gebruikt als onderwerp van de opdracht FIND in het volgende batchbestand CHKFRG.BAT:

```
@echo off
rem CHKFRG checkt op
file-fragmentatie
if "%1"=="" goto CHKALL
echo Bestand(en) %1 wordt
gecontroleerd...
chkdsk %1 | find "contiguous"
goto QUIT
:CHKALL
echo Alle bestanden worden
gecontroleerd
chkdsk *.* | find "contiguous"
:QUIT
```

Vernieuwde omgeving

De DOS-omgeving is een kleine opslagplaats waar de algemene instellingen van het systeem worden bewaard die door DOS en programma's worden gebruikt. De omgeving is ook een bron waaruit programmeurs van batchbestanden kunnen putten

Met de opdracht SET wordt de omgeving beheerd en bekeken. SET zonder schakelopties laat de volledige omgeving zien. Met de volgende syntaxis kunnen nieuwe gegevens aan een omgevingsvariabele worden toegekend:

```
set varnam=[reeks]
```

U kunt met de syntaxis *%envvar%* vanuit een batchbestand de waarde van een omgevingsvariabele op bijna dezelfde manier benaderen als een parameter met *%n*. Het volgende eenvoudige batchbestand geeft de waarde van de omgevingsvariabele PROMPT:

```
1 @echo off
2 echo %prompt%
```

Bij de batchtaal kunnen geen tijdelijke variabelen worden gemaakt, zoals dat bij de meeste programmeertalen wel mogelijk is. Met een handigheidje is dit toch te realiseren door gegevens met behulp van de omgeving op te slaan en terug te halen.

Wanneer het bericht verschijnt dat de omgeving vol is, kunt u deze omgeving groter maken met de schakeloptie /E bij COMMAND in een SHELL-opdracht. Zie voor meer informatie COMMAND in deel 4 *Alle DOS-opdrachten*.

Tijdelijk een ander zoekpad

Bij de opdracht PATH is het niet mogelijk met een schakeloptie een directory aan het zoekpad toe te voegen. Het pad is te wijzigen door PATH in AUTOEXEC.BAT aan te passen en het systeem te herstarten. Omdat het zoekpad echter in de omgeving ligt opgeslagen, wordt dit met het volgende batchbestand NEWPATH.BAT makkelijker gemaakt:

```
1 @echo off
2 path %1;%path%
```

Wanneer u een nieuwe directory wilt toevoegen aan het pad, hoeft u alleen dit batchbestand te starten met de nieuwe directory als parameter, bijvoorbeeld NEWPATH C:\WP. Het pad wordt ingesteld met deze parameter, gevolg door een puntkomma en het huidige pad (%path%).

U kunt het batchbestand verfijnen met de omleidingstekens > en >> waarmee een ander batchbestand OLDPATH.BAT wordt gemaakt, dat het originele pad terughaalt:

```
1  @echo off
2  if "%1"=="" goto HELP
3  if "%1"=="/?" goto HELP
4  if exist c:\oldpath.bat goto overwrite
5  :proceed
6  echo @echo off  c:\oldpath.bat
```

```
7  path > c:\oldpath.bat
8  echo echo Het oude pad is hersteld. >
c:\oldpath.bat
9  echo echo OLDPATH.BAT is verwijderd
(volgend bericht negeren) > c:\oldpath.bat
10 echo del c:\oldpath.bat > c:\oldpath.bat
11 echo Geef OLDPATH om het originele pad in
te stellen
12 goto changepath
13 :overwrite
14 choice Overwrite C:\OLDPATH.BAT
15 if errorlevel 2 goto changepath
16 goto proceed
17 :CHANGEPATH
18 path %1;%path%
19 goto quit
20 :HELP
21 echo Voegt een directory aan het pad toe
en creëert het batchbestand OLDPATH.
22 echo SYNTAXIS:      %1 pathdir
23 :quit
```

Op de tweede en derde regel wordt gecontroleerd of er een parameter, of anders de parameter /? is doorgegeven. In beide gevallen gaat het programma naar regel 21 en laat het een bericht zien.

