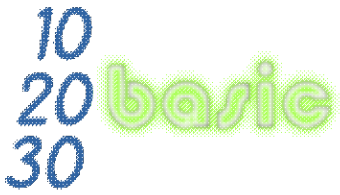
BASIC Programmierung mit Unix



by John Perr
<johnperr(at)Linuxfocus.org>

About the author:

Linuxuser seit 1994, er ist einer der französischen Editoren von LinuxFocus.

Abstract:

Entwickeln in BASIC mit Linux oder einem anderen Unix−System? Warum nicht? Verschiedene kostenlose Lösungen erlauben die Anwendung der BASIC−Sprache zur Entwicklung von interpretierten oder kompilierten Anwendungen.

Translated to English by:
Georges Tarbouriech
<gt(at)Linuxfocus.org>

Einführung

Obwohl BASIC später als andere Sprachen in der Computerlandschaft erschien, verbreitete es sich schnell in vielen Nicht−Unix−Systemen − als Ersatz für die unixeigenen Scriptsprachen. Hier liegt wahrscheinlich der Hauptgrund, warum diese Sprache selten von den Unixbenutzern angewendet wird. Unix hatte vom ersten Tag an eine mächtigere Scriptsprache. Wie andere Scriptsprachen ist BASIC grösstenteils eine interpretierte Sprache mit einer recht einfachen Syntax, ohne Datentypen − ausser der Unterscheidung zwischen Strings und Zahlen. Der Name entsprang der Einfachheit der Sprache und ihrer leichten Lehrbarkeit für angehende Programmierer.

Unglücklicherweise führte der Mangel an Standardisierung zu vielen verschiedenen Versionen, die meistens nicht miteinander kompatibel sind. Man kann sogar sagen, dass es so viele Versionen wie Interpreter gibt − dass macht BASIC nicht gerade portabel.
Trotz dieser und anderer Nachteile, an die uns die "echten" Programmierer erinnern werden, bleibt BASIC eine Möglichkeit, um schnell kleine Programme zu entwickeln. Das hat sich viele Jahre bewahrheitet, besonders dank des Integrated Development Environment in Windows, das ein grafisches Benutzeroberfläche−Design mit ein paar Mausklicks ermöglicht. Ausserdem sind diese Versionen ("Visual" genannt) als Makro−Sprachen für Produktivitäts−Anwendungen verwendet worden. Das führte zur Verbreitung unter Benutzern, die sich sonst nie ans Programmieren gewagt hätten.
Hier werden wir die verschiedenen für Linux verfügbaren Versionen vorstellen oder zumindest die bekanntesten und versuchen, sie miteinander zu vergleichen.

Etwas Geschichte

Aus Andrea M. Marconi's Historie in der KBasic Dokumentation.

Die Geburt von BASIC

Die Basic−Sprache (Beginners All−Purpose Symbolic Instruction Code) wurde 1964 im Dartmouth−College in New Hampshire (USA) geboren, wo sie von John G. Kemeney (1926−93) und Thomas E. Kurtz (1928−) entwickelt wurde. Kemeney arbeitete zuerst am Manhattan−Projekt (1945) und später (1948−49) als Albert Einsteins Assistent. Hier traf er 1956 Kurtz in Dartmouth.

Beide begannen an einer neuen Prgrammiersprache zu arbeiten. Und nach den sogenannten Darsimco und DOPE Versionen wandten sie sich einer Sprache mit der folgenden Spezifikation zu:

1. Allgemeine Anwendbarkeit
2. Leicht zu benutzen
3. Erweiterbar
4. Interaktiv
5. Debugger eingeschlossen
6. Leistungsfähig
7. Unabhängig von der Hardware
8. Unabhängig vom Betriebssystem

Um das zu erreichen, begannen sie mit FORTRAN und ALGOL. Die endgültige Version wurde Dartmouth BASIC genannt und enthielt 14 Instruktionen. Das Dartmouth Basic war eine kompilierte, für die damalige Zeit recht schnelle Sprache.

Die Anstrengungen von Kemeney und Kurtz fanden ihre Belohnung am 1. Mai 1964 um 4 Uhr morgens, als zwei BASIC−Programme gleichzeitig auf dem General Electric 225 UC des Dartmouth−College liefen.

BASICs Wachstum

Kemeney und Kurtz schüzten ihre Erfindung nicht mit einem Patent, sonders liessen sie in der Public Domain. Das ermöglichte deren Wachstum und führte zu mehreren Versionen. Unter den ersten Benutzern finden wir General Electric, die den GE−255 nach Dartmouth verkauft hatten.

