



(The Notorious SNAP Online)  
Visit me at: [www.tns.de.vu](http://www.tns.de.vu)

German - Einführung in QBasic V4.5 mit Compiler

Hallo, und wie geht es dir ???

Ich muß dir gleich von vorne herein gestehen, dass dies mein erstes Tutorial sein wird, also schnell dich besser an.

Nun ja, hier erstmal die wichtigsten QBasic-Befehle, die du unbedingt wissen solltest.

CLS > Löscht den Bildschirminhalt => neuer schwarzer Bildschirm.  
PRINT > Gibt einen Text oder Variable aus (schreibts auf den Bildschirm)  
INPUT > Man kann einen Text eingeben  
LET > Man kann etwas berechnen lassen  
END > Beendet das Programm

So, diese Befehle dürften vor erst etwas reichen.

Starte nun dein QBasic V4.5 mit Compiler. Was, du hast gar kein???  
Du kannst dir auf folgenden Websites einen runterziehen.

- [www.snap.to/tns](http://www.snap.to/tns) (unter der Rubrik: Downloads => Programme)
- [www.qbasic-cafe.com](http://www.qbasic-cafe.com) (unter der Rubrik: Downloads => Compiler)

zieh' dir am besten die Version 4.5 runter, da die am häufigsten gebraucht wird.

So! Programm gestartet??? Es kommt ein kleines Fenster, dass nach Parametern fragt? Dann drück einfach OK.=> Ist er im Vollbildmodus (der komplette Bildschirm ist ein alter DOS-Editor?). Wenn ja, dann Drück folgende Tastenkombination: ALT+EINGABETASTE, damit du QBasic in einem Windowsfenster erhältst. Denn jetzt kannst du dieses Tut neben deinen QBasic-Editor öffnen und hier meinen Schritten einfacher folgen.

Jetzt geh auf den Menüpunkt 'Optionen' und schau, ob der Punkt: 'Vollständiges Menü' einen Bobbel vorne dran hat. Wenn nicht klick es einfach an.

Jetzt dürftest du eine riesige Auswahl an Menüs haben!

So, hier sind die wichtigsten Menüs, die solltest du unbedingt kennen.

Datei > hier kannst du die Scripts speichern und laden

Ausführen > Start => Startet das geschriebene Programm  
Neustart => Startet das laufende Programm neu (dazu später)  
Weiter => führt ein Programm weiter aus.

Hast du es dir gemerkt?!? Ja, dann kann ich dir noch schnell was anderes geben.

Tastenkombinationen:

Ausführen > Start => Umschalt/Shift + F5  
> Neustart => wie oben (Umschalt/Shift + F5)  
> Weiter => einfach nur F5

Was tun, wenn ein Programm unendlich läuft????  
Ganz einfach, gut aufpassen, denn dies kann für dich sehr sehr sehr wichtig sein. Wenn es sich nicht beenden lässt oder nicht zum Ende steuert, dann mußt du folgende Tastenkombination drücken:

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!  
**!!!STRG/CTRL + PAUSE/BREAK!!!**  
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

So, jetzt haben wir das Wichtigste. Also schreiben wir jetzt mal unser erstes Programm. Es soll zwei eingegebene Werte miteinander addieren.

```
CLS                                'Löscht den Bildschirminhalt
INPUT "erste Zahl", Zahl1          'Frägt nach der 1. Zahl
INPUT "zweite Zahl", Zahl2        'Frägt nach der 2. Zahl
LET Ergebnis=Zahl1+Zahl2          'Addiert die Zahlen
PRINT Ergebnis                    'Gibt das Ergebnis aus
(END)                              'Muß nicht unbedingt eingegeben werden
(Drück F5 oder geh' über die Menüs)
```

und es geht! Es ist doch gar nicht mal so schwer, oder?

hier kommt etwas über Syntax (wie muß ich was in QBasic eingeben)

## Kapitel 1 INPUT - Befehl

=====

INPUT ("Text") (, /;) Variable

("Text") >Der Fragetext. Kann man aber auch weg lassen. Dieser muß aber in ">"<-Zeichen stehen (Bsp.: "Super")

(, /;) >Wenn man, wie wir vorher >, < schreiben, dann wird hinter dem Fragetext kein >?< gesetzt

Mit >; < wird hinter dem Fragetext ein >?< gesetzt.

Probiers einfach aus!!!

Variable >die Variable ist ein Platzhalter, die geändert werden kann.  
D.h., dass diese Variable keine feste Zahl ist.



!!  
!! Wir der Name der Variable ein 2. mal zum Speichern eines Wertes !!  
!! verwendet, dann wird der alte Wert aber gelöscht. !!  
!!

