

Programando en BASIC con Unix



Abstract:

by John Perr.
<johnperr@linuxfocus.org>

¿Desarrollando con Linux o con otro sistema Unix en BASIC? ¿Por qué no? Varias soluciones libres permiten usar el lenguaje BASIC para desarrollar aplicaciones interpretadas o compiladas.

About the author:

Usuario de Linux desde 1994,
es uno de los editores
franceses de LinuxFocus.

Introducción

Incluso aunque apareció más tarde que otros lenguajes en el ámbito computacional, BASIC rápidamente se extendió a muchos sistemas no Unix como un sustituto a los lenguajes de scripts encontrados nativamente en Unix. Esta es probablemente la razón más importante por la cual este lenguaje es poco usado por la gente de Unix. Unix tenía un lenguaje de scripts mucho más potente desde su aparición. Como otros lenguajes de scripts, BASIC es en su mayor parte interpretado y utiliza una sintaxis bastante simple, sin tipos de datos, aparte de una distinción entre caracteres y números. Históricamente, el nombre del lenguaje proviene de su simplicidad y del hecho de que permite enseñar fácilmente programación a estudiantes.

Desafortunadamente, la falta de estandarización condujo a versiones diferentes la mayoría incompatibles entre sí. Podemos decir incluso que hay tantas versiones como intérpretes lo que convierte al BASIC en difícilmente portable.

A pesar de estas desventajas y otras que los "verdaderos programadores" nos recordarán, BASIC es una opción a tener en cuenta a la hora de desarrollar pequeños programas. Esto ha sido así durante muchos años especialmente debido al Conjunto de Desarrollo Integrado encontrado en las versiones Windows permitiendo el diseño de interfaces gráficas con unos pocos clicks del ratón. Además, estas versiones, llamadas "Visuales", han usado lenguajes de macros en aplicaciones de productividad, lo que incrementó su extensión entre bastante gente que, de otra manera, no hubiera empezado nunca a programar.

Aquí, examinaremos las diferentes versiones disponibles para Linux, o al menos las más conocidas e intentaremos compararlas.

Un poco de historia

De la historia de Andrea M. Marconi, encontrada en la documentación de kbasic:

El nacimiento de BASIC

El lenguaje BASIC (Código de Instrucciones Simbólicas Multi-Propósito para Principiantes) nació en 1964 en la Universidad Dartmouth en New Hampshire (E. U. A.), donde ha sido desarrollado por John G. Kemeney (1926–93) y Thomas E. Kurtz (1928–). Kemeney, que trabajó primero en el Proyecto Manhattan (1945) y más tarde (1948–49) como el ayudante de Albert Einstein. Así fue como conoció a Kurtz en Dartmouth en 1956.

Ambos empezaron a trabajar en un nuevo lenguaje de programación simplificado, y después de las llamadas versiones Darsimco y DOPE, cambiaron hacia un lenguaje con las especificaciones siguientes:

1. Uso General.
2. Facilidad de uso.
3. Extensible.
4. Interactivo.
5. Con un sistema de depuración de ayuda.
6. Eficiente.
7. Independiente del hardware.
8. Independiente del Sistema Operativo.

Para lograrlo empezaron por FORTRAN y ALGOL. La versión final se llamó "Dartmouth BASIC" y venía con 14 instrucciones. El "Dartmouth BASIC" fue una versión compilada y era bastante rápido para la época.

Los esfuerzos de Kemeney y Kurtz fueron recompensados el 1 de mayo de 1964 a las 4 de la mañana, cuando dos programas BASIC corrieron simultáneamente en el 225 UC de General Electric de la Universidad Dartmouth.

El crecimiento de BASIC

Kemeney y Kurtz no protegieron su invención con una patente y la dejaron de dominio público. Esto permitió su crecimiento y también el incremento del número de versiones. Sobre los primeros usuarios podemos encontrar a la General Electric que vendió el GE-255 a Dartmouth.