Gordon Eubanks (CEO von Symantec) gehört zu den Schöpfern von mehren BASIC−Ausführungen, darunter E−BASIC aus dem Jahr 1970. Es benutzte einen Pseuodocode, wie das heutige Java. Danach entstand CBASIC und viele andere Versionen, was 1974 das ANSI veranlasste, Standards zu definieren. Diese wurden seit ihrem Erscheinen 1978 nie befolgt, BASIC war zu dieser Zeit schon recht verbreitet.

In der Zwischenzeit (1975) schufen Bob Albrecht und Dennis Allison TinyBASIC. Es lief auf 2kb RAM. Es erschien auch die erste interpretierte BASIC−Sprache, sie wurde von William S. Gates III (1955 −, auch als Bill bekannt) und Paul Allen (1953) entwickelt. Die Computerhersteller begannen eine Kopie von BASIC in der ROM ihrer Maschinen zu installieren. Am Ende der 70iger erhalten die ersten Personal Computer ihre

Version des BASIC:

- Radio Shack Level 1 BASIC (TRS 80)
- Apple Integer BASIC (Apple II, 1977)
- Timex Sinclair 1000 BASIC (Sinclair ZX80, 1980)
- Sinclair ZX81BASIC (Sinclair ZX81, 1981)
- PET BASIC (Commodore PET, 1977)
- Atari BASIC (Atari 400/800, both 1978)
- Commodore BASIC (VIC 20 in 1981 & C64 in 1982)
- TI−BASIC (Texas TI−99)
- etc...

Die Evolution von BASIC

Seit dem Beginn der 80iger ist die Geschichte von BASIC eng mit denen der Computern verknüpft. Zwei Produkte dominieren den Markt. IBM liefert BASIC A mit PC−DOS (interpretiert und in ROM, jedoch erweiterbar). MS−DOS erscheint mit Gee−Wiz BASIC (oder GW).
In 1984 taucht der Microsoft BASIC Compiler auf, gefolgt von zahlreichen Versionen der QuickBASIC−Serie, die 1985 begann und 1990 mit dem Microsoft BASIC Professional Development System 7.1 beendet wurde.

Wiederum führen die Betriebssysteme zu Änderungen der Sprache durch die Einführung der grafischen Oberfläche. Visual BASIC erscheint, entworfen, um grafische Benutzeroberflächen (GUI) zu entwickeln. Visual BASIC behauptet, eine Objektsprache zu sein, was von vielen Programmierern bestritten wird. Eine kürzliche Umfrage schätzt jedoch, dass für die Entwicklung von 90% der Programme für Windows 9x Visual BASIC eingesetzt wurde.

Die heutigen BASICs

Wenn wir uns bemühen, eine Bestandsaufnahme der BASIC−Versionen für Linux aufzustellen, finden wir etwa ein halbes Dutzend mehr oder weniger fortgeschrittene Projekte. In der "Basic−Foundry" in Sourceforge finden wir eine Kategorie mit der Anzahl der Downloads:

TOP DOWNLOADS vom Sourceforge.net

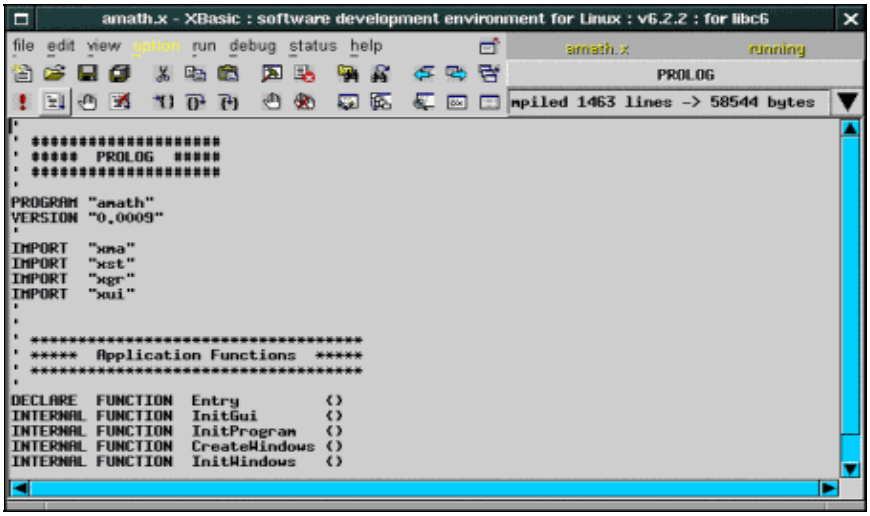
1. XBasic
2. SmallBASIC
3. wxBasic
4. GNU/Liberty Basic
5. YaBASIC
6. X11−Basic

Wir erfahren auch, dass Gnome und KDE beide ein Projekt haben, um Visual Basic zu ersetzen. Außerdem spricht Richard Stallman über die Notwendigkeit einer freien Alternative zu VB im Letter to "the register",vom 10. February 2002.
Anthony Liguori (ajl13−at−bellatantic.net), Autor von GLBCC (GNU/Liberty Basic Compiler Collection), dem einzigen BASIC−Project mit einer GNU Marke, fordert das ebenfalls auf der GLBCC Website (lbpp.sourceforge.net) die bei Sourceforge zu finden ist.

