Programmieren lernen für Kinder mit SMALL BASIC

Eine verständliche Einführung in die Welt der Programmierung

Dirk Hardy



Programmieren lernen für Kinder mit SMALL BASIC

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Informationen sind im Internet über http://dnb.d-nb.de abrufbar.

©2021 BMU Media GmbH
www.bmu-verlag.de
info@bmu-verlag.de

Lektor: Matthias Kaiser
Einbandgestaltung: Pro ebookcovers Angie
Druck und Bindung: Wydawnictwo Poligraf sp. zo.o. (Polen)

Taschenbuch-ISBN: 978-3-96645-153-6
Hardcover-ISBN: 978-3-96645-152-9
E-Book-ISBN: 978-3-96645-151-2

Dieses Werk ist urheberrechtlich geschützt.
Alle Rechte (Übersetzung, Nachdruck und Vervielfältigung) vorbehalten. Kein Teil des Werks darf ohne schriftliche Genehmigung des Verlags in irgendeiner Form – auch nicht für Zwecke der Unterrichtsgestaltungreproduziert, verarbeitet, vervielfältigt oder verbreitet werden.

Dieses Buch wurde mit größter Sorgfalt erstellt, ungeachtet dessen können weder Verlag noch Autor, Herausgeber oder Übersetzer für mögliche Fehler und deren Folgen eine juristische Verantwortung oder irgendeine Haftung übernehmen. Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären.

Vorwort

Kinder nutzen den Computer immer mehr. Dabei steht allerdings oft das Spielen im Vordergrund. Bei pädagogisch sinnvollen und altersgemäßen Spielen ist dagegen auch nichts einzuwenden. Leider sieht die Realität oft anders aus – stundenlanges Konsumieren von High-Tech-Produkten auf einer Spielkonsole ist vielleicht für die Kinder reizvoll, führt aber auf Dauer in eine Sackgasse. Deshalb spricht einiges dafür, den Computer auch einmal anders zu nutzen – und zwar, um zu programmieren.

Dabei ist es aber nicht das primäre Ziel, die Kinder zu reinen Programmierern auszubilden, sondern den Kindern vielmehr einen Zugang zum Computer zu ermöglichen, der über das reine Anwenden von Software hinausgeht. Sind die grundlegenden Aspekte der Programmierung erst einmal verinnerlicht, so können die Kinder später in vielen Bereichen (auch außerhalb der Informatik) davon profitieren.

Die Heranführung von Kindern an das Programmieren scheitert aber oft an den komplexen Entwicklungsumgebungen und den modernen Programmiersprachen, die zwar für professionelle Programmierer enorme Möglichkeiten bieten, aber ein Kind und deren Eltern (wenn sie keine Erfahrung im Programmieren haben) hoffnungslos überfordern.

Aus diesen Gründen wird in diesem Buch die Programmiersprache ***SMALL BASIC*** verwendet. Diese Sprache mit der dazugehörigen Entwicklungsumgebung ist sehr gut geeignet, um sich schnell in die Welt der Programmierung einzufinden. Das Buch ist selbstverständlich auch für Erwachsene geeignet, die sich in die Welt der Programmierung einarbeiten wollen.

Zur Benutzung des Buches

Das Buch ist so konzipiert, dass es zum selbstständigen Lernen animiert. Die Erläuterungen zu den einzelnen Befehlen sind anschaulich und mit vielen Beispielen hinterlegt. Jedes Kapitel schließt mit Übungsaufgaben. Der Schwierigkeitsgrad erhöht sich dabei von Kapitel zu Kapitel und bei den letzten Übungsaufgaben wird es dann auch relativ kompliziert. In manchen Kapiteln sind nach schwierigen Themen zusätzliche Verständnisübungen eingebaut. Zu allen Übungsaufgaben werden Hinweise und Tipps gegeben.

Neben dem selbstständigen Lernen wäre es auch vorstellbar, dass ein Elternteil gemeinsam mit dem Kind die Grundzüge der Programmierung mithilfe dieses Buches erarbeitet. Ebenso möglich wäre aber auch, dass dieses Buch im Rahmen des Computer-Schnupperunterrichts in der Grundschule als Unterrichtsmaterial genutzt wird. Von Vorteil ist auf jeden Fall die kostenfreie Entwicklungsumgebung, die im Internet downloadbar ist. Nach Einschätzung des Autors könnte dieses Buch für Kinder ab der vierten Grundschulklasse geeignet sein, einige fortgeschrittene Themen aber eher für Kinder ab der 7. Klasse. Sicheres Lesen und solide Grundkenntnisse im Rechnen sind schon nötig, sonst sind die Kinder zu schnell überfordert. Ansonsten bleibt nur noch, den Lesern des Buches viel Erfolg und viel Spaß bei der Erlernung einer ***Programmiersprache*** zu wünschen.

Inhalt

Vorwort 5

Zur Benutzung des Buches 6

Kapitel 1: Computer und Programme 13

1.1 Was ist eigentlich Programmieren? ... 13
1.2 Aufbau eines Programms ... 14
1.3 Programmiersprachen und Befehle ... 15
1.4 Das Übersetzungsprogramm ... 16
1.5 Aufgaben ... 17
1.5.1 Aufgabe 1: Fehler im Programm ... 17
1.5.2 Aufgabe 2: Mathematik-Hausaufgaben ... 18
1.6 Zusammenfassung ... 18

Kapitel 2: Das erste Programm schreiben 20

2.1 Das Programm SMALL BASIC ... 20
2.2 Das erste Programm ... 21
2.3 Der Befehl TextWindow.WriteLine ... 22
2.4 Fehler im Programm ... 24
2.5 Ein Programm speichern ... 25
2.6 Ein Programm öffnen ... 27
2.7 Aufgaben ... 28
2.7.1 Aufgabe 1: deine Adresse ausgeben ... 28
2.7.2 Aufgabe 2: Speichern und Laden ... 29
2.7.3 Aufgabe 3: Fehler beseitigen ... 20
2.8 Zusammenfassung ... 30

Kapitel 3: Platzhalter 32

3.1 Was sind Platzhalter? ... 32
3.2 Ein erstes Programm mit Platzhalter ... 34
3.3 Platzhalter anlegen ... 35
3.4 Der Befehl TextWindow.ReadNumber ... 36
3.5 Der Befehl TextWindow.Read ... 38

3.6 Aufgaben ... 38
3.6.1 Aufgabe 1: Fehler im Programm ... 41
3.6.2 Aufgabe 2: ein Begrüßungsprogramm ... 41
3.7 Zusammenfassung ... 42

Kapitel 4: Der Computer lernt rechnen 44

4.1 Die Zuweisung ... 44
4.2 Mit Zahlen rechnen ... 46
4.2.1 Zwischenübung ... 47
4.3 Rechnen mit Platzhalter-Werten ... 48
4.4 Mit Texten rechnen ... 51
4.5 Aufgaben ... 52
4.5.1 Aufgabe 1: das Doppelte und das Siebenfache ... 52
4.5.2 Aufgabe 2: ein kleiner Taschenrechner ... 52
4.5.3 Aufgabe 3: ein Satzverdreher ... 53
4.6 Zusammenfassung ... 54

