JavaScript | Inform 6 & 7 | Falcon | Sleep | Enlightenment | PHP

LINUX JOURNAL™

Since 1994: The Original Magazine

OCTOBER 2008 | ISSUE 174

THE STATE OF LINUX AUDIO SOFTWARE

Inform 7 Don't Get Eaten by a Grue!

REVIEWED

» **HP Media Vault 5150**

» **Scalent's Virtual Operating Environment**

Managing **PHP Code**

Guido van Rossum on PYTHON 3

Get Your Sleep from **Java**

www.linuxjournal.com

➕ **Martin Messner** Insights from SUSE's Security Team Lead

Enlightenment E17 Lightweight Alternative to KDE and GNOME

$5.99US $5.99CAN

10>

0 09281 03102 4

AMD
Opteron
64

THE QUIET REVOLUTION

32 processor performance in your desktop

HiPerSTATION 8000

System Highlights:

- Optimal multi-threaded application performance utilizing (8) eight native Quad-Core AMD Opteron™ processors
- Up to 256GB of DDR2-667 memory
- Next generation dual x16 PCIe technology offers scalable graphic bandwidth
- 2TB of storage
- Designed to be quiet enough to utilize in an office environment; ⅓ the noise of server version
- Ideal for usage in Computer Aided Designs, Visualization, and Digital Content Creation

HPC SYSTEMS INCORPORATED

www.hpcsystems.com 888-725-3472

AMD Solution Provider PLATINUM

GSA GS-35F-0596R Contract Holder

SBA 8(A) CERTIFIED

CONTENTS

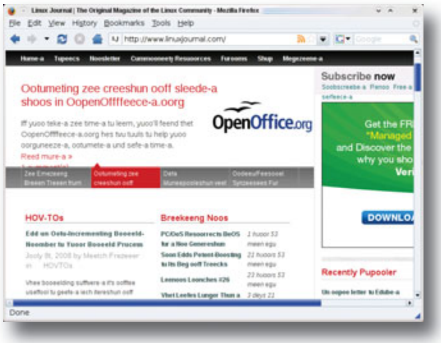
OCTOBER 2008
Issue 174

FEATURES

ON THE COVER

CONTENTS OCTOBER 2008 Issue 174

COLUMNS



Linux Journal | The Original Magazine of the Linux Community - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://www.linuxjournal.com/

Home-a Toopeecs Noosletter Cummooneety Resoorces Furoums Shup Megezeene-a

Ootumeting zee creeshun ooff sleede-a shoos in OopenOfffeece-a.oorg

Iff yuoo teke-a zee time-a tu leern, yuoo'll feend thet OopenOfffeece-a.oorg hes tuu tuuls tu help yuoo oorguneeze-a, ootumete-a und sefe-a time-a.

Reed mure-a »

Zee Emezeeng Breeen Treeen thum | Ootumeting zee creeshun ooff | Deta Muneepooleshun veet | Oodeeuffeesooel Syncessees Fur

HOV-TOs

Edd un Ootu-incrementing Booeeld-Noomber tu Yuoor Booeeld Proces
Jooly 8t, 2008 by Meetch Frezeer in HOVTOs
Vhee booeelding suffwere-a it's ooftee useffool tu geete-a iech itereshun ooff

Breekeeng Noos

PC/OoS Resoorrects BeOS fur a Noo Generashun — 1 huoor 53 meen egu
Soon Edds Petent-Booeting tu Its Beg ooff Treecks — 21 huoors 53 meen egu
Leenoos Looonches #26 — 23 huoors 53 meen egu
Vhet Leefes Lunger Thun a — 3 deys 21

Subscribe now
Soobscreebe a Fenoo Free a sefeece-a

Get the FR "Managed and Discover the why you sho Ver

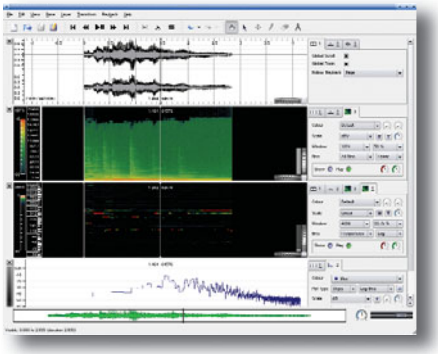
DOWNLO

Recently Pupooler
Un oopee letter tu Edube-a

Done

IN EVERY ISSUE

INDEPTH



REVIEWS

Next Month

HIGH-PERFORMANCE COMPUTING

High-performance computing means a lot more than fast CPUs in your desktop. If you want to do some serious number crunching, you need some serious processing power. Next month is our HPC issue, and we tell you all about the Roadrunner supercomputer, massively parallel computing with CUDA and even squeaking some extra oomph from the GPU with general-purpose programming languages.

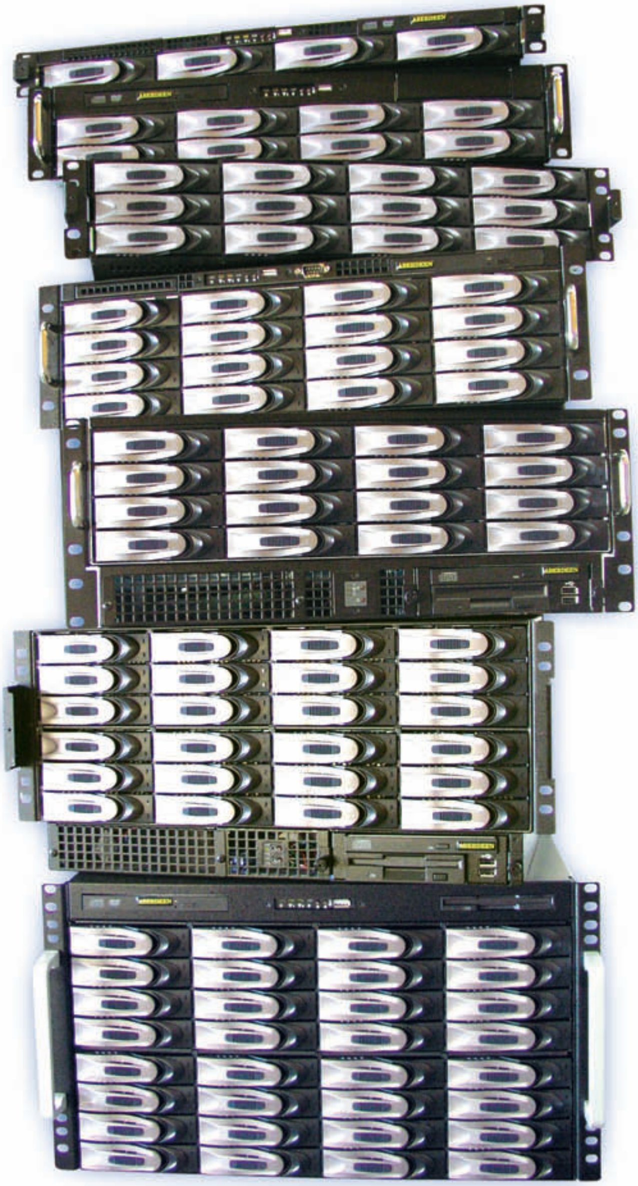
If petaflops and clusters are your bread and butter, November will be an issue that will make your mouth water. Add in our regular cast of columnists and product reviews, and it will be an issue you won't want to miss!

USPS *LINUX JOURNAL* (ISSN 1075-3583) (USPS 12854) is published monthly by Belltown Media, Inc., 2211 Norfolk, Ste 514, Houston, TX 77098 USA. Periodicals postage paid at Houston, Texas and at additional mailing offices. Cover price is $5.99 US. Subscription rate is $29.50/year in the United States, $39.50 in Canada and Mexico, $69.50 elsewhere. POSTMASTER: Please send address changes to *Linux Journal*, PO Box 980985, Houston, TX 77098. Subscriptions start with the next issue. Canada Post: Publications Mail Agreement #41549519. Canada Returns to be sent to Bleuchip International, P.O. Box 25542, London, ON N6C 6B2

The Straight Talk People℠
SINCE 1991

ABERDEEN

MIRROR MIRROR IN THE RACK



4-DRIVE 1U ABERNAS

Up to 4TB Capacity
- Dual-Core Intel® Xeon® Processor
- 2GB DDR2 Memory
- 300W Power Supply
- From 1TB to 4TB

Starting at **$2,495**

8-DRIVE 2U ABERNAS

Up to 8TB Capacity
- Dual-Core Intel Xeon Processor
- 2GB DDR2 Memory
- 500W Redundant Power
- From 2TB to 8TB

Starting at **$3,995**

16-DRIVE 3U ABERNAS

Up to 16TB Capacity
- Dual Quad-Core Intel Xeon Processors
- 2GB DDR2 Memory
- 650W Redundant Power
- Quad LAN and SAS Expansion
- From 8TB to 16TB

Starting at **$7,995**

24-DRIVE 5U ABERNAS

Up to 24TB Capacity
- Dual Quad-Core Intel Xeon Processors
- 2GB DDR2 Memory
- 950W Redundant Power
- Quad LAN and SAS Expansion
- From 12TB to 24TB

Starting at **$10,495**

32-DRIVE 6U ABERNAS

Up to 32TB Capacity
- Dual Quad-Core Intel Xeon Processors
- 2GB DDR2 Memory
- 1350W Redundant Power
- Quad LAN and SAS Expansion
- From 16TB to 32TB

Starting at **$13,495**

40-DRIVE 8U ABERNAS

Up to 40TB Capacity
- Dual Quad-Core Intel Xeon Processors
- 2GB DDR2 Memory
- 1350W Redundant Power
- Quad LAN and SAS Expansion
- From 20TB to 40TB

Starting at **$15,995**

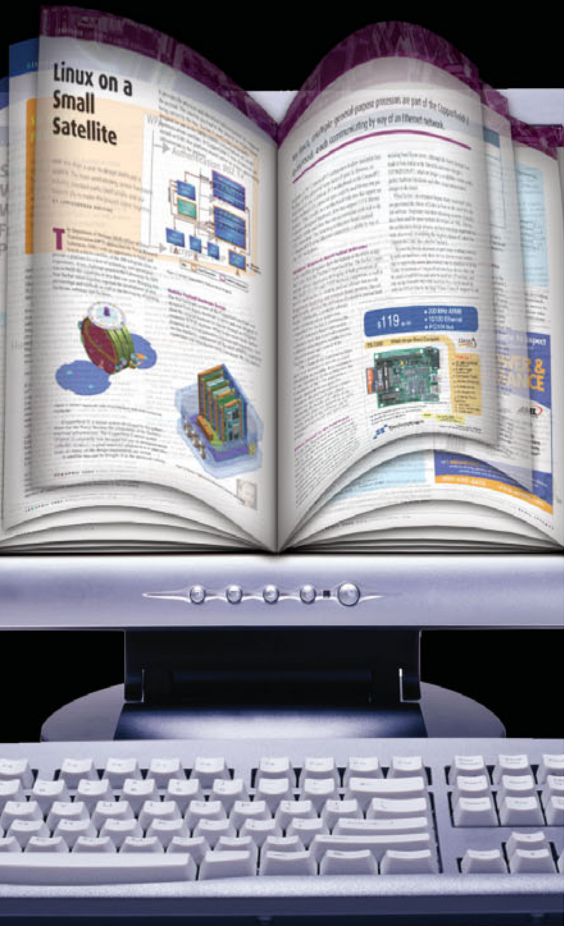
Features & Benefits:
- With available NAS-to-NAS Mirroring and Auto-Failover over LAN
- Native Linux-based OS featuring iSCSI
- Supports SMB, CIFS, NFS, AFP, FTP
- DOM-based (Disc-On-Module) OS
- Independent Data Drive RAID
- RAID 0, 1, 5, 6 Configurable
- Convenient USB Key for Recovery
- NAS-2-NAS Replicator
- PCBackup Utility
- iSCSI Target Capable
- All 3U+ models support expansion via cost effective JBODs

intel®
Xeon®
inside™
Quad-core.
Unmatched.

"Aberdeen surpasses HP … markedly higher scores … AberNAS 128 boasts outstanding features"
Network Computing—Aberdeen AberNAS 128

Intel, Intel Logo, Intel Inside, Intel Inside Logo, Pentium, Xeon, and Xeon Inside are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries. For terms and conditions, please see www.aberdeeninc.com/abpoly/abterms.htm. lj027

888-297-7409
www.aberdeeninc.com/lj027



LINUX JOURNAL

Executive Editor	Jill Franklin jill@linuxjournal.com
Associate Editor	Shawn Powers shawn@linuxjournal.com
Senior Editor	Doc Searls doc@linuxjournal.com
Art Director	Garrick Antikajian garrick@linuxjournal.com
Products Editor	James Gray newproducts@linuxjournal.com
Editor Emeritus	Don Marti dmarti@linuxjournal.com
Technical Editor	Michael Baxter mab@cruzio.com
Senior Columnist	Reuven Lerner reuven@lerner.co.il
Chef Français	Marcel Gagné mggagne@salmar.com
Security Editor	Mick Bauer mick@visi.com

Contributing Editors
David A. Bandel • Ibrahim Haddad • Robert Love • Zack Brown • Dave Phillips • Marco Fioretti
Ludovic Marcotte • Paul Barry • Paul McKenney • Dave Taylor • Dirk Elmendorf

Proofreader	Geri Gale

Publisher	Carlie Fairchild publisher@linuxjournal.com
General Manager	Rebecca Cassity rebecca@linuxjournal.com
Director of Sales	Laura Whiteman laura@linuxjournal.com
Regional Sales Manager	Joseph Krack joseph@linuxjournal.com
Circulation Director	Mark Irgang mark@linuxjournal.com
System Administrator	Mitch Frazier sysadm@linuxjournal.com
Webmistress	Katherine Druckman webmistress@linuxjournal.com
Accountant	Candy Beauchamp acct@linuxjournal.com

***Linux Journal* is published by, and is a registered trade name of, Belltown Media, Inc.**
PO Box 980985, Houston, TX 77098 USA

Reader Advisory Panel
Brad Abram Baillio • Nick Baronian • Hari Boukis • Caleb S. Cullen • Steve Case
Kalyana Krishna Chadalavada • Keir Davis • Adam M. Dutko • Michael Eager • Nick Faltys • Ken Firestone
Dennis Franklin Frey • Victor Gregorio • Kristian Erik • Hermansen • Philip Jacob • Jay Kruizenga
David A. Lane • Steve Marquez • Dave McAllister • Craig Oda • Rob Orsini • Jeffrey D. Parent
Wayne D. Powel • Shawn Powers • Mike Roberts • Draciron Smith • Chris D. Stark • Patrick Swartz

Editorial Advisory Board
Daniel Frye, Director, IBM Linux Technology Center
Jon "maddog" Hall, President, Linux International
Lawrence Lessig, Professor of Law, Stanford University
Ransom Love, Director of Strategic Relationships, Family and Church History Department,
Church of Jesus Christ of Latter-day Saints
Sam Ockman
Bruce Perens
Bdale Garbee, Linux CTO, HP
Danese Cooper, Open Source Diva, Intel Corporation

Advertising
E-MAIL: ads@linuxjournal.com
URL: www.linuxjournal.com/advertising
PHONE: +1 713-344-1956 ext. 2

Subscriptions
E-MAIL: subs@linuxjournal.com
URL: www.linuxjournal.com/subscribe
PHONE: +1 713-589-3503
FAX: +1 713-589-2677
TOLL-FREE: 1-888-66-LINUX
MAIL: PO Box 980985, Houston, TX 77098 USA
Please allow 4–6 weeks for processing address changes and orders
PRINTED IN USA

LINUX is a registered trademark of Linus Torvalds.

