The monthly GNU Column

BRAVE GNU WORLD

Pingus under Mandrake

Welcome to another issue of Georg CF Greve's Brave GNU World. As this marks the column's third anniversary you'll find a few words of celebration as well as some new projects, but we'll kick off the proceedings with another Free game

Pingus

Pingus is a game developed under the GNU General Public License by Ingo Ruhnke, Giray Devlet, Cagri Coltekin, David Philippi and Alberto Curro. The game was inspired by DMA Design's proprietary game Lemmings, in which a player had to direct a group of lemmings through perilous levels to the safety of the exit. The only way to influence the flow of the lemmings was to give some of them special jobs, such as diggers, climbers or even bombers. Pingus, as the name might suggest, lets you do all these things to little penguins, which bear a striking resemblance to Tux, the mascot of the Linux kernel.

Game development was started in 1998 and after an announcement on Slashdot, gained the input of some graphically adept users, which gave Pingus a very attractive look. The graphics even triggered spin-offs like Xpenguins, which lets Pingus' protagonists roam the desktop.

At the end of 2000 development came to an almost complete stop, but a year later some fresh programmers helped overcoming this involuntary pause. As it stands, the game is still not finished and according to Ingo Ruhnke "it still doesn't feel like a real game". This may also be partially because no sound effects have been implemented although there is plenty of music. Some more interesting levels are also needed.

Help is wanted in many forms: developers are as welcome as people contributing to sound effects or more levels. Level design does not require programming knowledge, by the way, since everything is done with XML. The game itself was written in C++ and runs under GNU/Linux. It may be possible to also run it on other Unix-based systems, but the developers are more interested in a Win32 port at the moment.

The immediate plan is to finish the Windows port and release a new, completely playable version. Afterwards multiplayer, as well as network support, and a consistent storyline are needed to finish the game.

At the moment, the game can only be recommended to users willing to play around with half-finished games and who might like to contribute parts to it.

Process View Browser

The Process View Browser (pvbrowser) by Rainer Lehrig provides a structure for process visualisation. This is important in all areas where technical processes are to be visualised or controlled. Examples of proprietary programs performing similar tasks are WinCC or Wonderware.

The project consists of a server and a browser, which communicates with the user. Unlike in comparable projects, all configuration is done on the server-side. Users can modify the server according to their own needs, i.e. they can write routines interacting with the hard or software defining which objects are to be displayed and controlled in which way. These components are then displayed by the browser according to the information provided by the server over the network.

The programming language used for this project was C++ with Qt as the graphical toolkit for the browser and ANSI C for the server. The project is very platform-independent: it runs on GNU/Linux as well as under Windows or VMS. According to the information provided by Rainer Lehrig, the browser is faster on a 330MHz GNU/Linux notebook than on a

1GHz Windows NT system, which he blames on the networking code.

The platform independence is also linked to the one downside of the project, which made me think very hard about whether I should feature it in the Brave GNU World. The Process View Browser is only Free Software under GNU/Linux, for which it is released under the GNU General Public License. The project is proprietary under Windows and VMS.

However, two factors convinced me to write about it in the Brave GNU World. First of all this area has been dominated by purely proprietary versions until now, so the project certainly takes a step into the right direction. Also the user is capable of using it as entirely Free Software as long as the GNU/Linux platform is being used.

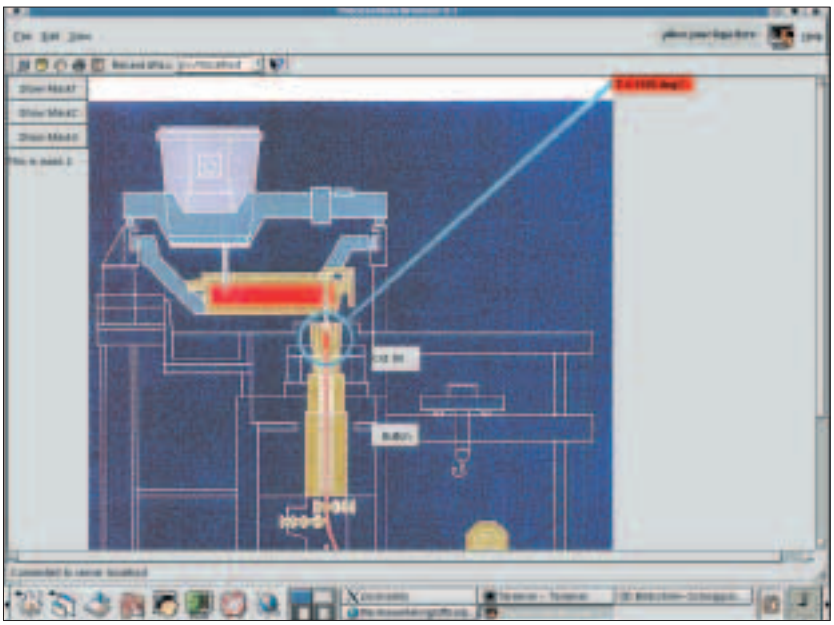
Additionally, the license-situation of the Qt toolkit is very much comparable since only the X11-Version is available as Free Software, while the Windows and Mac versions remain proprietary. Since Qt is being used by the Process View Browser, a Free Windows version of the pvbrowser would not really help a user since dependency on the proprietary Qt still remains.

