BASICODE 4 (Basicode.NET)
Paul F. Johnson
September 2008, update Oct 2008

Project aims & History

1. History.

Basicode is an old format which allowed software to be broadcast on MW radio and via an interpreter on a micro, read in and through the use of platform specific routines allow to run on the target machine (for example, a Basicode broadcast could run on machines as diverse as an IBM PC to a TI99/4A to an Acorn Atom to a ZX Spectrum).

Development by NOS (The Netherlands radio service) stopped after version 3/3C though version 4 was mooted, nothing came of it.

Three versions of Basicode were created, each successively having greater improvements than the last with version 3 including sound and (limited) graphics capabilities.

Broadcasts of Basicode programs ceased around 1992.

Reference : http://en.wikipedia.org/wiki/BASICODE

2. Project aims

To create new Basicode applications using the .NET framework and allow the import of old applications.

This will allow the use of old programs to run on modern architectures and platforms (using Mono for Mac OS and Linux) and breathe life back into the language.

Project license.

GPLv2

Source code

Released via the yahoo group or sourceforge

Programming Language

Either C# or VB.NET – in all likelihood, C#

How it works

Using Basicode3/3C as a basis for the language and extensions, a number of changes will be required for Basicode4 to operate in a windowing environment. I do not intend to make this list exhaustive, but it demonstrates the problems in keeping such a project portable between platforms.

For example, drawing tools have to be handled via gdiplus; there is no value or point in having the system use Tao or DirectX as this would not only cause a greater dependency chain but increase problems in maintaining the code.

Sound too would be an issue, though there are solutions (such as SoundPlayer [System.Media]) which could sort the problem (a generic PlaySound method would implement this with a pointer to the sound file). System beeps can be catered for directly.

Menus are not supported at this point.

New commands

Command	Parameters	Function and notes
		Windowing
WindowOpen	(size_x, size_y, title)	Creates a window of size x, y with a title bar name. If no title is given, it will be given a title bar containing the size of the window.
WindowClose	(window title)	Closes the window with a given title.
Message	(flags, title, text)	Creates an alert box with title and text. Flag — Meaning 0 — OK button, normal message 1 — OK button, alert 2 — OK and Cancel buttons, choice 3 — OK, Cancel, Retry All flags are OR'd
WindowColour	(colour)	Sets the window colour
TextColour	(colour)	Sets the text colour
		Audio
PlaySound	(filename)	Plays a sound asynchronously
PlayBeep		Makes a beep noise
		Graphics
SetLine	(width, colour)	Sets the line widths and colour
SetFill	(colour)	Sets the fill colour
DrawLine	(pos_x, pos_y, len, colour)	Draws a line of length len. If no colour is specified or set by SetLine, it will be black.
DrawCircle	(pos_x, pos_y, radius, line colour, fill colour)	Draws a circle of with a given radius. If no colour is specified or set by SetLine, it will be black. If no fill colour is specified, clear is used.
DrawBox	(pos_x, pos_y, length, width, line colour, fill colour)	Draws a box for a given length and width. If no colour is specified or set by SetLine, it will be black. If no fill colour is specified, clear is used.
DrawArc	(pos_x, pos_y, length, radius, line colour)	Draws an arc of a given length and radius. If no colour is specified or set by SetLine, it will be black

DrawTriangle	(pos_x, pos_y, len_1, len_2, angle, line colour, fill colour)	Draws a triangle. Given two lines and an angle, the third line and angles can be calculated. If no colour is specified or set by SetLine, it will be black. If no fill colour is specified, clear is used.
Font	(font name)	Changes the font to a new font
Menu	(name1,...,name6)	Creates a menu bar with upto 6 items
MenuItem	(parent, name)	Creates a menu item associated with the parent
		Filer
FileSave	(filename)	Opens a save file dialog box if filename is not given. If it is given, saves to the default directory.
FileLoad	(filename)	Opens a load file dialog box if filename is not given. If it is given, saves to the default directory.
SetFormat	(format)	Sets the BASICODE format to save out (1, 2, 3/3C or 4. Default = 4)
SaveData	(filename)	Saves data without opening a file dialog box
LoadData	(filename)	Loads data without opening a file dialog box
		Widgets
InputText	(pos_x, pos_y)	Creates an input box at x, y
Text	(pos_x, pos_y, text)	Creates text at x, y
NumberInput	(pos_x, pos_y, lower_lim, upper_lim)	Creates a number input box. If no limits are supplied, a default of 0 to 255 will be taken.
Button	(pos_x, pos_y, text)	Creates a button at x, y with some text in it. The button will automatically be sized for the text.
		Programming
foreach	(type as t)	Same as foreach in any other language
else		Adds else condition to if/then
|		Bitwise OR
&		Bitwise AND
stringcompare	(string1, string2)	Compares 2 strings. If they're the same, returns false.
stringcontains	(string, test)	Tests if a string contains a character. Returns first position.
length	(string)	Returns length of a string
quit		Quits the current program
DefProc	name(params)	Creates a new procedure (replaces gosub / goto)
EndProc		Ends the procedure and returns to the main process line
Proc	name(params)	Calls the procedure
start		Equivalent of the old GO TO 20 routine
string		Creates a string variable (as opposed to a character array)
local	name	Creates a variable local to a procedure. Outside of the procedure, it is unavailable.
global	name	Creates a global variable.
allocate	size	Allocates a chunk of memory
deallocate	name	Clears the chunk of memory
LET{F/S/I}		Creates a variable of type Float/String/Int