SHAWN POWERS

10 PRINT "Hello World"; 20 GOTO 10

There is a particularly cheesy scene in the movie *The Core*, in which the geeky dude claims to speak one language: one zero one zero zero. He also claims to require Hot Pockets in order to do any serious coding. Thankfully for us, our programming choices (along with dietary options) include much more than pure binary. This month, we tackle the subject of languages—specifically, programming languages.

In every issue of *Linux Journal*, we try to give you some useful tips and timely information on the programming scene. This month, we look at a few different languages to give you a better feel for some of the options out there. Although there is never just one way to solve a problem, some languages are a better fit for specific needs. The trick is picking the right tool for the job.

If scripting is your secret sauce, you might find Reuven M. Lerner's article on JavaScript event handlers useful. Or, for that matter, Dave Taylor's continuing series on scripting the Internet Movie Database might prove insightful. Because it's the programming issue, we have several other scripting articles as well. Raphael Mudge teaches us about the Sleep language, which uses Java and was inspired by Perl. Giancarlo Niccolai walks us through using Falcon, which is a language he wrote to fit a specific need. Thankfully, he's released it to open source, so we can all benefit.

Sometimes scripting just doesn't fit the bill, and here at *Linux Journal*, we're sensitive to such things. Federico Kereki shows us a great way to keep track of our code in PHP using Eclipse, an Integrated Development Environment (IDE).

When it comes right down to it, some of us have very little interest in learning to program. That's fine too. James Gray interviews the Python creator, Guido van Rossum. Whether you are a coder or not, it's pretty exciting to learn about the changes in version 3.0 of the extremely popular Python language. Heck, it's not even backward compatible! You won't want to miss the reasons why.

If you're not a programmer, that doesn't mean you have to use this issue for spit-wad ammunition, however. Maybe Kyle Rankin's column on integrating *Rock Band* controllers into your Linux machine is more up your alley. Combined with the open-source game *Frets on Fire*, you can take advantage of the *Rock Band* Wii controllers without even owning a Wii. Using the drum set, you can play a synthesized drum kit with Hydrogen. The amazing part is that Linux recognizes the controllers right out of the box! Thank you, Nintendo, for using standard USB ports.

We also have an interview with the SUSE Security Team Lead Marcus Meissner. You think *you're* worried about security exploits? Marcus worries for a living. His work helps protect our systems from unwelcome visitors. Speaking of which, what issue would be complete without Marcel Gagné's column? He does indeed stay true to the issue focus and discusses languages—specifically, Klingon. If that's too geeky for you, perhaps Pig Latin or even Swedish Chef-ese is more interesting. Marcel has it all and shows you how to translate for yourself.

If you really want to talk to your computer, you have to teach it how to interact with you. Daniel Bartholomew teaches us how to create our own Zork-like game using the Inform language. He includes instructions on using both Inform 6 and Inform 7. In fact, a downloadable version of the program he wrote for the article is available on our FTP site (see the article for details). If phrases like, "You're likely to be eaten by a Grue" spark some nostalgia, you won't want to miss it.

And, as we do every month, we have our regular cast of columnists, reviews and indepth articles. We hope that whether you're a programmer, a hacker or just a Linux enthusiast, you'll enjoy this issue. I know I sure have.

```
   010100110110010101100101001000000111100
101101111011101010010000001101110011001010
111100001110100001000000110110101101111011
01110011101000110100000100001■
```

Shawn Powers is the Associate Editor for *Linux Journal*. He's also the Gadget Guy for LinuxJournal.com, and he has an interesting collection of vintage Garfield coffee mugs. Don't let his silly hairdo fool you, he's a pretty ordinary guy and can be reached via e-mail at shawn@linuxjournal.com. Or, swing by the #linuxjournal IRC channel on Freenode.net.

servers ▶ DIRECT®

MORE PRODUCTS, BETTER SERVICE, GUARANTEED.

RELIABILITY WHEN YOU NEED IT. IN OTHER WORDS, ALL THE TIME.

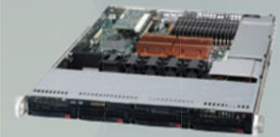
With ServersDirect® Systems based on the Intel® Xeon® Processor, you can count on high availability for your business-critical computing solutions.

GO STRAIGHT TO THE SOURCE! 1.877.727.7887 | www.ServersDirect.com

SDR-S1208-T00

STARTING AT **$599**

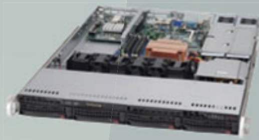
- Supermicro Mini 1U Rackmount Server with 260W Power Supply
- Supermicro Server Board w/Intel® 946GZ Chipset
- Support up to a Dual-Core Intel® Xeon® 3000 Series processor
- TPM Support
- 1x 3.5" Internal Drive Bay
- 2x Intel® 82573 PCI-e Gigabit LAN Port



SDR-S1305-T04

STARTING AT **$999**

- Supermicro 1U Rackmount Server with 520W Power Supply
- Supermicro Server Board w/Intel® 5000V Chipset
- Dual Intel® 64-bit Xeon® Quad-Core or Dual-Core 5400/5300/5200 sequence
- Support up to 16GB DDR2 667 & 533 FB-DIMM
- 4 x SATA Hot-swap Drive Bays
- Dual-port Gigabit Ethernet Controller



SDR-S1330-T04

STARTING AT **$1,519**

- Supermicro 1U Rackmount Server with 650W High-efficiency Redundant Power Supply
- Supermicro Server Board w/Intel® 5400 Chipset
- Dual Intel® 64-bit Xeon® Quad-Core or Dual-Core 5400/5300/5200 sequence
- Support up to 64GB DDR2 667 & 533 FB-DIMM
- 4 x SATA Hot-swap Drive Bays
- Dual-port Gigabit Ethernet Controller

SDR-C2205-T00

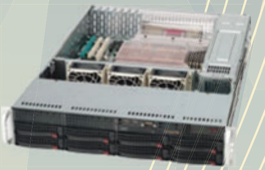
STARTING AT **$699**

- 2U Rackmount Server with 300Watt Power Supply
- Supermicro Server Board w/Intel® 3200 + ICH9R Chipset
- Support up to a Quad-Core Intel® Xeon® 3300/3200 series processor
- Up to 8GB unbuffered ECC / non-ECC DDR2 800/667 SDRAM
- 4x 3.5" Internal Drive Bay
- Dual-port Gigabit Ethernet Controller

SDR-C2300-T06

STARTING AT **$1,149**

- 2U Rackmount Server w/600W Power Supply
- Supermicro Server Board w/Intel® 5000P chipset
- Dual Intel® 64-bit Xeon® Quad-Core or Dual-Core
- Support up to 16GB DDR2 667 & 533 FB-DIMM
- 6 x 3.5" Hot-swap Drives Trays
- Dual-port Gigabit Ethernet Controller



SDR-S2301-T08

STARTING AT **$1,599**

- 2U Rackmount Server w/700W High-Efficiency Redundant Power Supply
- Supermicro Server Board w/Intel® 5000P chipset
- Dual Intel® 64-bit Xeon® Quad-Core or Dual-Core
- Support up to 64GB DDR2 667 & 533 FB-DIMM
- 8 x 3.5" Hot-swap Drives Trays
- Dual-port Gigabit Ethernet Controller

SDP-CP200-T04

STARTING AT **$899**

- Pedestal Server with 600W Power Supply
- Intel® Server board with 3000 Server Chipset
- Support up to a Quad-Core Intel® Xeon® 3300/3200 series processor
- Up to 8GB unbuffered ECC / non-ECC DDR2 800/667 SDRAM
- 4x 3.5" Internal Drive Bay (Hot-swap Optional)
- Dual-port Gigabit Ethernet Controller

SDR-IP300-T06

STARTING AT **$1,199**

- Intel Pedestal Server with Redundant (1+0) 650W Power Supply
- Supermicro Server Board w/Intel® 5000V chipset
- Dual Intel® 64-bit Xeon® Quad-Core or Dual-Core
- Support up to 16GB DDR2 667 & 533 FB-DIMM
- Support up to 6 x 3.5" Hot-swap Drives Trays
- Dual-port Gigabit Ethernet Controller

SDP-C5300-T24

STARTING AT **$2,969**

- 5U Rackmount Servers with Redundant 1350W Power Supply
- Supermicro Server Board w/Intel® 5000P chipset
- Dual Intel® 64-bit Xeon® Quad-Core or Dual-Core
- Support up to 32GB DDR2 667 & 533 FB-DIMM
- Support up to 24 x 3.5" Hot-swap Drives Trays
- Dual-port Gigabit Ethernet Controller

SERVERS DIRECT CAN HELP YOU CONFIGURE YOUR NEXT HIGH PERFORMANCE SERVER SYSTEM - CALL US TODAY!

Our flexible on-line products configurator allows you to source a custom solution, or call and our product experts are standing by to help you assemble systems that require a little extra. Servers Direct - your direct source for scalable, cost effective server solutions.

1.877.727.7887 | www.ServersDirect.com

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, Pentium, and Pentium III Xeon are trademarks of Intel Corporation or it's subsidiaries in the United States and other countries.

intel® Xeon® inside™

Dual-core. Do more.

letters

Billix | Rails | Gumstix | Zenoss | Wiimote | BUG | Quantum GIS
LINUX JOURNAL
Since 1994: The Original Magazine of the Linux Community
REVIEWED: Neuros OSD and Cradlepoint PHS300
WE'VE GOT BUGs AND OTHER COOL PROJECTS TOO
E-Ink + Gumstix Perfect Match?
Billix Kiss Install CDs Goodbye
How To: 16 Terabytes in One Case
Learn to Fake a UFO Landing Video
Wiimote Linux Interface HOW-TO

More Hardware

I gave renewing a lot of thought this time around. You see, it used to be automatic, but you folks seem to be catering primarily to the software crowd. That's okay, but spread yourselves a bit more freely, please. I am a hardware engineer and would appreciate some more from that side of the house. After thinking long and hard, I decided to give you one more try for one year. You see, I actually found that I am looking forward to my issue each month; it's just that the programming stuff is hard to get exited about after many years in hardware. I'm sure I am not alone.

--
Des Cavin

We greatly appreciate your feedback, and we will keep your request in mind.—Ed.

Linux Everywhere

In the Letters section of the last few issues, there has been mention of Linux being used in different consumer products. Today, while surfing the Web looking for a new car stereo, I came across the SoundStream VIR-4100N, which is listed on the Web site as a "4.3"-wide Touch Screen, Din Size, In-Dash, Fully Motorized TFT Monitor, with Navigation/DVD/AM/FM". The third bullet, in a long list of features, says "OS 2.6.x Linux-based software, 400MHz processor for fast recalculation times".

--
David Baldock

Correction

Regarding Dan Sawyer's review of the Cradlepoint PHS300 [titled "Hot and Bothered at Starbucks"], in the August 2008 issue: the CTR350 does *not* come with a battery, which is what separates it from the PHS300.

SATA RAID Problems

In response to the article "One Box. Sixteen Trillion Bytes" by Eric Pearce in the August 2008 issue of *Linux Journal*: I also was excited about the prospects of using larger, cheaper, SATA RAID solutions to cut costs on our ever-growing storage needs. I'm not saying it's a bad idea, but there are problems that are not apparent until after you make the investment. These problems can be dealt with, and I dare say that a niche market is waiting on someone to do this, but I've seen other companies fail to do it in the past.

The key problem that does not show up until later is that of SATA drive firmware compatibility. We had eight 400G drives in a box that got turned into the file server for the company. At first, we had a RocketRaid card. Some research suggested the problems we were

There's Always Another Way

Reading Dave Taylor's excellent article on extracting movie information from IMDb for a Twitter movie trivia game [*LJ*, July 2008], I could not help but think, "why doesn't he ...", on several occasions. To put my money where my mouth is, I rewrote his code snippets as a—IMHO—more readable bash script using more concise code snippets. Here it is:

```
#!/bin/bash

# imdb-top-250-movies.sh
#
#   Felix C. Stegerman <flx@obfusk.net>
#   2008-07-12 [14:15]
#
# --
#
# Output "title | year" for each of the top 250 movies at IMDb.
#
# --
#
# Usage: imdb-top-250-movies.sh > IMDB-TOP-250-MOVIES
#
# --

PREFIX='http://www.imdb.com'
  CHART="/chart/top"
  TITLE='/title/tt[0-9]+/'

function get_top_250_chart ()
{
  wget -O - "$PREFIX/$CHART" \
    | grep -E -o "$TITLE" \
    | sed 's!^!'"$PREFIX"'!'
}

function get_movie_and_year ()
{
  wget -O - "$1" \
    | grep '<title>' \
    | sed -r 's!^.*>(.*)<.*!\1!' \
    | sed -r 's!\(([0-9]{4})(/(I{1,3}|I?V|VI{1,3}|I?X))?\)$!| \1!'
}

for URL in $(get_top_250_chart); do
  get_movie_and_year "$URL"
done

# vim: set ft=sh tw=70 sw=2 sts=2 et:
```

Also, this version correctly parses—for example, "Crash (2004/I)" is changed to `"Crash | 2004"`.

--
Felix C. Stegerman

having were due to the drivers and the card being substandard. So, we invested in an eight-port, 9000 series 3ware card.