Qt is a respected Free Software library under GNU/Linux that is being used for many important projects like the K Desktop Environment (KDE) and has been mentioned in the Brave GNU World many times already. So it seemed unjust to not mention the Process View Browser because of a comparable licensing policy. Hope remains that the versions for systems other than GNU/Linux will also be available as Free Software in the long term – a hope that equally applies to Qt and the Process View Browser.

Rainer Lehrig has been working on the pvbrowser all by himself so far and is now looking for others who might be willing to help testing or contribute ideas and code. Volunteers are also sought for documentation.

If you are interested in this field, please feel free to participate in the Process View Browser. I recommend only doing so for the GNU/Linux version, however and authors of documentation should take care to release it under the GNU Free Documentation License or a similar license. Only in these cases will it be reasonably safe to assume that work contributed will continue to benefit the Free Software community.

In order to prevent possible misunderstandings I'd like to emphasize that the described problems do not lessen the contribution of Rainer Lehrig to Free Software. Bringing Free Software into a hitherto proprietary field is always a very important task. Still it remains important to be aware of the problems and understand what they mean.



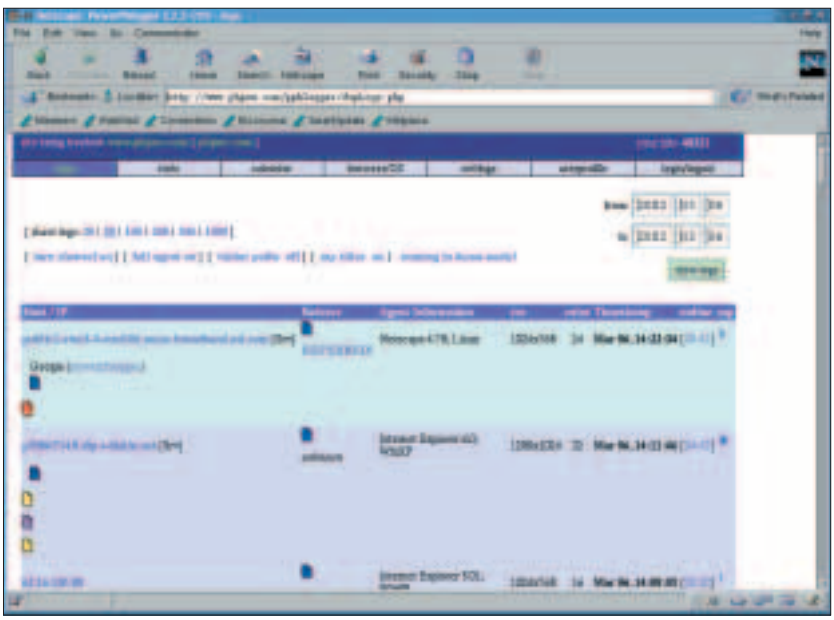
Process View Browser showing an image with widgets

PowerPhlogger

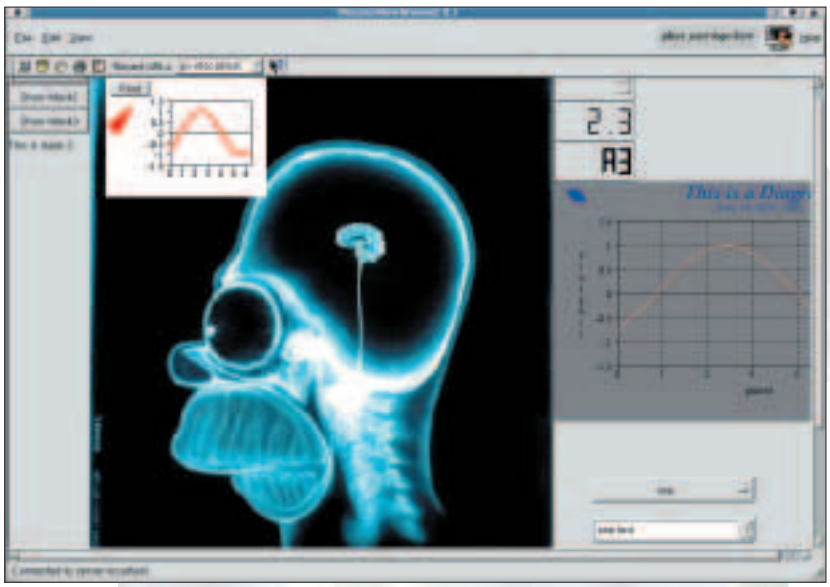
In June 2000 Philip Iezzi began working on a software to host Web page counters under PHP and the result of his work has been available since January 2001 as PowerPhlogger under the GNU General Public License.