Het batchbestand NEWPATH maakt een nieuw batchbestand, OLDPATH.BAT genaamd, in de hoofddirectory van station C. Op regel 4 wordt met een IF-opdracht vastgesteld of het bestand al bestaat. Als dat zo is, wordt er door CHOICE gevraagd op regel 14 of het oude bestand mag worden overschreven. Het omleidingsteken > op regel 6 zorgt voor een nieuw batchbestand (of overschrijft het bestaande bestand) met als inhoud de regel `@echo off`. Het omleidingsteken >> op de regels 7 tot en met 10 voegt nog eens vier regels toe aan het batchbestand. Als u het tijdelijke batchbestand OLDPATH start, haalt het pad de oorspronkelijke instelling terug en verwijdert het batchbestand zichzelf.

Als laatste wordt regel 18 uitgevoerd om de variabele %1 aan het begin van het pad te plaatsen.

Het bericht dat het batchbestand ontbreekt, verschijnt als u OLDPATH start. Deze mededeling wordt gedaan omdat het batchbestand zichzelf verwijdert terwijl DOS het bestand uitvoert.

De variabele TEMP beheren

Steeds meer programma's, waaronder ook Windows, bepalen met behulp van de

omgevingsvariabele TEMP welke tijdelijke bestanden er moeten worden gemaakt. Met het volgende batchbestand wijst u een directory aan TEMP toe, of verandert u een bestaande directory:

```
@echo off
if %temp%"=="" goto empty
choice Temp staat nu op %temp%. Overschrijven?
if errorlevel 2 goto quit
:EMPTY
set temp=%1
echo The TEMP staat nu op %temp%
:QUIT
```

Hoofdletters

Wanneer u met de syntaxis `%envvar%` een omgevingsvariabele vanuit een batchbestand benadert, maakt DOS geen onderscheid tussen hoofdletters en kleine letters, als er wordt bepaald of een variabele bestaat.

Het volgende fragment van een batchbestand illustreert de traditionele techniek om te controleren op zowel hoofdletters als kleine letters:

```
....
4 if "%1"=="ROOD" goto rood
5 if "%1"=="rood" goto rood
....
```

Wanneer u rood echter als Rood invoert, wordt dit niet door het batchbestand herkend. Bij een kort woord als rood zou er op elke mogelijke combinatie kunnen worden gecontroleerd, maar bij langere woorden is dit ondoenlijk.

Als u met een parameter een omgevingsvariabele maakt, kunt u met behulp van de syntaxis *%envvar%* in een IF-opdracht een test uitvoeren zonder onderscheid tussen hoofdletters en kleine letters. In het volgende batchbestand wordt deze techniek toegepast:

```
....
4 set $$%1=slim
5 if "%$$rood%"=="slim" goto ROOD:
6 if "%$$blauw%"=="slim" goto BLAUW:
....
20 set %1=
....
```

De parameter wordt voorafgegaan door $$ voordat deze wordt toegevoegd aan de omgeving. Zo is de

kans zeer klein dat een bestaande omgevingsvariabele wordt overschreven. Ik ben nog geen variabele tegengekomen die normaal gesproken met $$ begint.

In dit voorbeeld is de tekst *slim* toegewezen aan de omgevingsvariabele; dezelfde tekst wordt in de IF-opdrachten gebruikt. U kunt elke willekeurige tekst hiervoor gebruiken, maar houd deze wel kort om zo min mogelijk omgevingsruimte te gebruiken.

De opdracht IF op regel 5 onderschept een parameter rood, ongeacht of deze wordt geschreven als ROOD, Rood, RooD, enzovoorts.

Het enige nadeel van deze techniek is, dat de omgeving wordt gebruikt.

