KBasic WORLD'S MOST ADVANCED OPEN SOURCE BASIC

KBasic is a powerful programming language for Linux, Mac OS X and Windows, which is simply intuitive and easy to learn. It is a new programming language, a further BASIC dialect and is *related to* VB*.NET, Visual Basic 6 and Java.* It combines the best features of VB.NET, Visual Basic 6 and Java and comes with built-in backward support for VB.NET, VB6 and QBasic as it is *100% Syntax compatible to VB6 and QBasic. Additionally, it comes with support for* VB*.NET syntax, functions and similar objects and classes.*

But this is **NOT** a VB.NET or VB6 clone! It is a full featured object-orientated language, which supports the best modern programming techniches known with well designed objects, events and plenty of documentation.

Table of Contents

More exactly said KBasic is an object-oriented and event-controlled programming language, developed by KBasic Software (www.kbasic.com) and the open source community. KBasic is a multiplatform programming language, so almost the entire API is the same on all platforms and window systems. Thanks to Qt on which KBasic is based.

KBasic is an open source, easy-usable, object-oriented, interpreted, stable, platform-independent, fast and modern programming language.

KBasic uses Qt as its toolkit to provide cross-platform abilities. Qt is the best C++ cross-platform toolkit available and KBasic is the easiest way to get cross-platform development without the needs to learn C++ as it combines the expressive power of C++ with the familiarity and ease of use of VB6. The Qt API and tools are consistent across all supported platforms, enabling platform independent application development and deployment. Windows, Linux and MacOS X are supported platforms. Read more at Trolltech’s website http://www.trolltech.com/.

Together with KBasic you are able to create modern object oriented applications, e. g. for tasks such as GUI, SQL and more. **KBasic is available in English and German. Other languages such as French, Chinese and Spanish are in preparation.**

Since its commercial introduction in early 1996, Qt has formed the basis of many thousands of successful applications worldwide. Qt is also the basis of the popular KDE Linux desktop environment, a standard component of all major Linux distributions. Read more at KDE’s website http://www.kde.org/. That means that the internal functions of KBasic use the high-quality Qt library functions, which are the top of the art: fast, stable, bug-free and tested and used worldwide!

What does the name KBasic mean? KBasic means QtBASIC, but QBasic is used as most people know and QtBasic isn’t allowed at all, so KBasic is the best choice. Other possibilities could be KoolBasic or even KDEBasic. It’s your choice!

PLATFORMS

KBasic is supported on the following platforms:

- **Windows Vista, 2000 and XP** (created KBasic EXE programs run on older Windows versions: Win95, Win98, WinMe, WinNT)
- **Linux (KDE or GNOME)**, UNIX based like FreeBSD are comming soon
- **Mac OS X** (>= 10.2)

COMMUNITY

People around the world join KBasic - inspired by the idea to make software available for everybody: a programming language that is easy to use, and a development platform that is stable, reliable and available at a low price.

We communicate by different means, most of them on the Internet. The following selection enables you to stay up-to-date, extend and explore your KBasic experience, find new friends, and have fun in general.

The KBasic community rests on dedicated volunteers to further improve our programming language and development platform in a number of different ways. Whatever your skills, there are lots of places to start contributing.

Forum: http://www.kbasic.com/forum/index.php

DOWNLOAD

Full Version Professional Edition (English and German are supported languages)

- **Windows Vista/XP/2000**
  http://www.kbasic.com/installer_kbasic_professional_windows.exe
- **Mac OS X**
  http://www.kbasic.com/installer_kbasic_professional_mac.dmg
- **Linux (KDE/GNOME)**
  http://www.kbasic.com/installer_kbasic_professional_linux.bin

INSTALLATION

Easy and familiar installation: Windows Installer + Mac Installer + Linux Installer

These downloadable files are installer files, which must be executed by you. For Linux: make the installer file executable (right click –> properties).

Installation on Windows/Mac

succesfully tested on

- Windows Vista, Windows XP, Windows 2000: P2400/512MB
- Mac OS X Tiger 10.4
- Windows 95/98/Me/NT is not supported

Installation on Linux

succesfully tested on

- OpenSuse 10.2 (>=Qt 4.2.0) : P2400/512MB
- Fedora, Gentoo, KNOPPIX, (K)Ubuntu, Slackware, Xandros, Linspire are expected to work as well

If your PC does not show anything after clicking on the installer file, your system does not match the requirements or has not installed all needed software in the right location as Linux Standard Base (LSB) defines it!

Dependencies are (shared libraries, which you must have installed on your system):

- libqt-mt.so.3 ⇒ /usr/lib/libqt-mt.so.3
- libXft.so.2 ⇒ /usr/lib/libXft.so.2
- linux-gate.so.1 ⇒ ?

- libQtSql.so.4 ⇒ /usr/lib/libQtSql.so.4
- libQtGui.so.4 ⇒ /usr/lib/libQtGui.so.4
- libpng12.so.0 ⇒ /usr/lib/libpng12.so.0
- libSM.so.6 ⇒ /usr/lib/libSM.so.6
- libICE.so.6 ⇒ /usr/lib/libICE.so.6
- libXi.so.6 ⇒ /usr/lib/libXi.so.6
- libXrender.so.1 ⇒ /usr/lib/libXrender.so.1
- libXrandr.so.2 ⇒ /usr/lib/libXrandr.so.2
- libXfixes.so.3 ⇒ /usr/lib/libXfixes.so.3
- libXcursor.so.1 ⇒ /usr/lib/libXcursor.so.1
- libXinerama.so.1 ⇒ /usr/lib/libXinerama.so.1
- libfreetype.so.6 ⇒ /usr/lib/libfreetype.so.6
- libfontconfig.so.1 ⇒ /usr/lib/libfontconfig.so.1
- libXext.so.6 ⇒ /usr/lib/libXext.so.6
- libX11.so.6 ⇒ /usr/lib/libX11.so.6
- libQtCore.so.4 ⇒ /usr/lib/libQtCore.so.4
- libz.so.1 ⇒ /lib/libz.so.1
- libpthread.so.0 ⇒ /lib/libpthread.so.0
- libdl.so.2 ⇒ /lib/libdl.so.2
- libstdc++.so.6 ⇒ /usr/lib/libstdc++.so.6
- libm.so.6 ⇒ /lib/libm.so.6
- libgcc_s.so.1 ⇒ /lib/libgcc_s.so.1
- libc.so.6 ⇒ /lib/libc.so.6
- libXdmcp.so.6 ⇒ /usr/lib/libXdmcp.so.6
- libXau.so.6 ⇒ /usr/lib/libXau.so.6
- libexpat.so.1 ⇒ /usr/lib/libexpat.so.1
- /lib/ld-linux.so.2

DESCRIPTION

It comes with Java-like object orientation and backward support for VB6 and QBasic, as it is 100% syntax compatible, but it is not a VB6 or VB.NET clone! Though it comes with support for VB.NET syntax, functions and similar objects and classes as well. KBasic combines the expressive power of Object-Oriented languages like C++ with the familiarity and ease of use of VB6. It allows developers with an installed base of VB6 applications to start developing for a mixed Windows, Mac OS X and Linux environment without having to face a steep learning curve: KBasic uses the familiar visual design paradigm and has a full implementation of the BASIC language.