## Kapitel 3 PRINT - Befehl

=====

PRINT (Variable/"Text") (,;/;)

Variable > Die auszugebende Variable  
"Text" > Der auszugebende Text (muß in ">"< stehen (Bsp. "Super"))  
, > Der nächste Print-Ausdruck wird einen Tabstopp weiter  
ausgegeben  
; > Der nächste Print-Ausdruck wird direkt hinter dem diesigen  
ausgegeben

!!! Man kann den Print-Befehl auch einfach alleine Stehen lassen !!!  
(nur PRINT)

So, jetzt schreiben wir mal wieder ein kleines Programm. Es sollen  
wieder zwei Zahlen multipliziert werden, und dannach durch 2 geteilt  
werden.

Was??? Du findest das Schwer??? Du solltest einfach weiter machen, und  
nicht groß darüber nachdenken!!! => Es wird schon noch!!!

Und Programm schon fertig???

Hier wär meins:

```
CLS                'Löscht den Bildschirm
INPUT "Zahl 1: ", Zahl1    'Eingabe der 1. Zahl
INPUT "Zahl 2: ", Zahl2    'Eingabe der 2. Zahl
Ergebnis = Zahl1 * Zahl2 / 2 'Berechnung des Ergebnisses
PRINT Ergebnis           'Ausgabe des Ergebnisses
```

Wow, du hast das auch? Dann ist ja gut. Wenn nicht, dann lies dir  
den oberen Teil nochmals durch und versuche es zu verstehen. Und  
probiere es dann schließlich ohne zu kopieren das Ergebnis hin zu  
bekommen. (Falls du es nicht geschafft hast, dann melde dich bei mir,  
Adresse siehe unten)

Jippy, jetzt können wir schon mit 3 Zahlen (2 Varis) rechnen!!! :-)  
Jetzt noch schnell mit drei Varis. Also, machen wir ein Programm  
für folgendes: Zahl1 + Zahl2 und dieses Ergebnis geteilt durch die  
Zahl3

Schon wieder fertig???

Also, hier wäre meins:

```

CLS                                'Löscht den Bildschirm
INPUT "Zahl 1: ", Zahl1            'Eingabe der 1. Zahl
INPUT "Zahl 2: ", Zahl2            'Eingabe der 2. Zahl
INPUT "Zahl 3: ", Zahl3            'Eingabe der 3. Zahl
Ergebnis = (Zahl1 + Zahl2) / Zahl3 'Berechnung des Ergebnisses
PRINT Ergebnis                     'Ausgabe des Ergebnisses

```

Warum hab ich eine Klammer verwendet??? => Punkt vor Strich und Klammer vor Punkt!!!! (Das weiß sogar meine kleine Schwester ;-)

oder noch ein anderer Vorschlag

```

CLS                                'Löscht den Bildschirm
INPUT "Zahl 1: ", Zahl1            'Eingabe der 1. Zahl
INPUT "Zahl 2: ", Zahl2            'Eingabe der 2. Zahl
INPUT "Zahl 3: ", Zahl3            'Eingabe der 3. Zahl
Ergebnis1 = (Zahl1 + Zahl2)        'Berechnung des Ergebnisses1
Ergebnis = Ergebnis1 / Zahl3       'Berechnung des Ergebnisses
                                     (endgültig)
PRINT Ergebnis                     'Ausgabe des Ergebnisses

```

Ist doch auch eine Möglichkeit ;-)

So, jetzt haben wir aber genug gerechnet!!!

Jetzt kommt mal wieder etwas neues, was man auch brauchen kann.

## Kapitel 4 DO:LOOP - Befehl

## (Erzeugung einer Schleife)

=====

```

DO
:
: Rechen- oder Arbeitsfunktion
:
LOOP (UNTIL/WHILE Bedingung)

```

Syntax:

DO > steht immer allein (am Anfang der Schleife, die öfters wiederholt werden soll.

LOOP > steht immer am Ende der Schleife.

(UNTIL/WHILE)

(UNTIL) > Die Schleife wird solange ausgeführt, bis die Bedingung erfüllt ist.

(WHILE) > Die Schleife wird solange ausgeführt, bis die Bedingung nicht mehr erfüllt ist.

(Bedingung) > meist muß eine Variable einen bestimmten Wert erreichen.

Beispiel:

```

DO                'Anfang der Schleife
a=a+1             'a wird immer mit 1 addiert
LOOP UNTIL a=10  'Die Schleife wird beendet, wenn a=10 ist

```

Falls du es mir nicht glaubst, dann schreib hinter der LOOP-Anweisung ein PRINT a ein, somit kannst du dies kontrollieren.

```
DO          'Anfang der Schleife
a=a+1      'a wird immer mit 1 addiert
LOOP WHILE a=10 'Die Schleife wird beendet, wenn a nicht 10 hat.
```

So jetzt dürfte a nur 1 haben, da die Schleife nur einmal durchlaufen wurde. (Einmal  $a+1=1$ )

(!!!Zu Beginn sind alle Variablen immer 0!!!)

