Dieser Abschnitt behandelt QuickBASIC und dessen abgespeckte Version QBASIC gleichermaßen; der Einfachheit halber wird nur von QuickBASIC gesprochen. Wer noch nie mit QuickBASIC in Berührung gekommen ist, kann den Abschnitt getrost überspringen – er ist eher für Programmierer interessant, die von QuickBASIC zu FreeBASIC wechseln wollen.

Trotz hoher Kompatibilität zu QuickBASIC gibt es eine Reihe von Unterschieden zwischen beiden BASIC-Dialekten. Einige davon beruhen auf der einfachen Tatsache, dass QuickBASIC für MS-DOS entwickelt wurde und einige Elemente wie beispielsweise direkte Hardware-Zugriffe unter echten Multitaskingsystemen wie höhere Windows-Systeme oder Linux nicht oder nur eingeschränkt laufen. Des Weiteren legt FreeBASIC größeren Wert auf eine ordnungsgemäße Variablendeklaration, womit Programmierfehler leichter vermieden werden können.

Hinweis:
Mit der Compiler-Option **`-lang qb`** kann eine größere Kompatibilität zu QuickBASIC erzeugt werden. Verwenden Sie diese Option, um alte Programme zum Laufen zu bringen. Mehr dazu erfahren Sie im Kapitel **??**, S. **??**.

- **Nicht explizit deklarierte Variablen (`DEF###`)**
  In FreeBASIC müssen alle Variablen und Arrays explizit (z. B. durch **`DIM`**) deklariert werden. Die Verwendung von **`DEFINT`** usw. ist nicht mehr zulässig.

- **OPTION BASE**
  Die Einstellung der unteren Array-Grenze mittels **`OPTION BASE`** ist nicht mehr zulässig. Sofern die untere Grenze eines Arrays nicht explizit angegeben wird, verwendet FreeBASIC den Wert 0.

- **Datentyp `INTEGER`**
  QuickBASIC verwendet 16 Bit für die Speicherung eines Integers. In FreeBASIC sind es, je nach Compilerversion, 32 bzw. 64 Bit. Verwenden Sie den Datentyp **`SHORT`**, wenn Sie eine 16-bit-Variable verwenden wollen.

- **Funktionsaufruf**
  Alle Funktionen und Prozeduren, die aufgerufen werden, bevor sie definiert wurden, müssen mit **`DECLARE`** deklariert werden. Der Befehl **`CALL`** wird in FreeBASIC nicht mehr unterstützt.

- **Verwendung von Suffixen**
  Suffixe (z. B. ein $ am Ende des Variablennamens, um die Variable als String zu kennzeichnen) werden nicht mehr unterstützt. Jede Variable muss, z. B. durch **`DIM`**,

explizit deklariert werden.
Damit dürfen Variablen auch keinen Namen erhalten, der bereits von einem Schlüsselwort belegt ist.

- **Padding von `TYPE`-Feldern**
  Unter QuickBASIC wird kein Padding durchgeführt. In FreeBASIC werden UDTs standardmäßig auf ein Vielfaches von 4 Byte ausgedehnt. Dieses Verhalten kann durch das Schlüsselwort **`FIELD`** angepasst werden.

- **Strings**
  Unter FreeBASIC wird an das Ende des Strings intern ein **`CHR(0)`** angehängt. Strings können in der 32-Bit Version des Compilers eine maximale Länge von 2 GB besitzen, in der 64-Bit-Version eine maximale Länge von 8.388.607 TB.

- **BYREF**
  Alle Zahlen-Variablen werden standardmäßig **`BYVAL`** übergeben. Arrays werden immer **`BYREF`** übergeben; hier ist eine Übergabe mittels **`BYVAL`** nicht möglich.

- **Punkte in Symbolnamen**
  Punkte in Symbolnamen sind nicht mehr zulässig, da die Punkte für die objektorientierte Programmierung eine neue Bedeutung besitzen.

- **nicht mehr unterstützte Befehle**
  `GOSUB`/**`RETURN`**[1], **`ON...GOSUB`**, **`ON...GOTO`**, **`ON ERROR`** und **`RESUME`** sind nicht mehr erlaubt. Es gibt jedoch andere Befehlskonstrukte, um die gewünschten Ergebnisse zu erzielen.
  In Kommentare eingebundene Meta-Befehle **`'$DYNAMIC`**, **`'$STATIC`**, **`'$INCLUDE`** und **`'$INCLIB`** existieren nicht mehr. Verwenden Sie **`#INCLUDE`** bzw. **`#INCLIB`**, um diese Befehle zu ersetzen.
  `CALL` existiert nicht mehr und kann einfach weggelassen werden. **`LET`** kann nicht mehr für eine einfache Variablenzuweisung verwendet werden; lassen Sie dazu **`LET`** einfach weg. Stattdessen kann mit dem Befehl eine mehrfache Variablenzuweisung für die Records eines UDTs vorgenommen werden.

- **numerische Labels**
  Labels, die nur aus einer Zahl bestehen, werden nicht mehr unterstützt.

- **globale Symbole, die den Namen eines internen Schlüsselworts besitzen**
  Möchten Sie diese weiterhin benutzen, müssen Sie sie in einem **`NAMESPACE`** deklarieren.

[1] **`RETURN`** kann stattdessen eingesetzt werden, um Prozeduren vorzeitig zu verlassen.