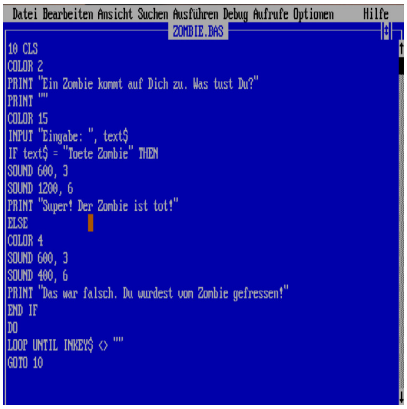


EnDOSkopie – QBasic

Description

Heute kennt man das eigentlich nicht mehr. Man startet seinen Computer und landet in einer Programmiersprache oder das Betriebssystem stellt zumindest eine zur Verfügung. Computer wie C64 und Amiga (AmigaOS 1.1 bis 1.3) lieferten BASIC mit und auch unter DOS bekam man, zumindest eine abgespeckte, BASIC-Version mitgeliefert.

Trostlose Gegenwart

Welche Programmiersprache ist bei Windows im

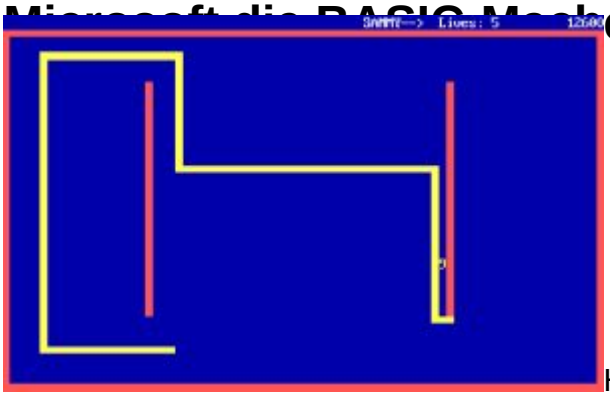
Lieferumfang? Die Frage ist nicht so einfach zu beantworten, wie sie es früher noch war. Wenn man heute Windows installiert, hat man nicht den Eindruck, sofort programmieren zu können. Klar, über den Browser könnte man gleich mit HTML und JavaScript starten und da Windows .NET mitliefert bzw. dies mehr oder weniger ein Bestandteil des Betriebssystems ist, haben wir so etwas wie eine Basis für viele Programmiersprachen.

Theoretisch können wir einen Texteditor starten und Batchdateien schreiben, aber letztlich ist das alles *nur* gescripte und eine Entwicklungsumgebung ist noch nicht in Sicht. Wer programmieren lernen möchte, bemüht die Suchmaschine seines Misstrauens und bekommt da hunderte mehr oder weniger schlechte Ratschläge. In den Anfängen der Heim- und Personalcomputern hatte man zwar weniger Programme und kein Internet, aber eine Entwicklungsumgebung war meistens inbegriffen.

Der Commodore 64 startete sofort mit dem BASIC (**B**eginner's **A**ll-purpose **S**ymbolic **I**nstruction **C**ode). Das heißt, dass man sofort programmieren konnte und im Prinzip jeder Aufruf am C64 ein BASIC-

Befehl war. Das verleitete früher oder später viele Benutzer dazu, sich in der Welt der Programmierung zu verirren.

Microsoft die BASIC Macher



Heutzutage kennen viele die Firma Microsoft nur als das große Imperium, welches Windows, Office-Programme und andere Software entwickelt, den halben Markt aufkauft und sich vor Gerichten wegen seinen Machenschaften rechtfertigen muss. Das Unternehmen begann aber einst sehr klein und sein Kerngeschäft bestand darin, BASIC für verschiedene Computer zu entwickeln. Nicht nur für das eigene Betriebssystem DOS, sondern auch das AmigaBASIC und C64 BASIC V2 stammen von Microsoft.

Keine Sorge, auch BASIC ist keine Microsoft Erfindung. Die Programmiersprache wurde 1964 von John G. Kemeny, Thomas E. Kurtz und Mary Kenneth Keller am Dartmouth College entwickelt. Erst 1975 entwickelte die neu gegründete Firma von Bill Gates das Altair BASIC für den gleichnamigen Computer Altair 8800.