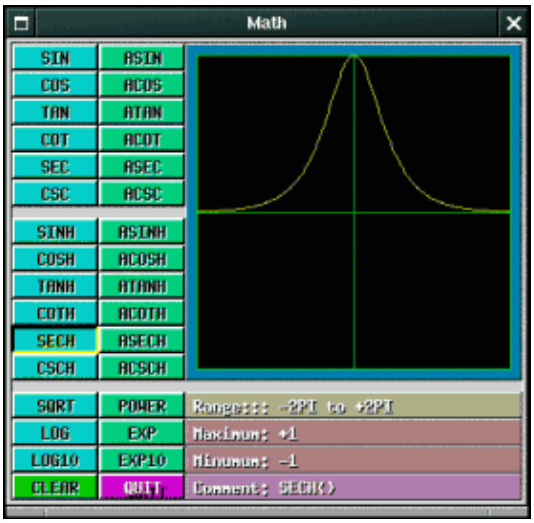
Werfen wir einen Blick auf die verschiedenen Basics, um einen Eindruck von deren Benutzeroberflächen und den Möglichkeiten dieser Sprachen zu erhalten:

XBasic

XBasic ist wahrscheinlich die Entwicklungsumgebung mit einer der fortgeschrittendsten Benutzeroberfläche. Es ist ein IDE, d.h. es enthält ein Tool zum Erstellen von grafischen Benutzeroberflächen, einen Debugger und einen Compiler. Ein "dot−commands"−System (Punkt−Befehle−System) erlaubt die Benutzung der Tastatur anstelle der Maus, um die Oberfläche zu verwalten, durch Schreiben von Befehlen gefolgt von einem Punkt (dot) im Texteingabefenster links oben im Hauptfenster:



XBasic, Darstellung 1a: Das Hauptfenster



XBasic, Darstellung 1b: Die Mathematik Anwendung

Was die Features angeht, so enhält XBasic alle notwendigen Bibliotheken, um grafische Benutzeroberflächen und viele Erweiterungen zu programmieren. Erwähnt sei die A Fähigkeit, Funktionen, die in C geschrieben sind, aufzurufen. Viele Features von C sind verfügbar wie z.B. Typdeklaration, Beziehungen von Variablen und Aufbau von Bibliotheken.
Abschliessend, XBasic ist unter der GPL für Windows oder Linux erhältlich bei: http://www.xbasic.org/

SmallBASIC

SmallBASIC ist ein Textmodus–Interpreter für Win32, Linux und PalmOS. Der Entwicklerteil ist sehr gut dokumentiert, um die Portierung zu anderen Betriebssystemen zu erleichtern. Der Interpreter kann für andere Oberflächen kompiliert werden:

- SVGALIB
- Frame Buffer
- SDL

Es kann im Text– oder Grafikmodus benutzt werden. Das folgende Beispiel führt das System_infos.bas–Programm aus:

Konsolemodus

```
$ sbasic  System_info.bas
SmallBASIC version 0.8.2, use -h for help
http://smallbasic.sourceforge.net

VMT Initialization...
File: System_info.bas
Section: Main
PASS1: Line 24; finished
SB-MemMgr: Maximum use of memory: 30KB

PASS2: Node 3/3
Creating byte-code...
Variables  20
Labels     0
Proc/Func  0
Code size  707

System Information

OS:     Unix/Linux version 2.4.18-6mdk (quintela @ bi.mandrakesoft.com)
       (gcc version 2.96 20000731 (Mandrake Linux 8.2 2.96-0.76mdk))
       #1 Fri Mar 15 02:59:08 CET 2002 204018
SB:     802
Display 99x58
Colors  16
Font:   1x1

Total free memory: 127728 KB
Stack size: 127728 KB
Largest free memory block: 127728 KB

 * DONE *

SB-MemMgr: Maximum use of memory: 30KB
$
```

Grafikmodus

```
$ sbasic -g  System_info.bas
SmallBASIC version 0.8.2, use -h for help
http://smallbasic.sourceforge.net


VMT Initialization...
File: System_info.bas
Section: Main
PASS1: Line 24; finished
SB-MemMgr: Maximum use of memory: 30KB

PASS2: Node 3/3
Creating byte-code...
Variables  20
Labels     0
Proc/Func  0
Code size  707

SB-MemMgr: Maximum use of memory: 30KB
$
```