KBasic is made up of the following programs:

a development environment with visual form designer (IDE)

- a compiler (KBC)
- an interpreter (VM)
- a graphical user interface component. (VM)

It’s about 5 MB source codes (C++)

130,000 program lines 100 classes 2,500 methods in 280 files

It is really easy to develop multi-platform GUI applications with well known BASIC syntax in a modern fashion, because KBasic uses the familiar visual design paradigm and has a full implementation of the BASIC language. KBasic uses Qt to provide cross-platform functionality (Qt is the leading cross-platform technology available worldwide).

- Write once and deploy native applications for Windows, Mac OS X and Linux
- OOP RAD features deliver high productivity
- Much cheaper than other BASIC’s
- “backward” support for VB6 and “forward” support for inheritance and other OOP features
- do it on multiple platforms
- Porting existing VB6 projects is easy, because KBasic is 100% syntax compatible
- Familiar development process, environment
- Same syntax as VB6
- Easy to learn: built-in Tips and language reference
- Be more productive with OOP RAD features
- Built-in memory management via reference counting
- True cross-platform deployment
- Familiar language features: OOP, single inheritance, exceptions, etc.
- Drag & drop GUI development
- Rich UI widgets set
- Familiar editing features: easy and fast browsing of your source code
- Familiar editing features II: Auto-Completion of builtin-functions and datatypes, even user defined functions and types
- Auto-Completion (Professional version only)
- Familiar debugging features: single step, showing variables’ values, local and global scope
- Familiar Install/Uninstall features: for Linux (KDE), Mac OS X and Windows
- **Lots of Documentation!**

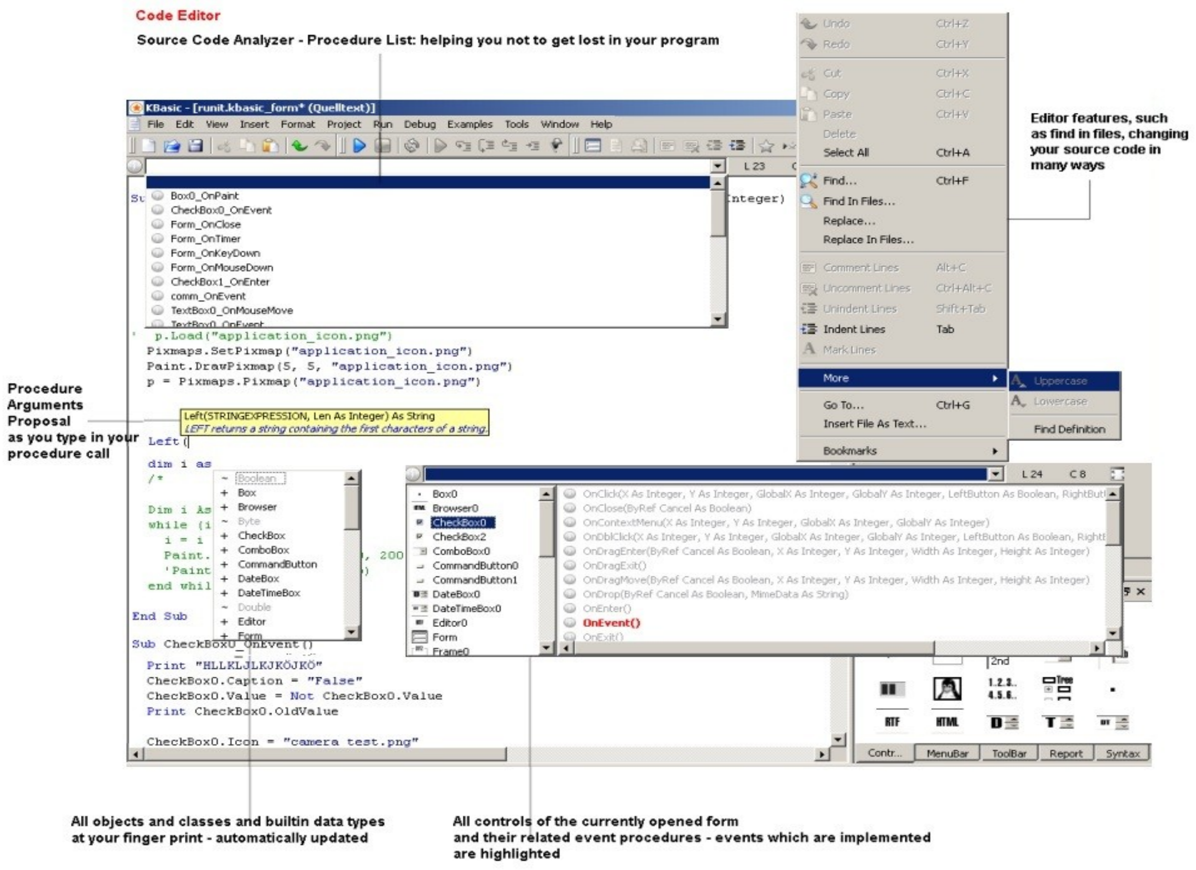
  KBasic comes with extensive documentation, with hypertext cross-references throughout, so you can easily click your way to whatever interests you. The part of the documentation that you will probably use the most is the KBasic Language Reference. Each link provides a different way of navigating the KBasic Language Reference; try them all to see which work best for you. You might also like to try The KBasic Book: this book contains detailed information about KBasic, and it provides a full text search facility. There are also a growing number of KBasic books.

- **Thousands of examples!**

  KBasic ships with lots of small and some medium-sized example programs that teach you how to implement various tasks with KBasic. Most of them will show how to use a certain class or module, others aim at programming techniques and KBasic basics, and some of them simply want to show you what is possible.

  Note that most of the examples assume that you have some experience with KBasic and Object-Oriented programming and therefore are not commented extensively. If you are interested in a line-by-line coverage please refer to the “Learning Coding for Beginners” and

“The KBasic Book”.



```
Su                                                                      nteger)
'  p.Load("application_icon.png")
  Pixmaps.SetPixmap("application_icon.png")
  Paint.DrawPixmap(5, 5, "application_icon.png")
  p = Pixmaps.Pixmap("application_icon.png")

  Left(
  dim i as
  /*

  Dim i As
  while (i
    i = i
    Paint.                      , 200
    'Paint                       )
  end whil

End Sub

Sub CheckBox0_OnEvent()

  Print "HLLKLJLKJKÖJKÖ"
  CheckBox0.Caption = "False"
  CheckBox0.Value = Not CheckBox0.Value
  Print CheckBox0.OldValue

  CheckBox0.Icon = "camera test.png"
```

Left(STRINGEXPRESSION, Len As Integer) As String
LEFT returns a string containing the first characters of a string.