Im August 1981 veröffentlichte Microsoft ihr MS-DOS Betriebssystem und war damit ihr erstes Betriebssystem für x86-PCs. Dabei war die erste DOS-Version auch keine Eigenentwicklung. Die Firma kaufte lediglich für 50 000$ alle Rechte an 86-DOS und verkaufte es in Paket mit MS-BASIC für 80 000$ an IBM weiter.

DOS wurde fortan von Microsoft weiterentwickelt und ab MS-DOS 2 lag 1983 GW-BASIC dem Betriebssystem bei. Obwohl GW-BASIC bis 1988 weiterentwickelt wurde und den DOS-Versionen bis 1991 beilag, erschien bereits 1985 QuickBASIC, welches nicht nur für DOS, sondern auch für MacOS und OS/2 erhältlich war.

Ab 1991 wurde statt GW-BASIC nun QBasic als DOS-Beilage mitgeliefert. QBasic ist dabei eine abgespeckte Version von QuickBASIC. Es sollte den ersten Einstieg bieten in der Hoffnung, dass begeisterte Programmierer irgendwann QuickBASIC kauften.

Beschränktes BASIC

Der größte Unterschied zu QuickBASIC bestand bei QBasic

darin, dass ein Compiler fehlte und der Code nur interpretiert wurde. Das bedeutete, dass man aus den .BAS Dateien keine EXE machen konnte und die Entwicklungsumgebung für kommerzielle Zwecke nahezu unbrauchbar wurde, da man den Code und nicht das Programm weitergeben musste.

Da der Code interpretiert und nicht kompiliert wurde, war er natürlich auch deutlich langsamer. Außerdem ließen sich keine Fremdbibliotheken einbauen und zu allem Überfluss wurde die Hilfe abgespeckt und lediglich vier Beispielprogramme mitgeliefert. Kaum vorstellbar, dass ausgerechnet bei der Anfängerversion an der Hilfe gespart wurde.

Es gab noch weitere Beschränkungen beim Systemzugriff, aber am meisten fiel tatsächlich der fehlende Compiler ins Gewicht. Kleinere Programme und sogar Spiele ließen sich ohne Schwierigkeiten verwirklichen, aber ohne eine EXE war man zum OpenSource-Lieferanten verdammt, obwohl es den Begriff damals noch nicht gab.

Der Code im Zentrum

Klassischer BASIC-Code wurde stur von oben nach unten geschrieben und gelesen, wobei jede Zeile (meistens in 10er Schritten) nummeriert wurde. Typisch für BASIC waren die GOTO-Befehle. Bei bestimmten Ereignissen sprang der Code in die vorgegeben Zeile. In QBasic war es nicht mehr nötig die Zeilen zu nummerieren, man konnte dies aber weiterhin für die Sprungmarker nutzen. Wenn man in die erste Zeile *10* schrieb und später im Programm *GOTO 10*, sprang man immer an den Anfang des Programms.

Nachfolgend ein kleines Beispiel:

```
10 CLS
COLOR 2
PRINT „Ein Zombie kommt auf Dich zu. Was tust Du?"
PRINT „"
COLOR 15
INPUT „Eingabe: „, text$
IF text$ = „Toete Zombie" THEN
SOUND 600, 3
SOUND 1200, 6
PRINT „Super! Der Zombie ist tot!"
ELSE
COLOR 4
SOUND 600, 3
SOUND 400, 6
PRINT „Das war falsch. Du wurdest vom Zombie gefressen!"
END IF
DO
LOOP UNTIL INKEY$ <> „"
```

Mangels Einrückung und den Befehlen in Großbuchstaben

wirkt der Code auf den ersten Blick ziemlich chaotisch. Tatsächlich lässt sich mit diesen Befehlen sogar ein kleines Textadventure basteln, wenn man gerade vierzehn Lenzen alt ist und sonst keine Ahnung hat. Was man nicht vergessen darf: Die Code highlight gab es damals im Editor noch nicht. Es war alles die gleiche farbliche Soße, was das Lesen zusätzlich erschwert.

In der ersten Zeile haben wir unseren Sprungmarker und den Befehl **CLS**, mit dem der Bildschirminhalt gelöscht wird. Der Befehl war auch unter DOS Standard. In der zweiten Zeile definieren wir mit **COLOR 2** die Farbe Grün für den nachfolgenden Text, den wir mit **PRINT** ausgeben.