Kapitel 5: Programme beginnen zu denken 56

5.1 Der Computer soll denken lernen ... 56
5.2 If-Then-Befehl ... 57
5.3 Vergleiche mit Zahlen anstellen ... 59
5.3.1 Zwischenübung ... 60
5.4 Vergleiche mit Texten anstellen ... 62
5.5 Und sonst? ... 63
5.6 Im Falle eines Falls ... 65
5.7 Vergleiche mit Und / Oder ... 68
5.8 Aufgaben ... 71
5.8.1 Aufgabe 1: ein kleiner Vokabeltrainer ... 71
5.8.2 Aufgabe 2: ein kleiner Rechentrainer ... 72
5.8.3 Aufgabe 3: Geheime Informationen schützen ... 73
5.9 Zusammenfassung ... 75

Kapitel 6: Wiederholungen 77

6.1 Wiederholungen sind langweilig? ... 77
6.2 Der While-Befehl ... 78
6.3 Eine Wiederholung steuern ... 82
6.3.1 Zwischenübung ... 86
6.4 Eine Wiederholung wiederholt sich ... 87
6.5 Die zählergesteuerte Wiederholung ... 88
6.6 Aufgaben ... 90
6.6.1 Aufgabe 1: Alle Zahlen ausgeben ... 90

6.6.2 Aufgabe 2: Einen Countdown programmieren ... 90
6.6.3 Aufgabe 3: automatische Reihenberechnung ... 91
6.6.4 Aufgabe 4: eine Sternchentreppe ... 92
6.7 Zusammenfassung ... 93

Kapitel 7: Programmier-Tricks 96

7.1 Trick 1: Kommentare sind wichtig ... 96
7.2 Trick 2: Farbe ins Spiel bringen ... 98
7.3 Trick 3: Die Ausgabe verbessern ... 100
7.4 Trick 4: Eine Zufallszahl erzeugen ... 102
7.5 Trick 5: Kommazahlen im Programm ... 105
7.6 Aufgaben ... 106
7.6.1 Aufgabe 1: Die Schriftfarbe auswählen ... 106
7.6.2 Aufgabe 2: zufällige Schriftfarbe ... 107
7.6.3 Aufgabe 3: Zahlen raten mit dem Computer ... 108
7.7 Zusammenfassung ... 109

Kapitel 8: Felder von Zahlen 112

8.1 Wettervorhersage und Zahlen ... 112
8.2 Felder von Zahlen ... 115
8.2.1 Zwischenübung ... 117
8.3 Felder und Wiederholungen ... 118
8.4 Weitere Feld-Befehle ... 123
8.5 Aufgaben ... 126
8.5.1 Aufgabe 1: Temperaturmesswerte prüfen ... 126
8.5.2 Aufgabe 2: ein Feld von Zufallszahlen ... 127
8.5.3 Aufgabe 3: Kopfrechnen trainieren ... 127
8.6 Zusammenfassung ... 130

Kapitel 9: Texte und Felder 132

9.1 Ein Text ist auch schon ein Feld ... 132
9.2 Weitere Funktionalitäten für Texte ... 133
9.3 Ein Feld von Texten ... 134
9.4 Aufgaben ... 135
9.4.1 Aufgabe 1: Texte rückwärts ausgeben ... 135
9.4.2 Aufgabe 2: ein richtiger Vokabeltrainer ... 136

Kapitel 10: Unterprogramme 140

10.1 Immer wieder dasselbe? ... 140
10.2 Ein Unterprogramm schreiben ... 142
10.3 Mehrere Unterprogramme schreiben ... 145

10.4 Unterprogramm ruft Unterprogramm 146
10.4.1 Zwischenübung 150
10.5 Werte mit Unterprogrammen austauschen 151
10.5.1 Zwischenübung 153
10.6 Aufgaben 154
10.6.1 Aufgabe 1: Worte rückwärts schreiben 154
10.6.2 Aufgabe 2: Geheime Botschaften erstellen 154
10.7 Zusammenfassung 156

Kapitel 11: Dateien **158**

11.1 Daten dauerhaft speichern 158
11.2 Texte in eine Datei schreiben 160
11.3 Texte und Zahlen aus einer Datei lesen 163
11.4 Weitere Datei-Funktionen 165
11.5 Aufgaben 168
11.5.1 Aufgabe 1: ein Text-Anzeigeprogramm 168
11.5.2 Aufgabe 2: Vokabeltrainer mit Datei 169
11.6 Zusammenfassung 171

Kapitel 12: Grafik programmieren **174**

12.1 Ein Grafikfenster nutzen 174
12.2 Linien, Rechtecke, Kreise und Co. 177
12.3 Texte im Grafikfenster ausgeben 184
12.4 Schriftart, Schriftgröße und Co. 185
12.5 Auf Ereignisse reagieren 187
12.6 Texte im Grafikfenster einlesen 191
12.7 Aufgaben 197
12.7.1 Aufgabe 1: Kreise zeichnen 197
12.7.2 Aufgabe 2: Malen mit Small Basic 197
12.7.3 Aufgabe 3: Tic Tac Toe 199
12.8 Zusammenfassung 202

Kapitel 13: Buttons und Co. **205**

13.1 Steuerelemente im Grafikfenster 205
13.2 Buttons anlegen 206
13.3 Ereignisse für Buttons 208
13.4 Textfelder anlegen 210
13.5 Steuerelemente programmieren 213
13.6 Aufgaben 215
13.6.1 Aufgabe 1: Vokabeltrainer Deluxe 215
13.6.2 Aufgabe 2: Tic Tac Toe mit Buttons 217
13.7 Zusammenfassung 218

Kapitel 14: Grafik für Profis 221

14.1 Bewegung mit dem Shapes-Befehl 221
14.2 Noch mehr Bewegung mit der Shapes.Animate-Funktion 227
14.3 Bewegen mit Zeitsteuerung 229
14.4 Ein Grafik-Spiel programmieren 232
14.5 Entspannung zum Schluss: Schildkrötengrafik 244
14.6 Aufgaben 249
14.6.1 Aufgabe 1: Punktesammeln-Spiel 249
14.6.2 Aufgabe 2: Schildkröten-Spiel 249
14.6.3 Aufgabe 3: Pac-Man-Variante 250
14.7 Zusammenfassung 252

Anhang: Small Basic – Befehle 256

Index 278

Kapitel 1

Computer und Programme

1.1 Was ist eigentlich Programmieren?

Diese Frage ist natürlich mehr als berechtigt. Deshalb ist es auch sinnvoll, sich mit dieser Frage ein wenig zu beschäftigen. Programmieren heißt eigentlich ganz einfach: Programme für den Computer schreiben. Damit sind wir bereits bei der nächsten Frage: Was ist ein Programm? Diese Frage beantwortet uns nun ein kleiner Helfer namens ***Rob_X***. Er wird sich immer dann zu Wort melden, wenn wichtige Fragen zu klären sind.