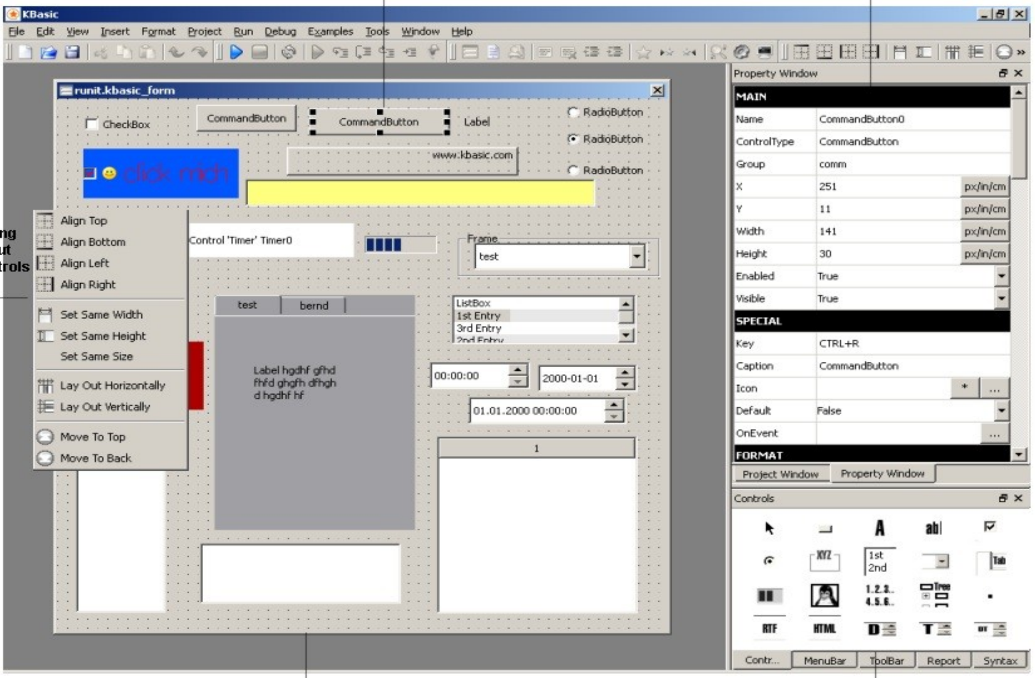
```
OnClick(X As Integer, Y As Integer, GlobalX As Integer, GlobalY As Integer, LeftButton As Boolean, RightBut
OnClose(ByRef Cancel As Boolean)
OnContextMenu(X As Integer, Y As Integer, GlobalX As Integer, GlobalY As Integer)
OnDblClick(X As Integer, Y As Integer, GlobalX As Integer, GlobalY As Integer, LeftButton As Boolean, RightB
OnDragEnter(ByRef Cancel As Boolean, X As Integer, Y As Integer, Width As Integer, Height As Integer)
OnDragExit()
OnDragMove(ByRef Cancel As Boolean, X As Integer, Y As Integer, Width As Integer, Height As Integer)
OnDrop(ByRef Cancel As Boolean, MimeData As String)
OnEnter()
OnEvent()
OnExit()
```

Form Designer

Selected Control - easy change size and position with the mouse or keyboard

Properties of the selected control(s)

Use fast changing of size and layout of selected controls



MAIN	
Name	CommandButton0
ControlType	CommandButton
Group	comm
X	251
Y	11
Width	141
Height	30
Enabled	True
Visible	True
SPECIAL	
Key	CTRL+R
Caption	CommandButton
Icon	
Default	False
OnEvent	
FORMAT	

WYSISWG - all property changes are instantly shown in the design view as well

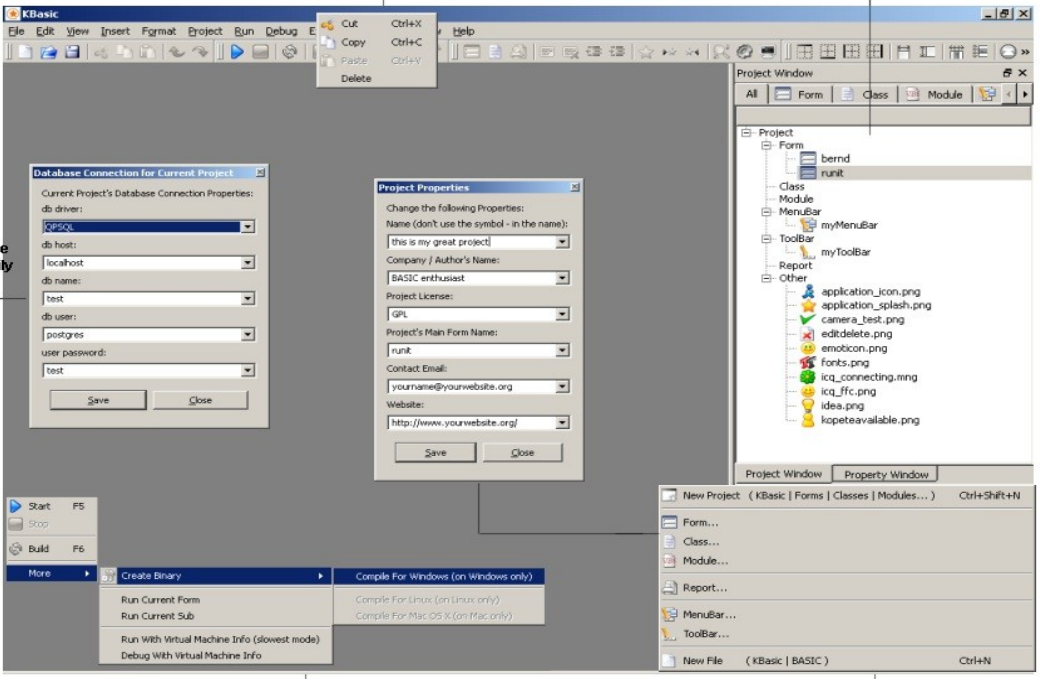
Controls to insert as you wish

Project Window

Manage your project files copy or delete them by one mouse click

File list organized in categories such as forms classes and modules

Set the datebase connection easily



Database Connection for Current Project

Current Project's Database Connection Properties:

db driver: QPSQL

db host: localhost

db name: test

db user: postgres

user password: test

Save | Close

Project Properties

Change the following Properties:

Name (don't use the symbol - in the name): this is my great project

Company / Author's Name: BASIC enthusiast

Project License: GPL

Project's Main Form Name: runit

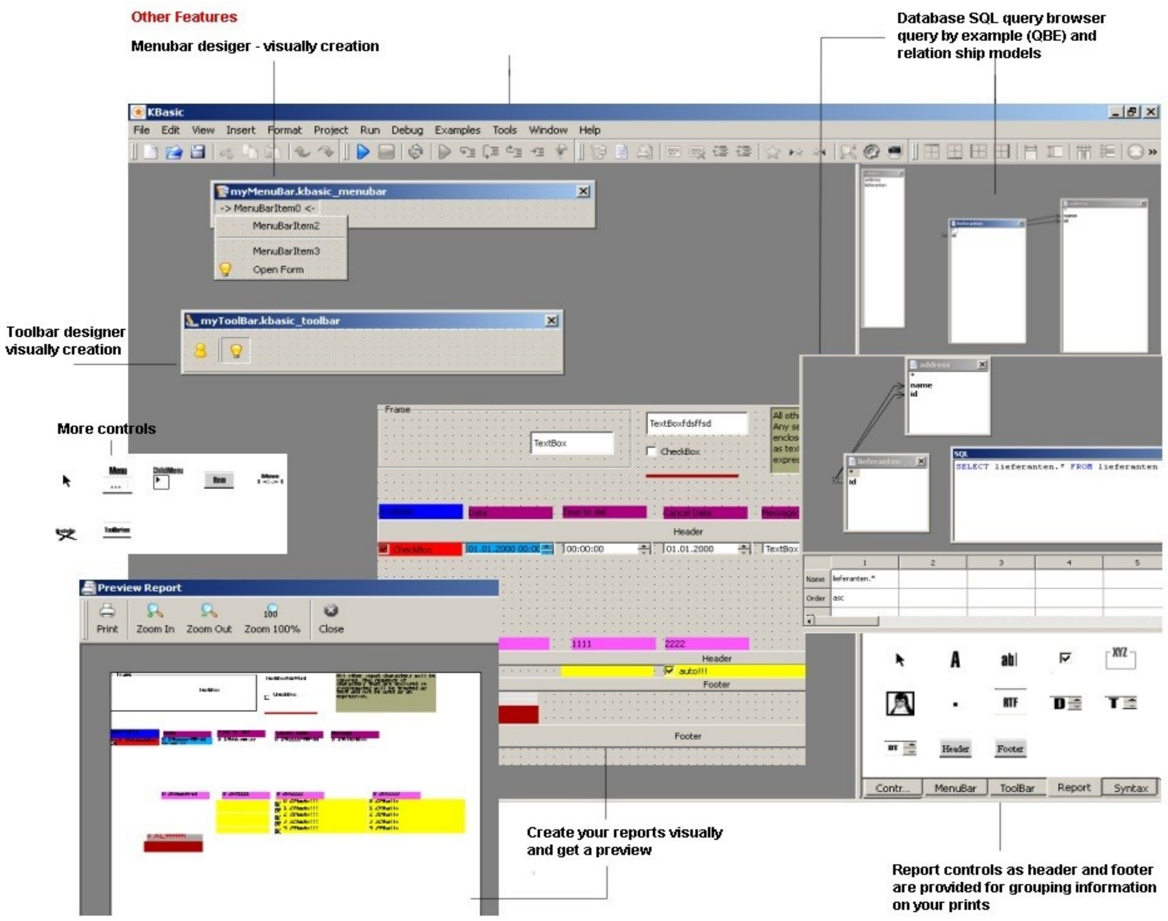
Contact Email: yourname@yourwebsite.org

Website: http://www.yourwebsite.org/

Save | Close

Run your code, create freely runable exe files, watch your code running in the virtual machine information window

Create new objects as you need them visually


Other Features
Menubar desiger - visually creation
Database SQL query browser query by example (QBE) and relation ship models
Toolbar designer visually creation
More controls
Create your reports visually and get a preview
Report controls as header and footer are provided for grouping information on your prints
KBasic
File Edit View Insert Format Project Run Debug Examples Tools Window Help
myMenuBar.kbasic_menubar
MenuBarItem2
MenuBarItem3
Open Form
myToolBar.kbasic_toolbar
Preview Report
Print Zoom In Zoom Out Zoom 100% Close
Header
Footer
Contr... MenuBar ToolBar Report Syntax

SELECT lieferanten.* FROM lieferanten

Feature Detail

KBasic

- **is object oriented with objects and classes, single inheritance and polymorphism and private, public, protected scope of objects' elements**
- **is syntax compatible to old BASICs like QBasic or VB6**
  - optional parameter
  - paramarray
  - named arguments [mySub(param1 := 23, param2 := 100)]
  - on error goto
  - label and goto
  - variable naming (with shortcuts like name$ or n%)
  - property handling
  - primitive variables also arrays and user defined types can be passed to functions by reference
- **contains a virtual machine**
  - automatic garbage collection
  - protected data and arrays
  - modern error management through exception handling

KBasic is not only one programming language but is also three languages.

Through one of the following command you can switch KBasic into one mode.

If you want to use KBasic newest features (Default) use

```
OPTION KBASIC
```

If you want to use old VB6 code use

```
OPTION OLDBASIC
```

For very old BASIC code like QBasic you should use:

```
OPTION VERYOLDBASIC
```

It is possible to use all three modes in one of your program, e.g. one module uses one mode and the other module of your program use another mode. Just place one of these lines in top of your module. Default is OPTION KBASIC.

Of course like other programming languages, KBasic comes with commands for control flow, conversion/casts, error handling, events and library functions like for gui, input or output, maths and so on. See the language reference for more. But besides this you should be familiar with the following main parts.

- **Class / Module**
- **Sub / Function / Method**
- **Variable / Constant / Property**
- **Array**
- **Type**
- **Enum**

If you start working on your first KBasic program keep in mind that is very similar to VB6. You have modules or classes and forms, which work together. Events in your forms are triggered by the user and you can react to them within your program inside its subs. That's it.

Annotations

- **Type**
- **Variable / Constant / Property**
- **Variable Scope**
    - global
    - module
    - class
    - local
        - module function or module sub
        - class static method
        - class instance method
- **Array**

    - static
    - dynamic
- **Sub / Method**
    - default arguments
    - arguments by reference or by value depending on primitive type are possible
    - function overloading possible
    - recursive calls are possible
    - non-primitive local variables (objects) will not be automatically destroyed if there is a reference to it
    - exception handling
- **Function / Method**
    - default arguments
    - arguments by reference or by value depending on primitive type are possible
    - function overloading possible
    - recursive calls are possible
    - (user defined types and arrays can be returned as well (in the future))
    - non-primitive local variables (objects) will not be automatically destroyed if there is a reference to it
    - exception handling
- **Scope**
    - global
    - module
    - class static
    - class instance
    - local
        - module
        - class static
        - class instance
- **Class**
    - Constructor (method overloading possible)
    - Default Constructor (automatically called if there is no constructor defined)
    - Destructor (automatically called by garbage collector)
    - instance methods (works on variables of an object of a class)
    - static methods (works without any relation to an object)
    - instance variables (private, public, protected), variables of an object of a class
    - static variables (works without any relation to an object)
    - static source code part of class (will be executed on start up of your program)
    - static constants
    - property support
    - variables (private, protected, public)
    - constants (private, protected, public)
    - types / enum can be private or public
    - all methods are 'virtual' (speaken in C++ terminology), which means that the child method is called by parent class instead the original parent method which was overwritten by the child class
- **Module**
    - subs or functions
    - variables (private, public)
    - constants (private, public)

    - types / enum can be private or public
- **Global**
    - subs or functions
    - variables
    - constants
    - types / enum

USEFUL FOR LINUX AND MAC!

WHY IS KBASIC USEFUL FOR THE LINUX, WINDOWS OR MAC OS X COMMUNITY?