Een alternatief is de parameter weer te geven met echo en door te sluizen naar de opdracht FIND. Door de schakeloptie /I bij FIND kunt u zoeken zonder onderscheid tussen hoofdletters en kleine letters. Ter illustratie een voorbeeld:

```
echo %1 | find "rood" /i
```

Deze opdracht moet hier en daar worden bijgesteld. Wanneer FIND de tekst vindt, wordt deze naar het scherm geschreven, maar dat is niet nodig bij dit voorbeeld. Als FIND de tekst niet vindt, wordt er een ERRORLEVEL (niet nul) ingesteld. Door de uitvoer van FIND naar NUL door te sluizen en te testen op een ERRORLEVEL die ongelijk is aan nul, kunt u zonder onderscheid testen:

```
echo %1 | find "rood" /i > nul
if not errorlevel 1 echo U hebt ROOD
ingevoerd!
```

Er is nog een aanpassing nodig. FIND zoekt de tekens rood overal in de invoerparameter op. De vorige test reageert op dezelfde manier op rood als op donkerrood, of op elke andere tekst met het woord rood. Door de zoektekst tussen twee tekens te plaatsen bent u er zeker van dat er naar een volledige overeenkomst wordt gezocht. U kunt de tekstparameter in het voorbeeld tussen dollartekens ($) plaatsen:

```
echo $%1$ | find "$rood$" /i > nul
if not errorlevel 1 echo U hebt ROOD
ingevoerd!
```

Datum, tijd en batchbestanden

Zo op het eerste gezicht lijkt het erop alsof taken met betrekking tot data en tijden niet worden ondersteund, maar dat is maar schijn. Er is een aantal technieken beschikbaar waarmee dergelijke taken wel kunnen worden uitgevoerd.

Ogenblikje geduld, a.u.b.

In veel programma's wordt een ogenblik gepauzeerd om een bericht of een logo te laten zien, waarna het eigenlijke programma wordt gestart. Met CHOICE kunt u in een batchbestand het programma een bepaalde tijd laten stoppen.

De schakeloptie /N onderdrukt de prompt `[Y/N]` en /T specificeert de standaardtoets en de tijd die wordt gewacht. Met deze twee schakelopties onderdrukt u alle uitvoer van CHOICE en laat u het bestand een aantal seconden wachten. De volgende opdracht wacht vijf seconden en gaat daarna verder met de rest van het bestand.

```
choice /c:a /n /t:a,5
```

Het enige nadeel hiervan is dat het programma blijft wachten als de gebruiker een ongeldige toets indrukt. Dit probleem kunt u verhelpen door met de schakeloptie /C alle standaardtoetsen te definiëren. Als er dan een toets wordt ingedrukt, zal CHOICE alleen sneller beginnen, wat door de meesten niet als een nadeel wordt gezien.

Wanneer er op een toets wordt gedrukt die niet in de lijst staat (zoals de spatiebalk of Enter), wacht het batchbestand totdat er op een geldige toets of op Ctrl-Break wordt gedrukt.

Hoe laat is het?

Zelfs voor een eenvoudig iets als het opvragen van de tijd, zijn enkele trucs nodig. De systeemtijd wordt met de opdracht TIME ingesteld. Deze opdracht laat de tijd zien zoals in het volgende voorbeeld:

```
Current time is 12:00:11.86a
Enter new time:
```

TIME laat de huidige tijd zien, maar wacht totdat er een nieuwe tijd wordt ingevoerd of op Enter wordt gedrukt. Omdat de ECHO-opdrachten altijd

om een harde return vragen, kunt u met ECHO een Enter aan de opdracht TIME toevoegen:

```
echo. | time
```

De opdracht `echo.` laat ECHO een lege regel schrijven en vervolgens een harde return uitvoeren. Deze harde return wordt naar TIME doorgesluisd, waardoor het programma ervan uitgaat dat er op Enter is gedrukt.

