LinuxFocus article number 265
http://linuxfocus.org

Gorm and ProjectCenter, the GNUstep RAD tools

by Georges Tarbouriech
<georges.t(at)linuxfocus.org>

About the author:

Georges is a long time Unix user. He loves GNUstep and the tools this great framework provides.

Abstract:

RAD stands for Rapid Application Development. At the end of the 80's, when NeXTstep was released, it came with an incredible tool, called InterfaceBuilder. Used in conjunction with another tool, named ProjectBuilder, it allowed to build graphical applications in a flash. GNUstep offers a free version of these tools, called Gorm.app and ProjectCenter.app.

At the beginning...

From the computers prehistory, software development has been a great challenge. Computers were quite big in size despite their very little power. They were quite expensive, not really numerous and developers were unable to use them as often as they wished since they had to share them with other people. Then, researchers tried to find a way to make computers execute more than one task at a time to improve efficiency. Obviously, they had to design and create programming languages from scratch, taking into account the poor resources of the available machines.
Thus, during the 60's various new programming languages appeared: LISP, FORTRAN, BASIC, Algol68, BCPL, etc. Next came the B language derived from the above mentioned BCPL, which soon became the C language. This last changed the world of programming.
The Object Oriented (SmallTalk, Objective C, C++, etc) languages appeared later, with the "graphical era".
In the 80's some machines were providing graphical OSes (Apple Macintosh, Commodore Amiga, Atari ST, etc) and the X Window System was in the works. At the same time, a company was working on a GUI for IBM OS2, called Presentation Manager. Before finishing that job, this company released its "own" GUI for its DOS, called... Windos. The first two versions were hardly usable, but... the third one started it all. The MvAI (Microsoft very Artificial Intelligence) was born !
That is, every user became a computer scientist. Since then we have seen "great" applications written using Excel or Word and Visual Basic:-(

Never mind !
Fortunately, long before we reached the above situation, NeXTstep was born and with it, came Interface Buider. This tool allowed you to create a GUI for your application in a very short lapse of time and with great ease. From there, this kind of tools has been spreading. Among others, let us mention Omnis, 4D, Delphi, Kylix, etc. A few of them are multiplatform while the vast majority is dedicated to Windos. Let us also mention that there are free toolkits using such a philosophy, Gtk (Gimp Tool Kit) for instance. Proprietary Unixes also provide these sort of tools.
The most important feature of these tools is that you do not have to write the code for the 200 windows of your application, but only the one to manage the data.
Whether you like this sort of tools or not is not the point. The development time is short: it is a fact (hence the name, "Rapid Application Development").
GNUstep provides us with free RAD tools. They are called Gorm and ProjectCenter. Of course, these tools are very "young" but they do work. Let us have a look at them.

GNUstep

To be able to use both Gorm and ProjectCenter, you need to install GNUstep. How to do this is beyond the scope of this article. You will find everything you need at the GNUstep website. This includes source code, HOWTOs, tutorials, etc.
You can also have a look at these articles: GNUstep, the open source OpenStep and GNUMail.app, the portability evidence.
The tests for the present article have been done under FreeBSD 4.7 with Window Maker 0.80.1, using gnustep-make-1.5.0, gnustep-base-1.5.0, gnustep-gui-0.8.2 and gnustep-back-0.8.2. These last are the latest GNUstep unstable versions. You can also use the stable versions if you wish. Last but not least, we used the gcc 3.0.4 compiler.

Gorm.app

Gorm stands for Graphic Object Relationship Modeler (or perhaps GNUstep Object Relationship Modeler, as said in the README file). It is a clone of the above mentioned NeXTstep Interface Builder (or today’s MacOS X).
Gorm is the work of Richard Frith-Macdonald who started the project. Today Gregory Casamento is the current maintainer and he does most of the work with Pierre-Yves Rivaille. The present version is 0.1.9. Newer CVS snapshots are available from http://savannah.gnu.org/projects/gnustep.
You can download the latest stable version from the GNUstep website.
The philosophy behind Gorm (and Interface Builder) is to provide the user with objects found in palettes and drag these objects to empty windows to design the graphical components of your application.
The objects can be buttons, fields, checkboxes, panels, etc. That is, everything you can add to a window to make it user-friendly. Next, you can modify them using inspectors. From the inspectors, you can change the attributes, define connections, size, help and manipulate classes for the selected objects. After creating a class, you can add outlets and actions to the objects.
Next you instantiate the class, what creates a new object (the instance) in the Gorm main window, and you connect the outlets and the actions to the corresponding components. You do this just by dragging

the mouse from the instance to the selected object to connect outlets and from the object to the instance to connect actions. Last, you create the skeleton of the class source files, and you’re done. More on this later.