The 21st century should welcome the concept that everybody is able to program an interface for their computer. Programming does not need to be reserved for those few who know how. Programming should be as easy as surfing the web:

- Point and click design
- Point and click to compile and install
- Click the icon to run

The REAL alternative

The modern computer is still inefficient as a labor saving device because it requires programming. Now, if more people could program everything that they need quickly and easily - that means not having to learn syntax, not having to learn keywords - maybe everyone will know how to use a computer in their own unique way.

You have probably heard about 'dirty inefficient code' and 'bad habits' that VB programmers gain. Those were the words of the 1960's, 70's and 80's when you were forced to optimize for minimal RAM and CPU usage. Nowadays, with nearly everyone running at least a Pentium-class processor with at least 32MB RAM, you need not worry about that.

Back to Basic!

Lose the discpline, embrace the usability and simplicity!

KBasic brings us closer to this reality.

Read more

What is it?

BASIC = Beginners All-purpose Symbolic Instruction Code

Standard BASIC

- BASIC is the most English-like programming language on Earth
- Familiar to native speakers of English
- No cryptic syntax - for new programmers, in particular
- Fewer Programming Rules
- some BASIC functions are very similar to SQL functions: LEFT, RIGHT and MID

KBasic is compatible with best known BASIC functionality:

Keywords: IF, THEN, ELSE, ELSEIF, ENDIF, FOR, EACH, NEXT, LOOP, DO, WHILE, GOTO, GOSUB, INTEGER, STRING, BOOLEAN, LONG, DIM, AS, SUB, FUNCTION, SELECT, CASE, END, TYPE, PRIVATE, PUBLIC, EXIT, OPTION, EXPLICIT, COMPARE, ON, ERROR, STOP, AND, OR, XOR, NOT …

File functions: ACCESS, APPEND, BINARY, INPUT, OUTPUT, RANDOM, CHDIR, MKDIR, RMDIR, FILES, CLOSE, EOF, FILEATTR, FREEFILE, KILL, LOC, LOF, PRINT, NAME, RESET, SEEK, SHELL, WITDH, WRITE …

Additional functions: BEEP, DATE, HEX, TIME …

Mathematical functions: ABS, ATN, COS, SIN, TAN, EXP, LOG, SGN …

Regular datatypes: INTEGER, STRING …

User definied datatypes: TYPE, ARRAY …

KBasic Extension

I want to allow the user to use modern programming techniques. To that end, I added some new features to the Standard BASIC language.

- Object-Oriented features like user defined classes and using the best ideas from other programming languages
- Database support

What can I do with it?

It is possible to do a lot with BASIC. Do you know what language most real scientists, not computer scientists, use to test their theories and build demos for colleagues? BASIC. It is easy to use and has enough graphics to prove the point with simple simulations. Not even VB, usually. Traditionally it has been QBasic for plain old DOS or a Macintosh variant of BASIC.

Who needs that?

- I think bringing BASIC to Linux would continue a trend in flexibility.
- For most Windows users, VB is the only language they know
- There are millions of VB developers out there in corporate jobs using it
- If 99% of the millions of programmers who switch to Linux to write KBasic programs are bad, we are still left with the 1% - how many thousand? - which are writing good programs which are useful.
- It is the easiest language for beginners, who do not want to get into low level programming
- If we want Linux to succeed in the mainstream, we need to provide the same functionality that exists in the mainstream.
- The system which has the most developers will win! Ignoring BASIC developers would leave most developers out of Linux.
- Professional programmers do not wish to learn a new language every 3-5 years. They do want their computer to do what they tell it to. “I do not care if it runs a little slower than C++, as long as it does the job I need done."
- Many people love BASIC

- Support broader clientele and languages for the masses
- All people who want to save money like schools, universities, developers with some spare time, and, of course, software vendors
- If you think a BASIC language would be useless send an e-mail to M$ and ask them why do they continue to develop VB.

Historical examples

“Where are the applications?"

Remember the situation with OS/2 (a bad example). Its measurements of “success” has languished for years.

There are hundreds of VB applications that are in use besides commerical VB applications. Business uses a lot of VB to get things done, because of its ease of use. Now, if they could port those apps to Linux …

Why BASIC?

- BASIC is the beginner’s programming language. And why not support Beginners?
- BASIC is not always the best, but it is certainly not the least.
- Not everyone has the time, patience and skill to learn C/C++ or Python.
- And I believe every programming language has its advantages.

Useful for Linux / Mac OS **X / other projects?**

Because the bigger the Linux/Mac user base becomes, the more willing hardware and software producers will be to port existing software, supply drivers, or write new software. They would see a possible profit instead of just a goodwill donation to the Linux/Mac community.

Everything that makes Linux/Mac what it is now and what it will become is a direct result of the people who work with it. And the more people that discover Linux/Mac and come on board, the bigger and better the possibilities become.

I think bringing BASIC to Linux/Mac would continue a trend in flexibility.

Why have I started a new BASIC project?

- Short answer: I have missed a good BASIC language/IDE in 2000.
- Long answer: I had searched the internet and found some BASIC languages for Linux. I looked at the source code, but no one could convince me.

MORE DOCUMENTATION

Learning Coding

First, you may be wondering what KBasic exactly is. Well, there is a simple answer - KBasic is a programming language based on well-known BASIC, which has been the accepted standard ever since. Why? Because of its ease of use, its English-like commands, and its power.

KBasic stands for Beginner's All-Purpose Symbolic Instruction Code. Several of its commands are pure English - PRINT, LET, and many others. It has a simple structure for its programs: its lines are numbered (10, 20, 30, etc.) and are executed in order.

But why should you use KBasic? What's in it for you? Many things are in it for you. When you learn KBasic, you also learn many of the fundamentals of other programming languages. You also can create programs easily. Once you get into it, you'll find that the fun in creating programs is worth coming back to.

Let me show you some theory about programming languages.

The CPU of your computer is only able to execute machine code, therefore a compiler is needed to translate from a high-level programming language (human friendly = KBasic) into machine code.

Programming languages (KBasic) give you the ability to create computer programs, meaning solving problems with algorithms. Through programming languages it is possible to run calculations of mathematical expressions (12 * 556 + 9), to give your computer different kind of commands ("Open file", "Print document") and to control your own computer program regarding the way it is executed, which command comes first and so on ("repeat this command 10times").

Other examples of programs are the programs of your local theater (defined order of the movies) or good old board games (game rules).

Important parts of the KBasic programming language.