It turns out that 3ware has to test every firmware revision of every SATA drive with its cards, because many perform very poorly. After losing three drives, we tried to replace a bad one with a 500G drive, because we couldn't get a 400G drive at local stores, and performance fell through the floor. I updated the firmware, researched on 3ware.com and called 3ware. It came down to poor quality control on the part of SATA manufacturers, and there was nothing we could do about the drives we had. In summary, if you use SATA RAID, make *sure* you check 3ware's compatibility list.

The bad news is that you probably don't know the firmware revision until after you open the drive and use it. Refunds get really tricky after that, so you'll probably go through a few drives trying to find good ones. It's probably a good idea to have multiple hot spares in your system and to have some pretested drives lying around.

--
JT Moree

Try Hyperic HQ

I'm writing in response to Jeremiah Bowling's article on Zenoss in the August 2008 issue of *LJ*. Although I'm sure it is a good product, there is another one that provides the same, if not more, capability with less setup time: Hyperic HQ. I believe *Linux Journal* mentioned it in the New Products section a while back.

Hyperic uses XML over HTTPS, using agents installed on hosts to provide metrics back to the HQ Server. It can monitor 10s of 100s of different metrics across different platforms, and it is open source. The process of adding devices and services is performed by the agent through its detection capabilities, which are reported back to the server automatically when the agent is registered. SNMP isn't even needed unless desired. It is open source, but Hyperic sells an enterprise version that comes with support. The only thing it seems that Hyperic doesn't have that Zenoss does have is the geographic mapping capability. Hyperic provides all the metric charts, alerting, roles and monitoring that Zenoss does. I urge those who try out Zenoss to take Hyperic for a spin also. I'm using it at work, and it makes monitoring systems a breeze.

One final note, Jeremiah discusses the setup of SNMP and the use of community strings. Although specifying a custom community string is better than the default, anyone still can sniff the network to determine what the string is, so it really doesn't provide much security. As I mentioned above, Hyperic agents communicate with the Hyperic server using HTTPS, which we all know really is secure.

I appreciate Jeremiah's article, despite my mention of a similar product. I wasn't aware of Zenoss and always am glad to have my awareness expanded.

--
Brandon McCombs

Look for an article on Hyperic HQ in an upcoming issue.—Ed.

Failed to Mention the Cost

I was interested to see an article about building E-Ink devices in the August 2008 issue of *LJ*. I acknowledge that the article involves a clever bit of engineering, but you fail to mention that display you used, AM200, costs $3,000 US (**store.nexternal.com/shared/StoreFront/default.asp?CS=eink&StoreType=BtoC&Count1=555858790&Count2=472999215**). It's disappointing this fact was left out to make the project sound more reasonable. Normally, I've found your articles are more honest about the costs.

--
Jim Putman

It's a Small World

Although I always enjoy reading about scientific applications for Linux, "One Tale of Two Scientific Distros" by Doc Searls in your August 2008 Upfront section, hit particularly close to home. My cousin, David Auty, is an astrophysicist, currently researching neutrino oscillations at Fermilab. Here's a link to a brief description of his work: **www.sussex.ac.uk/physics/profile104724.html**.

--
James Knott

More on SATA

The article "One Box. Sixteen Trillion Bytes" [August 2008] showed the use of a 3ware utility to interrogate performance of the RAID-connected SATA disks. Having some systems with problematic hard disks, that got me wondering what general utilities are available to support SATA disks—for example, what goes beyond hdparm?

I discovered that the newer sdparm utility is designed to support SATA disks. However, despite extensive Google searches, other than one good overview, there is a dearth of how-to guides that highlight what one can safely change, or the likely or possible implications of changing the various parameters.

I'm sure a future article in *Linux Journal* by an sdparm expert would be of great interest to many since most systems are now using SATA or SAS drives rather than PATA.

--
David Cartwright

PHOTO OF THE MONTH

Have a photo you'd like to share with *LJ* readers? Send your submission to publisher@linuxjournal.com. If we run yours in the magazine, we'll send you a free T-shirt.

Hannah C. Bigrigg, five years old, Canton, Ohio. Submitted by Travis Bigrigg.

LINUX JOURNAL

At Your Service

MAGAZINE

PRINT SUBSCRIPTIONS: Renewing your subscription, changing your address, paying your invoice, viewing your account details or other subscription inquiries can instantly be done on-line, **www.linuxjournal.com/subs**. Alternatively, within the U.S. and Canada, you may call us toll-free 1-888-66-LINUX (54689), or internationally +1-713-589-3503. E-mail us at subs@linuxjournal.com or reach us via postal mail, Linux Journal, PO Box 980985, Houston, TX 77098-0985 USA. Please remember to include your complete name and address when contacting us.

DIGITAL SUBSCRIPTIONS: Digital subscriptions of *Linux Journal* are now available and delivered as PDFs anywhere in the world for one low cost. Visit **www.linuxjournal.com/digital** for more information or use the contact information above for any digital magazine customer service inquiries.

LETTERS TO THE EDITOR: We welcome your letters and encourage you to submit them at **www.linuxjournal.com/contact** or mail them to Linux Journal, 1752 NW Market Street, #200, Seattle, WA 98107 USA. Letters may be edited for space and clarity.

WRITING FOR US: We always are looking for contributed articles, tutorials and real-world stories for the magazine. An author's guide, a list of topics and due dates can be found on-line, **www.linuxjournal.com/author**.

ADVERTISING: *Linux Journal* is a great resource for readers and advertisers alike. Request a media kit, view our current editorial calendar and advertising due dates, or learn more about other advertising and marketing opportunities by visiting us on-line, **www.linuxjournal.com/advertising**. Contact us directly for further information, ads@linuxjournal.com or +1 713-344-1956 ext. 2.

ON-LINE

WEB SITE: Read exclusive on-line-only content on *Linux Journal*'s Web site, **www.linuxjournal.com**. Also, select articles from the print magazine are available on-line. Magazine subscribers, digital or print, receive full access to issue archives; please contact Customer Service for further information, subs@linuxjournal.com.

FREE e-NEWSLETTERS: Each week, *Linux Journal* editors will tell you what's hot in the world of Linux. Receive late-breaking news, technical tips and tricks, and links to in-depth stories featured on **www.linuxjournal.com**. Subscribe for free today, **www.linuxjournal.com/enewsletters**.

GENSTOR

"Company With A Vision"

GNS 2450 Green Storage Appliance

Green
Scalable
Reliable
Performance
Plug & Play

Genstor Systems GNS 2450 is built on fully optimized Highly Efficient dual-core AMD Opteron™ processor model 2214 HE running Open-E® DSS™ (Data Storage Server).

GNS 2450 is an all-in-one IP-Storage with NAS and iSCSI (both target and initiator) functionality. Excellent price performance value, enhanced management and superior reliability is great for organizations of all sizes.

Hardware & Software

- One or two Highly Efficient dual-core AMD Opteron™ processor-based platform
- 8GB RAM (Expandable to **64GB**)
- Dual GbE NICs (Optional 10GbE)
- 2U rackmount chasis with 12 SATA HDD bays with redundant power supplies
- Open-E® DSS™ (Data Storage Server) with NAS and iSCSI functionality

Features & Benefits

- Administration: Powerful, secure and easy intuitive GUI for configuration and remote management
- Network Management: Along with standard Network Gateway support, the **GNS 2450** also supports DHCP Client, Multiple NIC, Adapter Fault Tolerance (AFT), Adaptive Load Balancing (ALB), Proxy Settings, IP-Sec, etc.
- Flexibility & Scalability: The **GNS 2450** storage appliance is highly scalable and flexible to take care of your future expandability needs

Price: $5200 (for 6TB raw storage)

GENSTOR SYSTEMS, INC
780 Montague Expressway # 604
San Jose, CA 95131
Phone: 408-383-0120
Fax: 408-383-0121

AMD Opteron 64

open-e

Please contact Genstor sales@genstor.com for your custom hardware needs

1-877-25-SERVER • www.genstor.com • sales@genstor.com

Open-E, Open-E DSS and the Open-E logo are registered tradedmarks of Open-E, Inc.

AMD, AMD Arrow logo, and AMD Opteron are trademarks or registered trademarks of Advanced Micro Devices, Inc.

UPFRONT

NEWS + FUN

diff -u

WHAT'S NEW IN KERNEL DEVELOPMENT

Keeping **firmware** in the kernel is a somewhat controversial issue, because it means keeping binary blobs of data in an open-source project. But removing it, as **David Woodhouse** is trying to do, turns out to be controversial also. Folks like **David S. Miller** are violently opposed to the idea on the grounds that it would break certain drivers (like tg 3) or make them much worse. Also, it takes a part of the kernel that essentially had been approved by **Linus Torvalds**—even in binary form—and puts it in the hands of someone else. Unlike GCC, libc and other projects on which the kernel depends, this external firmware project would not be anything like a universal tool; it just would be a key part of the kernel that was stored and maintained outside the official source tree.

But, now that the effort actually is underway, it may be unstoppable. A lot of folks find the problem interesting on a technical level, which generally means they'll eventually find solutions that satisfy everyone. For now, it seems as though firmware definitely will be relegated to its own little spot in the kernel sources. From there, it could end up getting a separate git tree altogether.

Willy Tarreau has adjusted the **2.4 release schedule**, based on the results of an informal survey of existing 2.4 users. He found that about 80% of current users didn't upgrade, because a lot of their employees or customers would be inconvenienced by any hiccups in the upgrade path or by the downtime required to do the upgrade successfully. The rest of the users could upgrade easily if they wanted to or if they knew how, but they either were running it on their home systems and simply didn't need the benefits of 2.6, or they were running it on firewalls, routers and so on, and the folks maintaining them didn't have a clearly defined upgrade path—although Willy reckoned it wouldn't be too hard to do.

From all this, Willy concluded that he would put out stability fixes more frequently, so people could get what they needed to fix whatever small problems remained on their 2.4 systems, but that he'd slow down on putting out major releases, and do that only for versions with new PCI IDs, big driver updates, compiler support changes and so on. Willy also suggested that someone write up a complete description of the differences between 2.4 and 2.6 and how to address all the problems of upgrading.

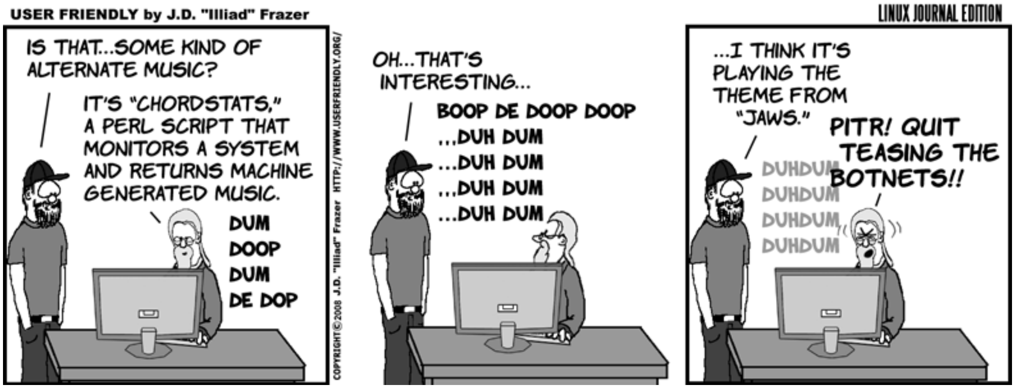
It's useful to be able to write to files under **CramFS**, **SquashFS** and even **mounted CDs**, knowing that the changes will not be permanent and will go away after a reboot. **Arnd Bergman** has recently implemented this kind of temporary write support for CramFS, and **Phillip Lougher** also is planning to write a similar feature for SquashFS. A lot of folks objected to Arnd's approach though, saying that this kind of thing should be accomplished by using **UnionFS** to stack something like **TmpFS** on top of CramFS. But, it turns out that UnionFS is kind of a mess and not really able to do the job well. Some folks, like Phillip, also feel that stackable filesystem support should go right in the VFS, instead of clunking around in a filesystem of its own.

The uproar against writing one-offs for each affected filesystem (CramFS, SquashFS and so forth) subsided when it became clear that CramFS and SquashFS would be the only two filesystems to do this, except for perhaps **ISO9660**. At least with a small number of filesystems, the amount of duplicated effort would not be too great. But, the people interested in making UnionFS the canonical method of accomplishing this sort of thing apparently have redoubled their efforts. So, it may be that both groups will get what they need soon enough.

Geert Uytterhoeven asked when SquashFS would be accepted into the kernel tree, and Phillip Lougher said that he actually was making some major changes, after which it would presumably take some time for the code to be reviewed by users and any remaining sizable bugs shaken out. Among the changes, Phillip is considering changing the on-disk data layout, which would solve certain problems but break backward compatibility.

The venerable **AdvFS** filesystem has been released by **Hewlett-Packard** under the terms of the GPL version 2. This filesystem has been around since the 1980s, supports journaling and file undeletion, and is very fast in general. Nowadays, there are plenty of journaling filesystems, but this release represents a solid and excellent body of code—and a really terrific body of documentation—available to the Open Source world for the first time. The AdvFS release is not, as **Linda Knippers** from HP has pointed out, a port of the filesystem to Linux. HP sees this release as providing a lot of cool stuff for open-source developers to use in any various projects where they might be useful. But, it's also likely that some kernel folks will want to do a real port. So, we probably can look forward to seeing AdvFS in the main kernel tree at some point.

—ZACK BROWN


USER FRIENDLY by J.D. "Illiad" Frazer
LINUX JOURNAL EDITION
IS THAT...SOME KIND OF ALTERNATE MUSIC?
IT'S "CHORDSTATS," A PERL SCRIPT THAT MONITORS A SYSTEM AND RETURNS MACHINE GENERATED MUSIC.
DUM DOOP DUM DE DOP
COPYRIGHT©2008 J.D. "Illiad" Frazer HTTP://WWW.USERFRIENDLY.ORG/
OH...THAT'S INTERESTING...
BOOP DE DOOP DOOP ...DUH DUM ...DUH DUM ...DUH DUM ...DUH DUM
...I THINK IT'S PLAYING THE THEME FROM "JAWS."
DUHDUM DUHDUM DUHDUM DUHDUM
PITR! QUIT TEASING THE BOTNETS!!