Und kapiert ??? Yeah, dann okay!!!

So programmieren wir mal wieder etwas.

Jetzt lassen wir mal den Computer auf 10.000 zählen, oder? Mal schauen, wie schnell deiner ist. (Meiner braucht nur einen Atemzug)  
Aber nur bis 10.000, okay?

So schon fertig?

K, hier ist meins!

```
CLS          'Löscht den Bildschirm
DO          'Anfang der Schleife
Zahl=Zahl+1  'Addition der Zahl mit 1
LOOP UNTIL Zahl=1000 'Ende der Schleife. Ende, wenn Zahl=1000
PRINT Zahl  'Ausgabe der Zahl, zur Kontrolle
```

Geht doch, oder???

Fragen? - Nicht?, dann ist es ja okay.

Neben bei noch schnell was. Alles was in  $>()<$  steht muß nicht unbedingt im Programm geschrieben werden. D.h., dass man DO und LOOP auch so schreiben kann, aber es ist dann kein Limit gesetzt, dann wiederholt der Rechner die Schleife unendlich oft.

So jetzt soll der Computer etwas tun, wenn eine Zahl 10 ist. ;-)  
Dafür brauchen wir aber ein neuen Befehl!

## Kapitel 5 IF THEN:ELSE:END IF - Befehl

=====

```
IF [Bedingung] THEN          'Anfang der Bedingung
:
: Rechen-/Arbeitsfunktion    'Wenn die Bedingung erfüllt ist
:
(ELSE)                        'Anfang der Bedingung nicht erfüllt
(:)
(:)Rechen-/Arbeitsfunktion  'Was tun, wenn Bedingung nicht erfüllt ist
```



German - Wie kann ich Grafik verwenden (erstellen)

Hallo, und wie geht es dir ???

Dies ist wiederum ein Tutorial von mir. Vielleicht kennst du ja schon einige von mir. Also hier wäre mein Thema für heute: "Wie kann ich Grafik verwenden (erstellen)?"

Klinkt doch ganz einfach, oder? Na egal, hohl dir schon mal was zum trinken und paar Chips, denn jetzt wird es sicherlich gleich spannend. ;-)

Nun, wo fang ich am besten an???

Okay, erstmal mit der Monitoreinstellung!!!

Merk dir schon mal folgenden Befehl!

## **Kapitel 6 SCREEN - Befehl**

=====

Syntax

SCREEN (Nummer)

(Nummer) > Die Bildschirmnummer gibt die Zeilen-/Spaltennummer und die Grafikauflösung an.

(Die Nummer werde ich noch im laufe dieses Tutorials nennen)

So, hast du das kapiert? Okay, also weiter...

Der Monitor steht am Beginn des Programms immer auf SCREEN 0 (nur Text-Mode). Um Grafik zu projekzieren müssen wir also den Mode umstellen und dafür verwenden wir den SCREEN-Befehl.

Ich empfehle den SCREEN immer auf 12 oder 9 zu stellen. Die Bildschirmauflösung dieser beiden ist 640x480 bei 16 Farben. Das dürfte reichen um einige Objekte darzu stellen. Die Zeilen-/Spaltenanzahl wird auf 80x24 eingestellt und dies dürfte eigentlich auch ausreichen.

So, jetzt kommt wieder etwas Grundlegendes! Das Koordinaten der Bildpunkte geht von oben links nach unten rechts. Das heißt der Punkt (0/0) liegt im linken oberen Eck. Und der Punkt (640/480) im unteren rechten Eck. => Nicht vergessen, es sei denn, ich sag etwas anderes oder wir verwenden den WINDOW-Befehl, dazu aber später.

So, ich gehe jetzt mal davon aus, dass du mein vorheriges Tutorial auch gelesen hast. (Einführung in QBasic V4.5). Mach jetzt mal dein QBasic auf. Bei den Parametern auf Okay. Und dann wenn es im Vollbildmodus (Der komplette Bildschirm ist ein Editor) gestartet wurde drückst du folgende Tastenkombination: ALT + EINGABETASTE  
So, jetzt dürfte es ein kleineres Fenster sein, das nur einen kleinen Teil deines Bildschirms einnimmt. => Somit kannst du dieses Tut neben hin machen und direkt arbeiten ;-)

Sorry für die vielen SO, aber es ist so einfacher für mich.

Okay, jetzt brauchen wir nur noch die Farbzahlen und die dazu gehörigen Farben.

0 => Schwarz (oder Hintergrundfarbe)  
1 => Dunkelblau  
2 => Dunkelgrün  
3 => Dunkelcyan  
4 => Dunkelrot  
5 => Dunkelmagenta  
6 => Braun  
7 => Hellgrau  
8 => Dunkelgrau  
9 => Hellblau  
10 => Hellgrün  
11 => Hellcyan  
12 => Hellrot  
13 => Hellmagenta  
14 => Gelb  
15 => Weiß

Einfach, oder??? => Nicht wirklich, aber man kanns bald auswendig, wenn man es etwas häufiger benutzt.