Gordon Eubanks (el CEO de Symantec) estuvo en el origen de varios BASICs, entre el E-BASIC que surgió en 1970. Usaba un pseudocódigo como Java hace hoy en día. Entonces apareció el CBASIC y muchas otras versiones que condujeron al ANSI a definir los estándares en 1974. Estos últimos no se siguieron hasta que aparecieron en 1978, en un tiempo en el que BASIC ya se había difundido.

Mientras tanto (1975), TinyBASIC es creado por Bob Albrecht y Dennis Allison. Puede funcionar con 2Kb de RAM. También aparece el primer BASIC interpretado. Es desarrollado por William H. Gates III (1955–) y Paul Allen (1953–). Los productores de ordenadores empezaron a introducir una copia del BASIC en la ROM de sus máquinas. Al final de los 70, los primeros ordenadores personales tenían su propia versión

BASIC:

- Radio Shack Nivel 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, ambos de 1978)
- Commodore BASIC (VIC 20 en 1981 y C64 en 1982)
- TI-BASIC (Texas TI-99)
- etc.

La evoluci3n de BASIC

Desde principios de los 80, la historia de BASIC est3 relacionada muy de cerca con la de las computadoras. Dos computadoras dominaban el mercado. IBM incluye BASIC A con PC-DOS, interpretado y en ROM, pero extensible. MS-DOS provee el Gee-Witz BASIC (o GW).

En 1984, el compilador BASIC de Microsoft aparece, seguido por muchas otras versiones entre las que se encuentran las series QuickBASIC, comenzada en 1985 y cuyo desarrollo se par3 en 1990 con el Sistema de Desarrollo Profesional de BASIC de Microsoft 7.1

Una vez m3s, los sistemas operativos llevan cambios al lenguaje introduciendo interfaces gr3ficas (GUI). Visual BASIC demanda ser un lenguaje de objetos, lo que es causa de disputa para muchos programadores. Sin embargo, una encuesta reciente estima que el 90% de los programas desarrollados para Windows 9x usan Visual BASIC.

BASIC hoy

Si intent3ramos hacer un inventario del n3mero de BASICs disponibles para Linux, podr3-amos encontrar una media docena de proyectos, m3s o menos avanzados. Hay una "Fundaci3n Basic" en SourceForge para dar una clasificaci3n al n3mero de descargas:

Los m3s descargados de SourceForge.net

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

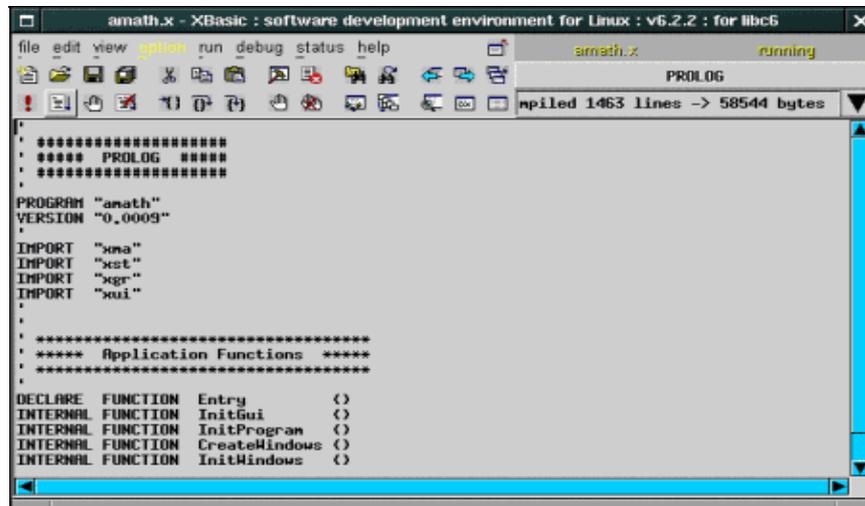
Tambi3n sabemos que GNOME y KDE tienen ambos un proyecto con la intenci3n de reemplazar a Visual Basic. Adem3s, Richard Stallman habla sobre la necesidad de una alternativa libre a VB en [carta a "The Register"](#), el 10 de febrero de 2002.