New Videos on LinuxJournal.com

As some of you may have noticed, there have been a number of new videos on LinuxJournal.com. If you haven't noticed, well, get over to LinuxJournal.com to learn something new in one of our new tutorial screencasts. When you are not enjoying Shawn "Gadget Guy" Powers' antics, you'll surely find something interesting and useful in our new series of videos. Visit **www.linuxjournal.com/video** to check out the selection. Perhaps you'll learn how to edit video with FFmpeg or something else to add to your bag of tricks.

As always, Shawn will continue to bring you reviews and commentary from his Gadget Guy studio, so keep watching those too. There's so much to enjoy!

—KATHERINE DRUCKMAN

They Said It

Our passion is about Linux for human beings, it's not Linux for Linux specialists, or Linux for anything other than the people who we care about.

...our goal, very simply, is to make sure the free software ecosystem can deliver a Mac OS-like experience, or an experience that will compete with the Mac OS. We see Apple as the gold standard of the user experience. We believe that, while it can be a challenge, the innovation inherent in the free software process can deliver an experience that is comparable and in many ways superior.

Proprietary drivers are a horrible kludge; they're a little bit like introducing a cast iron pot into a titanium machine. You have something that is inherently brittle and therefore reduces the value of the whole.

—Mark Shuttleworth, itmanagement.earthweb.com/osrc/article.php/12068_3757246_2

For every artificial scarcity, there's an equal and opposite artificial abundance.

—JP Rangaswami, a talk reported at Reboot10, twitter.com/dweinberger/statuses/844937650

If you provide your customer with solutions, you'll have a well-defined market. But if you provide your customer with the opportunity to create new solutions, your customer will create new markets for you.

—Bob Frankston, private e-mail message

Communications tools don't get socially interesting until they get technologically boring....a tool...has to have been around long enough that most of society is using it. It's when a technology becomes normal, then ubiquitous, and finally so pervasive as to be invisible, that the really profound changes happen....

—Clay Shirky, from his book *Here Comes Everybody*, page 105

It's All Like...What?

Google phrase searches can produce results that seem like random answers to a Rorschach test—only more amusing. Here are the top results (on a day in July 2008) for "The Internet is like...":

- "a vast uncataloged library"
- "a vagina"
- "Joey Bishop"
- "a series of tubes"
- "Microsoft"
- "alcohol in some sense"
- "a newspaper"
- "going down the Chinese road"

Closer to home, "Linux is like...":

- "switching from a car to a motorcycle"
- "a pizza"
- "ice cream—too many flavors to choose"
- "a fixed-wheel bicycle"
- "turning on your first computer and figuring out what all those weird boxes did"
- "a common cold"
- "a whole new species"
- "Ubuntu, only different"

Our hat tip for this observation goes to reader Mike Warot. (See "What They're Using?" on page 17 for more about Mike.)

—DOC SEARLS

Laundering Blog Layouts for Mobile Devices

Whenever I run across something new, cool and Web-based, I check Netcraft's "What's that site using?" to see whether the site's servers are running on Linux. You can't always tell, and the results can be misleading, but it's a good first sniff test.

That's what I did with MoFuse (**www.mofuse.com**), and it passed. "Linux Apache/2.0.63 (Unix) mod_ssl/2.0.63 OpenSSL/...", it said.

What's cool about MoFuse is that it launders the layout complexities out of blog posts and turns them into simple lists of linked headlines. It does this free if you like, or you can pay for extra services, such as advertising revenue sharing and URLs that don't have "mofuse.mobi" in them.

MoFuse isn't alone, of course. Mippin (**www.mippin.com**) has been doing similar stuff for a while, and Netcraft shows them running Linux too (Red Hat, in this case).

And Google, naturally, also has an interface that can turn any blog into "mobile.blogname.whatever". Of course, that's Linux too.

In any case, the percentage of Web surfing via mobile devices is going to skyrocket—especially after the next generation of unlocked, uncrippled Linux-based devices start hitting the market. Formatting for the hand screen will shift from exception to rule.

—DOC SEARLS

***LJ* pays $100 for tech tips we publish. Send your tip and contact information to techtips@linuxjournal.com.**

Did you know Linux Journal *maintains a mailing list where list members discuss all things Linux? Join* LJ*'s linux-list today:* http://lists2.linuxjournal.com/mailman/listinfo/linux-list.

LJ Index, October 2008

1. Number of Firefox 3.0 downloads in 24 hours starting June 17, 2008: 8,002,530
2. Position of the above download number among Guinness World Records: 1
3. Minimum millions of Firefox users: 180
4. Minimum number of countries in which Firefox is used: 230
5. Millions of dollars in VC funding for open-source vendors in 2007: 328.5
6. Millions of dollars in VC funding for open-source vendors in first half of 2008: 321.3
7. Percentage increase in VC funding in first half of 2008 over 2007: 62
8. Total billions in open-source funding through first half of 2008: 2.80
9. Age in years of a reported bug in BSD discovered in May 2008: 25
10. Linux's market share percentage of counted browser visits in January 2007: 35
11. Linux's market share percentage of counted browser visits in January 2008: 64
12. Linux's market share percentage of counted browser visits in June 2008: 80
13. Linux year-over-year percentage growth in server sales in Q1 2008: 8.4
14. Linux server revenue in billions of dollars for Q1 2008: 1.8
15. Linux-based server percentage of all server revenue: 13.7
16. Number of Linux-based laptops to be distributed to students in the Tamil Nadu state of India: 100,000
17. Price in dollars for Ubuntu Hardy Heron (8.04) in ValuSoft boxes at Best Buy and Amazon: 20
18. Number of Linux distros listed on DistroWatch's Page Hit Ranking: 100
19. Position of Ubuntu among top distros in the six months ending July 16, 2008: 1
20. Position of OpenSUSE among top distros in the six months ending July 16, 2008: 2

Sources: 1–4: SpreadFirefox.com
5–8: 451 Group | 9: TheInquirer
10–12: NetApplications.com
13–15: International Data Corp (IDC)
16: LinuxWatch | 17: DesktopLinux.com
18–20: DistroWatch.com

What They're Using
Mike Warot

Linux converts don't always come in groups. Among IT folks, they tend to come one by one. Case in point, Mike Warot, a reader I knew for insightful comments and fun ideas (for example, see "It's All Like...What?" on page 16). Mike's experience is a good sample of what must be happening in countless shops, even though each is different. Now, here's Mike's description of "what he's using":

> I'm a one-man IT staff in what used to be a Windows shop. Today I have two machines on my desk. The one I just let run is Ubuntu. It just runs, 24x7, and it's there when I need it.
>
> The change came once it became apparent that we were going to have to put our Windows servers behind a firewall. I decided to use Linux to make a firewall. I used Red Hat and EBtables to create a transparent bridge to avoid having to re-address our servers and let everything keep its public IP address. This was nice, because it still wasn't *necessary* to have a firewall, it was just an extra precaution at the time.
>
> That worked well for about a year, until I got worried I wouldn't be able to restart it. That's when a friend told me about IPcop in his work with nonprofits. IPcop is a *very* easy to configure and maintain Linux-based firewall, which does a great job in almost all circumstances. I used IPcop for a year or so, until problems with multiple Internet connections forced me to seek a better solution.
>
> At the same time I put in the IPcop box, I learned (eventually) two very important facts about Windows 2000 Servers and the Internet. First, if your domain controllers have multiple subnets from which to choose, they will get confused and not be able to find each other (meaning users might not be able to log in). And second, if you followed common sense and used your real domain name for your Windows domain name, you were FUBAR, because active directory then puts all of your local addresses in the DNS, and you can't remove them (or the domain stops working again).
>
> This required another Linux box, for DNS services to the world. It's running Fedora Core 5, but I'm about to replace it, because I don't know how to update the DNS to fix the current exploit. It'll get replaced with a VMware virtual machine running Ubuntu (more about VMware below). I now have an Ubuntu box with iptables (or ipchains, I always forget), which connects us to our three Internet connections.
>
> With the proliferation of servers, VMware's offer of the free VMware Server was a godsend. And, I've since learned that Linux makes a far better host than Windows does. So, I've got some of my VMware servers running on Ubuntu, with Windows and other things inside them.
>
> Ubuntu has made it pretty easy to set up a new machine—not that Red Hat was very hard. But with apt-get, installation of almost any program is trivial. I use an editor called joe to do editing when I don't have the X Window System set up (on some of the older servers), and it's just `sudo apt-get install joe`, and I'm up and running.
>
> I'm not a guru, but I've learned enough over the years to get around. Linux offers a wider set of tools than Windows ever could. Bottom line: Linux is amazing, and it's getting better all the time.

—DOC SEARLS

REUVEN M. LERNER

Unobtrusive JavaScript

Remove JavaScript event handlers from HTML files using Prototype and Lowpro.

JavaScript has gone through a number of changes in the past few years. Implementations have become faster, more standardized and more stable. The development and growth of open-source JavaScript libraries, such as Prototype and Dojo, has helped mask many of the remaining differences between JavaScript implementations, such as with AJAX and event handling. The final change has occurred in the minds of developers (including myself), who now approach JavaScript as a serious application development language, rather than as a toy for highlighting images or doing simple effects.

Most JavaScript is executed not when it is initially read into a browser window, but rather when a particular event takes place. The easiest, and most common, way to assign event handlers is inside the HTML itself. For example, we can create a submit button for an HTML form as follows:

```
<form method="POST" action="/action">
    <input type="submit" value="Submit the form" />
</form>
```

When a user clicks on this button, the browser submits the contents of the form to the URL specified in the form tag's action attribute, using the method specified in the method attribute. But, we can change this by adding an onclick attribute to the submit button:

```
<form method="POST" action="/action">
    <input type="submit" value="Submit the form"
           onclick="alert('hello!'); return false;" />
</form>
```

With the onclick handler in place, the button now opens a JavaScript alert box (saying "hello"). Moreover, because our event-handler definition returns false, the form will not be submitted.

Of course, we don't have to put our JavaScript literally in the event handler. We could define a function somewhere else—in the document's <head> or perhaps in an external JavaScript file altogether:

```
<form method="POST" action="/action">
    <input type="submit" value="Submit the form"
           onclick="do_something(); return false;" />
</form>
```

Now, none of this is new. But, there are problems associated with setting event handlers in "on___" attributes. For one, it becomes difficult to assign more than one handler to the same event on an object.

A second and more significant reason is that our HTML becomes full of JavaScript. Until a few years ago, it wasn't unusual for HTML to be mixed in with code and style information as well, but the growth of strictly separated MVC frameworks have removed most code from the HTML, and style information now is put in external CSS files.

A growing movement during the past few years has pushed for "unobtrusive JavaScript". Proponents of unobtrusive JavaScript argue that by placing JavaScript in a separate file and by defining event handlers in a separate file, the code becomes easier to read and understand and is cached by the browser. By using JavaScript unobtrusively, we also have the opportunity to make our HTML pages degrade gracefully, continuing to work with browsers that don't support JavaScript.

Listing 1. test.html, the Simplest Version

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
         "http://www.w3.org/TR/html4/strict.dtd">
<html>
 <head>
   <title>Unobtrusive JavaScript</title>
 </head>
 <body>
   <h1>Unobtrusive JavaScript</h1>

   <p>A paragraph of text.</p>

   <p>A <a href="http://www.nytimes.com" id="hyperlink">hyperlink</a>
   to The New York Times.</p>

   <form method="POST" action="/action">
       <input type="text" name="text_field" id="text_field" />
       <input type="submit" value="Submit the form" id="submit_button" />
   </form>

 </body>
</html>
```

This month, we look at unobtrusive JavaScript and the unobtrusive approach to defining functions and event handlers. We also examine the Lowpro library that works with the popular Prototype JavaScript library, allowing us to write JavaScript that is unobtrusive, clean and easy to read.

Observing Events

Above, I showed how to assign a piece of JavaScript (called "do_something") to an event on a particular HTML element. A fuller version of this HTML form, as well as some more content and tags, is shown in Listing 1 (test.html). This file contains a simple hyperlink, as well as our form.

I've already discussed how to handle an onclick event by setting the onclick attribute. However, there are at least two other methods for setting this event handler. One is to set the onclick attribute through JavaScript, treating onclick as a property of the DOM element associated with the hyperlink or button. Using Prototype's $() function, we can write:

```
$('hyperlink').onclick =
          function() { alert('clicked!'); return false; }
```

Notice how the event handler is an anonymous function, similar to "lambda" in Ruby and Python or an anonymous subroutine in Perl. The event-handling function can take an optional argument, whose value will be an event object. For example:

```
$('hyperlink').onclick =
          function(event) { alert(event); return false; }
```

With this alternate code in place, the alert (in Firefox, at least) indicates that the event was an "object MouseEvent". This object, like all objects in JavaScript, then has a number of properties we can query. For example, the pageX and pageY properties indicate the X and Y coordinates of the mouse cursor when the event took place. We can see these by specifying the following:

```
$('hyperlink').onclick =
          function(event) { alert(event.pageX + ", " +
                        event.pageY); return false; }
```

Each click on the link will give a slightly different result, depending on the coordinates of the mouse cursor at the time of the click.

Of course, we also can define non-anonymous functions as our event handlers:

```
function show_x_and_y(event) {
    alert(event.pageX + ", " + event.pageY); return false;
}

$('hyperlink').onclick = show_x_and_y;
```

Notice that our assignment is to show_x_and_y (that is, the name of a function), rather than to show_x_and_y() (that is, the result of executing the function). This is a particularly useful technique if we want to assign the same function to handle multiple events.

We can handle a number of different events. For example, the onmouseover and onmouseout events let us execute a function based on when the mouse

Listing 2. test-2.html, with Event Handlers Defined in <script> Tags

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
         "http://www.w3.org/TR/html4/strict.dtd">
<html>
 <head>
   <title>Unobtrusive JavaScript</title>

 </head>
 <body>
   <h1>Unobtrusive JavaScript</h1>
   <script text="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/prototype/
➥1.6.0.2/prototype.js"></script>

   <p>A paragraph of text.</p>

   <p>A <a href="http://www.nytimes.com" id="hyperlink">hyperlink</a>
     to The New York Times.</p>

   <form method="POST" action="/action" id="the_form">
     <input type="text" name="text_field" id="text_field" />
     <input type="submit" value="Submit the form" id="submit_button" />
   </form>

 </body>

 <script>
   function show_x_and_y(event) {
       alert(event.pageX + ", " + event.pageY); return false;
   }

   $('hyperlink').onclick = show_x_and_y;

   $('hyperlink').onmouseover =
               function() { $('the_form').hide(); }
   $('hyperlink').onmouseout =
               function() { $('the_form').show(); }


 </script>
</html>
</programlisting>
```