Ein Programm ist eine Reihe von Befehlen, die wir dem Computer mitteilen. Meistens werden die Befehle hintereinander aufgeschrieben und in einer Datei gespeichert. Das Speichern in einer Datei hat den Vorteil, dass die Befehle nicht immer neu eingegeben werden müssen, sondern bei Bedarf einfach aus der Datei geladen werden können.

ROB_X hat sich die Arbeit gemacht und ein Beispielprogramm aufgeschrieben:

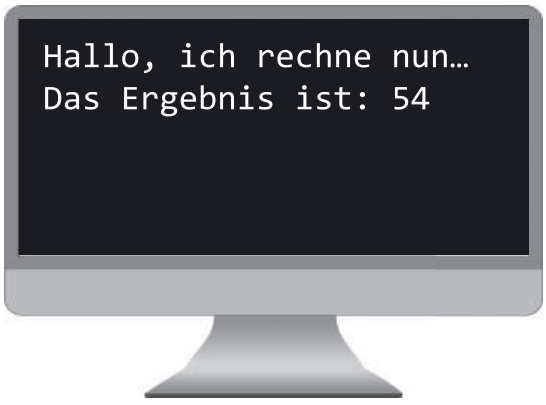
```
***********************************************
Befehl Nr. 1:     SCHREIBE AUF BILDSCHIRM: Hallo, ich rechne nun…
Befehl Nr. 2:     BERECHNE: 20 + 10 + 3*8
Befehl Nr. 3:     SCHREIBE ERGEBNIS AUF BILDSCHIRM
***********************************************
```

Das Beispiel zeigt schon wichtige Punkte, die bei einem Programm beachtet werden müssen. Die Befehle stehen untereinander und sollen natürlich auch in dieser Reihenfolge befolgt werden. Es würde natürlich keinen Sinn machen, das Ergebnis der Berechnung auszugeben, bevor eine Berechnung durchgeführt wurde.

Was sollen diese Befehle nun bewirken?

Der Computer bekommt die Anweisung, einen Text auf den Bildschirm zu schreiben und zwar „Hallo, ich rechne nun …". Dann soll eine Berechnung durchgeführt werden und danach soll das Ergebnis auf dem Bildschirm angezeigt werden. Der Computer arbeitet sozusagen als Taschenrechner, weil er so programmiert wurde. Der Computer startet das Programm und führt die Befehle aus.

Das Ergebnis könnte dann so aussehen:



1.2 Aufbau eines Programms

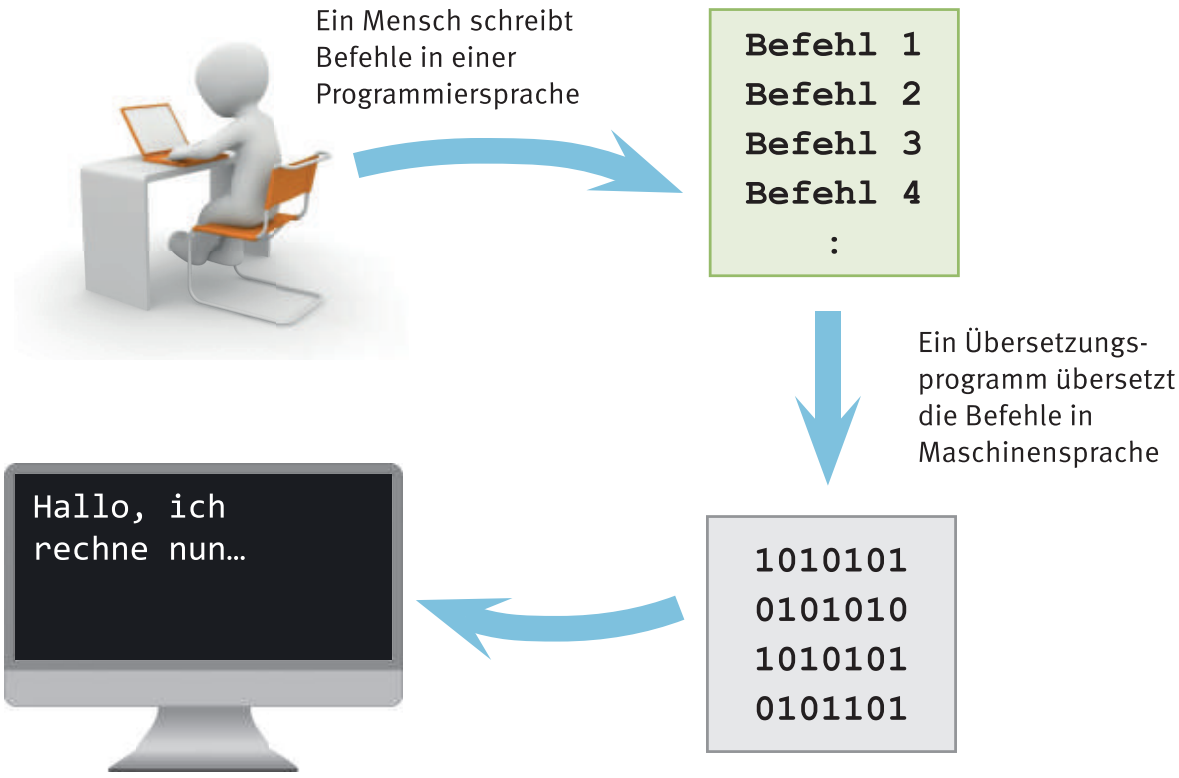
Das erste kleine Beispiel-Programm zeigt schon wichtige Merkmale eines Programms. Die wichtigsten Merkmale werden nun aufgeschrieben.

Die Merkmale eines Programms:

- Das Programm besteht aus Befehlen an den Computer.
- Die Befehle werden hintereinander vom Computer bearbeitet.

1.3 Programmiersprachen und Befehle

Welche Befehle es gibt und wie sie genau benutzt werden, hängt davon ab, welche Programmiersprache benutzt wird. Eine Programmiersprache ist eine Sammlung von Befehlen, die der Computer versteht. Heutzutage gibt es sehr viele verschiedene Programmiersprachen. Wenn man aber einmal eine Programmiersprache gelernt hat, dann ist es aber zum Glück so, dass man sehr schnell auch eine andere lernen kann. Ganz genau genommen versteht der Computer die Befehle einer Programmiersprache noch nicht. Sie müssen erst von einem speziellen Übersetzungsprogramm in die echte Sprache des Computers übersetzt werden – und zwar in die **Maschinensprache**. Diese Sprache versteht der Computer sofort. Ein Programm, das in Maschinensprache geschrieben ist, kann direkt gestartet werden. Bildlich kann man sich das so vorstellen:



Das Programm kann nun gestartet werden

Was ist eigentlich diese Maschinensprache?



Die Maschinensprache besteht nur aus den Ziffern Null und Eins. Das ist tatsächlich die einzige Sprache, die der Computer direkt versteht. Wenn du dieses Buch in Maschinensprache aufschreiben wolltest, dann müsstest du ungefähr 4 Millionen Nullen und Einsen aufschreiben. Das könnte natürlich kein Mensch lesen, aber du bekommst eine Vorstellung davon, mit wie vielen Nullen und Einsen ein Computer umgehen muss.