Anthony Liguori (ajl13-at-bellatantic.net), autor de GLBCC (GNU/Liberty Basic Compiler Collection), el 3nico proyecto BASIC con etiqueta GNU, tambi3n habla de ello en [en la p3gina de GLBCC \(lbp.sourceforge.net\)](#) hospedada por SourceForge.

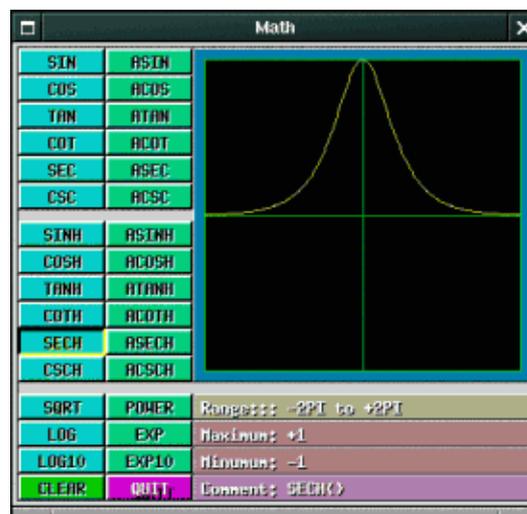
Revisemos los diferentes BASIC para hacernos una idea sobre el aspecto y sensaciones de cada interfaz y sobre las posibilidades de cada una:

XBasic

XBasic es probablemente el medio de desarrollo con uno de las interfaces de usuario más avanzado. Es un IDE, esto es, provee una herramienta de construcción GUI, un depurador y un compilador. Un sistema de "comandos de punto" permite utilizar el teclado y no el ratón para manejar el interfaz, escribiendo comandos seguidos de un punto en el área de escritura a la izquierda de la ventana principal.



XBasic, Imagen 1a: La ventana principal



XBasic, Imagen 1b: La aplicación matemática

En el lado de características, XBasic tiene todas las bibliotecas requeridas para programar interfaces gráficas y muchas extensiones. Debemos mencionar la habilidad de llamar funciones escritas en C. Muchas de las características del lenguaje C están disponibles como la declaración de tipo, asociación de variables o creación de bibliotecas.

Por último, Xbasic está disponible bajo GPL para Windows o Linux en: xbasic.org.

SmallBASIC

SmallBASIC es un intérprete en modo texto para Win32, Linux y PalmOS. La parte de desarrollo está muy bien documentada para animar el portarlo a otros SOs. El intérprete puede ser compilado para diferentes interfaces:

- SVGALIB
- Frame Buffer
- SDL

Puede correr en modo texto o modo gráfico. El siguiente ejemplo corre el programa System_info.bas:

Modo consola

```
$ 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
$
```

Modo gráfico

```
$ 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
$
```



Imagen 2: SmallBASIC. SDL modo gráfico.

El lenguaje SmallBASIC es suficientemente simple y ajustado a las funciones estándar como puedes esperar de BASIC. Las funciones gráficas no tienen nada nuevo y puedes encontrar las clásicas RECTANGLE y CIRCLE capaces de ser ejecutadas en cualquiera de los SOs mencionados arriba. No hay tipos de variable. Sin embargo, SmallBASIC es compatible con el viejo TINYBasic y QuickBasic y está muy bien integrado en PalmOS.