Mainly there are three parts:

1. math. calculations, variables, expressions, math. functions n = 1 * 2 + 3 + 4 * 5 n = ABS(-99)

2. commands, open/save file, move the mouse, play movie saveFile(c:\bernd\thesenpapier.sdw) playMovie(open hearts.mpg")

3. program control, order and organisation

Some other rules

1. commands are executed from top to bottom

2. it is possible to define if a command is executed at all using commands like: IF THEN ELSE, SELECT CASE

3. if a command should be repeated and how often using commands like: DO WHILE, FOR NEXT

4. if several commands should be grouped to a new single command using commands like: FUNCTION, SUB, CLASS

List of programming languages

Only a small list, probably there exist more than 65 different programming languages

Machine code (1943)	"in the beginning there was machine code"	10001111 10101110 10101110 10101100

Assembler	More human-friendly way of machine code	MOV AX, 13 JMP 66 INC BX
BASIC, (1964) Visual Basic (1990) and KBasic (2005)	Most known programming language, because it is also integrated in most office programs like word processors	IF i = 0 THEN GOTO 999
C (1969) und C++ (1984)	Most important programming languages, most important programs are written using one of them (operating system, office, games)	If (i == 0){ void startGame(void *p){
Java (1996)	For Internet und Server applications	public class Abspann { setFont(font);
C# (2001)	Yet another kind of C, C++ and Java	namespace TestParser { System.Console.WriteLine
Script languages	e.g. PHP / Perl (Internet)	Echo "hello"

Glossary

Only a small list of the important notions

- algorithm, order of commands to solve a problem
- syntax, describes how to write commands
- compiler, computer program for translation of high-level to low-level = machine code, the translation result is a complete runnable computer program in machine code
- interpreter, computer program, which while executing a high-level program translates the next command"on the fly" into machine code
- expression, e.g. 1 + 23 * 5
- variable, place to store the result of math. Expressions for later processing

Now, let us do something real. Here comes our first KBasic program:

```
OPTION VERYOLDBASIC
CLS
PRINT "Hello!"
PRINT
PRINT "This is my first program!"
```

This is just a list of commands that the computer will interpret and execute. Go up to the "RUN" menu and click "START". The screen will clear, and look something like this, with a "Press any key to continue" on the bottom of the screen.

```
Hello!

This is my first program!
```

Notice it doesn't say CLS, or PRINT, or anything. To begin with the explanation, the command 'CLS' stands for Clear Screen, and PRINT is quite self-explanatory. Just make sure that when you're PRINTing, have quotes on each side of what you want the screen to display. Fool around with the two commands until you get used to them.

IMPORTANT!! The following program line tells KBasic that it should run your program in old style code.

```
OPTION VERYOLDBASIC
```

These lines have to be placed on top in all example programs discussed in this 'Learning Coding' tutorial programs examples, otherwise you will run into a syntax error, because modern BASIC is expected. But for some reasons it is useful to start as a beginner at that old style, just to learn some simple rules and ideas of programming.

To be continued…

That's it! Thanks for reading and trying and have lot of fun with KBasic.

Using The Mouse

The mouse can be used for a lot more than just clicking.

Basic mouse usage

Operation	Examples of use
Click left button	Select a control
Click right button	Context menu
Click, move mouse, let go	Move control around
Hold Ctrl and click	Select multiple controls

Using The Keyboard

You can navigate a lot faster using the keyboard, so be sure to read this page. Some operations have multiple shortcuts; use those you find most convenient.

Standard keys Movement

Keypresses	Action

Backspace	Deletes the character to the left of the cursor.
Delete	Deletes the character to the right of the cursor.
Ctrl+Insert	Copy the selected text to the clipboard.
Ctrl+K	Deletes to the end of the line.
Ctrl+V	Pastes the clipboard text into text edit.
Shift+Insert	Pastes the clipboard text into text edit.
Ctrl+X	Deletes the selected text and copies it to the clipboard.
Shift+Delete	Deletes the selected text and copies it to the clipboard.
Ctrl+Z	Undoes the last operation.
Ctrl+Y	Redoes the last operation.
LeftArrow	Moves the cursor one character to the left.
Ctrl+LeftArrow	Moves the cursor one word to the left.
RightArrow	Moves the cursor one character to the right.
Ctrl+RightArrow	Moves the cursor one word to the right.
UpArrow	Moves the cursor one line up.
DownArrow	Moves the cursor one line down.
PageUp	Moves the cursor one page up.
PageDown	Moves the cursor one page down.
Ctrl+End	Moves the cursor to the end of the text.
Alt+Wheel	Scrolls the page horizontally (the Wheel is the mouse wheel).

System keys

Find text	Ctrl + F
Open file	Ctrl + O
Save file	Ctrl + S

Create A New Project

Projects keep your work together. When developing an appication in KBasic, you work mainly with projects. A project is a collection of files that make up your KBasic application. You create a project to manage and organize these files. KBasic provides an easy yet sophisticated system to manage the collection of files that make up a project. The project window shows each item in a project. Starting a new application with KBasic begins with the creation of a project. So before you can construct an application with KBasic, you need to create a new project. A project consists of many separate files collected in one project directory, where one *.kbasic_project file is and many other files:

- *.kbasic_module
- *.kbasic_class
- *.kbasic_form

In your KBasic-application forms are not only masks for inputting and changing data, but they are the graphical interface of your application. In the eyes of the beholder, they are the application! By creating your application using forms, you control the program flow with events, which are raised in the forms.

Each form in a KBasic application has got a form module with event procedures. Those event procedures react on events raised in the form. Additionally, every module can contain other non-event procedures. A form module is part of every form, when you copy a form, its form module is automatically copied, too. If you delete a form, its form module is deleted as well. KBasic creates a form module automatically. So you need only to write the event procedures and other procedures.

Forms hold your KBasic program together!

The KBasic development environment The KBasic development environment contains windows, toolbars, and editors that make developing your KBasic application easy. It is actually an integrated development environment (IDE). When a development environment is said to be intergrated, this means that the tools in the environment work together. For example, the compiler might find an error in the source code. In addition to displaying the error, KBasic opens the source file in the text editior and jumps to the exact line in the source code, where the error occurred. A large part of application development involves adding an arranging controls on forms. KBasic provides tools to make designing forms a simple process.

Windows KBasic windows are the eares on your screen that you use to develop your programs, including monitoring the status of your projects at any time. These windows include:

Form designer – drag and drop controls, as well as arrange them to this main development area. It is the primary tool you use to create your programs. Project window – work with the different parts of your project in this window. Its views include objects and files.

Property window – lists and lets you control the properties for your controls. You can resize, name, change visibility, assign values, and change colors and fonts of your controls.

Toolbox window – the central place where often used controls can be accessed

Source code editor – it is where you add and customize source code for your project in any phase of the development cycle.

Toolbars KBasic provides an extensive set of toolbars. Toolbars can be docked in the KBasic window or floated on the screeen.

Editor KBasic has one editor for creating and managing KBasic projects. You can use these editor to control the development of your projects, such as editing source code and manipulating classes. The source code editor is a tool for editing source code.

Create A Stand-Alone Application

There are several ways to run your programs by others. There are extended in the future: with installer for your programs for Windows, Mac OS X and Linux…

Single File/Many Files

Give away a binary file using the KBasic compiler (Professional Edition only). If you compile a project, the binary file may contain many files of that project. All files are included in one large file.

For Windows:

Attention! You must always deploy some DLL files with your program (you will find it in the kbasic installation directory): msvcr71.dll QtGui4.dll QtCore4.dll QtSql4.dll in fact all Qt*.dll files you will find there

as well the plugin directory.

For Mac:

Attention! You must always deploy some dylib files with your program (you will find it in the kbasic installation directory): qt.4.dylib and more in fact all qt*.dylib files you will find in Frameworks

as well the plugin directory.

Create An Installer For Your Application

For each platform you like to support with your application you need an installer. There are several free installers you can use.