Listing 3. test-3.html, All JavaScript Removed and Placed in atf-events.js

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
         "http://www.w3.org/TR/html4/strict.dtd">
<html>
 <head>
   <title>Unobtrusive JavaScript</title>

 </head>
 <body>
   <h1>Unobtrusive JavaScript</h1>
   <script text="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/prototype/
➥1.6.0.2/prototype.js"></script>
   <script text="text/javascript" src="atf-events.js"></script>

   <p>A paragraph of text.</p>

   <p>A <a href="http://www.nytimes.com" id="hyperlink">hyperlink</a>
     to The New York Times.</p>

   <form method="POST" action="/action" id="the_form">
     <input type="text" name="text_field" id="text_field" />
     <input type="submit" value="Submit the form" id="submit_button" />
   </form>

 </body>
</html>
```

Listing 4. atf-events.js, Broken JavaScript Code for test-3.html

```
function show_x_and_y(event) {
 alert(event.pageX + ", " + event.pageY); return false;
}

$('hyperlink').onclick = show_x_and_y;
$('hyperlink').onmouseover = function() { $('the_form').hide(); }
$('hyperlink').onmouseout = function() { $('the_form').show(); }
```

Listing 5. atf-events-2.js, JavaScript Code for test-3.html

```
function set_event_handlers () {

 function show_x_and_y(event) {
   alert(event.pageX + ", " + event.pageY); return false;
 }

 $('hyperlink').observe('click', show_x_and_y);
 $('hyperlink').observe('click', function() { alert('yay!'); return false;
});

 $('hyperlink').onmouseover = function() { $('the_form').hide(); }
 $('hyperlink').onmouseout = function() { $('the_form').show(); }
}

window.onload = set_event_handlers;
```

starts or stops pointing to a DOM element. Thus, we can do the following:

```
$('hyperlink').onmouseover =
            function() { $('the_form').hide(); }
$('hyperlink').onmouseout =
            function() { $('the_form').show(); }
```

When the mouse points to the hyperlink in test-2.html (Listing 2), the HTML form disappears. When the mouse moves away from the link, the form reappears. This might not be especially useful, but it does demonstrate the sorts of events (and event handlers) we can define.

Assigning events in this way has some advantages over using the onclick and related attribute-based event handlers. It lets us define all of our event handlers in a single place—typically at the end of the HTML file. Thus, we have some separation between our HTML and JavaScript.

But, what if we want to go one step further, putting all our JavaScript into a separate file? Listing 3 shows a new version of our HTML file, now called test-3.html. Instead of having the JavaScript at the bottom of the page, I put it in a separate file, called atf-events.js (Listing 4). However, if you try to load this file, you quickly will discover that it doesn't work. We get a JavaScript error upon loading the file (clearly evident and readable if you're using the wonderful Firebug debugger for Firefox), telling us that $('hyperlink') is null.

How can this be? If you look through Listing 3, you still will see an HTML element with an ID of hyperlink. And, we definitely have included the Prototype library, so $() should work. How can it be, then, that $('hyperlink') returns null?

The answer is subtle, but well known to JavaScript programmers: $('hyperlink') is available only after the HTML element with an ID of hyperlink has been loaded. Because our JavaScript file was loaded (in the <head> of the document) before the hyperlink element was defined, JavaScript threw us an error.

One solution to this problem is to load our JavaScript at the end of the file, right before the closing </body> tag. Another possibility is to define all of our event handlers in a function that itself is executed only after the entire document is loaded. In other words, we define a function (set_event_handlers) that defines all of our event handlers. Then, we attach this function to the window.onload event, which executes only after the entire document has been loaded. The code, shown in Listing 5, is exactly the same as Listing 4, except the functionality is wrapped in the set_event_handlers function, which is invoked based on an event.

Events in Prototype and Lowpro

Our event handlers are now unobtrusive. However, there still are some problems associated with them. For example, what happens if we want to assign multiple handlers to a single event? That is, what if we want to execute not one function, but two, for $('hyperlink').onclick? In our current paradigm, we don't have any options; to have two functions execute, we need to wrap them both into a single function and then make that single wrapper function the event handlers.

This isn't much of a solution, particularly if we are loading third-party libraries that might want to attach handlers to one or more events. Instead, we need to use a different paradigm—one that lets us attach a handler to an event, rather than set the handler.

Prototype lets us do this with the observe method, which is available to any extended element—including those returned by the $() and $$() functions. So, we can say:

```
$('hyperlink').observe('click', show_x_and_y);
```

Because of the way that Prototype's observe method works, we can attach multiple handlers to a single event:

```
$('hyperlink').observe('click', show_x_and_y);
$('hyperlink').observe('click',
           function() { alert('yay!'); return false;});
```

Of course, because this code still depends on the existence of $('hyperlink'), we still need to wrap it in a function that is then attached to window.onload. (We also can attach our function to the dom:loaded event, which fires before window.onload, but the idea is the same.)

An alternative solution is to use the Lowpro JavaScript library, which provides functions that facilitate easier writing of unobtrusive JavaScript.

By loading lowpro.js (after Prototype, but before any code that will use Lowpro), we gain access to the Event.addBehavior method, which lets us attach one or more events to any CSS selector. Listing 6 is a slight rewrite of our HTML file to include lowpro.js, and Listing 7 shows how we can set our event handlers using Event.addbehavior:

```
Event.addBehavior({
     '#hyperlink:click' : show_x_and_y,
     '#hyperlink:mouseover' : function() { $( 'the_form' ).hide() },
     '#hyperlink:mouseout' : function() { $( 'the_form' ).show() }
});
```

We see that Event.addBehavior is a function that takes a single parameter, a JavaScript object (which we can think of as a hash). Each of the object's keys combines a CSS selector (#hyperlink in this case) with the name of an event, with a colon separating the two. Note that the event name does not include a leading "on". So, what would be the onmouseover handler is called mouseover for Event.addBehavior.

As you can see in Listing 7, Event.addBehavior automatically wraps our event-handler definitions in code that waits for the entire page to load. So, we no longer need to set document.onload, for example.

Listing 6. test-4.html, Using Lowpro

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
             "http://www.w3.org/TR/html4/strict.dtd">
<html>
 <head>
   <title>Unobtrusive JavaScript</title>

 </head>
 <body>
   <h1>Unobtrusive JavaScript</h1>
   <script text="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/prototype/1.6.0.2/
➥prototype.js"></script>
   <script text="text/javascript" src="lowpro.js"></script>
   <script text="text/javascript" src="atf-events-3.js"></script>

   <p>A paragraph of text.</p>

   <p>A <a href="http://www.nytimes.com" id="hyperlink">hyperlink</a>
     to The New York Times.</p>

   <form method="POST" action="/action" id="the_form">
     <input type="text" name="text_field" id="text_field" />
     <input type="submit" value="Submit the form" id="submit_button" />
   </form>

 </body>
</html>
```

Listing 7. atf-events-3.js, Using Lowpro's Event-Adding Code

```
function show_x_and_y(event) {
 alert(event.pageX + ", " + event.pageY); return false;
}

Event.addBehavior({
         '#hyperlink:click' : show_x_and_y,
         '#hyperlink:mouseover' : function() { $( 'the_form' ).hide() },
         '#hyperlink:mouseout' : function() { $( 'the_form' ).show() }
   });
```

Finally, the CSS selector code means we can set events on multiple elements simultaneously. If we want all paragraphs, or all table headers or even all images, we can do that quickly and easily with Lowpro. Lowpro allows us to reduce the amount of event-handling code that we write dramatically, keeping it in a single location and removing it from the HTML file where we might have first considered putting it.

I should add that Lowpro used to include DOM-manipulation routines as well, allowing us to add and modify page elements using a variety of convenience functions. However, recent versions of Prototype include this functionality already, allowing Lowpro to stick to behavior not addressed by Prototype.

Conclusion

Unobtrusive JavaScript is an increasingly popular style for working with JavaScript, particularly when it comes to defining event handlers. Prototype makes it easier to work with events than with raw JavaScript, but the Lowpro library makes it even easier than that. With Lowpro, it becomes quite simple to assign event handlers to any combination of elements in our document, without having to clutter up our HTML page or worry about when the page has loaded.■

Reuven M. Lerner, a longtime Web/database developer and consultant, is a PhD candidate in learning sciences at Northwestern University, studying on-line learning communities. He recently returned (with his wife and three children) to their home in Modi'in, Israel, after four years in the Chicago area.

Resources

David Flanagan's *JavaScript: The Definitive Guide* is an excellent resource for JavaScript programmers, including both a tutorial and a reference section. Douglas Crockford's recent book, *JavaScript: The Good Parts*, is much shorter, but it's also excellent and provides useful advice on which parts of JavaScript we should avoid. Both books are published by O'Reilly. My opinion (and use) of JavaScript has improved dramatically since reading Crockford's writing, letting me concentrate more on writing code and less on problems associated with the specification or implementation of JavaScript.

You can read more about Prototype at its home page, **www.prototypejs.org**. I also enjoyed the book *Prototype and Scriptaculous*, written by Christophe Porteneuve and published by the Pragmatic Programmers.

Finally, the Lowpro library is written and distributed by Dan Webb, and it's best described on his blog, **www.danwebb.net/2006/9/3/low-pro-unobtrusive-scripting-for-prototype**. And, a Google group for discussing Lowpro is at **groups.google.co.uk/group/low-pro**.

TECH TIP Downloading an Entire Web Site with wget

If you ever need to download an entire Web site, perhaps for off-line viewing, wget can do the job—for example:

```
$ wget \
     --recursive \
     --no-clobber \
     --page-requisites \
     --html-extension \
     --convert-links \
     --restrict-file-names=windows \
     --domains website.org \
     --no-parent \
         www.website.org/tutorials/html/
```

This command downloads the Web site www.website.org/tutorials/html/.

The options are:

- --recursive: download the entire Web site.
- --domains website.org: don't follow links outside website.org.
- --no-parent: don't follow links outside the directory tutorials/html/.
- --page-requisites: get all the elements that compose the page (images, CSS and so on).
- --html-extension: save files with the .html extension.
- --convert-links: convert links so that they work locally, off-line.
- --restrict-file-names=windows: modify filenames so that they will work in Windows as well.
- --no-clobber: don't overwrite any existing files (used in case the download is interrupted and resumed).

—DASHAMIR HOXHA

EmperorLinux

...where Linux & laptops converge

Tablet Linux: The Raven

- Based on the ThinkPad X61 tablet by Lenovo
- Full tablet feature support in Linux:
  - Active digitizer pen with pressure sensitivity
  - Handwriting recognition
  - Dynamic screen rotation
- ThinkPad X series laptops are also available

Features include:

- 1.6 GHz Core 2 Duo
- 12.1" SXGA+ LCD w/ X@1400x1050
- 120-200 GB @ 7200 rpm or 64 GB SSD, up to 4 GB RAM
- DVD±RW (UltraBase), Ethernet, 802.11a/b/g, Bluetooth
- One year Linux tech support - phone and email
- Three year manufacturer's on-site warranty
- Choice of pre-installed Linux distribution:

Powerful Linux

Rhino D830/M6300

- Dell Latitude D830/Precision M6300
- Up to 17" WUXGA w/ X@1920x1200
- NVidia Quadro FX 3600M graphics
- 1.8-2.8 GHz Core 2 Duo/Extreme
- Up to 8 GB RAM
- 60-320 GB @ 7200 rpm / 64 GB SSD
- DVD±RW or Blu-ray
- Starts at $1350

Tablet Linux

Piranha P1620

- Fujitsu LifeBook P1620
- 8.9" WXGA TouchScreen w/ X@1280x768
- 1.2 GHz Core 2 Duo
- Up to 2 GB RAM
- 80 GB hard drive / 32 GB SSD
- Only 2.5 pounds
- Starts at $2650

Rugged Linux

Tarantula CF-30

- Panasonic Toughbook CF-30
- Fully rugged MIL-SPEC-810F tested: drops, dust, moisture, & more
- 13.3" XGA TouchScreen
- 1.6 GHz Core 2 Duo
- Up to 4 GB RAM
- 80-320 GB hard drive
- Call for quote

www.EmperorLinux.com 1-888-651-6686

Model prices, specifications, and availability may vary. All trademarks are the property of their respective owners.

MARCEL GAGNÉ

Imaginary Languages

Programming languages aren't languages per se. You can't go out into the street and say, "print 'Hello, world!\n';" to passersby without raising the occasional curious eyebrow. In the right crowd, however, that might work as well as boldly proclaiming "naDevvo' yIghos" in a roomful of Klingons. It's all in how you say it and to whom.

François! Our guests will be here any moment, and you are sitting back watching old *Star Trek* episodes! Yes, I know that the Klingons in the original series look different from how they do now, but why are you watching television when there is work to be done? As much as I enjoy the original series....*Quoi?* Yes, of course, it is silly to show every alien race as speaking English. It happens in most science fiction and in most fiction, for that matter. Having the characters on television spend weeks or months learning the basics of a language before getting down to action would make for pretty boring television. And, people constantly would have to invent new languages for us to hear. Yes, I know many people speak Klingon, but that's different. We will discuss this later, François. I can see our guests at the door.

Welcome, *mes amis*, to *Chez Marcel*. Your tables await, and François was just turning off the television, weren't you *mon ami*? Please, sit and make yourselves comfortable while my faithful waiter fetches the wine. There's a case of 2006 Cuvée Bacchus Pfaffenheim Gewurztraminer in the cellar's north wing. You remember, François—right next to the ancient Egyptian hieroglyphics tablets.

The world has numerous languages. Some languages are spoken by millions. English, for example, is perhaps the most popular language, spoken by countless billions throughout the galaxy—a little joke, *mes amis*. Other languages are more obscure, and some, although still spoken, are nearly forgotten or relegated to the pages of history. Then, there are the imaginary languages, which, amazingly, sometimes have a great many speakers—like Pig Latin or Klingon. Originally only a few words invented by Marc Okrand for the 1979 movie *Star Trek: The Motion Picture*, Klingon now boasts tens of thousands of speakers worldwide. The language is championed by the Klingon Language Institute.

Ah, François, you have returned. Please, pour for our guests. Enjoy, *mes amis*. This Gewurztraminer is a full-bodied white with deep fruit flavors, a nice floral bouquet and just a hint of spiciness. Don't forget my glass, François.