Está disponible en <http://smallbasic.sourceforge.net>

wxBasic

wxBasic es conocido por tener características Quick Basic y algunas específicas de Unix como las matrices asociadas que se encuentran en awk. Es un intérprete pequeño en tamaño: cabe en un disquete. La documentación es completa y está disponible como un manual PDF de 138 páginas. El lenguaje es una biblioteca gráfica permitiendo escribir programas para XWindow o Windows. Este lenguaje es cercano a lenguajes orientados a objetos como C++, al menos para el diseño GUI. Sin embargo, los interfaces gráficos tienen que ser diseñados a mano. No parece que haya disponible un interfaz de desarrollo integrado para este BASIC.

```

// Mi primera demo wxbasic...
option explicit

// Crear la ventana
dim frame=new wxFrame(0,-1,"wxBasic
App",wxPoint(10,10),wxSize(300,200))
frame.Centre()

// Colocar un panel en la ventana
dim panel = new wxPanel(frame, -1)

// AÑadir una barra de estado
dim sBar = frame.CreateStatusBar( 1 )
sBar.SetStatusText("wxBasic Frame Demo")

// AÑadir una barra de menÃ° a la ventana
dim mBar = new wxMenuBar()
frame.SetMenuBar(mBar)

// Construir el menÃ° extensible "archivo"
dim mFile = new wxMenu()
mBar.Append(mFile,"&File")
Trolltech
// Completarlo
mFile.Append(wxID_NEW, "&Nuevo", "Crea un nuevo
archivo")
mFile.Append(wxID_OPEN, "&Abrir", "Carga un
archivo
existente desde disco")
mFile.Append(wxID_SAVE, "&Guardar", "Guarda el
archivo
actual")
mFile.Append(wxID_SAVEAS, "Guardar &Como",
"Guarda el
archivo actual con un nombre diferente")
mFile.AppendSeparator()
mFile.Append(wxID_EXIT, "&Salir", "Salir de la
aplicaciÃ³n")

// Construir el menÃ° extensible "EdiciÃ³n"
etc.....

```

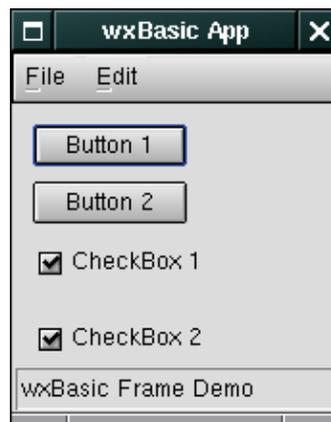




Imagen 3: wxBasic: uno de los programas demo

Sitio web: <http://wxbasic.sourceforge.net>

GNU/Liberty Basic

Tambi n llamado GLBCC (GNU Liberty Basic Compiler Collection) es un compilador o mejor dicho, una interfaz compiladora gcc GNU en C que convierte BASIC en C, como algunos compiladores C++ que transforman C++ en C antes de compilar. Como su nombre indica, este compilador est  dise ado para ser 100% compatible con Liberty BASIC. El Liberty BASIC es uno de los que aparecieron en los 90 en plataformas Windows y el cual ha sido bastante popular debido a su disponibilidad gratuita (de ah - su nombre). Para saber mas acerca del mismo, visita esta [p gina](#) que predica su alta calidad. Este lenguaje no es gratuito, aunque una versi n gratuita para Windows puede ser descargada de la [p gina](#).

El compilador GLBCC est  disponible para Windows y Linux y es capaz de crear ejecutables independientes llamados a ser tan r pidos como los programados en cualquier otro lenguaje. Los autores aseguran que el c digo de Liberty BASIC compilado con GLBCC puede hacer parecer el Visual BASIC rid culo en lo concerniente a velocidad.

Instalar GLBCC es bastante simple bajo Linux y s lo requiere el cl sico "tar" para descomprimir el archivo y hacer un "make install".

En el modo habitual, el programa se utiliza escribiendo en la l nea de comandos, y tecleando "glbcc hello.bas" producir  el siguiente ejecutable:

```
$ glbcc
/usr/local/bin/lbpc -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/lbcbt0.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!
```

Si se utiliza sin ning n par metro, GLBCC abre un di logo gr fico y pregunta al usuario por el nombre de un fichero BASIC y el nombre del ejecutable de salida. Por defecto el nombre del programa se pondr  a la salida, con un sufijo .exe para Windows y sin sufijo para Linux.