For Windows:

Inno Setup is a free installer for Windows programs. First introduced in 1997, Inno Setup today rivals and even surpasses many commercial installers in feature set and stability. http://www.jrsoftware.org/isinfo.php

For Linux:

Try the several available package managers for Linux.

For Mac:

With Mac OS X and Disk Utility together with Packager is a InstallShield / Demoshield like installation tool for your applications written for Mac OS X. You can use it for free. Create a self-executable installation binary for Mac in two minutes. See your Mac OS X documentation and installation for details.

Or try http://installjammer.com/

Migrate From VB6/VB.NET

KBasic supports 100% of the syntax of VB6. Many components are equal as well. It is possible to develop GUI applications with well known BASIC syntax in a modern fashion. It comes with truly Java-like object orientation and backward support for VB6 and QBasic, as it is 100% syntax compatible. KBasic combines the expressive power of object-oriented languages like C++ with the familiarity and ease of use of VB6. It allows developers with an installed base of VB6 applications to start developing for a mixed Windows, Mac OS X and Linux environment without having to face a steep learning curve: KBasic uses the familiar visual design paradigm and has a full implementation of the BASIC language.

Some information which might help you to migrate are listed in the following:

- do not use ( ) to access arrays, better use [ ]
- do not use 'Option OldBasic' or 'Option VeryOldBasic'
- do not use 'On Error Goto', better use 'Try Catch'
- do not use 'Nothing', better use 'Null'
- avoid the use of the data type 'Variant'
- do not use 'class_initialisize', better use 'Constructor', the same for the destructor
- do not use 'Call' when calling a sub or function
- do not use 'Optional' and 'IsMissing' with arguments in subs or functions, better use the default value of an argument
- use always 'Do While…Loop' and 'Do …Loop While' instead of the other loops
- always write many comments in your source code
- use in conditions 'AndAlso' and 'OrElse' instead of the binary operators 'And' and 'Or'
- avoid the use of 'ByRef' with primitive data types to get faster execution speed
- do not use 'Data' and 'Def*', like 'DefInt'
- use enumerations instead of many integer constants
- use constants, instead of the use of numeric literals many times in your source code
- avoid the use of 'GoSub', better use real functions or real subs
- avoid the use of 'GoTo', better use loops and other control flow language elements
- 'Let' and 'Set' are not needed
- use 'For Each', when it is possible
- use 'For i As Integer = 0 To 10 Next', instead of declaring the counter variable outside of the loop
- name the variable names without giving them suffixes
- use always ( ) when you call a sub or function

VB.NET vs. KBasic

WHAT IS THE DIFFERENCE BETWEEN VB.NET AND KBASIC?

Porting VB **Applications to Linux or Mac** OS **X**

See a comparision of Visual Basic and KBasic, the following paragraphs show you where they are different in ways that is related to porting your project. For the first time in computer history, with the release of Visual Basic.Net, many Visual Basic developers felt left behind. VB has changed so much that a Visual Basic implementation could not open and run old VB6 source code. It requires Visual Basic developers to modify their source code because .Net cannot run old Visual Basic code without re-working it. So, if you are a Visual Basic developer, you will be porting your projects for new ones or stick with old VB6. Porting means learning and using new keywords and new ideas in general. And keywords actually do not working as the old style, which means a lot of work. Actually, why should you switch to .Net? After learning a new language (.Net), significantly changing your source code base would be involved. Furthermore, you and your project would stick with a massive "framework" underneath and Windows, You would have a program that ran under Windows only. So what is the alternative? Instead of learning a new language and IDE, you better use your knowledge you already have. Use KBasic! It could use much of your Visual Basic code unchanged, and it can use most of your Visual Basic forms in a known way. And the best is, that KBasic is 100% syntax compatible, which means that KBasic and Visual Basic keywords work identically, including Mid, Left, Abs and so on.

Surprisingly

Surprisingly, in KBasic, unlike in other BASIC's, Mid can also assign a string as the Mid statement as VB6 does. Almost all of your Visual Basic knowledge translates directly into KBasic. Of course, there are some form controls that do not work the same. But in summary, code differences between KBasic and Visual Basic lie not in the syntax, but in the object model, which have sometimes different control or property names. The controls (forms and checkboxes and so on) of KBasic and Visual Basic are very equal. KBasic has a much more built-in keywords, provided for backward and forward compatibility with VB6 and object oriented programming in general. But as in VB6, String is exactly the same in KBasic, as in Visual Basic to keep old source working. KBasic and Visual Basic each have some unique keywords as well. For example, Visual Basic has Int, GoSub and Space. KBasic has them, too.

KBasic programs run under Linux and Mac as well

You do not have to write extra code, to simulate VB6 in your KBasic application. KBasic has it all already done for you, so you do not have perfomance problems like in other BASIC's. Going on, KBasic has Min, Max; where Visual Basic does not. To make it easier for you, be sure to spend time with the 'Manual - The KBasic Book' and language reference beforehand. Another tip is to test and play with KBasic to get the feel for it. Normally, there are no language differences that do require you to re-work all of your Visual Basic code. All Visual Basic code imports and runs without changing it much as long as not form controls and other objects are involved. By the way, there are line numbers in KBasic, if you need them, too. Visual Basic programs run under Windows, only. KBasic programs run under Linux and Windows. KBasic has a much richer set of data types than Visual Basic. As such, there are compatible, incompatible and unique data types between Visual Basic and KBasic, with KBasic offering a much wider array of data types. The Boolean datatype has 1 byte size in KBasic, in VB6 it has 2 bytes. The Integer (Long) is 32bit (64bit) in KBasic, in VB6 16bit (32bit). Additionally, KBasic has got new datatypes, such as Integer datatypes, which can be used for VB6 Integer: Int16/Int32 (which is 16/32bit size).

No much difference

There is also no difference between how data is related into structures between the two languages. They both offer modules, forms and classes. No difference is in creating data structures as well. Visual Basic has User Defined Types, KBasic, too. The language structure of KBasic and Visual Basic is similar, but not quite the same, if you want it. KBasic provides some extra new keywords and functionality, to use some of the new object oriented features (other languages like Java or C++ provides for many years). You can use these new features in KBasic or just do the old VB6 stuff.

Error handling is more robust and flexibel in KBasic than Visual Basic. KBasic offers two error-handling methods: Exception, which apply to an entire method, and Try-Catch blocks, which apply to specific sections of code. But if you would like the old way of handling errors, Visual Basic's On Error mechanism is supported as well. Actually, do not switch to .Net! Choosing instead to port to KBasic gives you full Windows, Mac and Linux support, no huge runtime, minimal system requirements, and portability to Windows and Linux.

At least, before starting with .Net, you should try KBasic. You might find KBasic right for you and a clear path to the future. By the way, a VB Project Converter is helping you converting your projects.