De herziene opdracht TIME geeft nog steeds twee regels met tekst weer. Deze tweede regel kunt u met FIND uitfilteren. Door het woord Current te zoeken wordt alleen de eerste regel weergegeven. De complete expressie voor het weergeven van de huidige tijd is als volgt:

```
echo.| time | find "Current"
```

Na het invoeren van deze opdracht wordt de huidige tijd op de volgende manier weergegeven:

```
Current time is 12:00:11.15a
```

Op een bepaalde tijd wachten

Met de hiervoor besproken techniek kunt u met de opdracht TIME een batchbestand onderbreken en laten wachten op een bepaalde tijd. Het volgende batchbestand WAITFOR.BAT wacht op een bepaalde tijd en voert dan één of meer opdrachten uit:

```
1  @echo off
2  if "%1"=="" goto help
3  if "%1"=="/?" goto help
4  set $$$$TIME=is %1
5  echo We wachten tot %1
6  :LOOP
7  echo.|time|find "%$$$$TIME%" > nul
8  if errorlevel 0 if not errorlevel 1 goto
ontime
9  goto loop
10 :ONTIME
11 set $$$$TIME=
12 rem Voeg hier de gewenste commando's in:
13 echo Alarm Alarm Alarm!!!
14 goto quit
15 :HELP
16 echo Pauzeert een batchbestand tot een
bepaalde tijd is bereikt
17 echo SYNTAXIS %0 HH:MM[:SS], bijv. %0 11:31
18 :QUIT
```

Aan het batchbestand wordt een enkele parameter doorgegeven die de begintijd aangeeft. Deze tijd kan gelijk aan de opmaak van TIME worden doorgegeven: uren:minuten:seconden(optioneel).

Op regel 4 wordt een tijdelijke omgevingsvariabele gemaakt `$$$$TIME`, die wordt ingesteld als `"is "` gevolgd door de tijd die door de gebruiker wordt ingevoerd. Met het voorvoegsel `"is "` wordt voorkomen dat FIND geen overeenkomst vindt als de minuten en seconden toevallig overeenkomen met de opgegeven uren en minuten. Als bijvoorbeeld de huidige tijd 10:20 zou zijn, zal een overeenkomst voor `is 10:20` niet worden gevonden:

```
Current time is 9:10:20.04a
```

Een zoekopdracht naar `10:20` zou een overeenkomst hebben opgeleverd waar niet om wordt gevraagd.

Op regel 7 wordt de uitvoer van TIME doorgegeven aan FIND, die de omgevingsvariabele zoekt. Deze test wordt steeds herhaald totdat FIND een ERRORLEVEL van 0 retourneert, waarmee wordt aangegeven dat het zo laat is. Het batchbestand springt vervolgens naar regel 11, de omgevingsvariabele wordt verwijderd en de opdrachten worden uitgevoerd die op die bepaalde tijd waren ingesteld.

Een potpourri van batchbestanden

Alle goede programmeurs houden een collectie (of *bibliotheek*) met routines bij in hun codeverzameling. In de volgende paragrafen worden er een aantal routines voor batchbestanden gegeven die u aan uw eigen verzameling kunt toevoegen.

Bestaat een directory?

Met een IF-opdracht kunt u heel makkelijk vaststellen of een bestand bestaat. Het volgende gedeelte van een batchbestand laat zien hoe IF controleert of het bestand CLOCK.WCH bestaat:

```
....
5 if exist CLOCK.WCH goto Nextstep
6 goto Error3
....
```

Voor een directory is het enigszins lastiger. Het volgende batchbestand genereert de tekst Found als er minstens één bestand in de directory staat (met uitzondering van de bestanden . en ..):

```
1 @echo off
2 if exist c:\empty\*.* echo Found
```

Als er geen bestanden in de directory staan, is het resultaat van de test negatief. Om te kunnen bepalen of een directory bestaat zonder dat er bestanden in staan, moet u met IF testen op Nul. De syntaxis hiervoor is als volgt:

if exist [*station***:]***padnaam***\nul**
opdracht

Het volgende batchbestand accepteert als parameter een enkel bestand, maakt een directory en schakelt vervolgens over naar de nieuwe directory:

```
1  @echo off
2  if "%1"=="" goto HELP
3  if exist %1\nul goto OK
4  md %1 > nul
5  if exist %1\nul goto OK
6  echo De directory kan niet worden gemaakt
7  goto quit
8  :help
9  echo Syntaxis is %0 drive:\newpath
10 goto QUIT
11 :OK
12 cd %1
13 :QUIT
```