ProjectCenter.app

ProjectCenter, as the name says, is the "heart" of a project. It is a clone of Project Builder found under NeXTstep and Mac OS X.
ProjectCenter is the work of Philippe C.D.Robert and the present version is 0.3.0. Like Gorm, you can download it from GNUstep website going to the Developer apps section. Of course you can get the latest CVS snapshot: we use it for this article and it is version 0.3.1.
From ProjectCenter you can create a project, its interface (using Gorm), write its source code; you can build this project and run it (debugging is not yet available). In short, you can manage all the resources required by the project: source code, documentation, libraries, subprojects, interfaces, etc.
When you create a new project, you can choose its type. You can select between application, bundle, tool, library and Gorm application.
Among other things, ProjectCenter provides you with an editor in which you will be able to complete the Gorm skeleton code.
How do Gorm and ProjectCenter work together ? Very well, thank you !
More seriously, we will use two examples to illustrate it.

A few notes

This article is NOT a tutorial. The idea is to show the ease of use of these tools while insisting on the fact that you will be able to use the same code for both GNUstep (that is, a lot of Unix platforms... and, if you like "struggling", under Windos too) and MacOS X. The only thing you will have to do is to design the interface on every platform, since the nib (InterfaceBuilder or Gorm) files are not portable (at least for now).
The above mentioned GNUMail.app article showed the portability from a user’s point of view. This one will focus on the developer’s point of view, still with portability in mind. That is, in GNUMail.app we used the work of Ludovic and friends and here we do create a GUI application for both GNUstep and MacOS X.
Many tutorials are available, either for MacOS X or GNUstep. You can reach most of the GNUstep’s ones from the GNUstep website or from http://www.gnustep.net, but let us mention a few of them.
- An application using Gorm and ProjectCenter by Pierre-Yves Rivaille.
- The Nicola Pero’s tutorial page
- An older tutorial on how to create a HTMLEditor:
http://stepwise.com/Articles/Technical/HTMLEditor/
To learn more, you can also check the source code, the nib files, etc, of the existing GNUstep applications (Gorm, ProjectCenter, GNUMail, GWorkspace, etc) and of course, the gnustep-examples.

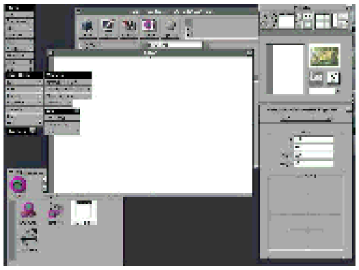
Just call it "VerySimpleEditor"

Among the numerous MacOS X tutorials for InterfaceBuilder available on the net, we will use the following one as a first model: http://www.macdevcenter.com/pub/a/mac/2001/05/18/cocoa.html. The author, Mike Beam wrote a lot of more sophisticated tutorials available from http://www.macdevcenter.com/pub/ct/37.
Why this one ? Because it provides you with a working text editor without writing a single line of code. This shows the power of these development tools, wether they work under MacOS X or under GNUstep.

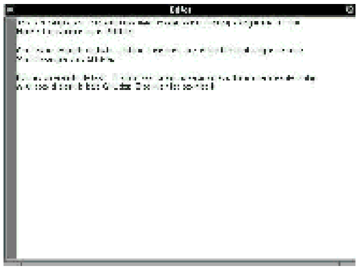
Using ProjectCenter.app and Gorm.app under GNUstep we create a very simple text editor able to cut, copy, paste. Obviously, you will not be able to save your work: remember, we will not write a single line of code. Using ProjectBuilder and InterfaceBuilder under MacOS X we will do the same. Obviously, there is a lot to do to improve this editor and we will leave this as an exercise for the reader. Again, this article is not a tutorial !
Here we go.

