Exkurs: Entwicklung der Programmiersprachen (I)

Gemessen an den Entwicklungszyklen der PC-Technik sind die meisten der heute verwendeten Programmiersprachen sehr alt. Gängige Sprachen wie BASIC, Pascal oder C haben ihre Wurzeln in den 60er- und 70er-Jahren. Erste objektorientierte Programmiersprachen wie z.B. Smalltalk wurden in den 80er Jahren entwickelt. Einzig die weit verbreitete Sprache JAVA ist ein echtes neues Produkt der Neunziger.

Selbst hochleistungsfähige Prozessoren (CPU) der Pentium-Klasse als Kernstück eines Computers verfügen nur über einen geringen Befehlssatz, etwa zum Lesen und Schreiben von Speicherzellen, zu wenigen mathematischen Berechnungen oder zum Verschieben von Speicherblöcken. Dieser Befehlssatz wird als Maschinensprache oder Assembler-Code bezeichnet.

Programmiersprachen sollen dem Programmierer ermöglichen, auch komplexere Probleme in einer Weise zu formulieren, die einerseits dem Denken des Menschen entspricht und andererseits noch für den Computer übersetzbar bleibt. Einen Anspruch, den Programmiersprachen bis heute nur sehr unzureichend erfüllen. Oder anders gesagt, es gibt derzeit kein Programmiersystem, das aus jeder in (normaler) menschlicher Sprache formulierten Aufgabenstellung ein ablauffähiges Programm erzeugen kann.

Anfangs standen den Programmierern nur maschinenorientierte Sprachen - unterschiedlichste Assembler-zur Verfügung. Der Programmieraufwand war gewaltig. Programme im Assembler-Code sind sehr hardwarenah und besitzen folglich eine hohe Performance. Die Hardwarenähe, d. h. die Anbindung an einen bestimmten Prozessortyp in Verbindung mit einer spezifischen (Arbeits-)Speicherkonfiguration, ist aber zugleich das wesentliche Handikap eines im Assembler-Code beschriebenen Programmes - es ist wenig portabel (Unter Portabilität wird die Fähigkeit eines Programmes bzw. einer Programmiersprache verstanden, unverändert auf unterschiedlichen Hardware- und Betriebssystem-Plattformen zu funktionieren)

Schon in den 60er-Jahren wurden dann Programmiersprachen entwickelt, die stärker von der zu lösenden Aufgabe als von dem zu lösenden Problem ausgingen. Es entstanden die problemorientierten Programmiersprachen, wie z. B. BASIC (***B**eginners **A**ll purpose **S**ymbolic **I**nstruction **C**ode*). Die Verwendung dieser Sprachen war und ist mit der Entwicklung der Methode der Strukturierten Programmierung verbunden und bedeutete für die Programmierer vor allem eine Schule hinsichtlich der exakten Darstellung eines Problems als Gesamtheit aller zugehörigen Teilprobleme und der Relationen zwischen diesen Teilstrukturen.

Die ersten (problemorientierten) Programmiersprachen hatten prozeduralen Charakter. Dies bedeutet nichts anderes, als dass die Programme die Lösungsprozedur beschreiben. Und dies sowohl im Gesamtprogramm als auch in jeder abgeschlossenen Einzelstruktur.

Ereignisorientierte Programmierung

Mit der Abkehr von der Befehlszeile, beispielsweise unter MS DOS, hin zur Verwendung grafischer, mausgesteuerter Benutzeroberflächen, wie z. B. MS WINDOWS, wurde ein anderer Ansatz von Programmierung wünschenswert, ja sogar notwendig.

Die Erfahrungen der PC-Nutzer bei der Arbeit mit Anwendungen unter MS WINDOWS wurden aufgegriffen. Ein Programm soll jetzt auf ein Ereignis reagieren - nämlich auf Mausklicks auf bestimmte Elemente einer (für das betreffende Programm spezifizierten) Nutzeroberfläche. Als Ergebnis dieser Entwicklung entstanden die objekt- bzw. ereignisorientierten Programmiersprachen.

Im Allgemeinen präsentiert sich ein ereignisorientiertes Programm seinem Nutzer über eine grafische Oberfläche - die so genannte UserForm. Sie beinhaltet alle zur Programmabarbeitung notwendigen Elemente. Solche Elemente sind vor allem Eingabefelder zum Belegen möglicher Variablen, Schaltflächen zur Steuerung durch das Programm (einschließlich einer Schaltfläche für das Programmende) sowie Ausgabefelder für die durch das Programm ermittelten Werte. Nachdem die erforderlichen Eingaben gemacht wurden, wird durch Mausklick (Ereignis) auf ein bestimmtes Element der UserForm das zugeordnete Programmsegment ausgeführt.

In ereignisorientierten Programmen werden daher Objekte definiert, die bestimmte Eigenschaften sowie ein spezifisches Verhalten besitzen. Diese Objekte verständigen sich mit Nachrichten und reagieren aufeinander. In der UserForm werden diese Objekte mit einzelnen grafischen Elementen verknüpft und können dann per Mausklick aktiviert werden.