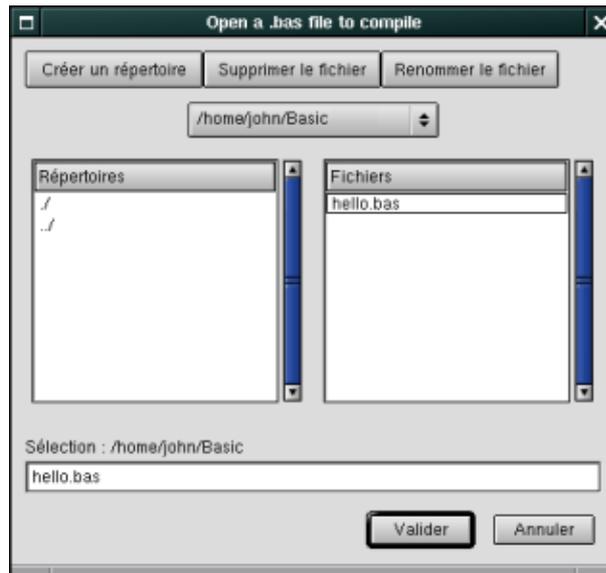


Imagen 4: GNU/Liberty Basic

Como lenguaje, este BASIC está completo y el GUI está basado en la librería GTK. Lo más divertido es que GLBCC en sí mismo, está escrito en BASIC.

YaBASIC

Yet another Basic for Unix and Windows/Otro Basic más para Unix y Windows

Aunque se comporta como un intérprete, YaBasic no es un intérprete. Más que nada es un compilador: cuando se le proporciona un código fuente para procesar, lo compila, lo cambia a código máquina y lo ejecuta inmediatamente. YaBasic se utiliza desde la línea de comandos. Con un fichero como argumento, ejecuta dicho fichero como acabo de describir. Sin argumento, se introduce en modo intérprete como se ve en el siguiente ejemplo:

```
$ 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
$
```

Esta forma de trabajar permite usar YaBasic para escribir scripts Unix o incluso scripts CGI, tan pronto como se introduce "#!/bin/yabasic" como la primera línea del archivo, como es clásico en las ventanas de comandos Unix y que hace que el fichero sea ejecutable.

Como lenguaje, YaBasic es un BASIC estandar (esto es, parecido a QuickBASIC) sin tipos de variables. Es suficiente con diferenciar entre las cadenas y los números, usando o no el carácter \$ al final del nombre de la variable. No hay programación orientada a objetos pero permite crear tus propias bibliotecas e instrucciones permitiendo acceder a los modos gráficos de X11 y Windows, según el SO. Hay disponibles algunas funciones útiles más como split(), que se encuentra en Perl o PHP, los bucles FOR, WHILE o REPEAT UNTIL y la opción IF ELSE ENDIF; estos últimos forman parte del lenguaje BASIC desde que en los 80 aparecieron los lenguajes estructurados.

X11-BASIC

X11-BASIC es un intérprete BASIC estructurado con habilidades X11. Utiliza las características del GFA BASIC del Atari ST. Se trata del BASIC usado en el sistema Atari ST GEM/TOS (finales de los 80) portado a Unix. Puede ser usado como un intérprete o para escribir scripts o CGIs. Está disponible un seudocompilador que permite crear o bien ejecutables independientes (modo estatico) o conectados a la biblioteca Basic X11 la cual ocupa alrededor de 200 Kb. (modo dinámico). Es un seudocompilador porque el archivo resultante no contiene el código máquina ejecutable directamente por el procesador sino una forma comprimida del lenguaje BASIC y su intérprete. Este seudocompilador está escrito en BASIC-X11. El lenguaje de este BASIC es muy rico, estructurado y con variables de tipo (entero, flotante, texto, vector, booleano). Contiene instrucciones para acceder a memoria como la C malloc() o para multiplicar matrices aplicable a vectores.

