LinuxFocus article number 277
http://linuxfocus.org

BASIC programming with Unix

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

About the author:

Linux user since 1994, he is one of the French editors of LinuxFocus.

Abstract:

Developing with Linux or another Unix system in BASIC ? Why not ? Various free solutions allows us to use the BASIC language to develop interpreted or compiled applications.

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

Introduction

Even if it appeared later than other languages on the computing scene, BASIC quickly became widespread on many non Unix systems as a replacement for the scripting languages natively found on Unix. This is probably the main reason why this language is rarely used by Unix people. Unix had a more powerful scripting language from the first day on. Like other scripting languages, BASIC is mostly an interpreted one and uses a rather simple syntax, without data types, apart from a distinction between strings and numbers. Historically, the name of the language comes from its simplicity and from the fact it allows to easily teach programming to students.

Unfortunately, the lack of standardization lead to many different versions mostly incompatible with each other. We can even say there are as many versions as interpreters what makes BASIC hardly portable. Despite these drawbacks and many others that the "true programmers" will remind us, BASIC stays an option to be taken into account to quickly develop small programs. This has been especially true for many years because of the Integrated Development Environment found in Windows versions allowing graphical interface design in a few mouse clicks. Furthermore, these versions, called "Visual", have been

used as macro languages in productivity applications, what increased their spreading among many people who otherwise, would never have started programming.
Here, we will review the different versions available with Linux, or at least the most known and we will attempt to compare them.

A bit of history

From Andrea M. Marconi's history, found in the kbasic documentation:

The birth of BASIC

The BASIC language (Beginners All-Purpose Symbolic Instruction Code) was born in 1964 at the Dartmouth college in New Hampshire (USA), where it has been developed by John G. Kemeney (1926-93) and Thomas E. Kurtz (1928-). Kemeney, first worked in the Manhattan Project (1945) and, later (1948-49) as Albert Einstein's assistant. This is when he met Kurtz in Dartmouth in 1956.

Both began to work on a new simplified programming language, and after the so-called Darsimco and DOPE versions, they turned towards a language with the following specifications:

1. General use
2. Easy to use
3. Extensible
4. Interactive
5. With a debugging help system
6. Efficient
7. Hardware independent
8. Operating System independent

To do this they started from FORTRAN and ALGOL. The final version was called "Dartmouth BASIC" and provided 14 instructions. The "Dartmouth BASIC" was a compiled one and was rather fast for the epoch.

The Kemeney and Kurtz efforts were rewarded on May 1st 1964 at 4:00 in the morning, when two BASIC programs simultaneously ran on the General Electric 225 UC of Dartmouth college.

The growth of BASIC

Kemeney and Kurtz did not protect their invention with a patent and left it in the public domain. This allowed its growth and also its increase in the number of versions. Among the first users we can find General Electric which had sold the GE-255 to Dartmouth.

Gordon Eubanks (the CEO of Symantec) is at the origin of various BASICs, among which the E-BASIC emerged in 1970. It used a pseudo code like Java does today. Then came CBASIC and many other versions, what lead the ANSI to define standards in 1974. These last have never been followed since they appeared in 1978, at a time when BASIC was already widespread.

In the meantime (1975), TinyBASIC is created by Bob Albrecht and Dennis Allison. It can run with 2Kb of RAM. The first interpreted BASIC also appears. It is developed by William H. Gates III (1955-, aka Bill) and Paul Allen (1953-). The computers makers start introducing a copy of BASIC in the ROM of their machines . At the end of the 70's, the first personal computers have their version of 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...

The evolution of BASIC

From the beginning of the 80's, the history of BASIC is closely linked to the one of computers. Two products were dominating the market. IBM provides the BASIC A with PC-DOS, interpreted and in ROM, but extensible. MS-DOS provides the Gee-Witz BASIC (or GW).
In 1984, the Microsoft BASIC compiler appears, followed by many versions among which the QuickBASIC series, started in 1985 and stopped in 1990 with the Microsoft BASIC Professional Development System 7.1

Once again, the operating systems bring changes to the language introducing the graphical interfaces. Visual BASIC appears, designed to create graphical interfaces (GUI). Visual BASIC claims to be an object language, what is contested by many programmers. However, a recent survey estimates that 90% of the in programs developed for Windows 9x use Visual BASIC.