Op regel 3 wordt bepaald of de directory bestaat door de NUL te testen. Als de directory inderdaad aanwezig is, springt het programma naar regel 12. Wanneer de directory niet bestaat, wordt de opdracht MD uitgevoerd en wordt de test herhaald. Bestaat de directory dan nog steeds niet, dan moet er een fout in de directory staan. Indien er geen problemen zijn, schakelt regel 12 over naar de nieuwe directory.

Op jokertekens controleren

De opdracht FOR is een gemakkelijke manier om te bepalen of er jokertekens in een bestandsnaam staan. Wanneer er een jokerteken wordt gebruikt in een opdracht FOR, wordt er een andere bestandsnaam gebruikt bij elke herhaling van de lus. Als de vervangen bestandsnaam in de FOR-lus exact gelijk is aan de naam die wordt getest, staan er in de geteste bestandsnaam geen jokertekens. Het

volgende batchbestand TESTWILD.BAT laat zien hoe dit in zijn werk gaat:

```
1 @echo off
2 if "%1"=="" goto NOTWILD
3 for %%W in (%1) do if %%W=%1 goto NOTWILD
4 :ISWILD
5 echo Jokerteken is ingevoerd
6 goto quit
7 :NOTWILD
8 echo Geen jokerteken ingevoerd
9 :Quit
```

De versie van DOS bepalen

Wanneer u een batchbestand maakt met de nieuwe DOS-opdrachten, wilt u misschien wel eerst controleren of de juiste DOS-versie wordt gebruikt. Een batchbestand met de nieuwe opdracht CHOICE werkt alleen onder DOS 6.

De opdracht FIND is in DOS 6 verbeterd en retourneert nu een ERRORLEVEL waarmee wordt aangegeven of een zoektekst wel of niet is gevonden. Een ERRORLEVEL van 1 wordt altijd geretourneerd wanneer de tekst niet wordt gevonden. Aangezien de opdracht FIND uit eerdere versies van DOS altijd een ERRORLEVEL van 0 (nul) retourneert, kunt u testen of de versie van DOS versie 6 is door tekst te zoeken die niet bestaat. Bij een ERRORLEVEL van 1 als resultaat weet u dat DOS 6 is geïnstalleerd.

Het volgende batchbestand ISDOS.BAT laat zien hoe USE met FIND kan worden gebruikt om niet-bestaande tekst op te zoeken om de versie van DOS te bepalen:

```
1 @echo off
2 ver | find "Sandra" > nul
3 if errorlevel 1 goto OK
4 echo Voor dit batchbestand is DOS
6 nodig
5 goto quit
6 :OK
7 echo DOS 6
8 :Quit
```

Op regel 2 wordt de uitvoer van de opdracht VER naar FIND doorgesluisd die zoekt naar de tekstreeks Sandra. (Ik had Sandra beloofd haar naam ergens in dit boek te vermelden!) U kunt elke tekst opzoeken die niet wordt gegenereerd door VER. Op regel 3 wordt de ERRORLEVEL getest

en als 1 wordt geretourneerd, moet DOS 6 zijn geïnstalleerd.

Op residente programma's controleren

In DOS 6 is de schakeloptie /M (or /MODULE) geïntroduceerd voor de opdracht MEM. Door deze schakeloptie vermeldt MEM de status van een bepaald programma dat in het geheugen staat. Wanneer het programma nog niet is geladen, wordt deze tekst weergeven:

programma `is not currently in memory`

Door de uitvoer van MEM /M naar FIND door te sluizen en te testen op de reeks `not currently` kan een batchbestand makkelijk controleren of een programma in het geheugen staat.

