GAMBAS - Gambas Almost Means BASic! (Linux Bangalore 2004)

Nalini Vasudevan * Arjun Jain †

1 Introduction

Gambas is a Linux replacement for Visual Basic (but by no means it's clone). It is a free development software based on a Basic interpreter. To put it in short, it is a Basic Language with object extensions. If a RAD tool is required for say designing a GUI or a front-end to access MySQL or PostgreSQL databases, then Gambas is the tool. KDE pilot application and network applications (with the Gambas Network ToolBox) can also be designed. Also Gambas has multilingual support and thus a program can be translated into many languages.

Gambas was not designed to be a VB clone, but a "better" Visual Basic like tool for the Linux environment. Gambas is not compatible with Visual basic and will never be..

Gambas comprises of a **Compiler**, an **Interpreter**, an **Archiver**, a **Graphical User Interface** component and a **Development Environment**.

A program written in Gambas has a number of classes, each described in a file.It uses the concept of Object programming. The .class files are compiled and then executed by an interpreter.

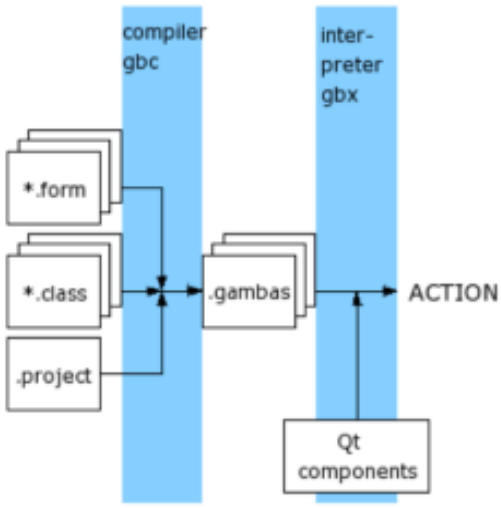

Figure 1: 'How Gambas Works'

2 Gambas - Features

The main advantage of Gambas is that it has a component architecture. This allows extension. We can write our own components as shared libraries that can be added dynamically. This architecture is useful for writing the GUI of the language. In fact, the Gambas interpreter is a text only program and the graphical interface itself is implemented as a Gambas-Qt component. The advantage is that Gambas can me made independent of any tool kit! A Gambas program can first be written and then the toolkit can be chosen

*R.V. College of Engineering, Computer Science & Engineering, Bangalore, India 560059 Tel: +91 94481 07482 Email: naliniv@gmail.com

†R.V. College of Engineering, Computer Science & Engineering, Bangalore, India 560059 Tel: +91 94483 74482 Email: arjunjain@gmail.com

later.(of course the toolkit needs to be written first)

A Gambas project is stored under one directory. The project directory structure is transformed into one sole executable file by the archiver. Only the modified classes are compiled during compilation. External references of a class are solved dynamically at the execution time. Finally, Gambas projects are easily translatable, in any language.

3 Gambas - Architecture

The Development Environment hides this machinery behind a pretty graphical interface.

3.1 Interpreter

The interpreter is a program named gbx. It executes the byte code of the compiled file generated by the compiler.

3.2 Project

A project is a set of files stored in one directory. A project can contain source files (forms, classes, modules) or any data files of any types. The project configuration is stored in a file named ".project".

3.3 Native Classes

The native classes can be used without loading any component. They are integrated in the interpreter, and can be looked upon as a part of the Gambas language.

3.4 Component Interface

The Component Interface is a set of routines and services used by the components to communicate with the interpreter. The interpreter's internals are hidden this way.