Under GNUstep

Open ProjectCenter.app and create a new project called Editor. Choose a Gorm Application project at the bottom of the window before saving its name. This will provide you with an Interfaces item in the left column of ProjectCenter.
Clicking *Interfaces* displays *Editor.gorm*. Double-click *Editor.gorm* and Gorm.app opens. Select the default window (MyWindow) and using the tool inspector change the name to *Editor* in Attributes.
From the palette, drag a TextView to the Editor window. The TextView is the biggest object found in the selected palette using the rightmost icon at the top of the Palettes window. Resize this object to make it fill the window, and you are done.
Again, using the GormInternalViewEditor inspector (while the TextView is selected), choose *Size* and change the values to make them match the Editor window size values. These last are obtained in the same way, that is, selecting the window and checking the size in GormNSWindow inspector. If you do not change the X and Y values, for instance, you will not be able to use the full width of the editor, whether you resize the window or not.
Save all in the Gorm Document menu and quit to go back to ProjectCenter. Select the Build icon and click in the new build icon in the second horizontal half of the window. Everything should go well if you defined the right preferences for your compiler, debugger, etc. For example, using FreeBSD, you must change make to gmake (including the path) clicking in the Settings icon of ProjectCenter. Check also the paths from the Preferences menu in ProjectCenter.
If the built succeeded (it should have !), just do the same with *Run* and you will see the Editor application. Just play with it, writing, cutting, pasting, etc. Obviously, you can restart it later using the openapp command.
How long did it take ? Well, I should say a few minutes.
Here is what it should look like during the development phase:

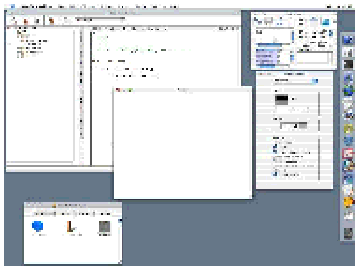


and the resulting application:

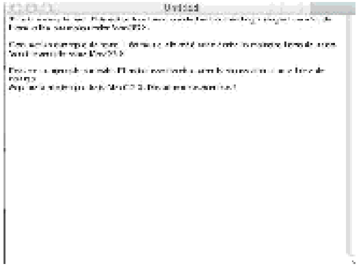


Under MacOS X

Nothing much to say since you will have to do the same as above. Here is what it looks like while designing the GUI:



Here is the editor "at work":

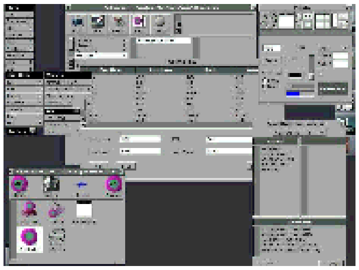


And now... a true working application

Now we choose another example from Mike Beam. This time is a full working application, able to manage data: an address book. Mike's tutorial about the address book (like every other) is recommended reading to understand how the "thing" works. Also check the tutorial list since Mike provides different steps of the developing process for one and the same application allowing to improve it.
Again we create and run the application on both GNUstep and MacOS X.

Under GNUstep

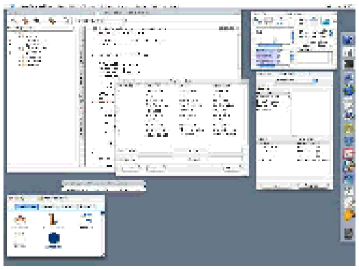
Like you did for the Editor example, start ProjectCenter.app. Select a Gorm application and call it AddressBook. From ProjectCenter launch Gorm by double-clicking in Interfaces -> AddressBook.gorm. Drag a TableView from the palette to the default window. In other words, follow Mike's tutorial like you would under MacOS X. You will have to adapt a few things since they work differently in Gorm and in InterfaceBuilder.
For example, the number of columns in the TableView cannot be defined from the attributes inspector in Gorm. To keep things simple, just copy a column and paste it next to it to get the required number (4 in our case). You should end with something like this:



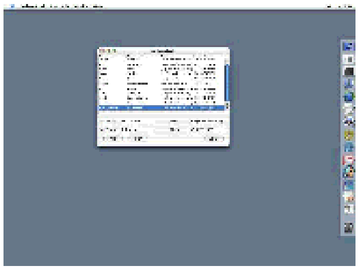
When you have finished with it, save everything and go back to ProjectCenter to type or modify the code. In case you made some mistakes, Mike provides you with the full application source. If you download it, enough to copy and paste the code to your own Controller.m and Controller.h files generated by Gorm. Do not to include (import in Objective C terminology) *Cocoa.h* since it is reserved for MacOS X. By the way, you should keep the skeleton "written" by Gorm and change a few things. For instance, replace the *void* with IBAction in Controller.h and Controller.m. Add IBOutlet before the outlets id in Controller.h.
Anyway, if you feeel like it, you can keep the whole provided code: just replace the cocoa inclusion with *#import <AppKit/AppKit.h>*. Now, you can build and run the application.
There you are: you can start playing with your new address book.
Here is what it looks like:

Under MacOS X

Mike Beam did the whole job: what else could I add ?
Here is a screenshot of the development process under MacOS X:



Here is the resulting application:



So what ? Well, it took a few minutes to design the GUI and the whole code represents a bit more than 2000 bytes. Not bad is it ?
Again, there are a lot of improvements that can be done but the whole thing works ! Now, you can play with the code to get a better address book...

GNUstep and MacOS X

Obviously, GNUstep development tools cannot be as ahead as Apple’s. Apple and NeXT represent a 15 years old experience with hundreds of developers. GNUstep is the work (for free) of a few individuals

having to do something else for a living. Accordingly, do not be surprised to find, for instance, much more available classes in InterfaceBuilder than in Gorm. Remember, Gorm is at version 0.1.9 (or 0.2.0). Furthermore, we did the tests the "hard" way. That is, we "ported" from OS X to GNUstep. The other way round would have been easier because of the above mentioned differences between the tools.
For example, porting applications developed under MacOS X 10.2 would be much more difficult since the new Apple development tools have much improved. As already said, there are many new available classes or more elaborated ones.
However, the tools rely on the same philosophy whether they work under GNUstep or MacOS X... and GNUstep improves every day. One thing looks very nice to me: GNUstep people really work together. They do help each other when individual projects are concerned and they also contribute in improving GNUstep core. This is the Free Software way of working I like. Congratulations for such a behavior Mr.Fedor and friends.

Imagine...

The goal of this article was to show the power of the GNUstep "RAD" tools, Gorm.app and ProjectCenter.app. Despite their "youth" they can help you develop nice applications in a very easy way.

Futhermore, these tools provide a very pleasant way of working while being very efficient. Objective C is a very compact language and, in my opinion, much easier to learn than C++ for someone with C knowledge (I know, I already said so !). This allows to design nice looking applications (well, it is a matter of taste, but I do love this look and feel) while keeping them rather small in size.
I must admit that I never recovered from the shock I received when I first met the NeXT machine. The fact Apple released a modern version of NeXTstep delights me. This is also why I am very fond of projects such as GNUstep or Window Maker. However, if I love free software I am not a "fundamentalist" and accordingly, I am not against proprietary software (well, maybe a bit against a specific editor... but, just a bit !).
GNUstep can benefit from Apple... but Apple can benefit from GNUstep too. GNUstep is not an Apple competitor, it is free software. As far as I know, free software is widely used in OS X. This to say that bringing even more free software to Apple cannot be a bad thing. What Ludovic and friends did with GNUMail.app is a very good example of what could happen.
"I had a dream"... Apple was providing most of its development tools source code to GNUstep. GNUstep and Apple developers were working together to bring great applications to Unix users. And slowly, people were realizing they could live without Windos...
Unfortunately, it was a dream ;-)
Anyway, if you do not know GNUstep and its applications, feel free to give them a try. Remember, GNUstep is a framework and tools such as Gorm and ProjectCenter provide you with everything to create, to invent. In other words, with a bit of imagination, you can develop "products" much different from what we can see nowadays: Windos applications clones !
We are living in a great time !

Thanks...

To the GNUstep people: A.Fedor, N.Pero, G.Casamento, P.Y.Rivaille, N.Roard, L.Marcotte, R.Frith-Macdonald, P. C.D.Robert, E.Sersale, A.Froloff, F.Kiefer, M.Viviani, M.Guesdon and all those I forgot for the very great job either for the framework or for the applications.
To Window Maker people: A.Kojima, D.Pascu and friends for bringing us a free NeXTstep interface for X.
To J.M.Hullot and B.Serlet for inventing InterfaceBuilder.
To "Steve Jobs INC." for bringing us NeXT, NeXTstep and MacOS X.
To all the people not mentioned here having contributed in making our professional life much less sad.

2005-01-14, generated by lfparser_pdf version 2.51