1.4 Das Übersetzungsprogramm

Das Übersetzungsprogramm, welches die Befehle einer Programmiersprache in die Maschinensprache übersetzt, ist enorm wichtig für das Schreiben von Programmen. Ohne ein solches Programm müsste man dem Computer die Befehle direkt in Maschinensprache mitteilen. Und das wäre sehr unangenehm, denn der Mensch ist es nicht gewohnt, sich nur mit den Ziffern Null und Eins zu verständigen. Alleine den eigenen Namen in Maschinensprache zu übersetzen, ist schon ziemlich mühsam und man braucht auch noch einiges Wissen über unsere Zahlensysteme – das bedeutet, dass man über den Aufbau unserer Zahlen genau Bescheid wissen muss. **`Rob_X`** hat seinen Namen einmal in Maschinensprache übersetzt. Das sieht schon recht gewöhnungsbedürftig aus:

```
Rob_X = 010100100110111101100010
        010111110101100000001010
```

Keine Angst, so etwas ist für das Erlernen einer Programmiersprache nicht nötig. Dafür gibt es ja das Übersetzungsprogramm, welches uns diese Arbeit abnimmt. Das Übersetzungsprogramm heißt in der Computerfachsprache *Compiler*. Das ist ein englisches Wort und wird „Kompeiler“ ausgesprochen. Es bedeutet so viel wie Übersetzer. Für das Lernen mit diesem Buch wird ein einfaches Übersetzungsprogramm genutzt, das im Internet kostenfrei herunterladbar ist (SMALL

BASIC). Damit soll ab dem nächsten Kapitel dann auch richtig programmiert werden. Vorher sollten aber noch einige kleine Aufgaben ohne den Computer erledigt werden.

Warum versteht der Computer eigentlich nur Nullen und Einsen?



Das liegt an dem Rechenwerk des Computers. Das Rechenwerk des Computers ist sozusagen das Gehirn des Computers, in dem alles verarbeitet wird. Damit dieses Gehirn funktionieren kann, muss elektrischer Strom fließen. Die Nullen und Einsen bedeuten nun ganz einfach, dass Strom fließt (Eins) oder nicht fließt (Null). Das ist natürlich etwas einfacher dargestellt, als es wirklich ist, aber um eine Vorstellung zu bekommen, völlig ausreichend.

1.5 Aufgaben

1.5.1 Aufgabe 1: Fehler im Programm

Rob_X hat ein Programm aufgeschrieben, das man gut im Mathematikunterricht der Schule gebrauchen könnte. Das Programm rechnet eine Länge in verschiedene Maße um. Leider funktioniert das Programm nicht ganz richtig. Kannst du die Anzahl der Fehler angeben?

```
*****************************************************************************************
Befehl Nr. 1:   SCHREIBE AUF BILDSCHIRM:   Das Programm von Rob_X

Befehl Nr. 2:   SCHREIBE AUF BILDSCHIRM:   Die Länge 3,51 m soll in cm umgerechnet
                                           werden

Befehl Nr. 3:   BERECHNE: 5,13 * 100

Befehl Nr. 4:   SCHREIBE ERGEBNIS AUF BILDSCHIRM

Befehl Nr. 5:   SCHREIBE AUF BILDSCHIRM:   Die Länge 3,51 m soll in dm umgerechnet
                                           werden

Befehl Nr. 7:   SCHREIBE ERGEBNIS AUF BILDSCHIRM

Befehl Nr. 6:   BERECHNE: 3,51 * 10
*****************************************************************************************
```

1.5.2 Aufgabe 2: Mathematik-Hausaufgaben

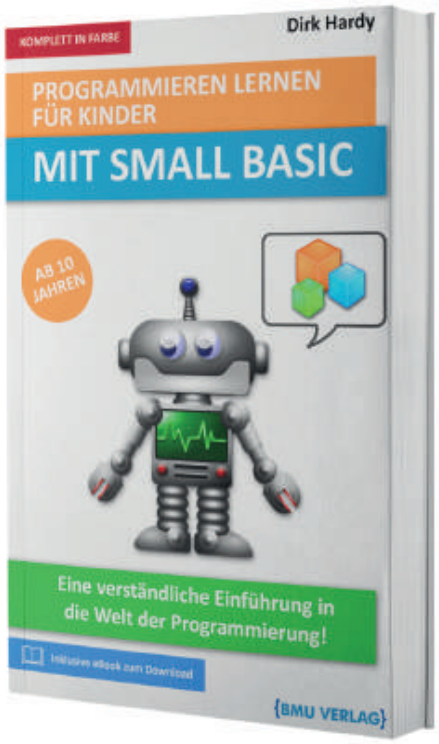
Die Hausaufgaben in Mathematik sind wieder sehr aufwendig. Mit der Zahl 50 sollen hintereinander einige Berechnungen durchgeführt werden. Zuerst soll durch 5 geteilt werden und dann das Ergebnis zu der Zahl 20 addiert werden. Zum Schluss soll dieses Ergebnis dann noch durch 2 geteilt werden. Kannst du die richtigen Befehle aufschreiben, sodass ein Computer dir die Arbeit der Hausaufgaben abnehmen könnte? Beachte dabei die Reihenfolge der Befehle.

1.6 Zusammenfassung

Nach jedem Kapitel fasst **Rob_X** noch einmal die wichtigen Dinge für dich zusammen. Ab dem zweiten Kapitel kommen noch Tipps zur Fehlersuche hinzu. Das hilft dir auch bei der Suche nach Fehlern in deinen eigenen Programmen.



- Ein Programm besteht aus einer Reihe von Befehlen, die wir dem Computer mitteilen. **Die Befehle werden dabei hintereinander aufgeschrieben.**
- **Der Computer versteht eigentlich nur die Maschinensprache.** Der Mensch versteht aber eine Programmiersprache besser. Mithilfe eines bestimmten Computerprogramms, dem Übersetzungsprogramm, wird die Programmiersprache in die Maschinensprache übersetzt. In der Computerfachsprache heißt das Übersetzungsprogramm ***Compiler*** (englisch für „Zusammenbauer“).