INPUT erwartet eine Eingabe des Benutzers, die wir in der Variable *text$* speichern. Anschließend wird mit **IF** *text$* = „Toete Zombie" die Eingabe geprüft. Egal ob richtig oder falsch, in beiden Fällen wird mit **SOUND** zwei Töne ausgegeben und eine Antwort geliefert. Am Ende wartet das Programm auf einen Tastendruck und mit **GOTO 10** springt das Programm wieder an den Anfang.

Was besonders auffällt: Es fehlen die aus anderen Programmiersprachen bekannten Klammern. Das If-Statement beenden wir deshalb mit **END IF**. Besonders für Programmierer, die aus der C-Ecke kommen (also auch C++, C#, Java, JavaScript, PHP und andere) ist das sehr ungewohnt und der Code deshalb schwieriger zu lesen.

Mit QBasic, bzw. mit BASIC allgemein, konnte man natürlich noch viel mehr machen. Man konnte Grafiken darstellen und mittels Schleifen, Switch-Statements und zahlreichen anderen Befehlen viel anstellen.

Um das ganze Chaos etwas besser in den Griff zu bekommen, gibt es in QBasic die Möglichkeit SUBs und Funktionen zu schreiben. Dazu ein simples Beispiel:

```
DECLARE FUNCTION zahlen! (s AS INTEGER)
CLS
p = zahlen(7)
PRINT p

FUNCTION zahlen (s AS INTEGER)
        zahlen = s + 5
END FUNCTION
```

Was hier in einer Liste erscheint, sind in QBasic zwei Fenster. Das Hauptprogramm geht bis **PRINT** *p*, die Funktion ist in einem eigenen Fenster. Die Ansicht ist zwar gewöhnungsbedürftig, aber immerhin lässt sich das Code-Chaos damit etwas besser verwalten.

Weiterentwickeltes BASIC

Der Einstieg in die Welt der Programmierung mag bei BASIC einfach gewesen sein, dennoch hatte es seine Nachteile. Je größer die Programme wurden, umso schwieriger waren sie zu lesen und somit zu warten. Außerdem war BASIC, im Vergleich zu Assembler, C, C++, Pascal und anderen Sprachen, sehr langsam und somit für komplexe Anwendungen oder Spiele ziemlich uninteressant.

Trotz der Nachteile entwickelte Microsoft die Sprache weiter und nutzte fast jede Gelegenheit, sie irgendwo zu implementieren. Bereits unter DOS, aber vor allem mit Windows wurde die Entwicklung von Visual Basic vorangetrieben. Trotz aller Bemühungen des Unternehmens verlor BASIC immer

mehr an Bedeutung. Als mit .NET eine einheitliche Schnittstelle entwickelt wurde und damit BASIC-Code kaum langsamer war als die anderer Sprachen, die von .NET unterstützt wurden, war es zu spät. BASIC wurde nur noch von *Fans* und von Firmen genutzt, die ihre alten Programme nicht komplett neu schreiben wollten. Im Gegensatz zu anderen Umgebungen wie Visual FoxPro hat Microsoft den Support für Visual Basic aber nie ganz aufgegeben.

Ein weiterer Nachteil an BASIC waren die zahlreichen Dialekte. Es gab viele verschiedene BASIC-Versionen, die kaum zueinander kompatibel waren. Das bedeutet, dass man sich in jedes BASIC neu einarbeiten durfte, was je nach Dialekt auch länger dauern konnte.

Über viele Jahre ging Microsoft einen eigenen Weg und versuchte dabei stets die Konkurrenz klein zu halten, wenn man sie schon nicht aufkaufen konnte. Da man die Programmiersprache bereits hatte und sie vergleichsweise einfach war, baute man sie auch in die eigenen Office-Produkte unter dem Namen VBA ein. Hier hatten Anwender die Möglichkeit, kleine Makros oder komplexere Programme zu schreiben. Bei umfangreichen Tabellen und Berechnungen konnte eine Eingabemaske durchaus eine große Hilfe sein. Ein Kernproblem bestand aber darin, dass diese Makros nicht zwingend zu anderen Office-Versionen kompatibel sein mussten. Wenn in einer Firma zwei oder drei verschiedene Versionen von Excel im Einsatz waren, konnte das durchaus in einer Katastrophe münden. Ähnliches galt auch für Ländereinstellungen bezüglich Punkt und Komma bei Zahlen. Ja, die Entwicklung kleinerer Programme war einfach, aber der Teufel steckte im Detail.