The BASIC(s) today

If we try to make an inventory of the number of BASIC available for Linux, we can find about half a dozen of projects, more or less advanced. There is a "Basic Foundry" in sourceforge to give a classification on the number of downloads:

TOP DOWNLOADS from Sourceforge.net

1. XBasic

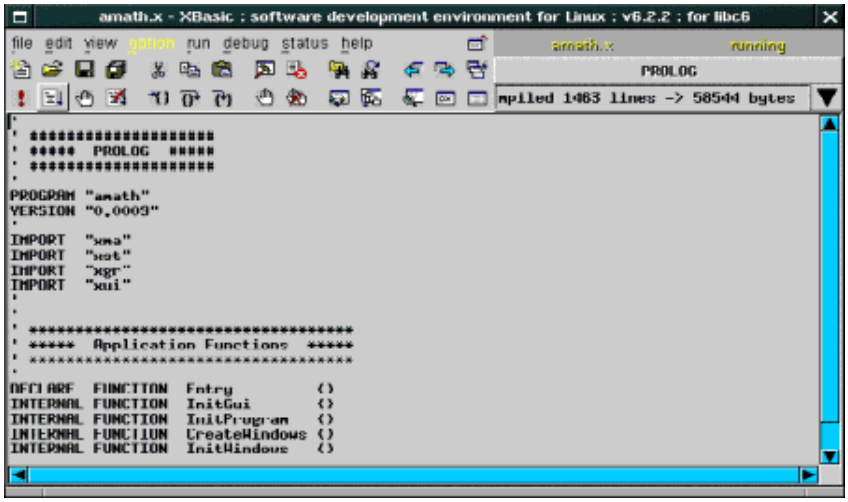
2. SmallBASIC
3. wxBasic
4. GNU/Liberty Basic
5. YaBASIC
6. X11-Basic

We also learn that Gnome and KDE have both a project intended for replacing Visual Basic. Furthermore, Richard Stallman talks about the need of a free alternative to VB in a letter to "the register", on February 10th 2002.
Anthony Liguori (ajl13-at-bellatantic.net), author of GLBCC (GNU/Liberty Basic Compiler Collection) the only BASIC project with a GNU label, also asks for this on the GLBCC website (lbpp.sourceforge.net) hosted by sourceforge.

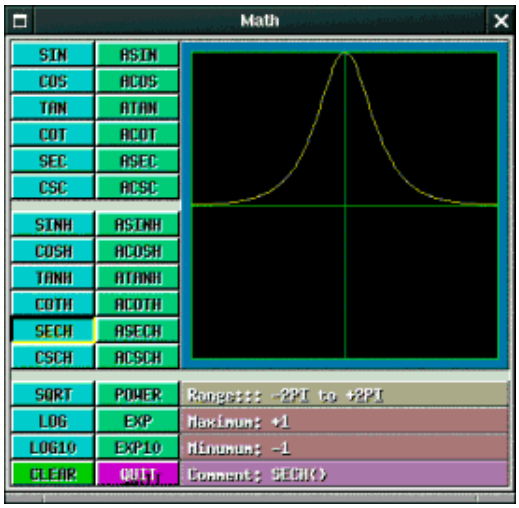
Let us review the different BASIC to get an idea about the look and feel of each interface and about the possibilities of the language:

XBasic

XBasic is probably the development environment with one of the most advanced user interfaces. It is an IDE, that is, it provides a GUI building tool, a debugger and a compiler. A "dot commands" system allows to use the keyboard and not the mouse to manage the interface, writing commands following a dot in the typing area on the top left of the main window:



XBasic, Picture 1a: The main window



XBasic, Picture 1b: The Math application

On the features side, XBasic has all the required libraries to program graphical interfaces and many extensions. Let us mention the ability of calling functions written in C language. Many C language features are available like declaration of type, association of variables or creation of libraries.
Last, Xbasic is available under GPL for Windows or Linux at: http://www.xbasic.org

SmallBASIC

SmallBASIC is a text mode interpreter for Win32, Linux and PalmOS. The developer part is very well documented as to encourage the port to other OSes. The interpreter can be compiled for different interfaces:

- SVGALIB
- Frame Buffer
- SDL

It can run in text mode or in graphical mode. The following example runs the System_infos.bas program:

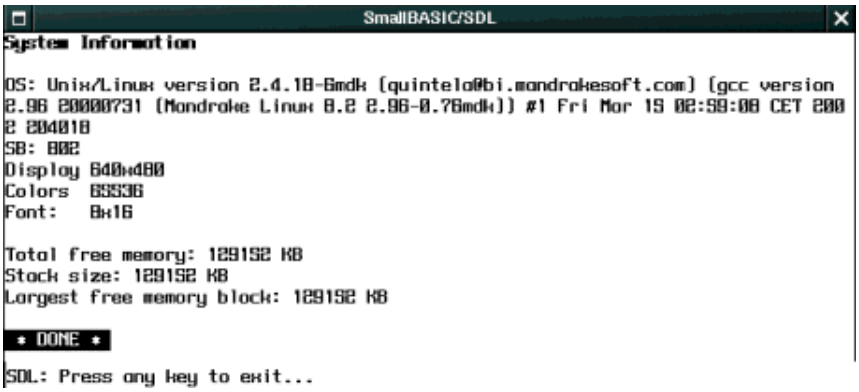
Console mode

```
$ 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
$
```

Graphical mode

```
$ 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
$
```



Picture 2: SmallBASIC. SDL graphical mode.

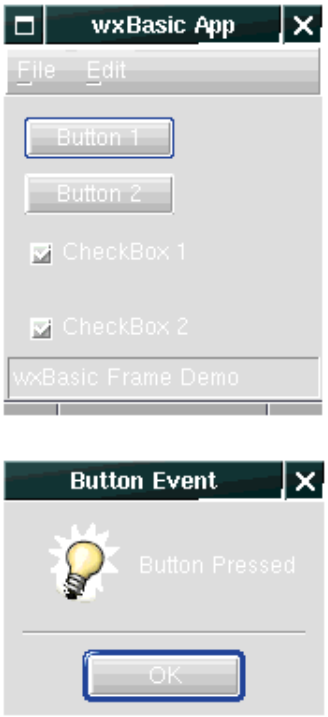
The SmallBASIC language is simple enough and close to standard functions you can expect from BASIC. The graphical functions have nothing new and you can find the classical RECTANGLE and CIRCLE able to be executed on any of the above mentioned OSes. There is no type of variables. However, SmallBASIC is compatible with the old TINYBasic and QuickBasic and is very well integrated into PalmOS.
It is available from http://smallbasic.sourceforge.net

wxBasic

wxBasic is supposed to have Quick Basic features and some Unix specificities such as the associated arrays found in awk. It is an interpreter small in size: it fits onto a floppy disk. The documentation is a complete one and it is available as a pdf manual of 138 pages. The language has a graphical library allowing to write programs for X Window or Windows. This language is close to object oriented languages such as C++, at least for GUI design. However, the graphical interfaces have to be designed by hand. No integrated development interface seems to be available for this BASIC.

```
// 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.....
```



Picture 3: wxBasic: one of the demo programs

Website: http://wxbasic.sourceforge.net

GNU/Liberty Basic

Also called GLBCC (GNU Liberty Basic Compiler Collection) it is a compiler or more exactly a C GNU gcc compiler interface which converts BASIC into C, a bit like some C++ compilers which transform C++ into C before compiling. As its name says, this compiler is dedicated to a language 100% Liberty BASIC compatible. The Liberty BASIC is one of those having appeared in the 90's on Windows platforms and which has been rather successful because of its free availability (hence the name). To know more about it, check this website which sings the praises of its high quality. This language is not free, but a Windows version of Liberty BASIC free of charges can be downloaded from the website.

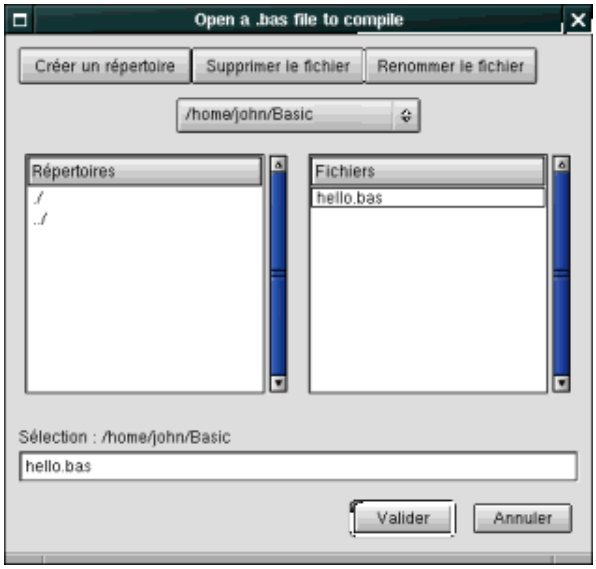
The GLBCC compiler is available for Windows and Linux and is able to provide standalone executables

said to be as fast as the ones programmed in any other language. The authors shout out that Liberty BASIC code compiled with GLBCC can make Visual BASIC "feel" ridiculous as far as speed is concerned.