Sie erhalten die eBook-Ausgabe zum Buch kostenlos auf unserer Webseite:

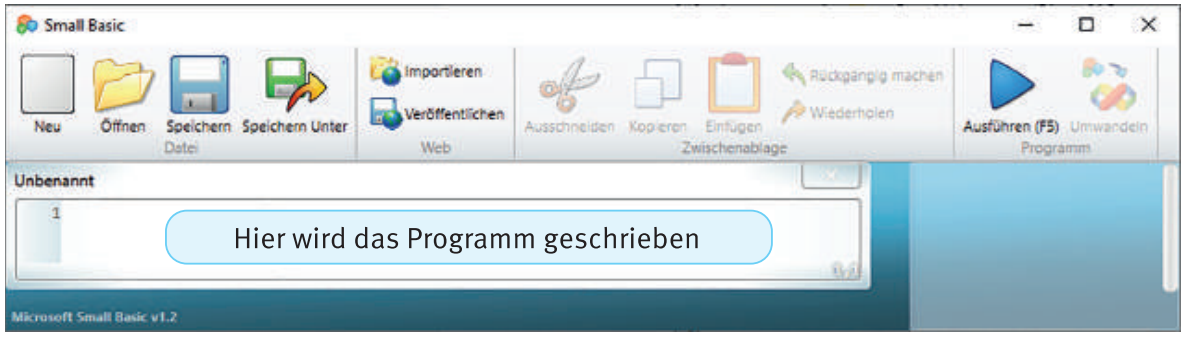
https://bmu-verlag.de/books/small-basic/
Downloadcode: siehe Kapitel 14

Kapitel 2

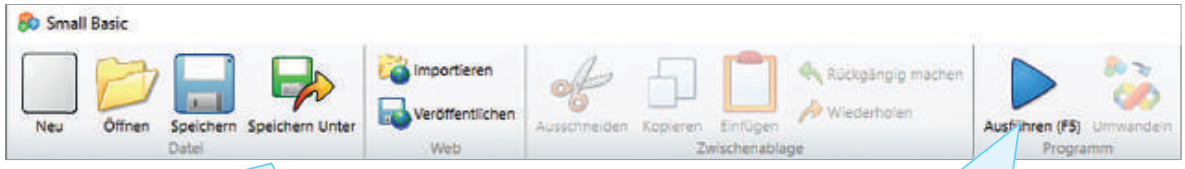
Das erste Programm schreiben

2.1 Das Programm SMALL BASIC

Bevor mit der Programmierung gestartet werden kann, muss natürlich noch das nötige Programm ***SMALL BASIC*** vorgestellt werden, welches von der Internet-Seite von ***Microsoft*** heruntergeladen werden kann. Nach der Installation von ***SMALL BASIC*** kann die Anwendung gestartet werden. Es erscheint dann das folgende Fenster auf dem Computermonitor:



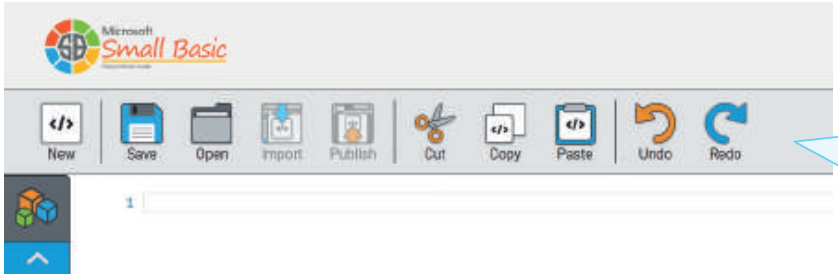
Das Programm ***SMALL BASIC*** hat einige Knöpfe, die für das Programmieren wichtig sind. Diese Knöpfe werden nun kurz vorgestellt:



Mithilfe der Befehle „Neu“, „Öffnen“, „Speichern“ und „Speichern unter“ werden die selbstgeschriebenen Computerprogramme verwaltet.

Der Befehl „Ausführen“ übersetzt das Programm in Maschinensprache und startet es.

Alternativ kann ***SMALL BASIC*** auch online im Browser programmiert werden. Dann muss das Programm nicht installiert werden:



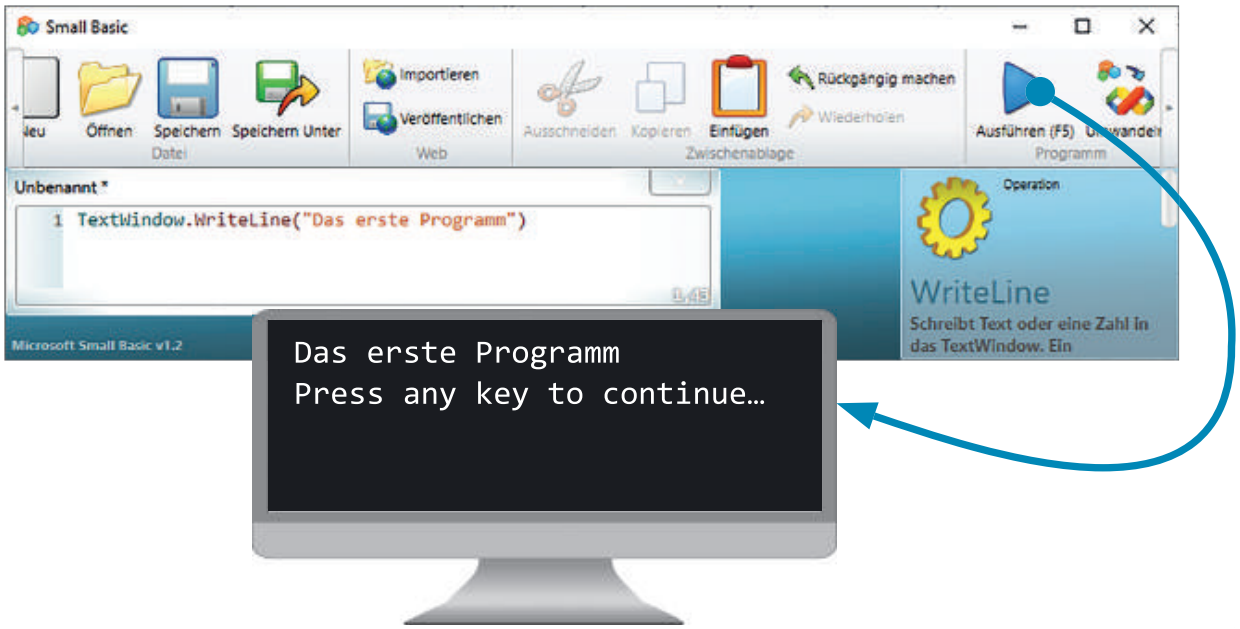
Online-Variante im Browser (leider nur in Englisch verfügbar).

2.2 Das erste Programm

Nun geht es endlich los. Das erste Programm wird geschrieben und gestartet. Öffne zuerst das Übersetzungsprogramm ***SMALL BASIC***. Schreibe danach den folgenden Text in den Schreibbereich (Editor):

```
TextWindow.WriteLine("Das erste Programm")
```

Anschließend drückst du auf den „Ausführen“-Knopf. Nun werden die Befehle übersetzt und das Programm wird gestartet. Wenn du alles richtiggemacht hast, dann erscheint ein neues Fenster. Dieses Fenster ist das gestartete Programm, also die Übersetzung und Ausführung deiner Befehle an den Computer. So sollte es nun nach dem Starten auf deinem Bildschirm aussehen:



Ist das Fenster so erschienen? **Dann herzlichen Glückwunsch, du hast soeben dein erstes Programm geschrieben und gestartet**. Falls das Fenster nicht erscheint, dann musst du alle Schritte, die oben beschrieben wurden, noch einmal überprüfen. In dem Fenster steht zusätzlich die Aufforderung, eine Taste zu drücken (in Englisch: ***Press any key to continue…***). Diese Aufforderung wird bei jedem Programm erscheinen, denn erst nachdem du eine Taste gedrückt hast, wird das Fenster wieder geschlossen und das Programm ist zu Ende.