Removed commands

go sub, go to

Other programming notes

go to and go sub have been removed in favour of DefProc/Proc/EndProc. This may cause problems with some older programs, but should make life easier for new programs. Gosub and go to will be replaced on loading old version code and new procedures created. They will be named newproc_xxx (where x = “A” to “Z”, you are free to rename them). Once converted, they will not run under old Basicode systems.

start – this replaces GO TO 20 and must always be called.

The code is case insensitive and all BASICODE 3/3C commands, logical tests and maths functions are maintained.

New function names

These replace the old GO SUB routines. These procedures are in addition to the new ones listed above

GO SUB	PROC	Notes
100	WindowOpen	
110	SetCursorPos	Sets the cursor position for a given x and y. If nothing is passed in then 0,0 is assumed.
120	GetCursorPos	Returns an array containing the x and y position
150	Text	Displays text at x,y
200	ReadChar	Returns the character pressed
210	WaitChar	Waits for any key to be pressed
220	ReadText	Reads text from an input box at a given position. If no position is given, the whole string (or NULL) is returned
250	PlayBeep	
260	RandomNum	Lower and upper limits can be passed in. If they are not, the default of between 0 and 1 is assumed.
270		Deprecated. No longer required
280		Deprecated. No longer required
300	NumToString	Assumes the number is an int
310	FloatToString	Converts float to string
330	ToUpper	
350	PrintLine	This may or may not work!
360	PrintCR	
400	PlaySound	
500		Replaced by FileSave/FileOpen/SetFormat
550	ReadFileString	FileOpen MUST be run before Read/Write/CloseFile
560	WriteFileString	
580	CloseFile	
600		Replaced by graphics procedures. There is no text or
620		graphics modes any more as it's a window system.
630		
650		
950	quit	

These are still open for change if users require them to be.

Variables

Traditionally, the following were not available to Basicode

- all variables starting with the letter "O"
- AS, AT, DI, DI$, DS, DS$, EI, EI$, EL, ER, FN, GO, GR, IF, LN, SQ, SQ$, ST, TI, TI$, TO
- PI

with

- A, CN, CT, FR, HG, HO, IN, IN$, NF, NF$, RV, SD, SP, SR, SR$, SV, VE, VG

available to the basicoder.

I'm intending to free most of these up with the exception of

IF, LN, TO, PI, IN, AT and IN$

Variables do not need to be initialised, but it's a good idea to. Pointers and References are not currently supported.

Variable types

Currently only string and integer variables are supported (traditionally, a string would be denoted by a $ after a variable name). These will now need to be altered as a new float type is being introduced.

To declare a variable

Again, the old style was to use LET <variable> = <value>. This was fine for integer and string variables, but with a third type a new format is required. For this LETF has been introduced (there are also LETS and LETI, but these can be omitted if you prefer just to use LET)

Allocating a memory block

This isn't so much of a new command, more a modifier.

Use:

LETI M = allocate(3000) and deallocate(M).

It can be considered to be like an char array, but that once it's no longer in use, can be deleted. The savedata/loaddata commands should be used to load information into the memory block.

The applications

Currently, the plan is not for an IDE, just a compiler which outputs intermediate code runnable by the CLR; create and edit your files using your favourite text editor! There will be a graphical front end to the command line programs.

The plan is to create 2 programs

basicode4compile – the command line compiler
basicode4convert – converts basicode (1 – 3/3C) to basicode4 format

The compiler includes a parser which will check the code to ensure that everything has been written correctly and that you're not doing anything dumb. It will not check file paths or anything like that, but will fill in the blanks if a blank exists (for example, not defining a colour on a line etc).

The converter will convert all old versions of the code to the new (removes the go to and go subs with procedure names and saves them back out).

Filename conventions

Standard files can be loaded in for converting using either .bas or .txt as the extension. .bas and .txt cannot be fed into the compiler. Once a file has been converted, it will be saved as .bc4 – this can be fed directly into the compiler (note : if you have created a basicode4 program directly, save as .bc4 and run it through the compiler).

From what I can see, there wasn't any way of determining the version of a Basicode program directly, therefore files need to be named like this for the parser to correctly alter routines (there were differences between Basicode 2 and 3/3C)

.bas / .txt = Basicode 1
.bc2 = Basicode 2
.bc3 = Basicode 3/3C

Don't rename them and there is no guarantee of the code compiling.

Compiler errors

Code compiles or doesn't. If it doesn't, the compiler will throw back the errors as a text stream to your standard output.

Runtime errors

(to be inserted)

Copyright attributions

Initial creation : March 2008
This version : October 2008