Installing GLBCC is rather simple under Linux and only requires the classical "tar" to uncompress the archive and a "make install".
In normal mode, the program is used with the command line and typing "glbcc hello.bas" will produce the executable like this:

```
$ 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!
```

When used without parameter, GLBCC opens a graphical dialog and asks the user for the name of a BASIC file and the name of the executable to be produced. As the default, the input name of the program will be added to the output, with an .exe suffix for Windows and no suffix at all for Linux.



Picture 4: GNU/Liberty Basic

As a language, this BASIC is a complete one and the GUI is based on the GTK library. The funny thing is that GLBCC itself, is written in BASIC.

YaBASIC

Yet another Basic for Unix and Windows

Even if it behaves like an interpreter, YaBasic is not an interpreter. It rather is a compiler: when you give it a source code to process, it compiles it, thus changing it to machine code and runs it immediately. YaBasic is used from the command line. With a filename as argument, it executes that file like we just described it. Without argument, it enters the interpreter mode like in the following example:

```
$ 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
$
```

This way of working allows to use YaBasic to write Unix scripts or even CGI scripts, as long as you insert "#!/bin/yabasic" as the first line of the file, like it is classical for Unix shells and making this file executable.

As a language, YaBasic is a standard BASIC (that is, close to Quick BASIC) without variables types. Enough to make a difference between strings and numbers, using or not the $ character at the end of the variable name. No object oriented programming here but the ability to create your own libraries, and instructions allowing to access the graphical modes for X11 and Windows, according to the OS. Some more useful functions are also available such as split(), found in Perl or PHP, the FOR, WHILE or REPEAT UNTIL loops and the IF ELSE ENDIF statement; these last are part of the BASIC language since the 80’s when structured programming appeared.

X11-BASIC

X11-BASIC is a structured BASIC interpreter with X11 graphical abilities. It uses the features of the GFA BASIC found on Atari ST. It is the port of the BASIC used in the Atari ST GEM/TOS system (end of the 80’s) to Unix. It can be used as an interpreter or to write scripts or CGIs. A pseudo compiler is available allowing to create either standalone executables (static mode) or linked to the X11 basic library which is about 200 KB (dynamic mode). It is a pseudo compiler because the resulting file does not hold machine code directly executed by the CPU but instead a compressed form of the BASIC language with its interpreter. This pseudo compiler is itself written in X11-BASIC.
The language of this BASIC is very rich, structured and with typed variables (integer, floating, text, array, boolean). Instructions are available to access memory such as the C malloc() or to multiply matrix applicable to arrays.