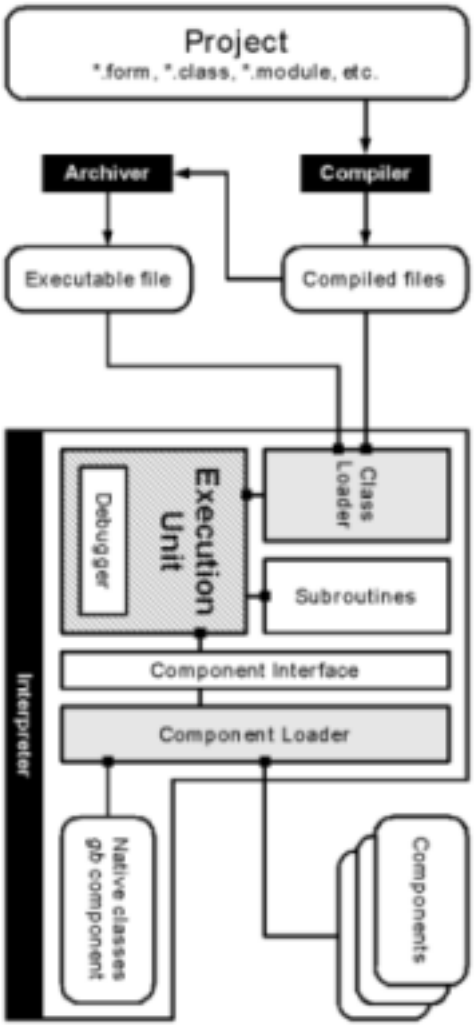

Figure 2: The Gambas Architecture

3.5 Components

Components are shared libraries that are loaded at run time by the interpreter. They can contain new classes and hook routines such as event loop management, shell arguments analyze, etc. They can publish a set of routines as an interface to other components.

3.6 Execution Unit

The execution unit is the heart of the interpreter. It dispatches and executes each bytecode instruction generated by the compiler.

3.7 Subroutines

The subroutines are the interpreter functions associated to the corresponding Gambas Basic functions like Sin( ), Left$( ), etc. or operators like +, &, etc.

3.8 Class Loader

The Class Loader loads compiled forms, classes and modules into the interpreter. If your project was compiled as an executable, i.e. as an archive, the interpreter maps the file into memory instead of loading it.

3.9 Executable file

A Gambas executable file is just an uncompressed archive of your project, the compiled files included. The archive file is marked as a script with the

```
"#!/usr/bin/gbx"
```

magic header, so that Linux executes it by calling the interpreter.

3.10 Compiled files

A compiled file is a binary representation of a class, that contains all information useful to the interpreter : functions transformed to byte code, constants, variables definitions, debugging information, etc.

3.11 The Archiver

The archiver is a program named gba. It transforms your project, the compiled files included, in one "sole executable file".

3.12 The Compiler

The compiler is a program named gbc. It transforms your project's forms, classes and modules files into binary compiled files that can be understood by the interpreter.

3.13 The Component Loader

The Component Loader is the part of the interpreter that loads components shared libraries, gives them access to the Component Interface, and that publishes their interface to other components.

4 Gambas v/s Microsoft Visual Basic

Gambas is not just a replica of Visual Basic. So, you can not expect to simply copy your VB code and compile it under Gambas. This is a great tool for VB developers and people wanting to port their legacy VB projects to the Linux environment. There are perhaps more similarities than differences and one to one relations between VB and Gambas, but the differences are the ones that need more attention.

4.1 Non-Language-Specific Differences

- VB embeds the class code for each form object into the same file as the form definition. Gambas keeps them separate, in a .form and a .class file
  File extensions:

VB	Gambas	Type of File
.vbp	.project	Project definition file
.bas	.module	Module
.cls	.class	Class File
.frm	.form	Form def^n file
.frx	(anything)	Binary files

- Gambas projects are defined as a directory with a .project file in it, and all of the files in that directory. VB can have multiple project files in each directory and can pull the same source file from a different directory into different projects, which has its benefits and disadvantages.

- Screen measurements in VB are done in

"twips", units of 1/1440 of an inch; in Gambas they're done in actual pixels.

- Form controls in Gambas programs are private by default. This can be changed by going into the Project Properties dialog and checking the Make Form Controls Public checkbox.

- Str$(), Val(), CStr()... conversion functions behave differently. For example, Str$() and Val() use the localization settings in Gambas, whereas they don't in Visual Basic. Note that Gambas behavior is more logical :-)

4.2 VB Has It, Gambas Doesn't

- Currently code can't be edited in Break mode in Gambas; the program needs to be terminated first.

- In Gambas, simple datatypes (integer, string, etc.) are passed by value to procedures and functions. They cannot be passed by reference as in Visual Basic. Note that VB passes parameters by reference if the ByVal keyword is not used. Also, the contents of object datatypes (array types, collections, objects) are always passed by reference in both languages!

- There is no such thing as a project-wide global variable in Gambas. (As a workaround, a class called Global can be created and global variables can be declared as static public variables in that class, and then they can be referred to as global variablename in your project.)

- Unless Option Explicit is included in a VB module, variables need not be declared prior to using them. Gambas behaves as if Option Explicit were always turned on, which makes for much better code at the expense of a bit more work.

- There's no direct Gambas equivalent to the Index property of VB form controls. Arrays of controls can be created easily, but it has to be done in code. There's currently no way to do it graphically. Thus, when a control is copied and pasted back on the form it came from, rather than prompting you to create a control array it automatically renames the copied control to an appropriate name.

- Currently transparent labels can't be created in Gambas; the background is always opaque.