Die Programmier-Befehle in ***SMALL BASIC*** sind in englischer Sprache. Das ist erst einmal gewöhnungsbedürftig, aber nach kurzer Zeit sollte das kein Problem sein. Die Befehle werden auch alle übersetzt und erklärt.

2.3 Der Befehl TextWindow.WriteLine

Nachdem das erste Programm nun erfolgreich umgesetzt wurde, muss der Befehl `TextWindow.WriteLine` nun näher betrachtet werden, denn er war ja sehr wichtig für das erste Programm. Mit diesem Befehl wird eine Ausgabe von Text auf dem Bildschirm ermöglicht.

```
TextWindow.WriteLine
```

`TextWindow` heißt übersetzt „Text-Fenster“. Damit wird verdeutlicht, dass der folgende Befehl in einem Text-Fenster arbeiten soll.

`WriteLine` heißt übersetzt „Schreibe eine Zeile“. Damit wird ein Text in das Text-Fenster geschrieben.

Der auszugebende Text muss allerdings in Anführungsstrichen und in Klammern stehen, sonst kann der Befehl nicht richtig übersetzt werden. Der Befehl zeigt schon den grundsätzlichen Aufbau von Befehlen in ***SMALL BASIC***. Der Befehl enthält zwei Angaben, die durch einen Punkt getrennt werden. Die erste Angabe steht für die Stelle, an welcher der Befehl wirken soll (in diesem Fall in dem Text-Fenster) und die zweite Angabe istder konkrete Befehl (hier die Ausgabe eines Textes).

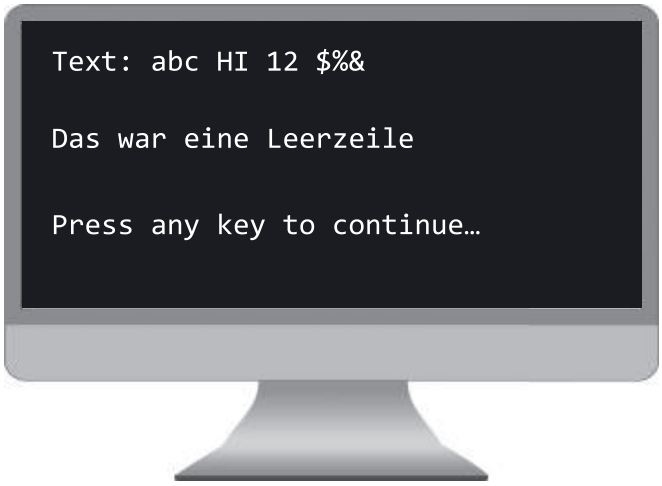
Was ist eigentlich genau ein Text? **Rob_X** erklärt es dir:



Später werden wir sehen, dass der Befehl auch noch für andere Dinge benutzt werden kann, und zwar für die Ausgabe von Platzhalter-Inhalten. Dazu aber dann mehr, wenn die Platzhalter vorgestellt werden. Das folgende Beispiel zeigt, wie der Befehl benutzt werden kann.

```
TextWindow.WriteLine("Text: abc HI 12 $%&")
TextWindow.WriteLine("")
TextWindow.WriteLine("Das war eine Leerzeile")
TextWindow.WriteLine("")
```

In diesem Beispiel werden verschiedene Möglichkeiten der Ausgabe gezeigt. Zuerst werden einige Buchstaben, Ziffern und Zeichen ausgegeben. Danach wird der Befehl ohne einen Text angegeben. Das hat zur Folge, dass eine Leerzeile ausgegeben wird. Nach dem Ausführen sieht es dann so aus:

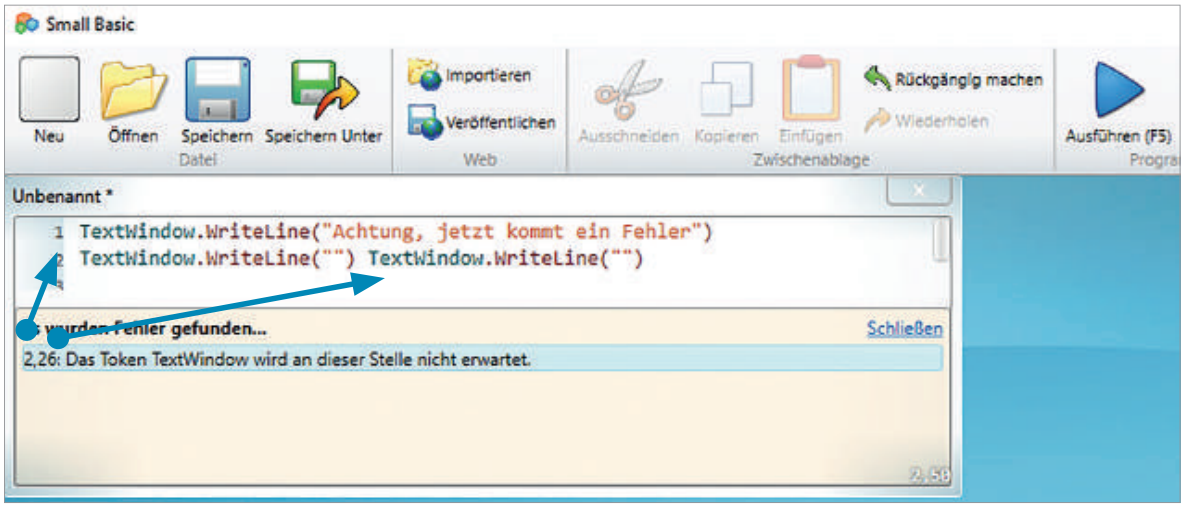


2.4 Fehler im Programm

Genauso wie im richtigen Leben kann man auch beim Programmieren Fehler machen. Beispielsweise schreibt man aus Versehen falsche Befehle oder vergisst die Anführungsstriche bei der Textausgabe. Das ist überhaupt nicht schlimm und gehört zum Programmieren einfach dazu. Die meisten Programmierer verbringen viele Stunden damit, Fehler im Programm zu suchen. Das folgende Programm würde einen Fehler verursachen, denn der Ausgabe-Befehl wurde versehentlich zweimal hintereinandergeschrieben:

```
TextWindow.WriteLine("Achtung, jetzt kommt ein Fehler")
TextWindow.WriteLine("") TextWindow.WriteLine("")
```

Nach dem Ausführen würde das Übersetzungsprogramm den Fehler erkennen. Es erscheint dann automatisch ein Fenster mit dem Hinweis, dass ein Fehler aufgetreten ist. Zusätzlich zeigt die Meldung auch an, in welcher Zeile (hier 2) und Spalte (hier 26) der Fehler aufgetreten ist.



Nun kannst du den Fehler einfach beseitigen und das Programm erneut ausführen. Dann sollte alles in Ordnung sein. Es kann aber auch sein, dass noch mehr Fehler in einem Programm sind. Dann müssen alle Zeilen überprüft und geändert werden. So werden Schritt für Schritt alle Fehler im Programm korrigiert und das Programm kann erneut gestartet werden. Falls nach dem Starten erneut Fehler

auftauchen, werden sie natürlich wieder angezeigt. Das geht so lange, bis das Programm fehlerfrei ist und das schwarze Fenster mit der Ausführung des Programmes beginnt.