Las instrucciones gráficas son las mismas que en Atari ST GFA BASIC pero tienen ahora un resultado similar bajo X. La instrucción MENU crea un menú en una ventana gráfica, por ejemplo. El intérprete se ve enriquecido con documentación y muchos ejemplos, además en algunos puntos. El lado oscuro es que la adaptación aún no está terminada y los errores pueden aparecer incluso con los ejemplos que se adjuntan. Consideremos que este intérprete se comporta mayormente como una versión beta para ser utilizada por la categoría de usuarios para los que el lenguaje BASIC está orientado.

```
$ 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: la casa del intérprete

Sitio web: <http://www-cip.physik.uni-bonn.de/~hoffmann/X11-Basic/>

HBasic

Aquí tenemos un BASIC que da buena impresión a primera vista, bien por sus características o la calidad de la documentación que ocupa 7.7 MB. La instalación requiere QT-3.* disponible desde la página de [Trolltech](#) si no dispone de una distribución Linux reciente. Es un entorno completo de desarrollo que provee todas las características con las que sueña todo programador (a pesar de su "joven": versión 0.8):

- Interfaz de desarrollo integrada, creador de GUIs y editor de propiedades.

- Carga y uso de paquetes predefinidos para incluir en formularios y objetos grÁficos en los programas.
- Editor de cÁdigo fuente con texto destacado, autocompletado y administraci3n de mÁdulos.
- Incluye un int3rprete para evitar compilar durante las pruebas.
- El compilador estÁ integrado y crea ejecutables reales.
- Un compilador de cÁdigo .NET que puede correr en un entorno .NET.
- Depurador: provee puntos de ruptura, un visor del contenido de las variables (bien durante la ejecuci3n o cuando se mueve el cursor sobre el nombre de la variable en el editor).
- Posibilidad de crear objetos C++ para extender los programas HBasic o la interfaz de desarrollo en tiempo de ejecuci3n.
- Lenguaje orientado a objetos para las clases definidas en el cÁdigo fuente HBasic o para los componentes cargados como paquetes.
- Herramientas para la administraci3n de las bases de datos integradas permitiendo acceder bien a la informaci3n desde el IDE (Entorno de Desarrollo) o desde el programa.
- Creaci3n y uso de estancias de clases Qt.
- Soporte .NET: permite hacer uso de una amplia gama de bibliotecas .NET para acceder a sus mÁtodos, propiedades o campos.
- Edici3n de programas C# y compilado desde el IDE HBasic.
- Hojas de cÁlculo y grÁficos (todaví en versi3n alpha).

Los autores de HBasic te avisan diciendo: *"La versi3n presente de HBasic no es suficientemente estable para ser usada por desarrolladores en BASIC. Todaví-a tendrí;n que esperar a la salida de la primera versi3n estable 1.0"*.

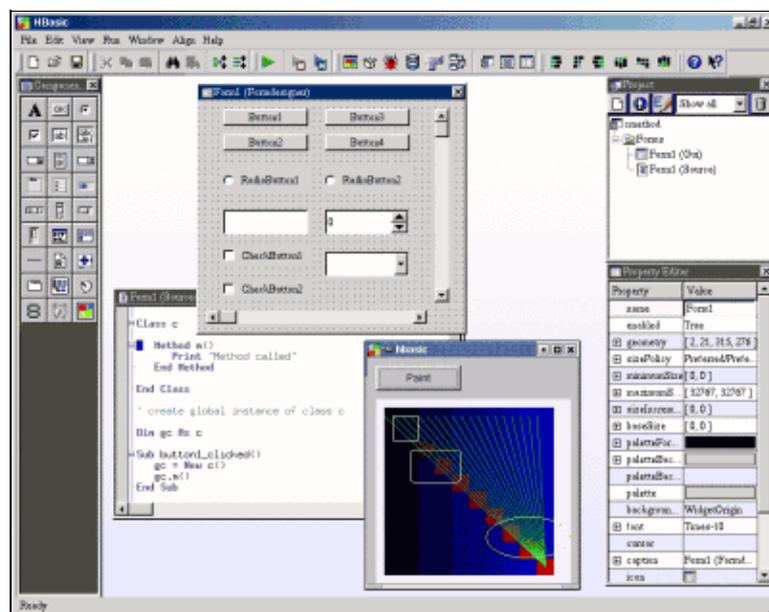


Imagen 5:Hbasic

GNOME Basic

GNOME Basic es un proyecto con la intenci3n de ser 100% compatible con Visual BASIC, VBA, VBscript, y otros lenguajes asociados. Al principio estaba pensado para hacer las macros VBA ejecutables desde Gnumeric, pero debido a cambios inesperados sÁlo ejecuta unos pocos formularios, comprueba casi todo el cÁdigo VB y corre ASPs sueltas. Con una mejor integraci3n en Bonobo, se puede esperar un entorno

completamente compatible con Microsoft.

Por ahora, Gnome Basic es un intento de proporcionar capacidades de compatibilidad VB para el proyecto GNOME, especialmente para aplicaciones de productividad (VBA).

El proyecto está todavía en estado pre-alpha y debe ser reservado para los desarrolladores del proyecto GNOME.

Página web: <http://www.gnome.org/gb/>

KBasic

KBasic es otro intento de desarrollar un BASIC compatible con Visual Basic. Los desarrolladores esperan sacar la primera versión estable 1.0 para el verano del 2003. Por ahora, sólo hay una versión inestable reservada al desarrollo. KBasic podrá usar el entorno Kdevelop.

Así es como la versión descargable trabaja por el momento:

```
$ 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
```

Como puedes ver, KBasic funciona pero está lejos de ser operativo, los mensajes de depuración del intérprete y el código ensamblador son inevitables por ahora.

Lugar: <http://www.kbasic.org>

Conclusión

Este análisis de los intérpretes BASIC y compiladores nos da una idea. Primero que todo, sorprendentemente, BASIC siendo tan criticado está todavía vivo, incluso en sistemas Unix donde su uso se mantiene bastante confidencial por los competidores (Perl, Python, Tcl/Tk, PHP....) que también corren de forma interpretada. La actividad que este lenguaje genera en los desarrolladores de software libre es significativa. El número de proyectos es bastante largo y todos ellos son bastante activos. La demanda es probablemente alta y esto puede contribuir a ayudar el desarrollo de software libre, Unix libres atrayendo a programadores BASIC.

En sistemas Windows, VisualBasic se utiliza ampliamente debido a su integración en el SO de Microsoft, las ERPs y las *suites* para productividad. Sin embargo, debemos destacar que muchos de los BASIC que hemos comprobado funcionan en ambas plataformas y algunas veces en algunas máquinas.

Lo siguiente es, que BASIC tiene todavía algunas desventajas. La falta de estándar condujo a muchas versiones, la mayoría incompatibles entre sí. El deseo de algunos nuevos proyectos como GNOME Basic o KBasic para reunir el estándar de facto podría ser una buena idea si VB fuera libre, lo que obviamente no es así. El monopolio podría ser más conveniente en este caso...

Para el desarrollador BASIC, hay una opción real y unas cuantas herramientas bastante cercanas. Como un IDE, HBasic es probablemente el más prometedor. Mientras esperamos por una versión estable, XBasic es el más conseguido. Sin embargo, SmallBasic, YaBasic permiten a los que se acaban de mover a Unix redactar scripts o CGI's sin tener que elegir entre las numerosas opciones disponibles por este sistema. GLBCC también lo permite pero tiene la ventaja de crear programas compilados con el gran GCC. Debido a la falta de un IDE y de un depurador fácil de utilizar, será difícil desarrollar y mantener grandes programas con esta herramienta. Por último, wx-Basic, el cual no es parte de ninguna categoría pero tiene algunas ventajas, como la habilidad para crear interfaces gráficos.

Otro buen punto sobre estos BASICs es la fácil migración de programas de un SO a otro sin tener que recompilar. Como algo habitual, la mayoría están disponibles para plataformas Win32 y Unix.

Los desarrolladores BASIC están entonces ante una dura tarea: Elegir la herramienta correcta para su objetivo.

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>

en --> es: David Losada. <losadaz(en)euskalnet.net>