KBasic is the BASIC language alternative compiler

Why is KBasic a BASIC language alternative compiler for Windows, Mac OS X and Linux? KBasic is extremely well built, is a powerfull programming language and has a complete IDE: a very professional working environment RAD, which is very similar to VB6, a valid alternative to VB6. It has hundres of commands and functions. KBasic can compile Linux or Mac OS X applications from the Windows versions (or the other way around) and is a cross platform Basic programming language. It is currently available for Windows XP/2000, Mac OS X and Linux/i386. It generates stand alone EXE/BIN and has a great IDE to help beginners creating their applications. KBasic language includes a visual designer to build GUI with all the major elements such as windows, forms, menu and data aware controls, such as buttons, labels and frames, textboxes, radio buttons, combo boxes, list boxes, check boxes… The only true official alternative to VB6: The KBasic project started in 2000 as an open source project… KBasic is a professional development tool, cause this language offers all the elements to design and create professional products. KBasic gives a single, easy-to-use API for writing GUI applications on multiple platforms and the application will adopt the look and feel appropriate to that platform.The IDE is really well thinked and is completed with all the tools the programmer need such as sensitive-to-the-context help.

Well defined language

The language structure, the statements and functions are really simple to understand and the documentation is well written and complete. It comes with a very well done printable manual with more than 140 pages which describe in detail the great number of commands, statements and functions. A powerful and fully featured Basic-like language: It is completely object-oriented and byte-code compiled. The syntax is very similar to VB6 and the language supports common methods and properties. It is a fully object-oriented languages which uses inheritance and polymorphism. The IDE offers also project managment and property editing, syntax highlighting of the source code, code completion, and debugging mode. The IDE provides a source level debugger with breakpoints, and single stepping: step-through code, display values of variables in special windows or by moving the mouse over the variable name in the editor. The main characteristic of KBasic is that it has been created to allow developers with VB6 experience to start programming for Linux, Mac OS X and Windows without having to learn deeply a new language.

KBasic is a high level professional Basic for Linux, Mac OS **X and Windows and its very good Basic language is constantly improved and updated.**

VB6 vs. KBasic

WHAT IS THE DIFFERENCE BETWEEN VB6 AND KBASIC?

In general both are very equal

- KBasic is 100% syntax compatible (keywords like DIM, IF, SELECT CASE…)
- KBasic's IDE is very similar to VB6 (Form Designer, Syntax Highlighting, Auto Completion, Builtin-Help…)
- KBasic supports many controls, which are very equal to the controls of VB6 (Form, CommandButton, TextBox…)
- KBasic comes with a rich set of objects like VB6 (Application, …)
- KBasic is able to create standalone applications (EXE, BIN…)
- KBasic is able to use many databases (in the near future: MySQL, ODBC…)
- KBasic does not support ActiveX, because ActiveX is limited to Windows

Getting away from VB6

KBasic supports 100% the syntax of VB6. Additionally, many GUI components are familiar. And it is possible to develop GUI applications with well known BASIC syntax in a modern fashion. So it comes with truly Java-like object orientation and backward support for VB6 and QBasic, as it is 100% syntax compatible. KBasic combines the expressive power of object-oriented languages like C++ with the familiarity and ease of use of VB6. It allows developers with an installed base of VB6 applications to start developing for a mixed Windows, Mac OS X and Linux environment without having to face a steep learning curve: KBasic uses the familiar visual design paradigm and has a full implementation of the BASIC language.

Why KBasic succeeds

Traditionally, BASIC languages have suffered form the attitude that you should abandon everything you know and start from scratch with a new set of concepts and a new syntax, arguing that it is better in the long run to lose all the old baggage that comes with it. This may be true, in the long run. But in the short run, a lot of that baggage was valuable. The most valuable elements may not be the existing code base, but instead the existing mind base. If you are functioning VB6 programmer and must drop everything you know about VB6 in order to adopt a new language, you immediately become much less productive for many months, until your mind fits around the new praradigm. Whereas if you can leverage off of your existing VB6 knowledge and expand on it, you can continue to be productive with what you already know while moving into the world of object-oriented proramming.

As everyone has his or her own mental model of programming, this move is messy enough as it is withouht the added expens of starting with a new language. The problem with learnig a new language is producitvity. No company can afford to suddenly lose a productive software engineer, because he or she is learning a new language. KBasic is very similar to VB6, not a complete new syntax and programming model. It allows you to continue creating useful code, applying the features gradually as you learn and understand them. This may be one of the most important reasons for the success of KBasic. In addition, most of your existing VB6 code is still viable in KBasic.

Surprisingly

Why is KBasic a BASIC language alternative compiler for Windows, Mac OS X and Linux? KBasic is extremely well built, is a powerfull programming language and has a complete IDE: a very professional working environment RAD, which is very similar to VB6, a valid alternative to VB6. It has hundres of commands and functions. KBasic can compile Linux or Mac OS X applications from the Windows versions (or the other way around) and is a cross platform Basic programming language. It is currently available for Windows XP/2000, Mac OS X and Linux/i386. It generates stand alone EXE/BIN and has a great IDE to help beginners creating their applications. KBasic language includes a visual designer to build GUI with all the major elements such as windows, forms, menu and data aware controls, such as buttons, labels and frames, textboxes, radio buttons, combo boxes, list boxes, check boxes…

The only true official alternative to VB6

The KBasic project started in 2000 as an open source project… KBasic is a professional development tool, cause this language offers all the elements to design and create professional products. KBasic gives a single, easy-to-use API for writing GUI applications on multiple platforms and the application will adopt the look and feel appropriate to that platform.The IDE is really well thinked and is completed with all the tools the programmer needs such as sensitive-to-the-context help .The language structure, the statements and functions are really simple to understand and the documentation is well written and complete.

It comes with a very well done printable manual with more than 140 pages which describe in detail the great number of commands, statements and functions. A powerful and fully featured Basic-like language: It is completely object-oriented and byte-code compiled. The syntax is very similar to VB6 and the language supports common methods and properties. It is a fully object-oriented languages which uses inheritance and polymorphism. The IDE offers also project managment and property editing, syntax highlighting of the source code, code completion, and debugging mode. The IDE provides a source level debugger with breakpoints, and single stepping: step-through code, display values of variables in special windows or by moving the mouse over the variable name in the editor. The main characteristic of KBasic is that it has been created to allow developers with VB6 experience to start programming for Linux, Mac OS X and Windows without having to learn deeply a new language.

Professional Basic Language

KBasic is a high level professional Basic for Linux, Mac OS **X and Windows and its very good Basic language is constantly improved and updated.**

ABOUT

www.kbasic.com.

Quality by Bernd Noetscher Made in Germany (European Union)

KBasic Software is a small software company with headquarter in Frankfurt am Main / Germany. Our flagship product is KBasic Professional, the multi-platform BASIC programming language and environment.

KBasic Software was founded in 2003, although, development of KBasic started in 2000.

I know that it is crucial for my customers to have good tools for making good software. Therefore, I do not compromise my demands for superior design and technical quality when I develop my *products. At KBasic Software, I continously work to improve and expand KBasic Professional to ensure that it always represents the state of the art in usability, look and feel, performance, and stability.*

KBasic has received international recognition from users, industry experts and media.

Address

Bernd Noetscher's KBasic Software
Boseweg 9
60529 Frankfurt am Main
Germany

email: info@kbasic.com
sales related issues: sales@kbasic.com