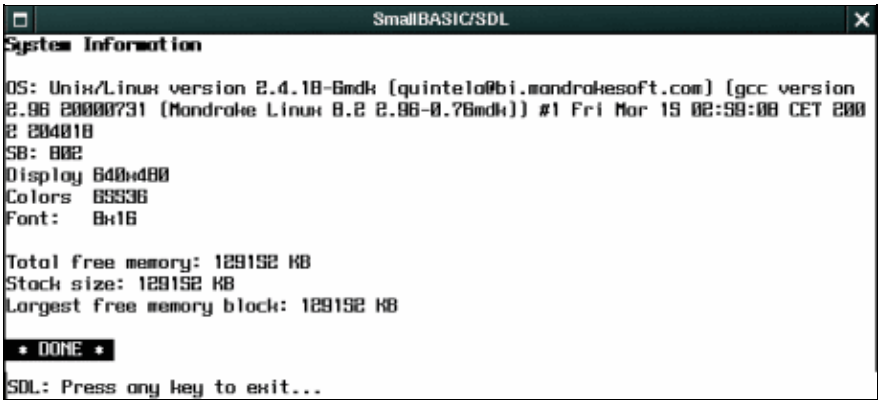


Abbildung 2: SmallBASIC. SDL Grafikmode.

Die SmallBASIC−Sprache ist recht einfach und hält sich eng an die BASIC−Standardfunktionen. Die grafischen Funktionen enthalten nichts Neues und wir finden die klassischen RECTANGLE und CIRCLE ausführbar auf jedem der oben genannten Betriebssysteme. Es gibt keinen Variablentyp.
SmallBASIC ist jedoch kompatibel mit dem alten TINYBasic und QuickBasic und es ist auch bestens in das PalmOS integriert.
Es ist erhältlich bei http://smallbasic.sourceforge.net/

wxBasic

wxBasic sollte QuickBasic − und einige Unix−Eigenschaften haben, wie die Assoziative Arrays, die man in awk findet. Es ist ein kleiner Interpreter: es passt auf eine Floppy. Die Dokumentation ist vollständig und ist als 138−seitiges pdf−Handbuch erhältlich. Die Sprache enthält eine Grafikbibliothek und ermöglicht Programme für X Window oder Windows zu schreiben. Sie ähnelt objektorientierten Sprachen wie C++, zumindestens beim GUI−Design − diese müssen jedoch von Hand entwickelt werden. Eine integrierte Benutzeroberfläche scheint für dieses BASIC nicht erhältlich zu sein.

```
// My first wxBasic demo...
option explicit

// create the window
dim frame=new wxFrame(0,-1,"wxBasic App",wxPoint(10,10),wxSize(300,200))
frame.Centre()

// place a panel in the window
dim panel = new wxPanel(frame, -1)

// add a status bar
dim sBar = frame.CreateStatusBar( 1 )
sBar.SetStatusText("wxBasic Frame Demo")

// attach a menubar to the window
dim mBar = new wxMenuBar()
frame.SetMenuBar(mBar)

// build the "File" dropdown menu
dim mFile = new wxMenu()
mBar.Append(mFile,"&File")

// populate it
mFile.Append(wxID_NEW, "&New", "Creates a new file")
mFile.Append(wxID_OPEN, "&Open", "Loads an existing file from disk")
mFile.Append(wxID_SAVE, "&Save", "Saves current file")
mFile.Append(wxID_SAVEAS, "Save &As", "Saves current file with new name")
mFile.AppendSeparator()
mFile.Append(wxID_EXIT, "&Exit", "Exit Application")

// build the "Edit" dropdown menu
etc.....
```