Welche Arten von Fehlern gibt es eigentlich?



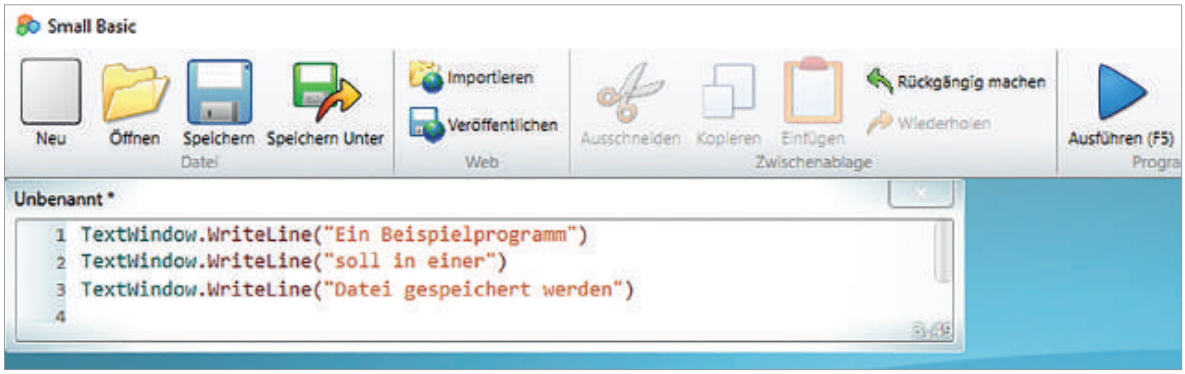
Es können grundsätzlich zwei Arten von Fehlern unterschieden werden. Auf der einen Seite die sogenannten formalen Fehler. Das sind Fehler wie in dem obigen Beispiel – ein Befehl wird falsch geschrieben oder doppelt verwendet oder bei dem Text werden Anführungsstriche vergessen. Diese Fehler kann das Übersetzungsprogramm sehr gut finden. Auf der anderen Seite gibt es die sogenannten logischen Fehler. Bei diesen Fehlern ist es so, dass das Programm übersetzt und gestartet werden kann, aber trotzdem nicht das macht, was der Programmierer will. Dann stimmt etwas mit der Programmlogik nicht. Diese Begriffe sind wahrscheinlich neu für dich und etwas schwer zu verstehen. Stell dir einfach vor, du schreibst einen Aufsatz für den Deutschunterricht. Die Rechtschreibfehler bei dem Aufsatz sind eher formale Fehler. Wenn der Inhalt des Aufsatzes nicht so richtig passt, dann ist es eher ein logischer Fehler.

2.5 Ein Programm speichern

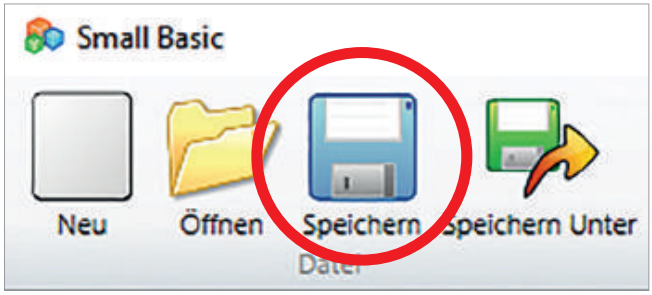
Ein Programm für den Computer zu schreiben ist eine tolle Sache. Wenn es fehlerfrei startet und genau das macht, was man möchte, dann kann man als Programmierer zufrieden sein. Aber was passiert

mit dem Programm, wenn das Übersetzungsprogramm beendet und der Computer ausgeschaltet wird? Es ist für immer verschwunden. Beim nächsten Mal müsste der Programmierer das Programm erneut schreiben. Das wäre unheimlich mühsam. Deshalb gibt es die Möglichkeit, das Programm in einer Datei zu speichern. Das bedeutet, dass das Programm auch nach dem Ausschalten des Computers noch vorhanden ist – und zwar als Datei auf der Festplatte des Computers. Das kann man gut damit vergleichen, dass man sich eine tolle Geschichte ausdenkt. Einige Tage später kann man sich nicht mehr so richtig an die Geschichte erinnern und man muss sich (fast) alles neu ausdenken. Deshalb wäre es sinnvoll, die Geschichte sofort aufzuschreiben. Dann kann man sie jederzeit lesen und vergisst keine Details. So ähnlich ist es mit den Dateien auf der Festplatte des Computers. Der Benutzer eines Computers speichert seine *Geschichten* (also Programme oder Texte) in Form von Dateien auf dem Computer. Das Programm ***SMALL BASIC*** bietet deshalb die Möglichkeit, die Programme in Dateien zu speichern. Wie das funktioniert, wird an dem folgenden Beispiel Schritt für Schritt gezeigt.

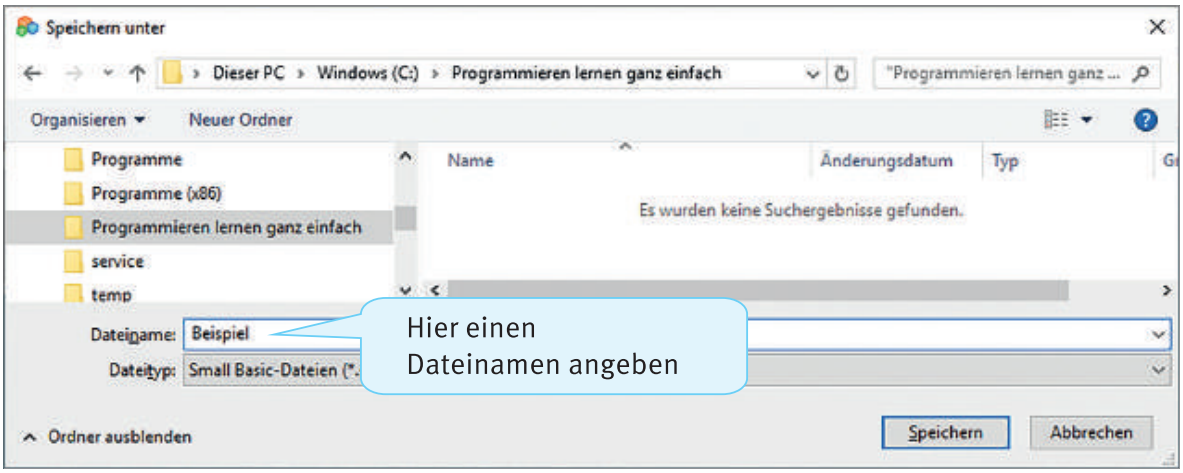
Schritt 1: Ein Programm schreiben



Schritt 2: Den Speichern-Knopf drücken



Nach dem Drücken dieses Steuer-Knopfes öffnet sich ein neues Fenster. In diesem Fenster kann der Speicherort (hier Festplatte C: und Ordner „Programmieren lernen ganz einfach“) und anschließend der Dateiname gewählt werden:



Schritt 3: Noch einmal einen Speichern-Knopf drücken

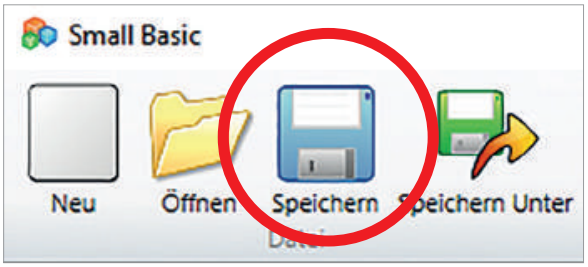
Nachdem ein Dateiname angegeben wurde, muss jetzt nur noch der Speichern-Knopf des „Speichern unter"-Fensters gedrückt werden. Danach ist das Programm gespeichert und man könnte den Computer ruhigen Gewissens ausschalten.