Diese Nummern und Farben solltest du Tag und Nacht beherrschen ;-)

Okay, genug blödsinn gemacht!!! ;-)

Zentrieren wir mal einen kleinen hellroten Punkt in die Mitte des Monitors.

Hier mein Programm:

```
SCREEN 12          'Screen-Mode 12
CLS               'Clear Screen
PSET (320, 240), 12      'Zeichnet Punkt
DO               'Anfang der Schleife
LOOP UNTIL INKEY$ <>""    'Ende der Schleife
```

Hast du es eingegeben und gestartet? Jetzt dürfte kurz vor dem Beenden die Nachricht "Taste drücken um fortzufahren ..." nicht erscheinen. Jetzt mußt du eine Taste drücken, um diese Nachricht zu bekommen. Und dann noch einmal, um raus in QBasic zu gelangen.

Appropo, ist ein Punkt erschienen? Ja, dann ist gut, wenn nicht, dann muß ich das Tut hier noch einmal überarbeiten, da ich jetzt noch nicht einmal mein QBasic auf habe => Alles im Kopf. Mail mir, falls nicht.

Nun die neuen Befehle:

## **Kapitel 7 PSET - Befehl: (Setzt eine Punkt an eine Stelle ;-)**

=====

PSET (x, y), Farbe (Ganz einfach, oder?)

## Kapitel 8 LINE - Befehl: (Zeichnet eine Linie)

=====

LINE (xAnfang, yAnfang)-(xEnde, yEnde),Farbe,(B/BF)

xAnfang => Wo soll ich auf der x-Achse anfangen?

yAnfang => Wo soll ich auf der y-Achse anfangen?

xEnde => Wo soll ich auf der x-Achse aufhören?

yEnde => Wo soll ich auf der y-Achse aufhören?

Farbe => klar, oder?

(B/BF) => malt ein Kästchen mit den Eckkoordinaten.

=> B nur der Rahmen der Box

=> BF gefüllte Box mit der selectierten Farbe

## Kapitel 9 PAINT - Befehl: (Füllte einen Bereich mit einer Farbe aus)

=====

Paint (x,y), Farbe1, Farbe2

Farbe1 => Mit welcher Farbe soll gefüllt werden

Farbe2 => Welche Farbe soll die neue Farbfläche eingrenezn. Nur eine möglich :-(

x,y sollte klar sein.

## Kapitel 10 CIRCLE - Befehl: (Zeichnet einen Kreis/Elypse)

=====

CIRCLE (x,y), Radius, Farbe, Anfang, Ende, X/Y-Verzerrung

Anfang => Wird im Bogenmaß gerechnet. Zur Zeichnung von Halbkreisen

Ende => Wird im Bogenmaß gerechnet. Zur Zeichnung von Halbkreisen

X/Y-Ver=> Gibt das Verhältnis zwischen X und Y Achse wieder. Man sollte damit einfach mal rumspielen.

Der Rest sollte klar sein.

So ... damit kann man jetzt nette Zeichnungen machen! Es braucht zwar immer ein bisschen, aber es funktioniert.

Um den Hintergrund einzufärben sollte man ganz zu beginn des Zeichnens auf (0/0) den PAINT-Befehl verwenden, damit alles gleich die richtige Farbe hat. Es gibt zwar noch folgende Möglichkeit, diese wird aber von mir nicht gerade in positives Licht gerückt, weil bei dieser Möglichkeit die Farbe Schwarz verschwindet :-(

Diese Möglichkeit hat aber auch etwas positives, und dazu jetzt.

## Kapitel 11 COLOR - Befehl:

=====

COLOR Textfarbe, Hintergrundfarbe

Textfarbe => hiermit kann man die Textfarbe der Print/Input Texte ändern  
Hintergf. => hiermit wird die Hintergrundfarbe geändert, aber ist nicht unbedingt zu empfehlen, es sei denn es handelt sich um ein eher Text gebundenes Programm.

Nun ja, jetzt kann man ein kleines Bildchen malen. Wow, super und wie kann ich das jetzt animieren ???

=> Kein Problem, das kommt jetzt

Dies geht mit den sogenannten GET/PUT Befehlen, aber dazu etwas später, denn ich muß jetzt erst wieder etwas Grundwissen aufbauen ;-)

Die gemalte/gezeichnete Grafik wird mit einem speziellen Befehl (GET) in den Cachespeicher geladen. Und mit dem sogenannten PUT-Befehl auf dem Monitor nach belieben ausgegeben. Klingt nicht schlecht, oder? Auf Grund, dass sich die Grafik im Arbeitsspeicher befindet, muß man einen Speicherplatz dort reservieren. Dieses geschieht mit dem sogenannten DIM-Befehl (dieser ist auch für andere Sachen gut, aber dazu im nächsten Tutorial).