Where was I? Ah...Klingon is unique in that it originally was an imaginary language. Google, as we all know, runs thousands of Linux servers to do its magic, and in performing those searches, it tries to support as many languages as possible—including Klingon. For a Klingonese Google start page (Figure 1), visit **www.google.com/intl/xx-klingon**.

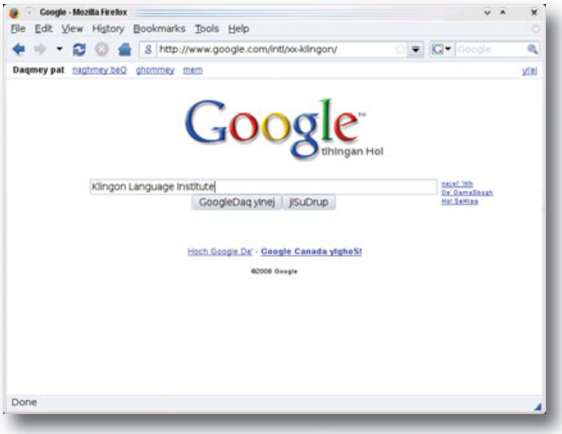
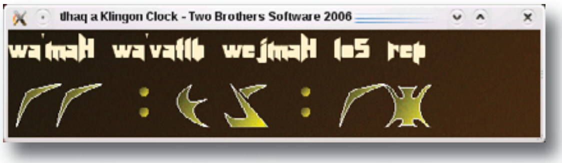


Figure 1. Google offers a start page for those who are fluent in Klingon.

If you happen to be among those who speak Klingon, you might want to check out tlhaq, the Klingon clock, at Two Brothers Software (Figure 2). This graphical clock displays the time textually and numerically, using, of course, Klingon characters. Even if you can't speak the language, it's a pretty cool addition to the desktop, worthy of a warrior. Visit **tlhaq.twobrotherssoftware.com** and download the package.



tlhaq a Klingon Clock - Two Brothers Software 2006
wa'maH wa'vatlh wejmaH loS rep

Figure 2. tlhaq, the Klingon clock—I'm not even going to pretend I know what time it is.

This is a binary distribution, so there's no compilation required. You need the SDL_image and SDL_ttf packages installed on your system, however. Create a directory called tlhaq (or kclock, if you prefer), and extract the tarred and gzipped bundle:

```
mkdir kclock
cd kclock
tar -xzvf kclock.tar.gz
```

From that directory, you simply can run the `tlhaq` binary. When the clock is running, you can use some single keystrokes to modify its behavior. For instance, the S key toggles the text for the seconds on and off. Pressing + or - increases or decreases the font size. Should you find the text difficult to read, press F to switch between different font types (it still will be in Klingon though). If you want tlhaq to remember your settings, press C to save them.

I alluded to the idea that programming languages are, if not imaginary, at least artificial languages. Combining Klingon with programming hasn't escaped some Klingon enthusiasts. For instance, Unicode support for the Klingon language was added to the Linux kernel. Seriously. If you don't already have the Linux kernel source installed on your system, do so, and then check out the included file called /usr/src/linux/Documentation/unicode.txt.

Given this Linux Klingon support, there's var'aq, the Klingon programming language created by Brian Connors. He describes var'aq as "PostScript with a dash of Lisp thrown in". Hmm...one wonders if Brian has ever worked in a kitchen. A programming language for warriors, var'aq can be downloaded from **www.geocities.com/connorbd/varaq**. Once you have it, extract the programming source. This is a zip file, which you can extract in the following way:

```
unzip varaq-current.zip
```

Once done, change to the varaq directory, and start the interpreter:

```
perl varaq-engl
```

You also can start varaq in the native Klingon language version if you happen to speak Klingon:

```
perl varaq-kling
```

var'aq is an unforgiving language. As if you would expect otherwise. There is no prompt to provide hints, but var'aq has no trouble telling you when you are wrong. Documentation is provided, so you can start writing your own operating system should you so desire. You might want to start with the classic "Hello world" program, keeping in mind that Klingons don't say hello:

```
"nuqneH 'u'?" cha' pong
```

The English version might make a little more sense:

```
"What do you want, Universe?" disp name
```

All programming languages, some might argue, are essentially fancy adding machines. On that note, you can use var'aq to discover the nature of the universe:

```
52 10 boqHa' cha'
```

This most certainly looks like the right place to take a break and have François refill our glasses. While you enjoy that lovely wine, let me direct you to another imaginary language—one of the very first learned by English-speaking children. *Yesay, itway'say igpay atinlay*. Pig Latin, as most kids will tell you, is as simple as taking the initial consonant, putting it at the end of the word, and adding ay to it. In that way, Linux becomes Inuxlay and Cooking becomes Ookingcay. Easy, huh? If you want to write large amounts of text in Pig Latin, it can, however, become fairly tedious. Luckily, there's a great little program to help do the job. It's called pig, and it comes with the classic bsdgames package (or bsd-games, in some distributions). To translate large phrases into Pig Latin, simply type `pig` at the command line, then type the phrase you want translated:

```
$ pig
Linux is the world's greatest operating system.
Inuxlay isway ethay orldway'say eatestgray
 operatingway ystemsay.
```

On a similar note, that same package comes with another tool for speaking in tongues. It's called rot13, and this is how you use it:

```
$ rot13
Linux is cool redefined.
Yvahk vf pbby erqrsvarq.
```

As it turns out, rot13 is actually a cypher, albeit a very simple one. It takes the letters of a word, such as linux, and changes it by moving 13 letters forward (or backward) in the alphabet. In that way, linux becomes yvahk. To translate from rot13, simply re-enter the encrypted phrase using the same command. If you break the alphabet up into two rows of 13 letters, it's extremely easy to see how rot13 works:

```
abcdefghijklm
nopqrstuvwxyz
```

Also in that bsd-games package is a translator for the computer world's most powerful language, acronyms. The program, wtf (which no doubt stands for "What's that frase?") can help you decipher those strange words you find scattered on IRC and inside instant-message conversations. For example:

```
$ wtf rotfl
ROTFL: rolling on the floor laughing
```

Cool, *non*? Although, it's arguably nowhere near as cool as the greatest imaginary language of all time. I'm talking about mock-Swedish.

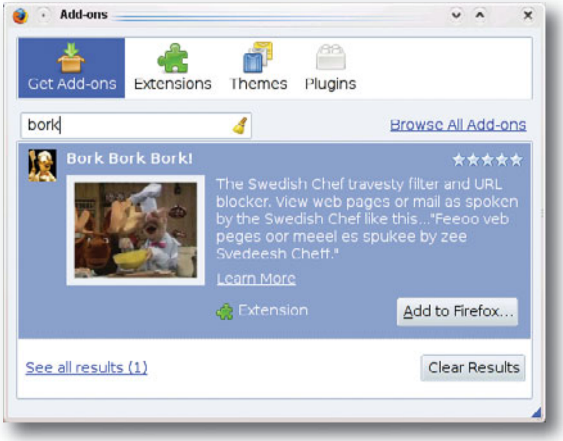
The greatest (and funniest) television Chef of all time is without a doubt the Swedish Chef from the legendary Jim Henson's brilliant *Muppet Show*. If you never have watched the *Muppet Show*, please, do yourself a huge favor and buy yourself the DVD boxed set. You won't regret it. But, as I have been known to do from time to time, I digress...the Swedish Chef was funny largely in part because of his rather strange form of mock-Swedish—that and his hilarious antics and over-the-top recipes. Nevertheless, many thousands of people have, over the years, attempted to duplicate the language of the Swedish Chef. Now, your Firefox browser can bring the experience to any page you visit.

Firefox is an excellent browser on many counts, but one of its coolest features is the ability to add features and capabilities through a system of add-ons and extensions. Extensions are program enhancements that can change how you work with your browser dramatically. This framework of extensions makes Firefox not only a great browser, but also a superior browser. To experience Firefox extensions, click Tools on the menu bar and select Add-ons.

A window labeled Add-ons appears with a list of buttons to access installed extensions, themes and plugins already on your system. On a fresh install, there usually isn't much here. Most likely, the Get Add-ons button will be highlighted with a selection of recommended add-ons listed in the larger pane below. You can choose to add the recommended extensions, browse through a rather huge list of other extensions, or type something in the search bar to narrow the search. What does all this have to do with the Swedish Chef? Let me tell you.

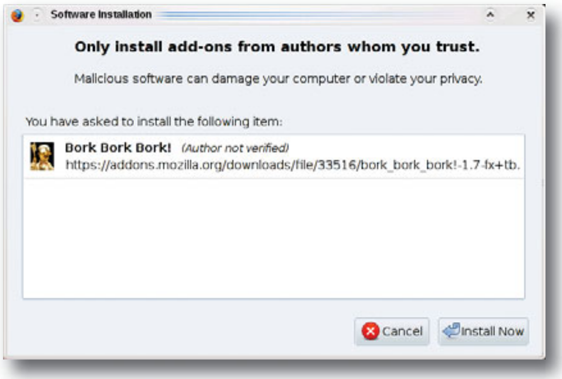
Type `bork` in the search box, and press Enter. The dialog should display Anthony Howe's "Bork Bork Bork!" extension along with a description (Figure 3). There's also a nice, friendly button labeled Add to Firefox.

When you click that button, a new window appears asking for confirmation before going ahead and installing the extension (Figure 4). There's also a warning about installing malicious software. If you are comfortable with your choice, click the Install Now button.



The Swedish Chef travesty filter and URL blocker. View web pages or mail as spoken by the Swedish Chef like this..."Feeoo veb peges oor meeel es spukee by zee Svedeesh Cheff."

Figure 3. The Firefox Add-ons dialog not only gives you access to installed extensions, but it also lets you search for many others.



Only install add-ons from authors whom you trust.

Malicious software can damage your computer or violate your privacy.

You have asked to install the following item:

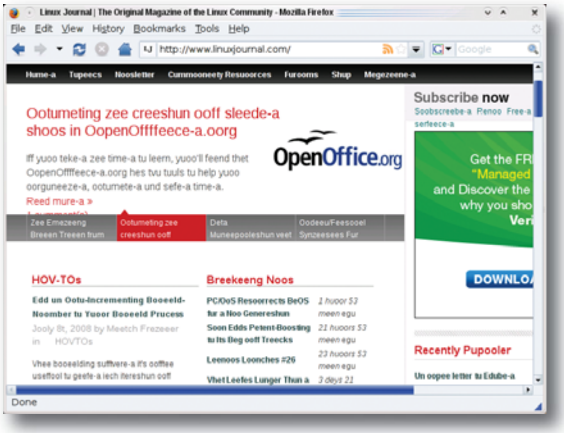
Bork Bork Bork! *(Author not verified)*
https://addons.mozilla.org/downloads/file/33516/bork_bork_bork!-1.7-fx+tb.

Figure 4. Do you know where that extension has been? Even so, installing it is merely a click away.

That's it. Once the extension is installed, you'll see a message appear above the description telling you to restart Firefox before the extension actually can take effect. Firefox also provides a handy Restart Firefox button. Once the browser comes back to life, the Add-ons dialog appears once more to confirm that you have indeed installed the "Bork Bork Bork!" extension. Now, when I surf to a Web site, such as **www.linuxjournal.com**, I can right-click on the page and select View Bork Text from the menu. In a few seconds, my page is translated into something only the Swedish Chef could understand (Figure 5).

Another way to bork the text is to click View on the menu bar and select Bork text. To return your browser to a normal view, simply repeat the process and uncheck Bork text.

Sadly, the clock tells us that our time for speaking in various strange tongues is nearly over. Yes, *mes amis*, it's closing time. If you need to have François refill your glasses, which he will do happily,


Linux Journal | The Original Magazine of the Linux Community - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.linuxjournal.com/
Google
Hume-a Tupeecs Noosletter Cummooneety Resuoorces Furooms Shup Megezeene-a
Done

Ootumeting zee creeshun ooff sleede-a shoos in OopenOffffeece-a.oorg

Iff yuoo teke-a zee time-a tu leern, yuoo'll feend thet OopenOffffeece-a.oorg hes tvu tuuls tu help yuoo oorguneeze-a, ootumete-a und sefe-a time-a.

Reed mure-a »

Zee Emezeeng Breeen Treeen frum | Ootumeting zee creeshun ooff | Deta Muneepooleshun veet | OodeeuFeesooel Synzeesees Fur

HOV-TOs

Edd un Ootu-Incrementing Booeeld-Noomber tu Yuoor Booeeld Prucess

Jooly 8t, 2008 by Meetch Frezeeer in HOVTOs

Vhee booeelding suffvere-a it's ooffee useffool tu geefe-a iech itereshun ooff

Breekeeng Noos

PC/OoS Resoorrects BeOS fur a Noo Genereshun — 1 huoor 53 meen egu

Soon Edds Petent-Boosting tu Its Beg ooff Treecks — 21 huoors 53 meen egu

Leenoos Loonches #26 — 23 huoors 53 meen egu

Vhet Leefes Lunger Thun a — 3 days 21

Subscribe now

Soobscreebe-a Renoo Free-a serfeece-a

Get the FR
"Managed
and Discover the
why you sho
Veri

DOWNLOA

Recently Pupooler

Un oopee letter tu Edube-a

Figure 5. Once borked, Web text becomes something only the Swedish Chef could understand.

remember that he only understands English and French. Asking for a refill in Klingon is likely to frighten him. Please, *mes amis*, raise your glasses, and let us all drink to one another's health. *A votre santé! Bon appétit!*■

Marcel Gagné is an award-winning writer living in Waterloo, Ontario. He is the author of the *Moving to Linux* series of books from Addison-Wesley. Marcel is also a pilot, a past Top-40 disc jockey, writes science fiction and fantasy, and folds a mean Origami T-Rex. He can be reached via e-mail at marcel@marcelgagne.com. You can discover lots of other things (including great Wine links) from his Web sites at www.marcelgagne.com and www.cookingwithlinux.com.

Resources

BSD Games (check your distribution's repositories)

Klingon Clock: **tlhaq.twobrotherssoftware.com**

Klingon Language Institute: **www.kli.org**

var'aq, a programming language for warriors: **www.geocities.com/connorbd/varaq**

Marcel's Web Site: **www.marcelgagne.com**

Cooking with Linux: **www.cookingwithlinux.com**

DAVE TAYLOR

Movie Trivia—Finally!

Use the shell to generate movie trivia from a movie database.