Similar services are relatively common on the net, but they are usually proprietary and also rather unsatisfactory. PowerPhlogger allows everyone to set up such a service, even if the relevant pages do not support PHP. The creation of accounts with these services can either be done by an administrator or the users themselves. An example for this is the gratis PowerPhlogger service, Freelogger.

The functionality of PowerPhlogger surpasses that of most proprietary solutions. Among the features is the counting of real visitor numbers through "unique hits" instead of counting every page access. This is done by IP-comparison in combination with a cookie-check and a timeout defined by the user.



PowerPhlogger showing statistics



Process View Browser showing diagrams and other widgets

PowerPhlogger also offers so-called "visitor paths", enabling administrators to trace the path a user has taken to reach the page. It also keeps track of the time a user spent browsing the page.

Of course the PowerPhlogger is capable of displaying counters on pages that can be fitted to the layout of the page through TTF and user-defined colours. Even the layout of the statistics page can be modified to suit the user's taste with CSS modifications. Additionally, the project has been internationalised for 16 languages and supports different time zones.

All data is currently stored in a mySQL database, but version 3 of the PowerPhlogger, which is pencilled in for an October 2002 release, will contain a database abstraction layer. Also some of the less desirable sections of code will be cleaned up and rewritten for object orientation.

Help is welcome in any form, including financial support. Philip also needs volunteers to provide support in the online-forum.

GNU Stow

GNU Stow is an extremely useful project for everyone installing software that's either not available for the distribution you're using or has to be installed from source for other reasons.

Under normal circumstances such activities tend to act as proof that the second law of thermodynamics applies to computer systems, as well: entropy remains the same or rises, but it never decreases. In other words this means that systems have the tendency to become increasingly messy. GNU Stow offers a solution for this.

Stow has its own directory tree, which usually resides at */usr/local/stow*. New packages are installed into their own subdirectories in this directory tree. Calling Stow will create symbolic links, making sure all files of the package appear in the standard filesystem hierarchy where other programs look for them. If the package is to be uninstalled, one can simply delete the install directory and/or remove the links by calling Stow again.

Stow was originally written by Bob Glickstein in 1993 using Perl, but lack of time forced him to suspend development. GNU Stow is now maintained by Guillaume Morin, who has had only mainly minor changes to implement since the project has been stable for some years now.

If you haven't tried out GNU Stow yet, I can only recommend taking a look.

GNU gettext

GNU gettext is a project probably known to most developers already and also a package that only developers and translators will ever come in direct contact with. It does play a crucial role for users, however, since it allows programs to communicate with them in their native language. So I'd like to introduce this important component here.

Although details should be spared, a short introduction into the functional concept seems useful at this point: when developing programs, all output is normally written in English. All user interaction strings are collected by GNU gettext in a single file.

If a program is to be localised, translators can make a copy of this file, translate all the strings in this simple ASCII file into their native language and mail it back to the developer. If this file is then copied into the right directory under the right name, the program supports that language after the next compilation.

When the user runs the program, GNU gettext will try to supply him or her with the messages in the user's preferred language. Whenever this isn't possible because the translation is not complete or doesn't exist at all, gettext falls back to the original/English version.

Supporting incomplete translations was one of the design goals of GNU gettext, because programs evolve step by step and very often the translators are one or two steps behind the developer.

GNU gettext consists of several tools under the GNU General Public License as well as libraries under the GNU Lesser General Public License. It complies with the Unix-standards X/Open and Li18nux2000, was originally written in 1995 by Ulrich Drepper. It has since rapidly become the de-facto standard for software internationalisation in and outside of the GNU Project.

Bruno Haible recently took over as the GNU gettext maintainer. He currently focuses on expanding GNU gettext to more languages and is considering integrating simple spell checking sometime in the future.

Bruno felt two anecdotes were worth sharing with the Brave GNU World audience. First of all there is a distinguished and apparently quite active team

working on the translations from American to British English. This seemed somewhat easier than a translation to Japanese to him.

He also warns other programmers against trying to translate their programs in to another languages themselves – especially if this is not their native language. As far as he was concerned, some of the translations he has encountered are worse than no translation at all.

Experience shows that translations into French, Swedish, German and Spanish are provided quite often, other languages could use more volunteers, however. Localising a program for your own language is a very good way of furthering Free Software in a practical way that needs little technical expertise.