Die Befehle:

## Kapitel 12 DIM - Befehl:

=====

DIM Variable (Speichergröße)

Variable => Gibt den Variablennamen an, in dem die Grafik gespeichert werden soll.

Speichergröße => Ganz harmlos. Man kann zwar nicht den ganzen Monitorinhalt darin quätschen, aber doch einiges. Dieser Wert sollte zwischen 500 (ganz kleine Grafik) und 30000 liegen (ganz große Grafik)

Okay soweit? Dann kann's ja weiter gehen.

## Kapitel 13 GET - Befehl:

=====

GET (xAnfang, yAnfang) - (xEnde, yEnde), Variable

Box genauso wie mit dem LINE-Befehl (B/BF)

Variable muß mit DIM festgelegt werden, sonst bekommt man eine nette Fehlermeldung.

-> Als Hilfe: mahl erst mit dem LINE (B)-Befehl eine Box um die Grafik,

um zu sehen, ob es in das GET-Fester hinein passt.

## Kapitel 14 PUT - Befehl:

=====

PUT (x,y), Variable, PSET

x,y => gibt die Koordinaten der Grafik an, an die es projiziert werden soll. Wobei x,y ist die linke obere Ecke !!!

Variable => In der zuvor die Grafik gespeichert wurde.

PSET => Überschreibt die umliegende Grafik. Macht es plan. Beim weglassen, kann man durch die Grafik hindurchsehen, falls ein grafischer Hintergrund verwendet wird.

So jetzt haben wir alles.

Programmieren wir mal was kleines, oder???

Ich bin dafür, dass ein kleines Auto von rechts nach links über den Bildschirm fährt. ;-)

Nee, nehmen wir erst einmal eine Box!!!

Also male eine Box, die dann mit den gelernten Befehlen von links nach rechts über den Monitor flitzt.

Hier wäre mal meine kleine Box:

```
SCREEN 12           'Screen-Mode 12
CLS                'ClearScreen
DIM Box(2000)      'Dimensionierte Variable
LINE (100,100)-(110,100),12, BF 'Die Box (hellrot)
GET (99,99)-(110,100),Box 'Die Box im Cache
CLS                'ClearScreen
FOR i = 10 to 600  'Anfang der Zählschleife
  PUT (i,100),Box,PSET 'Setze Grafik
  LET Starzeit=TIMER 'Nimmt TIMER-Wert
DO 'Anfang der Schleife
LOOP UNTIL TIMER - Startzeit > .000001 'Ende der Schleife
NEXT i             'Ende der Zählschleife
```

Und wie sieht's bei dir aus? Jaja, wohl alles abgeschrieben. Versuch es doch einfach mal selbst. => Es ist nicht schwierig!!!

Doch es ist eigentlich schwierig, wenn man folgendes nicht weis.

Die Grafik darf nicht über das Bildschirmauflösungslimit hinaus projiziert werden. => Gibt sonst eine Fehlermeldung.

Links neben der Box (weil sie von links fährt) sollte man einen Pixel Luft lassen, damit die Box keine Spur zieht. (Probiers einfach mal ohne Luft aus!). Wie man eine Zählschleife benutzt, oder bedient. Wie man ein Zeitverzögerung einsetzt.

Eine Zählschleife

## Kapitel 15 FOR:NEXT - Befehl:

=====

FOR Variable Anfang TO Ende (STEP Schrittgröße)

Variable => Die Variable, in der gezählt wird, man kann diese auch dann für andere Befehle verwenden (mit ihrem Wert => siehe PUT-Befehl im Bsp.)

Anfang => Wo zu zählen angefangen werden soll. \ Von Bis

Ende => Wo zu zählen aufgehört werden soll. /

Schrittgröße => In was für einer Schrittgröße gezählt werden soll.  
Voreingestellt (wenn man es weglässt) = 1

andere Möglichkeit für eine Zählschleife (Aber umständlich)

Hier das kleine Prog

```
Variable=Anfangswert      'Setzt Variable auf Anfangswert
DO                          'Beginn der Schleife
  Variable=Variable + Schrittweite 'Zählt in gewählten Schritten nach oben
LOOP UNTIL Zahl = Endwert  'Ende der Schleife
```

Ist doch ganz easy, aber es geht halt mit FOR:NEXT einfach schneller.  
Und glaube ich sogar einfacher.

