PureBasic: Hello, world!

Die Tradition des “Hello, world!"-Programmes wurde von Brian Kernighan eingeführt, als er eine Dokumentation über die Programmiersprache B schrieb. Durch das Buch *The C Programming Language* erlang das Beispiel Bekanntheit.

“Hello, world!" ist hervorragendes Beispiel für jede Programmiersprache, da man mit jenem die wesentlichen Aspekte dieser erfassen und darlegen kann. Von daher ist es nur angebracht, es hier ebenfalls zu verwenden.

; Listing 1: Hello, world! OpenConsole() PrintN(“Hello, world!") Delay(2000)

Ausgabe:

Hello, world!

Nachdem man PureBasic installiert hat, öffnet man die IDE über das Startmenü (Windows) bzw. über den Befehl *purebasic* (Linux & Mac OSX). Eine IDE ist eine Entwicklungsumgebung, in der alle Werkzeuge, die man zum Programmieren benötigt, organisiert und schnell verfügbar sind. Man gibt nun den oberen Codeabschnitt ein und drückt F5. Für dieses und alle anderen Kapitel soll gelten, dass man die Programme am Besten von Hand eingibt, da man durch simples Copy&Paste keine Programmiersprache lernt, sondern indem man selber schreibt und ausprobiert!

Hat man die Anleitung beachtet, müsste nun ein Fenster offen sein, in welchem *Hello, world!* steht und das sich nach 2 Sekunden wieder schließt. Unter Linux und Mac wird die Ausgabe direkt auf der Shell sichtbar, über die man die IDE öffnete. Warum geschieht dies? Die erste Zeile stellt einen sogenannten Kommentar dar, d.h. eine Befehlszeile, die vom sogenannten *Compiler* ignoriert wird. Dies sieht man an dem Semikolon, dass am Anfang der Zeile steht. Kommentare können überall im Code stehen, auch hinter einer Codezeile. Sie werden genutzt, um den Code *sinnvoll* zu kommentieren. Code soll für sich selber sprechen und Kommentare stellen eine Ergänzung dar, um sich in diesem einfacher zurecht zu finden. Der Compiler ist das Programm, dass den für den Programmierer lesbaren Code in eine für den Computer ausführbare Datei übersetzt. Nach dem Kommentar kommt der erste Befehl: *OpenConsole()*. Dieser tut nichts anderes, als unter Windows eine Konsole zu öffnen, unter Linux und Mac wird er ignoriert, da Konsolenprogramme, wie dieses eines ist hier immer über die Shell gestartet werden, trotzdem muss er auch auf diesen Betriebssystemen immer geschrieben werden, wenn es sich um ein Konsolenprogramm handelt.

Es werden an dieser Zeile wesentliche Eigenschaften von PureBasic klar: Der PureBasic-Compiler arbeitet zeilenorientiert, d.h. pro Zeile steht ein Befehl. *PrintN(“Hello, world!")* tut nichts anderes, als die Zeile *Hello, world!* auf die Ausgabe zu schreiben und danach einen Zeilenumbruch einzufügen. *“Hello, world!"* ist hierbei das Argument bzw. ein Parameter von *PrintN()*, also etwas, das von *PrintN()* zur Ausführung benötigt wird, in diesem Fall wird ein String (Zeichenkette) benötigt. Was das ist, wird in einem anderen Kapitel erklärt. *Delay(2000)* hält die Ausführung für 2000 Millisekunden, also 2 Sekunden, an. *2000* ist wieder ein Argument, diesmal jedoch ein Zahlenwert. Es fällt auf, dass das Argument von *Delay()* nicht in Anführungsstrichen steht. Dies liegt daran, dass *Delay()* einen Zahlenwert erwartet und keinen String wie *PrintN()*, denn Strings stehen in PureBasic immer in Anführungsstrichen, ansonsten werden sie vom Compiler nicht als solche interpretiert.

Nachdem nun klar ist wie das Programm funktioniert, sollte man ein bisschen mit diesem herumspielen, um sich mit der Syntax, also dem Aufbau eines PureBasic-Programms, vertraut zu machen. Was passiert zum Beispiel, wenn man die Anführungstriche bei *PrintN(“Hello, world!")* weglässt? Man sollte sich mit den Ausgaben des Debuggers vertraut machen; diese helfen weiter, wenn etwas falsch programmiert wurde oder andere Probleme auftreten.

Wenn man Fragen zu einem Befehl hat, kann man auf diesen klicken und dann F1 drücken. So wird direkt die Hilfeseite zu diesem Befehl geöffnet. Ansonsten gelangt man über F1 in die allgemeine Referenz, die auch im Internet eingesehen werden kann. Der Link ist im Inhaltsverzeichnis verfügbar.

0.1 Der Debugger

Der Debugger zeigt, wie schon gesagt, Fehler an. Er wird durch das Fenster unter dem Editor repräsentiert. Lässt man z.B die Anführungszeichen bei *PrintN(“Hello, world!")* weg, zeigt er beim Kompilieren an: “Zeile 4: Syntax-Fehler.” Aus dieser Information kann man erschließen, dass das Programm nicht kompiliert werden konnte, weil in Zeile 4 etwas falsch ist und ausgebessert werden muss.

Der Debugger kann auch in der Entwicklung eines Projekts nützlich sein, denn mit ihm kann man die Ausführung des Programmes dokumentieren.

; Listing 2: Hello, world! mit Debugger OpenConso-

le() Debug “Jetzt kommt die Ausgabe” PrintN(“Hello, world!") Debug “Nun wartet das Programm 2 Sekunden” Delay(2000)

Ausgabe:

Hello, world!

Man drückt abermals auf F5, nachdem man das Programm eingegeben hat. Es kommt wieder die Ausgabe aus dem ersten Programm, jedoch passiert noch etwas anderes: Es öffnet sich ein Fenster in dem die Strings hinter *Debug* angezeigt werden. Die Strings sind hierbei aber keine Argumente, sondern sogenannte Ausdrücke, also etwas, das ausgewertet werden kann. *PrintN(“Hello, world!")* ist ein Ausdruck, denn er kann zu einer Ausgabe von *Hello, world!* auf der Konsole ausgewertet werden. Es gibt noch viele weitere solcher Debugger-Schlüsselwörter. Das Wort “Schlüsselwörter” wird in einem anderen Kapitel erläutert. Das Interessante an diesen Schlüsselwörtern ist, dass sie nur bei aktivierten Debugger kompiliert werden. Dieser ist standardmäßig aktiviert. Sie stellen also eine großartige Hilfestellung bei der Projekterstellung dar, ohne dass man sich in der finalen Version darum kümmern muss alle Befehle zu entfernen.

Der Debugger kann über den Reiter *Debugger* deaktiviert werden, in der Demo-Version lässt er sich jedoch nicht ausschalten.

0.2 Aufgaben

1. Niemals vergessen: Programmieren lernt man nur durch selbst schreiben und ausprobieren, also auch durch bewusstes Fehler einbauen und Verändern des Codes.

2. Was ist an folgendem Programm falsch?

OpenConsole() PrintN(Hello, world!) Delay(2000

1 Text- und Bildquellen, Autoren und Lizenzen

1.1 Text

- **PureBasic: Hello, world!** *Quelle:* https://de.wikibooks.org/wiki/PureBasic%3A_Hello%2C_world!?oldid=813729 *Autoren:* Inkowik, Raymontag, Hgzh und Anonyme: 1

1.2 Bilder

1.3 Inhaltslizenz