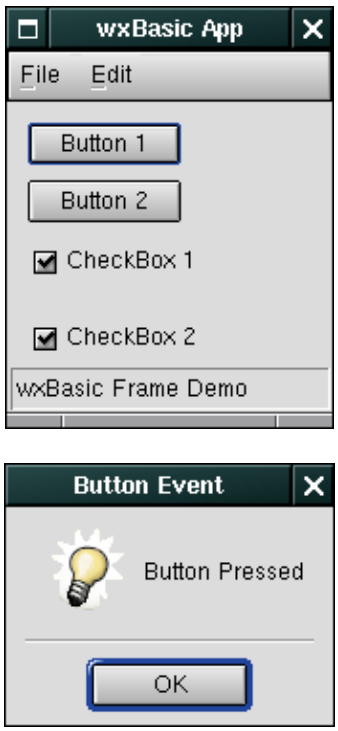


Abbildung 3: wxBasic: Eines der Demo–Programme

Webseite : http://wxbasic.sourceforge.net/

GNU/Liberty Basic

Auch GLBCC (GNU Liberty Basic Compiler Collection) genannt. Es ist ein Compiler oder besser gesagt ein "C GNU gcc"−Compiler−Interface, das BASIC in C konvertiert − etwa wie einige C++ −Compiler, die C++ in C umwandeln, bevor sie kompilieren. Wie der Name schon andeutet, ist dieser Compiler nur für Sprachen geeignet, die 100% LibertyBasic−kompatibel sind. LibertyBASIC ist eine von den Sprachen, die in den 90igern auf der Windowsplattform erschien und sich dank ihrer freien Erhältlichkeit (daher der Name) weit verbreitete. Wer mehr darüber wissen will, findet es auf dieser Webseite Dort wird sie für ihre grossartige Qualität gepriesen. Die Sprache ist nicht frei, aber eine Windowsversion von LibertyBASIC kann gratis von der Webseite heruntergeladen werden.

Der GLBBC−Compiler ist für Windows und Linux erhältlich, er ist fähig, ausführbare, selbstständige Programme zu erzeugen, deren Geschwindigkeiten sich mit denen jeder anderen Sprache messen können. Die Autoren verkünden lauthalsig, dass LibertyBasic−Code, kompiliert mit GLBCC, VisualBasic − was Geschwindigkeit anbetrifft − in den Schatten stellt.

GLBCC ist recht einfach in Linux zu installieren, es benötigt nur das klassische "tar" zum Dekomprimieren und ein "make install" .
In Standardmodus wird das Programm mittels Befehlszeile benutzt und die Eingabe von 'glbcc hello.bas' führt zu der folgenden ausführbaren Datei:

```
$ glbcc
/usr/local/bin/lbpp -I/usr/local/lib/glbcc-lib/0.0.7/include -o out.c hello.bas
gcc -g -I/usr/local/lib/glbcc-lib/0.0.7/include `gnome-config --cflags gnomeui`
    -o hello out.c /usr/local/lib/glbcc-lib/0.0.7/lib/lbcrt0.o
    -L/usr/local/lib/glbcc-lib/0.0.7/lib -lLB
    -lm `gnome-config --libs gnomeui`
$ ls -l hello*
-rwxr-xr-x  1 john  john  339671 oct 13 21:55 hello
-rw-r--r--  1 john  john      22 avr 14 17:41 hello.bas
$ cat hello.bas
print "Hello, world!"
$ ./hello
Hello, world!
```

Ohne Eingabe von Parametern öffnet GLBCC ein Dialogfeld und fragt den Benutzer nach dem Namen einer BASIC−Datei und den Namen der ausführbaren Datei, die erzeugt werden soll. Als Default wird der Eingabename der Datei mit einer .exe−Erweiterung für Windows und ohne Erweiterung für Linux ausgegeben.

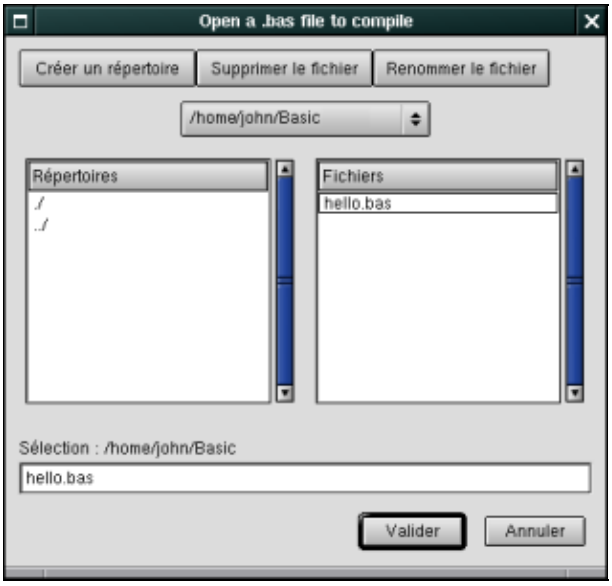


Abbildung 4: GNU/Liberty Basic

Dieses BASIC ist als Sprache vollständig, die Benutzeroberfläche ist auf der GTK−Bibliothek aufgebaut. Der Spass an der Sache ist, dass GLBCC selbst in BASIC geschrieben ist.