So und jetzt machen wir das noch etwas Zeitabhängig ;-)

## Kapitel 16 TIMER - Befehl:

=====

Den Timer kann man nur in eine Variable speichern oder direkt auf dem Monitor ausgeben.

```
Variable = TIMER
      oder
PRINT TIMER
```

Der Timer ist die Uhr im Computer, d.h. dass dies ein Zähler, der in 1/100 Sekunden zählt. Von daher kann man damit die Zeit zwischen 2 Vorgängen messen oder festlegen. Dafür gibt es Folgendes.

```
LET Anfangswert=TIMER
DO
LOOP UNTIL TIMER - Anfangswert > Die zu wartende Zeit
```

Klingt doch logisch, oder?

Hier mit kann man also eine kleine Ruhepause ins Programm einbauen.

So das waren jetzt die wichtigsten Grafik-Befehle.

Jetzt gibt es als Hausaufgaben ;-)) Die Aufgabe ein Auto zu programmieren, dass von einer Bildschirmseite (rechtsoben -> linksunten) fährt. Ich wünsche dir viel Spaß. Schicks mir dann, wenn es fertig ist. Und der erste, der mir solch ein Programm schickt, der darf mit mir eine Runde Outlaws spielen ;-))

Also halt dich ran...

CU, next time

- Regards

The Notorious SNAP

PS: Falls dir dieses Tutorial etwas gebracht hat, dann meld dich bitte bei mir, da es immer toll ist zu sehen, dass man etwas wichtiges vermittelt hat.

Homepage: <http://www.tns.de.vu>  
Email: [TheNotoriousSNAP@GmX.Net](mailto:TheNotoriousSNAP@GmX.Net)  
ICQ: #109612445

Dies ist wiederum ein Tutorial von mir. Vielleicht kennst du ja schon einige von mir. Also hier wäre mein Thema für heute: "In Dateien speichern und aus Dateien laden". Klingt doch cool, oder?  
Nun ja, also fangen wir doch einfach mal mit dem Shit an. ;-)  
Soo, jetzt wo soll ich anfangen. K, um in eine Datei zu schreiben, oder um aus ihr zu lesen muß man sie erst einmal öffnen. Also lernen wir jetzt erstmal den Befehl um Dateien zu öffnen.

## Kapitel 17 OPEN:CLOSE - Befehl:

=====

```
OPEN [Dateiname] FOR [OpenParameter] AS #[Dateinummer]
:
: Die zu speichernden/auszulesenden Befehle
:
CLOSE #Dateinummer
```

[Dateiname] => Dateiname, mit Suffix. Gegebenenfalls auch mit Pfad

[OpenParameter] =>

INPUT => Datei nur lesen

OUTPUT => Datei nur schreiben

APPEND => Etwas am Ende der Datei anfügen

[Dateinummer] => Gibt die Dateinummer an, falls man mehrere Dateien gleichzeitig geöffnet hat.

!!

!!!Bei Output wird die komplette Datei überschrieben !!!

!!!D.h., dass man nicht irgendwo zwischendrin ändern kann!!!

!!

So jetzt kann man eine Datei zum Lesen und zum Schreiben öffnen.  
Jetzt brauchen wir nur noch die Befehle, um in die Datei zu schreiben, oder eine Zeile daraus zu entnehmen.

INPUT - Befehl

=====

(LINE) INPUT #Dateinummer, Variable

(LINE)           => Ist besser so, so wird das Einlesen durch keine Leerschritte unterbrochen.

#Dateinummer => Die Nummer, unter der die Datei geöffnet wurde.

Variable   => Die Vari, in die die einzulesende Zeile gespeichert werden soll.

## Kapitel 18 PRINT - Befehl

=====

```
PRINT #Dateinummer, (Text/Variable)(, / ;)
```

#Dateinummer => Die Nummer, unter der die Datei geöffnet ist  
 (Text/Variable) => Den Text/die Variable, die gespeichert werden soll.  
 (,;/) => Der nächste Text, oder Variable wird bei >, < mit einem  
 Tab in der gleichen Zeile gespeichert und bei >; < wird  
 der Text/Variable direkt dahinter abgespeichert.

So, ja, dass war's. Nicht viel, oder? Du mußt dir nur den Scheiß einfach merken. ;-)

Programmieren wir nun mal ein kleines Programm, das einen Text in eine Datei schreibt und ihn danach aus der Datei wiederholt und auf dem Bildschirm ausgibt.

Dazu öffnest du am besten dein QBasic und machst es zu einem DOS-Fenster mit ALT+EINGABETASTE => somit kannst du mein Tut neben her ohne Probleme lesen. Also los geht's!