It's been one of those proverbial journeys of a thousand steps, but I think we're finally ready to start generating some movie trivia after spending the past few months doing all the underlying tool development. You'll recall that we're grabbing the top 250 movies list from Amazon's IMDb site, then getting the release year of each movie and storing it in a database.

Separately, we chewed on the interesting problem of coming up with adjacent years for a given year in time, recognizing that the older the movie, the more of a spread we want between years, because precious few people will guess incorrectly that a movie released in 2007 was released in 1983, but a movie released in 1947 could stymie people who might think it came out in 1931.

Now, it's time to put the pieces together.

Two Random Years

The last column dug in to the year spread, ending with a script that produced a likely adjacent year for a given year. We need to refine this script, because what we want to produce are three different year possibilities, two that are likely but wrong and one that's the correct year, without duplicates.

First, let's make the code that generates a reasonable adjacent year a script function:

```
get_random()
{
  delta="$(( $RANDOM % $factor + 1))"
  add="$(( $RANDOM % 2 ))"
  if [ $add -eq 1 ] ; then
    closeyear="$(( $releasedate + $delta ))"
  else
    closeyear="$(( $releasedate - $delta ))"
  fi
  if [ $closeyear -gt $thisyear ] ; then
    closeyear="$(( $releasedate - $delta ))"
  fi
}
```

Next, given that we can't gracefully return a value short of using a global variable, here's how we can leverage the function:

```
get_random
match1=$closeyear
```

That gets us the first year guess, easily enough. But, the next guess needs to be different from the first. How to do that? In a while loop:

```
match2=$match1        # needs an initial value
while [ $match2 -eq $match1 ] ; do
  get_random
  match2=$closeyear
done
```

This is slightly risky, because there is the possibility of an infinite loop if the code never finds a random year value that differs, but I'll ignore that for now.

Now we have three year values: two incorrect ones, $match1 and $match2, and the correct value, $releasedate. How to give them back to the calling routine sorted? Easy:

```
echo "$match1 $match2 $releasedate" | sort -n
```

And, that's the function. Give it a year, and it'll return three: two that are close but wrong, and one that's correct. For example:

```
$ ./year-delta.sh 1975
1981 1971 1975
$ ./year-delta.sh 1999
2000 1998 1999
$ ./year-delta.sh 1938
1948 1935 1938
```

That's exactly what we want. Now, how to integrate this into the bigger script that grabs a random line from the IMDb database and then presents it in a workable fashion?

Extracting Data and Displaying It

Once you remember the trick of `$(( $RANDOM % some-value))`, it should be straightforward to get a random line from a data file:

```
lines="$(wc -l < $filmdb | sed 's/ //g')"

randline=$(( $RANDOM % $lines + 1 ))

match="$(sed -n "${randline}p" < $filmdb)"
```

As I've written about before, wc is one of your best friends in script writing, because it's easy. But, it's also frustrating that there's no way to turn off the superfluous white space it generates. That's why the first line includes a call to sed to axe any spaces that are added. Somewhere, in a parallel universe to our own, there's an -n flag to wc that says "no padding" and makes this forevermore unnecessary. Sadly, we aren't in that universe, so just about every time you use wc, you have to strip out the white space at the same time.

The result of these three lines is that match has a value similar to:

```
The Lord of the Rings: The Two Towers|2002
```

Now we need to split it into two fields, which is easily, if tediously, done:

```
title="$(echo $match | cut -d\| -f1)"

relyear="$(echo $match | cut -d\| -f2)"
```

And, finally, it's time to invoke the random years function that will, if you recall, generate one correct and two incorrect years:

```
years=$($randomyears $relyear)
```

Finally, let's pull the three years into separate variables and then output an attractive trivia query:

```
year1="$(echo $years | cut -d\  -f1)"

year2="$(echo $years | cut -d\  -f2)"

year3="$(echo $years | cut -d\  -f3)"

echo "IMDb Top 250 Movie #$randline: Was $title
released in $year1, $year2 or $year3?"
```

Not too shabby! Let's see how it works:

```
$ ./generate-trivia-question.sh

IMDb Top 250 Movie #82: Was "Some Like It Hot"
released in 1950, 1959 or 1963?

$ ./generate-trivia-question.sh

IMDb Top 250 Movie #118: Was "Mononoke-hime"
released in 1994, 1995 or 1997?

$ ./generate-trivia-question.sh

IMDb Top 250 Movie #250: Was "Planet of the Apes"
released in 1967, 1968 or 1969?
```

Perfect, perfect!

That's about all we have space for in this column, but we've come a long, long way from the URL for a Web page that lists some top movies to a nice little trivia engine that's fast and fun!

Next month, we'll look at how to inject the trivia into the Twitterstream. Want to see it in action? By the time you read this column, it'll be live at **twitter.com/FilmBuzz** (along with movie commentary and much more).■

Dave Taylor is a 26-year veteran of UNIX, creator of The Elm Mail System, and most recently author of both the best-selling *Wicked Cool Shell Scripts* and *Teach Yourself Unix in 24 Hours*, among his 16 technical books. His main Web site is at www.intuitive.com, and he also offers up tech support at AskDaveTaylor.com. Follow him on Twitter if you'd like: twitter.com/DaveTaylor.

Interview with Marcus Meissner

MICK BAUER

Insights from SUSE's Security Team Lead.

Recently, after receiving one of SUSE's regular security-update e-mail messages, it occurred to me that people who track security vulnerabilities for Linux distributions really are in the front lines of Linux security. I have the luxury of examining one command here, one architecture there, but these people track threats and vulnerabilities for *entire operating systems*.

And, for a moment, I was fearful. What if this occurs to my editors, who might reach the obvious conclusion that the Paranoid Penguin really ought to be authored by a real expert, like SUSE Security Team Lead Marcus Meissner, rather than this goofball Mick?

But, of course, if they fired me, I'd be forced to release those photos I took of them back in 2003 on the Linux Lunacy cruise, and nobody wants that to happen. Think of the disillusionment and embarrassment, not to say indigestion, this might cause! I, for one, do not wish to be party to such a vile media circus. (Unless that party features the TCP/IP Drinking Game, like at Def Con—good times! Then again, that's the kind of thinking that landed my editors in this pickle.)

So, you're stuck with me for the foreseeable future, gentle reader. But, take heart! This month you get both me and wisdom: Marcus Meissner graciously agreed to an interview, in which we talked about practically every major topic in Linux security I could imagine.

As you can see, Marcus has useful and interesting insights on all of these topics. Read, learn and enjoy.

Introduction

MB: Thanks so much for taking the time to chat with *Linux Journal*! By way of introducing you to our readers, I'd like to start with a question or two about your background.
MM: Thank you for interviewing me!

MB: Could you describe your duties at SUSE as Security Team Lead?
MM: My duties as Security Team Lead are mostly overseeing security-related work done for the SUSE Linux/OpenSUSE product lines. My team has three people and one trainee currently, so there still is time left for myself to be involved in the groundwork.

What my team does can be split into two parts: reactive work and proactive work.

Reactive work is what you know of as coordinating security updates for security incidents for the products currently under support. This involves reading mailing lists, watching the bug tracker, watching the internal update release processes and giving answers to packagers, QA and others wanting to know about the updates.

Marcus Meissner

Proactive work consists mostly of reviewing security-related changes for future products, adding new security features for those and, of course, actively reviewing and auditing source code for flaws and fixing them.

My typical day is mostly lots of communication spread over the whole day: reading and answering e-mail, instant messages, phone calls and, of course, meetings. All this is done mostly to get the above-listed security tasks taken care of, so there are not specific phases of the day. Occasionally, I still do some programming and handling of the packages I maintain in the distribution.

MB: You started out at SUSE on its PowerPC Maintenance Team, right? Why did you go to the Security Team?
MM: Historically, I had been working in the security field already in my last year (2001) at Caldera. After coming to SUSE, there was no need for another security engineer there, so I had different tasks.

I started out doing various non-PowerPC-related development my first year at SUSE, but gravitated in the end to the PowerPC Development Team.

In 2004, however, I wanted a bit more responsibility, and then SUSE started looking for a lead for the Security Team. So, the offer of interesting work, more responsibility and my first people management job got me into it in the end.

MB: Do you have any formal training in information security? (This is sort of a trick question: many of my most accomplished colleagues are mostly self-taught.)
MM: Portions of it I got from university, such as basics on cryptography, protection models like Bell-LaPadula and information leaks via side channels.

But, most of my information security training was on the job and by self-education, because it was not very high on the list of must-haves in the universities in the middle of the 1990s.

MB: How has your background in software development informed your work as a security professional?
MM: Lots of the low-level work of security in the Linux distribution area is actually bug fixing and helping others design and write good code to avoid bugs. So, yes, my background in software development was a must for this job.

Secure Coding

MB: In my own day job at the bank, I frequently interact with software developers who have very little understanding of security or of secure coding. Although ignorant is the last word that I, being a humble nonprogrammer, would use in this context, it seems that security has not traditionally been an important part of programming culture and training. Have you found that to be the case yourself? Has this situation changed in recent years?
MM: I have found this to be the case, yes. In former times, it was mostly "get things working". Now, it is a bit better. Everyone knows about viruses, worms, trojans and so forth. However, it is not optimal yet; the hard part is convincing people that their code might be exposed to such problems.

I am regularly hearing, "but then it will just be broken into the user account, not root" or "it will run behind the firewall here" as excuses. But, I see definite improvements.

MB: We seem to be stuck in an endless parade of security vulnerabilities in the full range of applications on which we depend—graphics editors, compression libraries, server dæmons, even security tools. I think some of us assumed that over time, as security awareness in the development and user communities increased, the rate of security bugs would decrease. Why, do you think, it hasn't? Or, is this rate, in fact, decreasing?
MM: This has several factors at play.

First, the classes of problems have changed due to security research over time.

The trivial exploit technologies, like stack and heap overflows and format string problems, are explored and can be captured by technology today, and programmers understand them.

But, integer overflows still are an uncharted sea for most—"Why is there a difference between 'signed' and 'unsigned'? Why is this multiplication bad?"—and are hard to find using today's compiler technology.

The new kind of applications on the Web has brought entirely new problems for things like SQL injection, command injection, cross-site scripting and others. Due to the startup boom, lots of those applications were not designed with security in mind. Here, the easy Web development languages are a bit at fault too—on one hand, allowing beginners to develop applications easily, but on the other hand, making it trivial to make mistakes.

Also, the increased integration of the network and Internet has brought more code up to the light, as in "can now be accessed from the network". Just think of all the image libraries written with a command-line local user in mind that now run as a back end to Web galleries.

I personally see no reduction of security problems in the near future. At most, it will level off.

MB: It might seem non-intuitive for developers of non-networked desktop applications, like gPhoto, to have to worry about security. But, of course, such applications frequently are used as stepping-stones in attacks against other processes and for gaining access to system data or other local resources. Do you ever find yourself lecturing other developers on threat scenarios like that?
MM: Hmm, not for gPhoto or Wine at least, but for other distribution-related packages.

Application Architecture

MB: What changes, especially improvements, have you seen lately in application architecture? What changes would you like to see?
MM: I see different languages being used, which bring different architecture concepts. The trend away from C toward Java, C#, Ruby and so on with stricter typing is helpful.

I would like to see people not re-inventing/reprogramming everything existing anew, but trying more to reuse existing, proven code.

MB: How do you think the race to Web-oriented application architectures has affected application and operating system security?
MM: It makes securing the process more difficult. As an OS vendor, we now are just one part of the chain—the secure base system used to browse the Web.

MB: Is it just me, or has the emergence of cross-site scripting (XSS) added a whole new dimension to the security landscape? I don't remember many pre-Web attack vectors that allowed you to use one person's server to attack other users. The older paradigm was that you attacked the system, not the users per se.
MM: Definitely, the advent of Web applications has brought new problems, and seeing that there still are dozens of XSS issues fixed per week shows that the problems are not contained yet.

And yes, when interacting between the network, the user is way more interactively involved than before. Consider phishing problems, for instance, which even though they fall under social engineering, do endanger system security.

MB: How has the emergence of Web services affected the SUSE Security Team's work? Is it easier to isolate and address bugs in Web applications, which are usually written in scripting languages, than in compiled applications?
MM: Fortunately, not much, as we do not include very many. Quite a lot of fixes went into the PHP core packages in the last few years, and the Web applications we include had their share of problems.

What we also shipped is an increasing rate of Web browser security fixes, as the browsers now get good reviews too. More than a hundred Mozilla bugs were fixed in the last few years, for instance.

As for isolating and addressing bugs, it's still at the same difficulty level. You need to understand the code and fix the problem. Also, as we usually back-port patches, we need to understand those, and there the language does not matter much. Plus, Web applications usually are not smaller than regular programs.

Operating System Architecture

MB: In the Linux world, we've seen less malware (viruses, trojans and worms) than the Microsoft Windows world has been subject to. Why is this, do you think?
MM: First, Microsoft Windows just has more installations, and so it is a more valuable target; thus, it gets more research into exploitability.

Second, Windows has quite a high integration level. You can do lots of stuff from everything, and this was seen as good thing—easy embedding of document/image viewing and so on. Although on the one hand, this is a good thing, it also exposes a lot more code to the attackers.

Plus, the Windows software development community before the Internet was not really programming with security in mind, and so there were large holes. The same goes for reviewing the code; it was hard without source for externals.

It's something like a mix of all those things, I guess.

MB: My own opinion for several years has been that Linux isn't inherently more or less secure than Windows; their underlying security models are very similar. What are your thoughts on this?
MM: UNIX/Linux has, for example, the advantage that we separated (the concept of) the user from the administrator right from beginning, which Windows still has problems with.

Due to less integration, or integration at different levels, Linux has perhaps a better chance of resisting those attacks.

Linux also has less of a monoculture in programs and libraries, and it is also more rapidly changing than perhaps on Windows.

MB: What kind of potential do you see in mandatory access control (MAC) systems, like AppArmor and SELinux, in improving Linux security for the masses? To what extent do you think they're already helping?
MM: It's difficult to say. I have no experience with SELinux, but with AppArmor, I see a bit of acceptance issues in default settings, and then it does not catch everything.

MB: When SUSE incorporated Novell AppArmor into its general releases, this caused a bit of controversy. It seemed like some people involved with SELinux felt that this undermined their efforts. As a SUSE employee, I assume you're pro AppArmor, but what do you think about the controversy? Isn't it healthier for multiple MAC options to be available to people?
MM: There surely was controversy, but most of it seems to have died down now.

It is healthier to have more than one MAC system, especially in exploring the MAC problem from different angles.