Persönliche Meinung

QBasic war meine erste Programmiersprache. Ich stieß eher zufällig darauf und die Einarbeitung war mangels Bücher und einer schlechten Hilfe die Hölle. Ohne Internet, Lektüren und erfahreneren Freunden macht der Einstieg in die Welt der Programmierung keinen Spaß, egal welche Sprache. In der örtlichen Bücherei gab es zwar ein paar Schriften zu Basic, aber diese waren nicht kompatibel zu QBasic und vermittelten deshalb nur wenige Grundlagen.

Der Vorteil war, dass man sehr schnell kleine Programme schreiben konnte und sei es nur als Taschenrechner-Ersatz. Nachdem ich mich einige Monate mit einem Freund abmühte, konnten wir unser erstes QBasic Spiel veröffentlichen und waren extrem stolz darauf. Eine Fortsetzung, sogar mit Sprite-Animationen, wurde zwar begonnen aber nie fertig gestellt.

In den nachfolgenden Jahren schrieb ich ein paar Sachen in Pascal, testete Visual Basic unter Windows aus, was das erstellen von Anwendungen super einfach machte, aber vertieft befasste ich mich damit nicht mehr. Erst einige Jahre später in der Arbeitswelt rutschte ich in die VBA-Welt hinein und lernte sie zunächst lieben und dann, aufgrund der beschriebenen Probleme, hassen. Später befasste ich mich vorwiegend mit Websprachen wie PHP und (natürlich) mit GML. Spätestens seit meiner PHP Zeit hasse ich BASIC.

Wer sich einmal an die Klammern, die Einrückung (ist bei BASIC möglich aber nicht üblich) und die ganze Syntax gewöhnt hat, kann mit BASIC kaum noch etwas anfangen. Wenn ich BASIC-Code sehe, wirkt es auf mich oft als wäre jemand mit dem Gesicht auf die Tastatur gefallen. Hier ein kleines Beispiel:

BASIC

```
FOR i = 1 TO 3 STEP 1
 FOR a = 1 TO 3 STEP 1
  PRINT i; „ und „; a
 NEXT a
NEXT i
```

PHP

```
for($i=1; $i<4; $i++)
{
 for($a=1; $a<4; $a++)
 {
  echo $i . „ und „ . $a . „<br>";
 }
}
```

Klar, in PHP brauche ich mehr Zeichen (in diesem Beispiel sind es 12), aber dennoch fühlt es sich für mich erheblich aufgeräumter an. Gerade bei komplexen Angelegenheiten ist das für mich ein Seegen. Man kann natürlich beide Versionen noch kompakter machen, aber es geht um Lesbarkeit. Gerade wenn man alten oder fremden Code lesen darf, hat das eine sehr hohe Priorität.

Laut PYPL-Index vom März 2018 ist derzeit die beliebteste Programmiersprache Java, dicht gefolgt von Python. Auf Rang drei und vier kommen JavaScript und PHP, gefolgt von C# und C. VBA und Visual Basic sind abgeschlagen auf Platz 13 und 14. Es gibt zahlreiche Statistiken zum Thema, bei denen vor allem die vorderen Plätze stark variieren können, aber gemein haben sie, dass Visual Basic (neben VBA der einzige BASIC-Vertreter) entweder nicht mehr auftaucht oder weit hinten liegt.

Dennoch hat es seinen Reiz, weil es so schön Oldschool ist. Seit einigen Jahren gibt es in Ungarn die QBParty. Das ist eine Demoszene-Party mit der Besonderheit, dass Projekte die in QBasic geschrieben wurden, speziell gewürdigt werden. Während andere alte Sprachen wie Assembler und C meiner Meinung nach heute noch ihre Daseinsberechtigung haben, ist vor allem alter BASIC-Code nur noch für die Nostalgie-Ecke zu haben. Manchmal kann es durchaus Spaß machen, ein kleines Projekt „wie früher“ zu realisieren. Nicht nur, aber auch, weil einem die Entwicklungsumgebung nicht jede Zeile komplettiert und auch sonst kaum durch *Hilfen* und Kommentare nervt. Am Wochenende, mit einer Tasse Kaffee und Retro-Musik kann das durchaus Spaß machen.

Date Created
26. Juni 2018
Author
sven