Als Beispiel sei hier ein Objekt mit dem Namen Kreis definiert. Das Objekt Kreis hat die Eigenschaften Durchmesser und Mittelpunkt und das Verhalten Zeichnen. Die Eigenschaften sind als Variablen im Objekt gespeichert, das Verhalten als Programmcode. Das Objekt Kreis ist somit in sich abgeschlossen. Dies bezeichnet man auch als Verkapselung.

Exkurs: Entwicklung der Programmiersprachen (II)

Das oben gekennzeichnete Objekt Kreis wird im Programm nun so verwendet, dass der Nutzer die Variablen Durchmesser und Mittelpunkt eingibt und dann durch ein Ereignis (Mausklick auf eine Schaltfläche) der Programmcode Zeichnen aktiviert wird. Im Ergebnis dieses Programmes wird dann ein Kreis mit dem eingegebenen Durchmesser um den festgelegten Mittelpunkt gezeichnet.

Das Objekt Kreis kann beliebig oft, an den verschiedensten Steller und in den verschiedensten Zusammenhängen verwendet werden, indem es durch ein entsprechendes Ereignis (Mausklick auf eine Schaltfläche) aufgerufen wird. Die Schaltfläche selbst wiederum ist auch ein Objekt und zwar mit dem Ereignis Click.

Die Geschichte von Visual Basic for Applications

Im Laufe der Entwicklung fügten die Softwareentwickler BASIC immer neue Leistungsmerkmale hinzu. Gleichzeitig wurde der Name des Produktes wiederholt geändert. 1992 brachte Microsoft dann VISUAL BASIC for WINDOWS auf den Markt. Diese Version von BASIC war stark an MS Windows orientiert und verfügte deshalb bereits über alle notwendigen Befehle zur Erstellung und Steuerung von Elementen eines Windows-Programmes wie Menüleisten, Dialogfelder, Symbolleisten, Befehlsschaftflächen, Drop-down-Listen und so weiter.

Gleichzeitig wurden Möglichkeiten für die Objektverknüpfung und -einbettung (OLE), für den dynamischen Datenaustausch (DDE) sowie für die Datenübertragung in andere Windows-Anwendungen bzw. die gemeinsame Nutzung von Daten geschaffen. Die Entwicklung hatte damit eine Stufe erreicht, auf der man VISUAL BASIC als wirklich neue Programmiersprache für Windows bezeichnen kann, trotz ihres Ursprungs in BASIC.

Die Programmiersprache VISUAL BASIC wurde zudem die Basis der Makrorekorder der verschiedenen Host-Anwendungen (WORD, EXCEL ACCESS usw. - ***host***-englisch für Wirt; hier also im Sinne von Wirtsanwendung). Dies führte wegen des Wunsches der Benutzer dieser Anwendungen nach immer universelleren und doch einfach zu handhabenden Makros dazu, dass einerseits die Makrosprachen immer komplexer und ausgereifter wurden und andererseits eine stärkere Differenzierung von Anwendung zu Anwendung erfolgte. So entstand WordBasic als Sprache für Word für Windows vor der Version WORD '97 und z. B. AccessBasic für Access 2.0.

Damit der Anwender nicht faktisch mehrere Sprachen lernen muss, entwickelte Microsoft eine spezielle Version von VISUAL BASIC, das VISUAL BASIC for APPLICATIONS (VBA). Als erste Software wurde EXCEL in der Version 5.0 mit dieser Makrosprache ausgeliefert. Heute ist VBA im MS OFFICE gemeinsame Makrosprache für WORD, EXCEL, ACCESS und PowerPoint. VBA und Visual Basic für Windows stimmen im Hinblick auf Strukturen, wesentlicher Lexik und Syntax weitgehend überein.

Betrachtet man die verschiedenen Ansprüche, die die einzelnen Anwendungen an die Makrosprache stellen, ist es darüber hinaus nur folgerichtig, dass verschiedene Implementierungen von VBA existieren. So enthält das WORD-VBA Befehle, die nur der Veränderung von Texten in Dokumenten dienen und EXCEL-VBA verfügt über Befehle, die ausschließlich für Arbeitsblätter interessant sind. Im Hinblick auf die Speicherung werden VBA-Makroprogramme als Objekte der Host-Anwendung behandelt, aus deren Umgebung sie entwickelt wurden. Folglich lassen sich Makroprogramme nur aus der jeweiligen Host-Anwendung heraus starten.

Nutzen der Hilfe in VBA-Excel

Ein guter Weg sich VBA zu nähern ist das Nutzen der online-Hilfe. Mit ein wenig Gewöhnung kann der interessierte User sich schrittweise der gewünschten Problemlösung nähern.

Aufgabe:

Erstellen Sie die Prozedur test () in einem Modul des VBA-Editors. Starten Sie die online-Hilfe (F1) und kopieren sie das gegebene Beispiel in den Quelltext der Prozedur test (). Zum besseren Nachvollziehen fügen Sie die Befehlszeile Range("A4") = Wert1 ein.

Starten Sie die Prozedur (z. B. Zuweisen des Makros auf eine Schaltfläche im EXCEL-Tabellenarbeitsblatt) und erläutern Sie die einzelnen Schritte der Prozedur.