In het volgende batchbestand wordt deze techniek toegepast om te controleren of het opgegeven programma in het geheugen is geladen:

```
@echo off
if "%1"=="" goto help
if "%1"=="/?" goto help
mem /m %1|find "not currently" > nul
if errorlevel 1 goto loaded
:NOTLOADED
echo %1 is niet geladen
goto quit
:LOADED
echo %1 is al geladen
goto quit
:HELP
echo Checkt of een programma in het geheugen
staat
echo SYNTAXIS %0 prognaam, bijv. %0 share
:QUIT
```

Door middel van deze eenvoudige techniek kunt u op residente programma's controleren, waaronder DOSKEY, SHARE, EMM386 en zelfs Windows (zie de paragraaf *Batchbestanden en Windows 3.1* verderop in dit hoofdstuk).

Met de opdracht MEM /D /P kunt u de naam opvragen van het programma dat in het geheugen staat.

Batchbestanden en Windows 3.1

Veel gebruikers die met DOS 6 werken, werken ook met Windows 3.1. In deze paragrafen worden enkele voorwaarden besproken waaraan batchbestanden moeten voldoen om vanuit Windows 3.1 te kunnen worden gestart.

Instellingen van DOS-sessie besturen

Wanneer u een DOS-sessie start, meldt Windows normaal gesproken dat u onder Windows draait en geeft een beschrijving van enkele handige toetscombinaties, zoals Exit, Alt-Tab en Alt-Enter. U kunt dit standaardbericht onderdrukken door de volgende regel in het bestand SYSTEM.INI te bewerken:

```
[386Enh]
DosPromptExitInstruc=false
```

Door een omgevingsvariabele WINPMT toe te voegen voordat u Windows start, kunt u een speciale prompt definiëren die in elke DOS-sessie actief is. De volgende opdracht in AUTOEXEC.BAT definieert deze speciale prompt van Windows

```
set winpmt=We zitten in
Windows!$_$p$g
```

Standaard wijst Windows dezelfde omgevingsruimte toe als in CONFIG.SYS is ingesteld. Windows kan een grotere of kleinere omgeving toewijzen als u de opdracht CommandEnvSize= plaatst in de sectie [NonWindowsApp] van het bestand SYSTEM.INI. De volgende opdracht maakt een omgeving van 800 bytes in elke DOS-sessie:

```
[NonWindowsApp]
CommandEnvSize=800
```

WINSTART.BAT

Wanneer u WIN invoert, zoekt Windows naar het bestand WINSTART.BAT in de Windows-directory. Windows voert dit bestand uit alvorens over te schakelen naar de grafische modus en de shell van Windows te lanceren.

Het hoofddoel van dit bestand is residente programma's te laden die wel noodzakelijk zijn voor Windows, maar niet voor een DOS-sessie die vanuit Windows wordt gestart. Alle residente programma's die worden geladen voordat Windows wordt gestart, worden automatisch in elke DOS-sessie geïnstalleerd. Door residente programma's vanuit het batchbestand WINSTART.BAT te laden wordt er geheugen uitgespaard voor elke DOS-sessie. Er zijn maar weinig residente programma's die alleen voor Windows zijn ontworpen. Gewoonlijk hebben zij ook te maken met netwerken of communicatieprogramma's.

In het standaard batchbestand WINSTART.BAT kan elke opdracht staan. De volgende opdrachten laten een herinnering zien dat er een cartridge in station D moet zijn geplaatst, voordat er verder kan worden gewerkt:

```
@echo off
Echo Zorg dat een cartridge in
station D aanwezig is.
pause
```

U kunt met het bestand ook de systeembestanden WIN.INI en SYSTEM.INI kopiëren, zodat zij niet door het een of andere programma kunnen worden gewijzigd:

```
@echo off
echo Systeembestanden worden gekopieerd...
copy c:\windows\win.ini c:\windows\oldwin.ini
copy c:\windows\system.ini
c:\windows\oldsys.ini
echo Windows wordt geladen...
```

De nieuwe schakeloptie /K voor COMMAND

Wanneer u op het pictogram MS-DOS klikt, wordt het bestand DOSPRMPT.PIF gestart. Normaal gesproken start dit PIF-bestand COMMAND.COM voor een DOS-sessie.