Noch eine Information für Computerexperten:



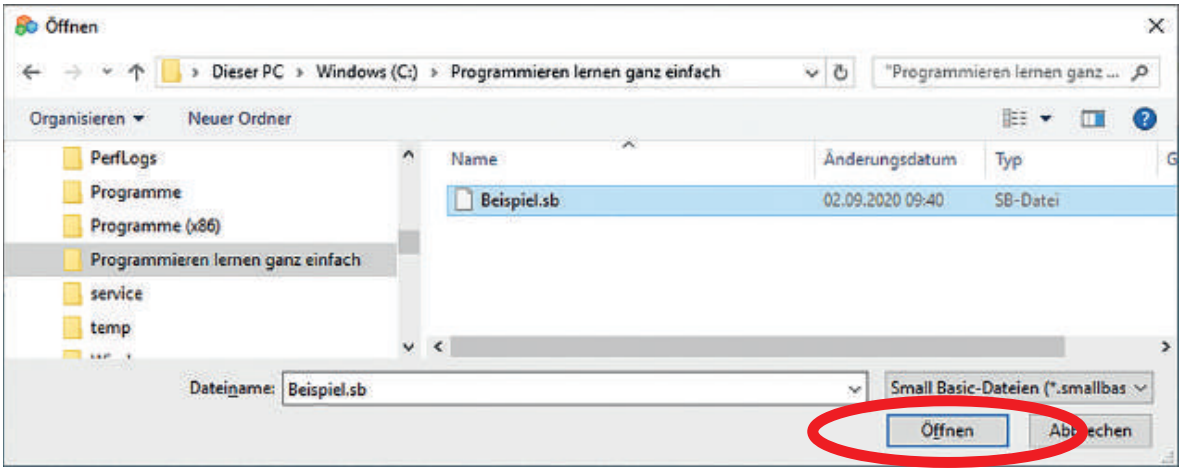
Die Programmier-Dateien werden in einem einfachen Format gespeichert (Small-Basic-Datei *.sb). Eine solche Datei kann auch mit einer Textverarbeitung wie Word für Windows oder Windows-Editor bearbeitet werden.

2.6 Ein Programm öffnen



Nachdem nun klar ist, wie ein Programm in einer Datei gespeichert werden kann, geht es nun darum, wie ein Programm aus einer Datei gelesen bzw. geöffnet werden kann. Das geht ebenfalls sehr einfach. Zuerst muss der Steuer-Knopf für das Öffnen gedrückt werden.

Anschließend öffnet sich ein „Öffnen“-Fenster, welches eigentlich fast genauso wie das „Speichern unter“-Fenster aussieht.



Nun kann einfach eine Datei ausgewählt werden, indem mit der Maus auf den Dateinamen geklickt wird. Automatisch erscheint der Dateiname in dem unteren Feld *Dateiname*. Zum Schluss muss nur der Öffnen-Knopf gedrückt werden und die Programm-Datei wird geladen. Das Programm erscheint vollständig im Eingabebereich von ***SMALL BASIC***.

2.7 Aufgaben

Nun wird es wirklich Zeit für einige praktische Aufgaben, damit du die vielen Informationen aus diesem Kapitel auch richtig verarbeiten und verstehen kannst.

2.7.1 Aufgabe 1: deine Adresse ausgeben

Schreibe ein Programm, das den folgenden Text auf dem Bildschirm ausgibt. Ersetze dabei den Namen von *Max Mustermann* durch deinen Namen und die Adresse von *Max Mustermann* durch deine Adresse. Achte auch auf die Trennzeilen.



2.7.2 Aufgabe 2: Speichern und Laden

Speichere das Programm aus Aufgabe 1 unter dem Dateinamen „Visitenkarte“ auf der Festplatte des Computers. Beende dann das Übersetzungsprogramm ***SMALL BASIC*** und starte es anschließend neu. Öffne nun die gespeicherte Datei erneut. Im Eingabebereich von ***SMALL BASIC*** sollte nun dein Visitenkarten-Programm erscheinen.

2.7.3 Aufgabe 3: Fehler beseitigen

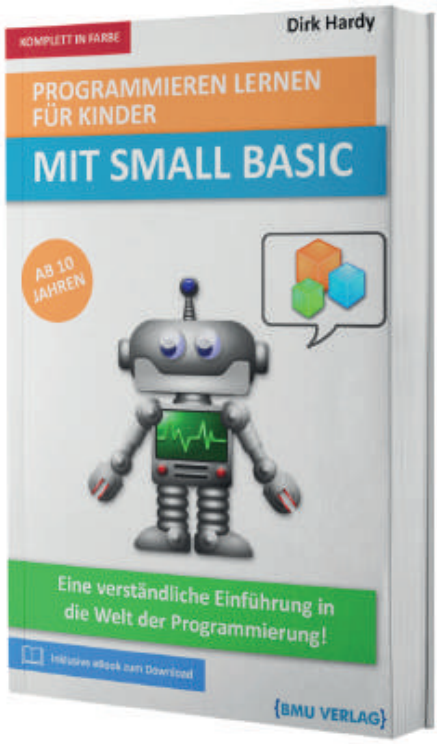
Das folgende Programm enthält einige Fehler. Markiere alle Fehler handschriftlich.

```
TextWindow.WriteLine("Das ist ein Text!)
TextWindow.WriteLine(TextWindow.WriteLine(""))
TextWindows.WriteLine("Noch ein Text!")
TextWindow.WriteLine("ENDE")
```

2.8 Zusammenfassung



- Das Programm ***SMALL BASIC*** übersetzt die Befehle in die Maschinensprache und führt das übersetzte Programm aus.
- Mit dem `TextWindow.WriteLine`-Befehl können Buchstaben, Zahlen und andere Zeichen auf den Bildschirm geschrieben werden.
- Das Übersetzungsprogramm erkennt Fehler und zeigt alle fehlerhaften Zeilen an.
- Ein fertiges Programm sollte in einer Datei gespeichert werden, damit es auch noch vorhanden ist, wenn das Programm ***SMALL BASIC*** beendet oder der Computer ausgeschaltet wird.
- Gespeicherte Programme können immer wieder geladen werden.



Sie erhalten die eBook-Ausgabe zum Buch kostenlos auf unserer Webseite:

https://bmu-verlag.de/books/small-basic/
Downloadcode: siehe Kapitel 14