- The MouseMove event only occurs when a mouse button is depressed in Gambas. The exception is the DrawingArea control, which has a Tracking property that allows getting mouse move events even if no mouse button is pressed.

- In VB two strings can be put together with the symbol + . Because the + sign is only used for mathematical addition in Gambas, '&' should instead be used, when one string needs to be added to another string.

- In the print command the colon ':' does not work to separate the code. A newline must be taken instead. The print command in VB 3.0 did not make a Linefeed. If it was used it to print out some text with printer.print, then text got lost. The Print Command in Gambas puts everything in one line. There is nothing lost.

- In VB, Mid$() can be used as an instruction to cut out a substring and put in

some other. In Gambas, it can not used it to assign a new substring. For example, in VB: MyString = "the dog jumps": Mid$(MyString, 5, 3) = "fox" results in MyString = "The fox jumps". That does not work in Gambas.It must be done like: MyString = Left$(MyString, 4) & "fox" & Mid$(MyString, 8) . One or more characters which are legal for use in identifiers in VB code, such as underscore ("_") are not acceptable in Gambas.

- Thankfully, in Gambas you GOTO can not be used to trap errors! Instead, CATCH, FINALLY or TRY should be used.

4.3 Gambas Has It, VB Doesn't

- Unlike VB, GUI support need not be compiled in if a Gambas command-line application needs to be written. Unselection of the gb.qt component in Project Properties and making the Sub Main definition does the job.

- Gambas has the concept of control groups, which allows handling of events from any number of different controls with one handler subroutine. This reduces redundant code and can be used to do many of the things VB's control indexes can do, and some things that VB can't.

- Whereas VB makes it impossible to run a program synchronously and receive its output without learning how to do API calls (Shell merely launches the program in the background), Gambas allows doing so using SHELL and EXEC, control the processes started using the Process object, and even read from and write to them, allowing you to easily add functionality with helper applications. This makes it incredibly easy to write Gambas front-ends for almost any command-line procedure.

- As of Gambas 0.60, all of the above can be done with Unix devices and special files as well, such as serial or parallel ports. The /proc filesystem can be used to write a RAID monitor, for example, or named pipes to get multiple channels of information from a back-end program in any other language! To make an odd-shaped window, just the ME.Mask property of the current window needs to be set to a picture that has transparent areas. VB requires API calls and a lot of more work..

- Controls and menu can now be created dynamically, just by instantiating them with the NEW instruction. Gambas forms can be embedded one inside another one: when the first is instantiated, the second should be specified as the parent.

- Controls have Enter and Leave events, which allows to know when the mouse pointer enters a control and when it leaves. This feature can be conveniently be exploited to implement mouse-over effects.

- Data can be easily read from binary files and endianness of its format can be easily managed by using the BIG and LITTLE keyword with the OPEN instruction.

- Gambas uses UTF-8 charset internally, and so projects are fully and easily internationalizable.

- Gambas is Free Software whose development environment is written in itself, allowing its customization to a large degree by using just once BASIC skillset!

5 Installation Notes

If you are still not satisfied with Gambas refer http://gambas.sourceforge.net/ To download the software, visit http://gambas.sourceforge.net/download.html

- Before installing GAMBAS ensure that your system has the X11 development packages, the Qt3 development packages and the KDE3 development packages if you want to compile the KDE component.
- PostgreSQL, MySQL or SQlite development packages is needed if you want to compile database drivers and the libcurl development packages (version 7.10.7 or greater) if you want to compile the network-curl component.
- The SDL and SDL_mixer development packages are required to compile the SDL component. In addition you require the libxml and libxslt development packages if you want to compile the xml components.
- Qt 3.2 is now required because of one Qt functionPicture.Copy() that was missing in older versions of Qt.
- Gambas does not compile with gcc 3.0.x., 3.2 is used instead.
- You must have the right to write to /tmp, otherwise Gambas will not work.

6 The Future...

In the very far future many improvements are planned and soon to be implemented in Gambas.

- A XML component, based on the libxml2 shared library.
- A Perl compatible regular expression component, based on the libpcre shared library.
- A GTK+ component, so that you can really choose the toolkit for any program.
- Manage 64 bits integer in the interpreter with a new datatype named LONG
- Make objects persistent, i.e. store them in databases automatically. A little bit obscure for me at the moment...
- A SDL component, based on the libsdl shared library. The goal is to make games with Gambas !
- And many many more ...

References

[1] *http://gambas.sourceforge.net/*, Gambas Home

[2] *http://www.binara.com/gambas-wiki/*, Gambas Wiki

[3] *http://www.theeasygambasdoku.de/*, The Easy Gambas Documentation