In DOS 6 is de nieuwe schakeloptie /K voor COMMAND.COM geïntroduceerd waarmee u de naam van het batchbestand of programma kan opgeven dat wordt gestart als COMMAND.COM start. Met behulp van deze schakeloptie kan de DOS-sessie op een makkelijke manier worden aangepast. Het is zoiets als één speciaal batchbestand per DOS-sessie.

Start PIFEDIT om deze schakeloptie te kunnen benutten en laad het bestand DOSPRMPT.PIF dat in de Windows-directory staat. Typ op de regel Options Parameters `/K` gevolgd door een spatie en de naam van een batchbestand. Het batchbestand zal worden uitgevoerd wanneer een DOS-sessie wordt gestart.

Dit voorbeeld batchbestand verandert de prompt en laat zien hoeveel geheugen er nog beschikbaar is:

```
1 @echo off
2 prompt WINDOWS DRAAIT$_$P$G
3 mem | find "executable"
```

Op Windows 3.1 controleren

Veel opdrachten moeten niet vanuit Windows worden gestart, zoals CHKDSK en DEFRAG. U moet in DOS-sessies ook niet bepaalde residente programma's laden, bijvoorbeeld diskcaches en programma's voor het geheugenbeheer. Daarom kan het nodig zijn te weten of Windows is geladen (of het batchbestand wordt uitgevoerd vanuit een DOS-sessie) voordat één of meer opdrachten worden gestart.

Het is u waarschijnlijk opgevallen dat Windows in elke DOS-sessie automatisch een omgevingsvariabele *windir* maakt. Met deze variabele wordt de hoofddirectory van Windows opgeslagen. Daarbij heeft men waarschijnlijk iets over het hoofd gezien: Windows zet de naam van de omgevingsvariabele in *kleine letters*. Het opdracht-vertolkingsprogramma zet de naam van omgevingsvariabelen altijd in *hoofdletters*, zodat een batchbestand op geen enkele manier kan testen of *windir* aanwezig is. Handig.

Gelukkig kunt u door middel van MEM met de nieuwe schakeloptie /M makkelijk testen of Windows is geladen. Zie hiervoor ook de paragraaf *Op residente programma's controleren* eerder in dit hoofdstuk. Het volgende batchbestand test of Windows in het geheugen is geladen:

```
mem /m win | find "not currently" > nul
if errorlevel 1 goto WindLoad
echo Windows is niet geladen
goto quit
:WINLOAD
echo Windows is geladen
:QUIT
```

Samenvatting

In dit hoofdstuk wordt beschreven hoe met een beetje fantasie en vindingrijkheid batchbestanden allerlei taken kunnen uitvoeren. De volgende punten zijn hierbij aan de orde gekomen:

- Met de nieuwe opdracht CHOICE wordt de gebruiker om invoer gevraagd. Ook kan er met deze opdracht een standaardwaarde worden geretourneerd als er een bepaalde tijd is verstreken.
- Tekens voor het omleiden en doorsluizen bieden extra mogelijkheden bij het maken van batchbestanden en het overbrengen van gegevens van en naar programma's.
- De omgeving is een opslagplaats voor tijdelijke gegevens en kan worden gebruikt voor het valideren van gegevens zonder onderscheid tussen hoofdletters en kleine letters.
- Door de uitvoer van DATE en TIME naar FIND door te sluizen kunt u batchbestanden maken die op een bepaalde tijd worden uitgevoerd.
- Een batchbestand kan controleren of een programma in het geheugen is geladen door de uitvoer van een opdracht MEM /C naar FIND door te sluizen.

U kunt DOS-opdrachten met batchbestanden beheren, maar het is geen programmeertaal met veel mogelijkheden. In hoofdstuk 11 wordt beschreven hoe u batchbestanden krachtiger kunt maken door utility's die met DEBUG worden gemaakt.