YaBASIC

Yet another Basic for Unix and Windows (Noch ein weiteres Basic für Unix und Windows)

Obwohl YaBasic sich wie ein Interpreter benimmt, ist es doch keiner. Es ist eher ein Compiler: geben wir Quellcode zum Verarbeiten ein, kompiliert es diesen in Maschinencode und führt diesen unverzüglich aus. YaBasic wird über die Befehlszeile benutzt. Mit dem Dateinamen als Argument führt es die Datei wie beschrieben aus. Ohne Argument geht es – wie im nachfolgenden Beispiel – in den Interpretermodus :

```
$ yabasic
Enter your program, type RETURN twice when done.
for i=1 to 10
   print i,"hello, world"
next i

1hello, world
2hello, world
3hello, world
4hello, world
5hello, world
6hello, world
7hello, world
8hello, world
9hello, world
10hello, world
$
```

Diese Arbeitsweise erlaubt es, Unixscript oder sogar CGI−Scripts mit YaBasic zu schreiben, solange wir "#!/bin/yabasic" als die erste Zeile der Datei einfügen, wie klassisch in Unix−Shells und sie ausführbar machen.

Als eine Sprache ist YaBasic ein Standard−BASIC (d.h. sie ist nahe an Quick BASIC) ohne Variablentypen. Es reicht aus, die Unterschiede zwischen Strings und Zahlen durch Anwendung des $ −Zeichen am Ende der Variablenamen zu markieren. Hier gibt es kein objektorientiertes Programmieren, es besteht jedoch die Fähigkeit, eigene Bibliotheken aufzubauen. Gemäss des OS besteht die Möglichkeit des Zugangs auf den Grafikmodus von X11 und Windows. Wir finden weitere nützliche Funktionen wie split() − in Perl oder PHP zu finden − die FOR−, WHILE− oder REPEAT UNTIL − Schleifen, sowie die IF−, ELSE− und ENDIF− Anweisungen, letztere sind seit den 80igern, als sich strukturierte Programmierung verbreitete, Teil der BASIC−Sprache.

X11−BASIC

X11−BASIC ist ein strukturierter BASIC−Interpreter mit Grafikeigenschaften von X11. Es nutzt die Möglichkeiten des GFA BASIC vom Atari ST und ist der Port den BASIC im Atari ST GEM/TOS − System (am Ende der 80iger) zu Unix benutzte. X11−BASIC kann als Interpreter, zum Schreiben von Scripts oder CGIs verwendet werden. Ein Pseudo−Compiler ist verfügbar und gestattet eigenständige ausführbare Programme (Statikmodus) zusammenzustellen oder zu der X11−BASIC−Bibliothek (etwa 200kb, dynamischer Modus) zu linken. Es ist ein Pseudo−Compiler, da die entstehende Datei keinen Maschinencode, der direkt von der CPU verarbeitet werden kann, enthält, sondern eine komprimierte Art von BASIC mit seinem Interpreter. Dieser Pseudo−Compiler selbst ist in X11−BASIC geschrieben.
Dieses BASIC hat eine sehr reiche, strukturierte Sprache mit Variablentypen (integer, floating, text, array, boolean). Es enthält Befehle zum Speicherzugang, z.B. C malloc() oder um Matrix' für Arrays zu vervielfältigen.
Die Grafikanweisungen sind die vom Atari ST GFA BASIC, aber ergeben jetzt das gleiche Resultat in X. Die MENU−Befehle z.B. erstellen ein Menü in einem Grafikwindow. Dieser Interpreter enthält eine Dokumentation mit vielen Beispielen, einige davon noch nur in Deutsch. Ein Nachteil: das Portieren ist noch nicht komplett und Bugs können selbst bei den beigefügten Beispielen auftreten. Sagen wir mal, dieser Interpreter benimmt sich wie eine Betaversion, die ohne grosse Mühe von der Kategorie von BASIC−Benutzern benutzt werden kann, auf die sie abzielt.

```
$ xbasic
************************************************************
*               xbasic                 V. 1.07             *
*                    by Markus Hoffmann 1997-2002 (c)      *
*                                                          *
* version date:           Wed Sep 25 10:26:29 CEST 2002    *
* library V. 1.07 date:   Wed Sep 25 10:26:29 CEST 2002    *
************************************************************
```

X11 Basic: des Interpreters Heimat

Webseite : http://www−cip.physik.uni−bonn.de/~hoffmann/X11−Basic/

HBasic

Hier haben wir ein Basic, das sofort einen guten ersten Eindruck macht, entweder durch seine Möglichkeiten oder durch die Qualität der Dokumentation, welche 7,7 MB umfasst. Zur Installation benötigt man Qt−3.*, erhältlich von der Trolltech−Webseite (http://www.troll.no/), falls wir keine nagelneue Linuxdistribution haben. Das ist eine vollständige Entwicklungsumgebung mit allen Features von denen ein Programmierer nur träumen kann (trotz seiner "Jugend": Version 0.8) :