The Graphical instructions are the same ones as the Atari ST GFA BASIC but have now a similar result under X. The MENU instruction creates a menu into a graphical window, for instance. This interpreter is bundled with documentation and many examples, still in German in a few places. The dark side is that porting is not yet finished and bugs may appear even with the provided examples. Let us consider that this interpreter mostly behaves like a beta version to be easily used by the category of users the BASIC language is aimed at.

```
$ 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: the interpreter’s home

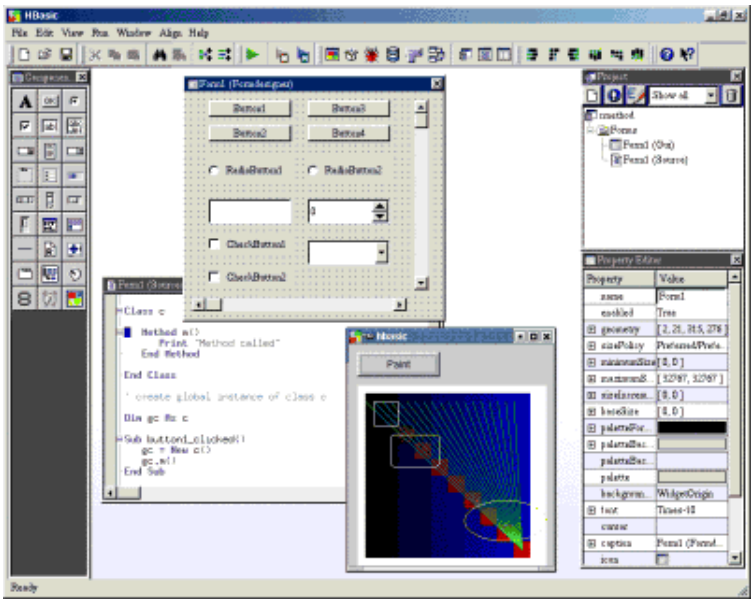
Website: http://www-cip.physik.uni-bonn.de/~hoffmann/X11-Basic/

HBasic

Here is a BASIC making a good impression at first sight, either because of its possibilities or the quality of the documentation which weighs 7.7 MB. The installation requires Qt-3.* available from Trolltech website (http://www.troll.no) if you do not have a brand new Linux distribution. It is a complete development environment which provides all the features a programmer can dream of (despite its "youth": version 0.8):

- Integrated development interface, GUI creator and properties editor.
- Load and use of predefined packages for inclusion of forms and graphical objects into the programs.
- Source code editor with syntax highlighting, completion and modules wrapping.
- Provides an interpreter to avoid compilation when testing.
- The compiler is an integrated one and it creates real executables.
- A .NET code compiler can be run in a .NET environment.
- Debugger: provides breakpoints, a viewer of variables content (either at run time or when moving the mouse over the variable name in the code editor).
- Ability to create C++ objects to extend HBasic programs or the development interface at run time.
- Object oriented language for the classes defined in the HBasic source code or for the components loaded as packages.
- Tools for the management of integrated databases allowing to access the data either from the IDE or from the program.
- Creation and use of instances for Qt classes.
- .NET support: allows to use the full range of .NET libraries to access their methods, properties or fields.
- C# program editing and compiling from the HBasic IDE.
- Spreadsheets and graphs (still in alpha version).

The HBasic authors warn you saying: *"The present version of HBasic is not stable enough to be used by BASIC developers. They will have to wait for the release of the first 1.0 stable version". .*



Picture 5:Hbasic

GNOME Basic

GNOME Basic is a project aiming at being 100% compatible with Visual BASIC, VBA, VBscript, and other associated languages. At the beginning it was dedicated to make VBA macros executable from gnumeric, but because of unexpected changes it only executes a few forms, checks most of the VB code and runs single ASPs. With a better integration in Bonobo, a fully compatible MS compatible environment can be expected.
For now, Gnome Basic is an attempt to provide VB compatible features for the Gnome project, especially for productivity applications (VBA).
The project is still in pre-alpha state and must be reserved for the developers of the Gnome project.

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

KBasic

KBasic is the other attempt to develop a BASIC compatible with Visual Basic. The developers hope to release the first 1.0 stable version for summer in 2003. For now, there is only an unstable version reserved for development. KBasic should use the Kdevelop environment.

Here is what the downloadable version looks like at the moment:

```
$ 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
```

As you can see it, kbasic works but it is far from being operational, the interpreter debugging messages and the generated assembly code are unavoidable for now.

Site: http://www.kbasic.org

Conclusion

This overview of BASIC interpreters and compilers provides us with some information. First of all, surprisingly, BASIC this so discredited language is still alive, even on Unix systems where its use stays rather confidential because of the many competitors (perl, python, Tcl/Tk, php....) also running in interpreted mode. The activity this language generates among the developers of free software is significant. The number of projects is rather high and all of them are very active. The demand is probably strong and this may contribute in helping free software development, free Unixes thus

attracting BASIC programmers.
On Windows systems, VisualBasic is widely used because of its integration into the Microsoft OS, the ASPs and the productivity suites. However, let us note that most of the BASICs we have tested work on both platforms and sometimes on a few others.
Next, BASIC still has a few drawbacks. The lack of standard lead to many versions mostly uncompatible with each other. The wish of some new projects such as Gnome Basic or KBasic to join the VisualBasic de facto standard could be a good idea if VB was free, what it is obviously not. Monopoly would be more convenient in that case...
For the BASIC developer, there is a true choice and a few tools are quite ahead. As an IDE, HBasic is probably the promising one. While waiting for the stable version, XBasic is the most achieved. However, smallbasic, yabasic allow the ones who have just moved to Unix to write scripts or CGIs without having to choose among the numerous solutions provided by this system. GLBCC also allows this but has the advantage of creating programs compiled with the great GCC. Due to the lack of IDE and thus of an easy to use debugger, it will be difficult to develop and maintain big programs with this tool. Last, wx-Basic, which is not part of any category but has some advantages, such as the ability to create graphical interfaces.
Another good point about these BASICs is the easy migration of programs from an OS to the other without recompiling. As a matter of fact, most of them are available for Win32 and Unix platforms.
The BASIC developer is then in front of a heavy task: choose the right tool for the purpose.

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>

2005-01-14, generated by lfparser_pdf version 2.51