Hier das erste kleine Programm für heute ;-)

```
CLS                                'Clear Screen
INPUT "Text: ", Text$              'Eingabeabfrage nach dem Text
OPEN "test.dat" FOR OUTPUT AS #1  'Öffnen der Datei test.dat (Speichern)
PRINT #1, Text$                    'Speichern der Variablen Text$
CLOSE #1                            'Schließen der Datei test.dat
OPEN "test.dat" FOR INPUT AS #1    'Öffnen der Datei test.dat (Laden)
LINE INPUT #1, Text$              'Lesen der ersten Zeile in in die Variable
                                   'Text$
CLOSE #1                            'Schließen der Datei test.dat
PRINT Text$                        'Ausgabe der Variablen Text$
```

So, alles soweit kapiert ??? Ja, dann ist ja gut.  
 Jetzt sollten wir die Datei nur wieder löschen, sonst hängt irgendwo  
 in irgendeinem Verzeichnis eine TEST.DAT rum und das wollen wir ja nicht,  
 oder?

## Kapitel 19 KILL - Befehl

=====

KILL Dateiname

Dateiname => Der Name, der zu löschenden Datei. Kann auch mit Pfad angegeben werden.

Soo, jetzt schreiben wir mal ein kleines nettes Programm, das etwas nützliches an deinem Computer vollbringt :-).

Du weißt doch sicherlich, dass wenn du deinen Rechner startest und das Windows95/98-Logo erschienen ist, dass dann der Bildschirm kurzzeitig schwarz wird und DOS-Befehlszeilen angezeigt werden. Dies soll nicht mehr passieren. Deshalb sollte man am Beginn der 'Autoexec.bat' ein sogenanntes echo off einfügen (Befehl: '@echo off')

Aber nicht, dass wir etwas kaputt machen solltest du zu erst von hand eine Kopie der 'Autoexec.bat' machen. Hier zu öffnest du unter Start->Programme -> MS-DOS-Eingabeaufforderung. Und gibst folgenden Text ein:

```
copy c:\autoexec.bat c:\autoexec.tns
```

Hast du es gemacht? Wirklich? Dann ist ja okay.

Also, zuerst müssen wir eine temporäre Autoexec-Datei erstellen, in der wir ganz zu Beginn das Echo-off schreiben, und dannach dann die Parameter aus der alten. Die originale (autoexec.bat nicht die Sicherungskopie) wird mit der temporären dann überschrieben.

Das alles heißt, dass man eine 'Autoexec.tmp' für's Schreiben öffnet. Die 'Autoexec.bat' für's lesen. In die 'Autoexec.tmp' das Echo-off schreibt und dannach die 'Autoexec.bat' anhängt. Darauf hin alle Dateien schließt und den Inhalt der 'Autoexec.tmp' in die 'Autoexec.bat' schreibt. (Vor den Dateinamen muß der Pfad stehen (C:\...))

Alles klar, was zu tun ist?

Jetzt hätte ich fast gesagt, dann kann's ja jetzt los gehen, aber dafür brauchen wir noch ein Befehl, der uns zeigt, dass die 'Autoexec.bat' bis zum Ende gelesen ist. Hier kommt er.

## Kapitel 20 EOF - Befehl:

=====

Syntax: EOF(n)

n => Dateinummer der geöffneten Datei

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! Dieser Befehl steht nie alleine meist ist er in eine DO:LOOP UNTIL !!!
!!! Schleife, oder in eine IF-Bedingung gefasst. Ich persönlich bevor- !!!
!!! die DO: LOOP UNTIL Schleife                                     !!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

???  
===

Jipp, ich weis, dass du gerade kein Wort verstanden hast. Hier kommt die Erklärung und die Syntax!

DO	'Anfang der Schleife
LINE INPUT #n, Var\$	'Einlesen der Daten
LOOP UNTIL EOF(n)	'Ende der Schleife, am Dateiende

Sieht doch einfach aus. Aus der Datei wird solange gelesen, bis sie komplett

gelesen wurde. (Bis zur letzten Zeile) Und keine Weiteren Daten in der Datei vorhanden sind.

Soo, das müsste jetzt eigentlich reichen, um das kleine Prog zu schreiben.

Denkmal nach. Hast du es? Soll ich dir meins zeigen, oder willst du deins erst einmal ausprobieren und dann bei mir spicken???

Hier kommt meins (kurz und schmerzlos)

```
OPEN "c:\autoexec.tmp" FOR OUTPUT AS #1      'öffnen der temporären Datei
OPEN "c:\autoexec.bat" FOR INPUT AS #2      'öffnen der originalen Datei
PRINT #1, "@echo off"                      'Einfügen des Echo offs in die temp.
DO                                           'Beginn der Lese-Schreib-Schleife
  LINE INPUT #2, a$                          'Einlesen der Daten aus der orig.
  PRINT #1, a$                               'Schreiben der Daten in die temp.
LOOP UNTIL EOF(2)                          'Ende der Lese-Schreib-Schleife
                                           'wenn Dateiende der orig. erreicht ist
CLOSE #1, #2                               'Schließen beider Dateien

OPEN "c:\autoexec.tmp" FOR INPUT AS #1      'öffnen der temporären Datei
OPEN "c:\autoexec.bat" FOR OUTPUT AS #2    'öffnen der orig. Datei
DO                                           'Beginn der Lese-Schreib-Schleife
  LINE INPUT #1, a$                          'Einlesen der Daten aus der temp.
  PRINT #2, a$                               'Schreiben der Daten in die orig.
LOOP UNTIL EOF(1)                          'Ende der Lese-Schreib-Schleife
                                           'wenn Dateiende der temp. erreicht ist
CLOSE #1, #2                               'Schließen beider Dateien
KILL "c:\autoexec.tmp"                    'löschen der temp. Datei
```

War doch ganz einfach, oder?