- Integrierte Entwicklungsumgebung, GUI –Werkzeug und Eigenschafteneditor.
- Laden und Benutzung von vorgegebenen Paketen, um Formen oder grafische Objekte in Programme einzuschliessen.
- Quellcode–Editor mit Syntax–Hervorhebung, – Vervollständigung und Modulzusammenfassung.
- Bietet einen Interpreter, um das Kompilieren während des Testens zu vermeiden.
- Der Compiler ist integriert und erstellt wirkliche ausführbare Dateien.
- Ein .NET–code Compiler kann in der .NET–Umgebung benutzt werden.
- Debugger: ermöglicht Haltepunkte, einen Viewer für Variableneigenschaften (bei der Ausführung des Programms oder wenn die Maus über Namen der Variablen im Code–Editor geführt wird)
- Die Fähigkeit C++ –Objekte zu schaffen, um HBasic–Programme oder die Entwicklungsoberfläche während der Ausführung zu erweitern.
- Objektorientierte Sprache für die Klassen, die durch den HBasic–Quellcode oder für die Komponenten der geladenen Pakete definiert sind.
- Werkzeuge für das Management der integrierten Datenbanken. Diese erlauben den Zugriff auf die Daten aus dem IDE oder durch das Programm.
- Erstellung und Benutzung von Instanzen der Qt–Klassen.
- .NET–Unterstützung : gestattet durch die Benutzung des gesammten Umfangs der .NET–Bibliotheken den Zugang zu deren Methoden, Eigenschaften und Feldern.
- Editieren und Kompilieren von C# –Programmen aus dem HBasic–IDE.
- Tabellenkalkulation und Grafiken (noch im Alpha–Status).

Die HBasic–Autoren warnen uns: *"Die gegenwärtige Version von HBasic ist nicht stabil genug, um von BASIC–Entwicklern benutzt werden zu können. Sie werden auf die Herausgabe der ersten stabilen Version1.0 warten müssen."* .

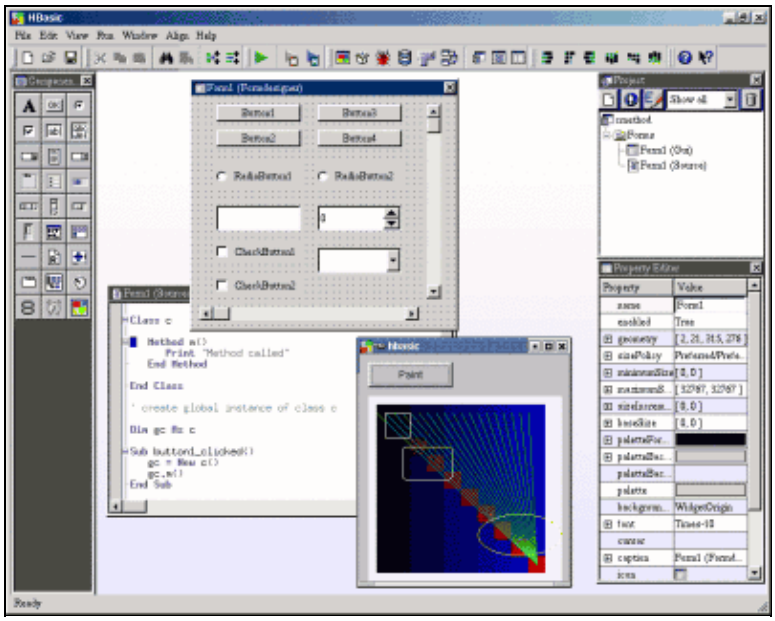


Abb. 5:Hbasic

GNOME Basic

Das GNOME Basic–Projekt setzt sich die 100%ige Kompatibilität mit VisualBASIC, VBA, VBscript und anderen verwandten Sprachen zum Ziel. Am Anfang war vorgesehen, sie zum Erzeugen von gnumeric–ausführbaren VBA–Makros einzusetzen. Wegen unerwarteter Änderungen kann sie jedoch nur

einige Formen ausführen, prüft den grössten Teil des VB−Code und führt einzelne ASPs aus. Volle Kompatibilität mit einer MS−kompatiblen Umgebung kann durch eine bessere Integration in Bonobo erwartet werden.
Gegenwärtig ist GNOME Basic der Ansatz, dem Gnome−Projekt VB−kompatibile Eigenschaften zu geben, besonders für VBA−Anwendungen.
Das Projekt ist noch in der Vor−Alpha−Phase und wird für die Programmierer des Gnome−Projekts reserviert.

Webseite : http://www.gnome.org/gb/

KBasic

KBasic ist ein weiter Versuch, ein Visual Basic−kompatibiles BASIC zu entwickeln. Die Entwickler hoffen, die erste stabile Version 1.0 im Sommer 2003 herauszugeben. Gegenwärtig besteht nur eine unstabile Version, die sich auf die Entwicklung beschränkt. KBasic wird wohl die Kdevelop−Entwicklungsumgebung benutzen.