Three years of Brave GNU World

What began as an experiment is now three years old, so I'm tempted to take a quick glance back.

The column initially began as a wild idea between Tom Schwaller and me on the 512-node GNU/Linux Cluster "CLOWN", at which I gave my first appearance as European speaker for the GNU Project.

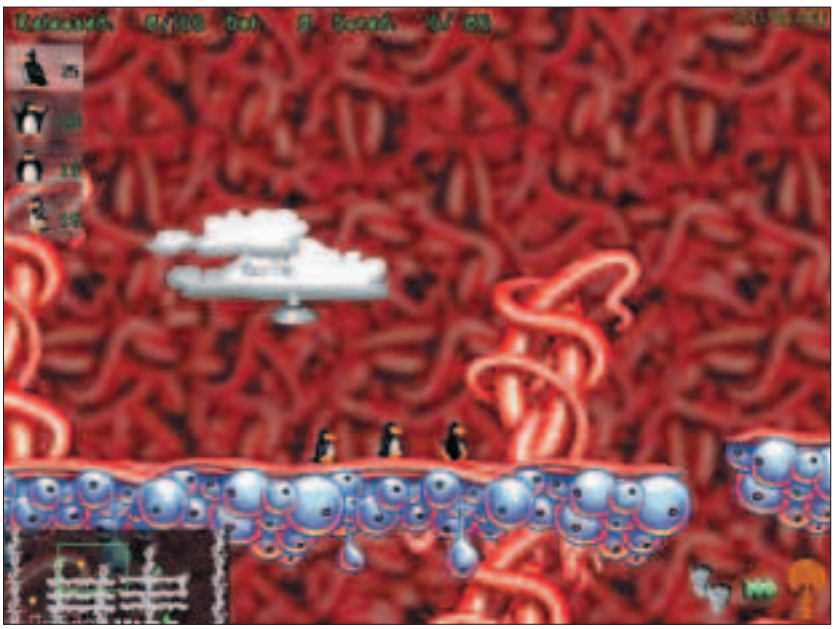
Tom approached me with the idea of a GNU column. Arriving back home I knew that I wanted to try writing a column that would also have the mix of technical and philosophical issues making the GNU Project so special. Still, I was sceptical whether this was possible and whether I'd be able to fill the column each and every month in time for the print-issue.

From the very first moment it was clear that the column should also be published on the Net in order to make it available to as many people as possible. Doing this only in German seemed of limited use, so the initial issue was first written in German and then translated into English by me in order to be released online together.

After releasing issue two, something remarkable happened. Within a few days, Okuji Yoshinori and Francois Thunus contacted me and asked whether I'd agree to them translating the column into Japanese and French. Of course I was quite happy about that and immediately included them in the production process of the Brave GNU World.

The dam broke; more volunteers contacted me for other translations and soon other magazines requested permission to print the Brave GNU World. Today the column appears in up to 7 languages online and 4 magazines worldwide. Without the help of so many volunteers, this would never have been possible.

The companions of the early days mentioned above all went their own ways by now, their jobs being taking over by others. I'd like to list everyone helping with the Brave GNU World, but there is hardly enough place for it. On a single issue you'll easily find 30 people helping as scout, proofreader, translator, Web master and so on. Even if some have participated for a long time now, there is always a certain amount of fluctuation.



Pingus version 0.4.1

To all these people and the other supporters of the Brave GNU World I would like to express my heartfelt thanks to for the past three years. I'd also like to thank all those who contacted me in person or via email to tell me about interesting projects, give feedback or discuss topics they had a different opinion about. Their involvement was a seminal part in filling the Brave GNU World with life.

...to another year

Enough said, I hope we'll see another good year for Brave GNU World and of course I don't want to finish without the mandatory request for feedback, comments, new projects, questions and ideas. Which project is incredibly useful, funny or good and still unknown to many users? Please send answers to the usual address.

Info

Send ideas, comments and questions to Brave GNU World	*column@brave-gnu-world.org*
Homepage of the GNU Project	*http://www.gnu.org*
Homepage of Georg's Brave GNU World	*http://brave-gnu-world.org*
"We run GNU" initiative	*http://www.gnu.org/brave-gnu-world/rungnu/ rungnu.en.html*
Pingus homepage	*http://pingus.seul.org*
XPenguins homepage	*http://xpenguins.seul.org*
Process View Browser homepage	*http://pvbrowser.sourceforge.net*
PowerPhlogger homepage	*http://www.phpee.com*
Freelogger homepage	*http://www.freelogger.com*
GNU Stow homepage	*http://www.gnu.org/software/stow*
GNU gettext homepage	*http://www.gnu.org/software/gettext*
"History and Philosophy of the GNU Project"	*http://www.gnu.org/philosophy/greve-clown.en.html*