Nebenbei noch was! Falls man aus einem Programm mit CTRL/STRG + PAUSE/Break aussteigt und dann das Programm neustartet, wenn noch eine Datei geöffnet ist, dann bekommt man eine Fehlermeldung. Man sollte in diesem Fall die Taste F6 drücken und den CLOSE(n)- Befehl in der unteren Bildschirmhälfte eingeben. (n in diesem Fall die Dateinummer, der geöffneten Datei) Und dann die Eingabe am Ende der Zeile drücken. Somit wird dieser Befehl ausgeführt. => Dann sollte man das Programm mit F5 weiter ausführen. (Jetzt müsste es eigentlich funktionieren. Falls nicht, dann beende einfach QBasic und starte dieses dann neu.

nun ja, jetzt haben wir mal was in einer Datei geändert. Dies ist eigentlich das schwierigste an dem ganzen Laden und Speichern. Aber nun machen wir mal ein anderes Prog, das uns die Zahlen zwischen 1 und 20 in eine Datei speichert. (Die Datei wird jedes mal dazu neu geöffnet)

```
CLS                                         'Clear Screen
FOR i=1 TO 20                              'Anfang der Schleife 1 -> 20
  OPEN "test.dat" FOR APPEND AS #1        'Dateiöffnen für Ab-Ende-Schreibweise
  PRINT #1, i                             'Schreibt den Zähler in die Datei
CLOSE #1                                   'Schließt die Datei
```

NEXT i 'Ende der Schleife

War doch ganz einfach, oder?

Okay, dann schreiben wir mal etwas anspruchsvolleres, oder?

Also hier kommt mein Auftrag für dich!!!

Erstelle in QBasic 2 Dateien (4 Output). Lass eine Eingabe abfrage ablaufen (10 abfragen). Die Eingaben, die vom Benutzer kommen werden abwechselnd in die eine und die nächste dann in die andere geschrieben. Danach werden die Dateien neu geladen, und dann wird das Eingegebene wieder rekonstruiert. Viel Erfolg!!!

Schon fertig???

Ja, hier wäre mein Beispiel, wie ich es geschrieben hätte!!!

```
CLS
OPEN "datei1.txt" FOR OUTPUT AS #1
OPEN "datei2.txt" FOR OUTPUT AS #2
FOR i=1 TO 5
  LINE INPUT "", a$
  PRINT #1, a$
  LINE INPUT "", b$
  PRINT #2, b$
NEXT i
CLOSE #1, #2
OPEN "datei1.txt" FOR INPUT AS #1
OPEN "datei2.txt" FOR INPUT AS #2
FOR i=1 TO 5
  LINE INPUT #1, a$
  PRINT a$
  LINE INPUT #2, b$
  PRINT b$
NEXT i
CLOSE #1, #2
```

Und wie sieht deins aus??? Komm, sag schon! Willst nicht sagen? Awwh...

Wieder einmal eine Aufgabe, die auf dieses Tutorial basiert. So hier kommt, die kleine Aufgabe: Jimm, erstellen Sie ein Programm, dass die Inhalte folgender Dateien ausgibt: 'c:\windows\win.ini'; 'c:\windows\system.ini'; 'c:\autoexec.bat'; 'c:\config.sys'

ja ich denke, dass diese Files reichen. Also, diese Files sollen nacheinander auf dem Bildschirm angezeigt werden, wobei man den Inhalt auch lesen können sollte!!! (Nicht, dass einfach nur der Inhalt einmal kurz über den Bildschirm flimmert)

Und wie immer gibt es heute auch das gewisse AddOn. Wer mir das Programm wie immer als erster einsendet, der darf mit mir eine Runde NFS4 spielen.

Also halt dich rann...

CU, next time

- Regards

The Notorious SNAP

PS: Falls dir dieses Tutorial etwas gebracht hat, dann meld dich bitte bei mir, da es immer toll ist zu sehen, dass man etwas wichtiges vermittelt hat.

Homepage: <http://www.tns.de.vu>  
Email: [TheNotoriousSNAP@GmX.NeT](mailto:TheNotoriousSNAP@GmX.NeT)  
ICQ: #109612445