So sieht z.Z. die Download−Version aus :

```
$ kbasic1 --help
Usage: kbasic1 [OPTION]... FILE
      --help                 display this help and exit
  -V, --version              print version information and exit
  -c, --copyright            print copyright information and exit
$ kbasic1 -V

KBasic version 0.8

Copyright (C) 2000, 2001, 2002 The KBasic Team
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE,
to the extent permitted by law.

Visit us on http://www.kbasic.org/

$kbasic1 hello.bas
-- scan --

LINE 00001 PRINT STRING "Hello, world!" EOL
LINE 00002 END

-- parse --

STRING in line 1
EOL in line 1
parsed PRINT

parsed END


-- interpret --

0000:  KBASIC   KBASIC
0006:  VER      1.000000
0011:  OPTION R OFF
0013:  JMP      36
0036:  DIM_STAT "Hello, world!", 1
0055:  JMP      18
```

```
0018:  PUSHS    1 , " Hello, world! "
0023:  PRINT
Hello, world!
0034:  END      execution ended
```

Wie man sehen kann, funktioniert KBasic – es ist jedoch keineswegs einsatzbereit. Die Debuggingmeldungen und der erzeugte Assemblercode sind zur Zeit unvermeidbar.

Webseite: http://www.kbasic.org/

Fazit

Diese Übersicht von BASIC– Interpretern und – Compilern liefert uns einige Informationen. Erstens ist die diskreditierte BASIC–Sprache überraschender Weise noch am Leben, selbst auf Unixsystemen, wo ihr Gebrauch eher im Verborgenen stattfindet – Dank der vielen Konkurrenten (perl, python, Tcl/TK,php ...), die auch im Interpretermodus laufen. Die Tätigkeiten mit dieser Sprache unter den Entwicklern von freier Software sind bemerkenswert. Es bestehen sehr viele aktive Projekte. Eine grosse Nachfrage scheint zu bestehen, was der freien Softwareentwicklung zu gute kommt – freie Unixsysteme ziehen BASIC–Programmierer an.
Auf Windowssystemen ist VisualBasic sehr verbreitet wegen der Integration im Microsoft OS, der ASPs und der Produktivitäts–Suites. Laßt uns jedoch bemerken, dass die meisten BASICs, die wir getestet haben, auf beiden Plattformen funktionieren und manchmal auch noch auf einigen anderen.
Weiterhin hat BASIC noch einige Nachteile. Der Mangel an Standardisierung führte zu vielen Versionen, die nicht miteinander verträglich sind. Der Wunsch einiger neuer Projekte, wie Gnome Basic oder KBasic, den de facto –Standard von Visual Basic anzunehmen, wäre eine gute Idee, wenn VB frei wäre – was jedoch nicht der Fall ist. Das Monopol wäre in dem Fall recht angenehm...
Für die BASIC–Developer bestehen jedoch echte Wahlmöglichkeiten und mehrere Werkzeuge werden bald zur Verfügung stehen. Als IDE scheint HBasic das vielversprechendste zu sein. Während wir noch auf die stabile Version warten, ist XBasic am fortgeschrittendsten. Für Unix–Neuentdecker gestatten SmallBASIC und YaBASIC Scripte oder CGIs zu schreiben, ohne sich in den vielen Lösungsmöglichkeiten des Systems zu verlieren. GLBCC gestatttet das auch und hat zudem den Vorteil, mit dem vorzüglichen GCC kompilierte Programme zu erzeugen. Da jedoch kein IDE und damit kein einfach benutzbarer Debugger vorhanden ist, wird es schwierig, grössere Programme mit diesem Werkzeug zu entwickeln und zu erhalten. Zuletzt bleibt wx–Basic, welches in keine Kategorie fällt, jedoch einige Vorteile aufweist, wie die Fähigkeit grafische Oberflächen zu erzeugen.
Eine weitere Eigenschaft dieser BASICS ist die einfache Überführung von Programmen von einem OS zu einem anderen ohne erneut Kompilieren zu müssen. Die meisten sind für Win32– und die Unix– Plattform erhältlich.
Der BASIC– Entwickler steht vor der schweren Aufgabe: das richtige Tool für die gewünschte Anwendung auszuwählen.

Webpages maintained by the LinuxFocus Editor team © John Perr "some rights reserved" see linuxfocus.org/license/ http://www.LinuxFocus.org	Translation information: fr –––> –– : John Perr <johnperr(at)Linuxfocus.org> fr –––> en: Georges Tarbouriech <gt(at)Linuxfocus.org> en –––> de: Jürgen Pohl <sept.sapins(at)verizon.net>

2005–01–11, generated by lfparser_pdf version 2.51