That AppArmor was much more usable than SELinux also has caused lots of thinking and usability improvements in SELinux (think targeted policies,

booleans and so on), and the other way around. AppArmor now can contain more things than in earlier times. We currently see both as solutions that even could co-exist to some degree.

Other new MAC approaches, like SMACK and so forth, also are appearing now.

MB: So, are there any plans for SUSE to support SELinux, as an alternative to AppArmor?
MM: I cannot say at this time, especially since partner requirements are still open for future products.

Virtualization

MB: When Linux virtualization first started to emerge into the mainstream a few years ago, it seemed to me that the whole concept of a hypervisor—an intelligence logically above the guest-OS kernel that manages system resources and monitors VM behavior—has a lot of security potential. Nowadays, I wonder whether I wasn't overly optimistic. The additional layer of abstraction might introduce other attack vectors. Your thoughts?
MM: Virtualization environments, unfortunately, were/are sold as security solutions, but the breakout possibilities are only now being investigated, and there likely was no formal containment design from the ground up.

Several ways also have been found for almost all virtualization technologies to break out of confinement.

So yes, I think its being used as security containers is overly optimistic.

Embedded Linux

MB: One of the most remarkable developments in Linux, it seems to me, is its rapid inroads in the embedded systems market. All kinds of consumer electronic devices are now Linux-powered. Does SUSE ever show up in this space? Do the particular challenges and ramifications of embedded operations figure into your team's work?

And, from a security perspective, how good of an idea is it to use a general-purpose operating system like Linux (or Windows) for embedded applications?
MM: We are not really showing in this space, even though we are working to bring the enterprise desktop more into the thin-client space. But, it's not the real embedded market.

What matters most for security in those devices is how they get updates and what security processes are there from their vendors. If the vendor just gives up support after six months for a device, but the device lives for five years or longer, it's bad. You have lots of unpatched devices out there.

Cryptography and Identity Management

MB: Your team, of course, digitally signs its communications. But 17 years after Phil Zimmerman gave us PGP, only a tiny percentage of ordinary users employ any kind of e-mail encryption. Any thoughts as to why, and what to do about it?
MM: It's too hard to use and, more so, too hard even to understand why to use cryptography. "Why does Aunt Emily need to encrypt letters to her niece Tina? Who cares about them anyway? And, how do I do it?"

MB: Maybe the real issue here is identity management. We haven't yet figured out any kind of universal identification on the Internet, which is part of the problem space that PGP, S/MIME, x.509 and LDAP are all supposed to address. But the paradox is that although such an identification infrastructure would greatly simplify all sorts of security problems—single sign-on, directory services, encryption and the like—the technology itself is very complicated.
MM: Yes, definitely. Perhaps a hardware solution might help here—something that Aunt Emily and niece Tina could physically exchange and so would physically grasp.

One could imagine doing premade USB tokens that can be torn off a strip and distributed for every family member involved, in a size that fits in regular letters. Or, using cell phones to pass encryption keys back and forth, as everyone owns cell phones now.

MB: Any time you talk about centralized identity management in the US, for which the logical starting point is the federal government, the discussion gets very strange very quickly. Americans are reluctant to trust their government not to abuse this information (which is perhaps strange given that they've got all sorts of information about us already). Are things different in Europe?
MM: They are better. People trust the government more, because they already hand out our passports and ID cards. But, with the current government trying to enter into our privacy more and more, I think even in Germany we will see more mistrust.

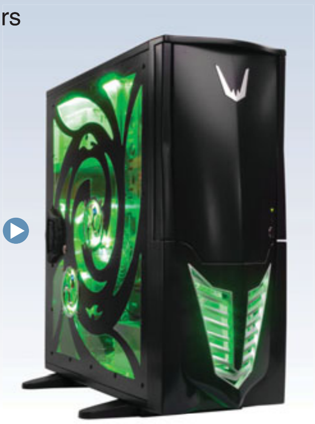
Conclusion

MB: We've amply filled this month's allotted space with a very wide-ranging discussion indeed. Thanks so much, Marcus, for a thoughtful and fun conversation!■

Mick Bauer (darth.elmo@wiremonkeys.org) is Network Security Architect for one of the US's largest banks. He is the author of the O'Reilly book *Linux Server Security*, 2nd edition (formerly called *Building Secure Servers With Linux*), an occasional presenter at information security conferences and composer of the "Network Engineering Polka".

Resources

Novell's SUSE Security Team: **www.novell.com/linux/security/team.html**



KYLE RANKIN

Wii Will Rock Linux

Why should your Wii have all the fun? Find out how to connect all those *Rock Band* instruments to your Linux machine and use them with a number of different audio programs.

In my August 2008 column, I wrote about how to use a Wiimote from a Nintendo Wii on a Linux system as a general-purpose wireless joystick. In that column, I covered how to bind use buttons not only on the Wiimote, but also on the Nunchuck and Classic Controller, so that you could use them with a number of different video game emulators. Well, since that column, *Rock Band* for the Wii was released, and with it three extra peripherals: a wireless guitar, a microphone and a drum set.

Everyone knows that only old people play real guitars, so I couldn't pass up the opportunity to rock out with an entire band of plastic instruments on my Wii. I hadn't read too much beforehand about the instruments that came with Wii's *Rock Band*, so when I unpacked everything, I was surprised to note that all three instruments were connected to the Wii via USB. That left me with only one question, do these instruments work in Linux?

It turns out that not only do all three *Rock Band* instruments work in Linux, they also all work with very little extra effort. In this column, I describe how to configure Linux to see these instruments and highlight some applications you can use them with.

The Microphone

Probably the simplest instrument to get working with Linux was the microphone. The moment I plugged it in, I got dmesg output that identified it:

```
[  188.006918] usb 1-1: new full speed USB device
     using uhci_hcd and address 2
[  188.132102] usb 1-1: configuration #1 chosen
     from 1 choice
[  188.474088] usbcore: registered new interface
     driver snd-usb-audio
```

I then fired up Audacity, one of my favorite sound recording programs, to see if the microphone would work. By default, Audacity was set to my system microphone, so I clicked Edit→Preferences, and under the recording section of the Audio I/O window I chose ALSA: Logitech USB Microphone: USB Audio from the Recording Device drop-down menu. I also changed it to be a Mono device.

After I clicked OK to accept my changes, I clicked the big red Record button on the main Audacity window and started talking. I was able to see my voice in the output immediately, and once I clicked the Stop button and played it back, I definitely was able to hear myself.

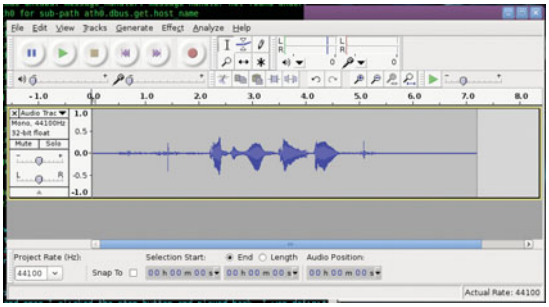
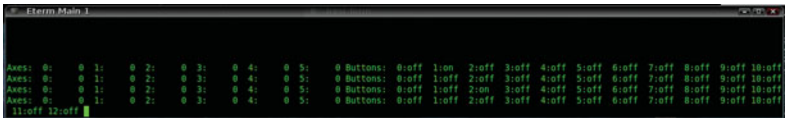


Figure 1. Audacity

The Guitar

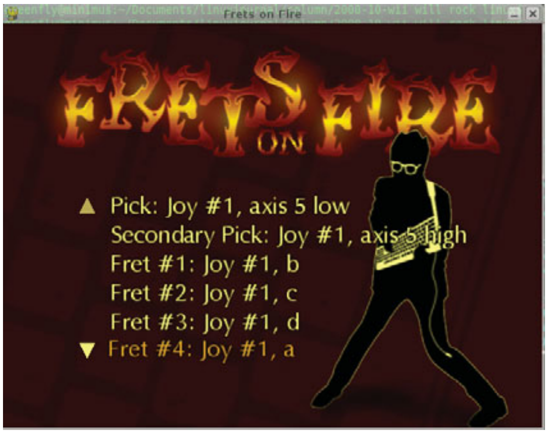
Considering how easy it was to use the microphone, I wondered what Linux would make of the wireless guitar. It appeared to connect a lot like a wireless mouse or keyboard with a small USB dongle that had a connect button you could use to sync with the wireless device. When I connected the dongle, I could see in the dmesg output that my Ubuntu Hardy install had detected the device as some sort of USB Human Interface Device (HID):

```
[  775.322361] usb 1-1: new full speed USB device
     using uhci_hcd and address 3
[  775.369009] usb 1-1: configuration #1 chosen
     from 1 choice
[  775.525791] usbcore: registered new interface
     driver hiddev
[  775.531822] input: Licensed by Nintendo of America
     Harmonix Guitar Controller for Nintendo Wii as
     /devices/pci0000:00/0000:00:1d.0/usb1/1-1/
➥1-1:1.0/input/input10
[  775.545411] input,hidraw0: USB HID v1.11 Gamepad
     [Licensed by Nintendo of America  Harmonix Guitar
     Controller for Nintendo Wii] on usb-0000:00:1d.0-1
[  775.545444] usbcore: registered new interface
     driver usbhid
[  775.545451] /build/buildd/linux-2.6.24/drivers/hid/
➥usbhid/hid-core.c: v2.6:USB HID core driver
```



```
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0 Buttons:  0:off  1:on   2:off  3:off  4:off  5:off  6:off  7:off  8:off  9:off 10:off
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0 Buttons:  0:off  1:off  2:off  3:off  4:off  5:off  6:off  7:off  8:off  9:off 10:off
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0 Buttons:  0:off  1:off  2:on   3:off  4:off  5:off  6:off  7:off  8:off  9:off 10:off
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0 Buttons:  0:off  1:off  2:off  3:off  4:off  5:off  6:off  7:off  8:off  9:off 10:off
 11:off 12:off
```

Figure 2. jstest Output



▲ Pick: Joy #1, axis 5 low
Secondary Pick: Joy #1, axis 5 high
Fret #1: Joy #1, b
Fret #2: Joy #1, c
Fret #3: Joy #1, d
▼ Fret #4: Joy #1, a

Figure 3. *Frets on Fire* Key Settings

Figure 4. *Frets on Fire* Gameplay

It appeared like a new gamepad device had been installed under /dev/input/js0, so I used the useful jstest utility (packaged by a number of distributions) to test whether the buttons on the guitar generated events. To use jstest, simply execute the program with the joystick device to test as an argument (in my case, /dev/input/js0). Each time a joystick event is registered, the output in the terminal updates. The four lines shown in Figure 2 are examples of the output when I pressed and released the green and red buttons on the guitar, respectively.

If you compare the lines, you can see that the green button corresponded to button 1, and the red button corresponded to button 2. Because the guitar interfaces directly with Linux like a regular joystick device, that means I can use its buttons with any game that supports joysticks.

Of course, probably the best game for the *Rock Band* guitar on Linux is *Frets on Fire*. *Frets on Fire* is an open-source guitar game written in Python and packaged for a number of distributions and operating systems. By default, it is designed to be used with your regular keyboard held in your hands somewhat like a guitar. The F1–F5 keys are frets on the guitar, and the Enter key can be used to strum. That works okay, but it certainly is nicer to use a guitar intended for the purpose, and sure enough, *Frets on Fire* supports remapping the default keyboard keys to joystick buttons.

To configure *Frets on Fire* for my guitar, all I needed to do was start the game, go into Settings and then modify the key settings. I just went through each key configured for the game, selected it, and then when it asked me to press a new key to set it to, I chose the corresponding key on the guitar. After you change the keys in this method, you will notice that you can navigate the *Frets on Fire* game completely from your guitar. You can strum up or down to move through the menus and use the green button to make selections.

The Drums

The final *Rock Band* instrument is also my favorite—the drums. Although you could argue, I suppose, that the microphone is the closest to a real instrument in the game, the drums feel the most real to me. The big question, of course, was whether the drums registered in Linux. Upon connecting the drums to my machine, I had hope from the dmesg output:

```
[  400.997524] usb 1-1: new full speed USB device
    using uhci_hcd and address 7
[  401.059524] usb 1-1: configuration #1 chosen
    from 1 choice
[  401.078667] input: Licensed by Nintendo of America
    Harmonix Drum Controller for Nintendo Wii as
    /devices/pci0000:00/0000:00:1d.0/usb1/1-1/
➥1-1:1.0/input/input14
[  401.104320] input,hidraw0: USB HID v1.11 Gamepad
    [Licensed by Nintendo of America  Harmonix Drum
    Controller for Nintendo Wii] on usb-0000:00:1d.0-1
```

It turns out the drums show up as a joystick device, just like the guitar. I ran jstest (as with the guitar), pointed to the new joystick device, hit a few of the drum pads, and was able to see that they definitely generated button events. Specifically, I saw that blue was button 0, green was button 1, red was button 2, yellow was button 3, and the foot pedal was button 4.

Now, although I could presumably use the drums with *Frets on Fire*, or really any game that

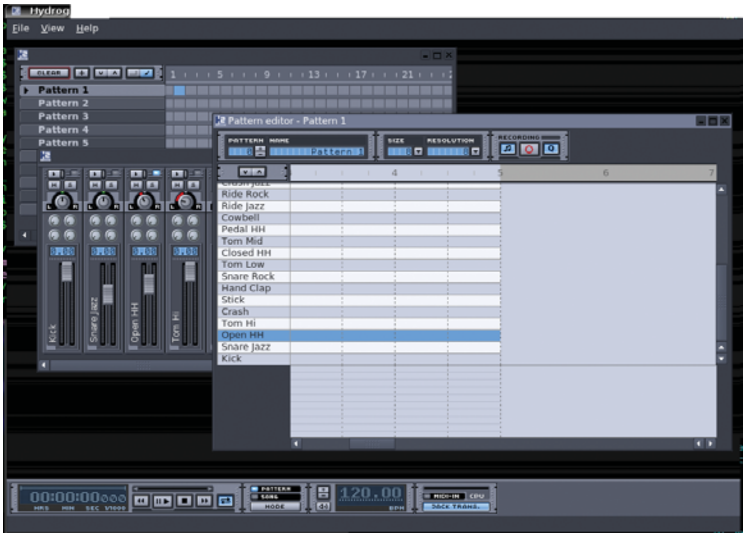


Figure 5. Hydrogen with the Pattern Editor Window Selected

supported joysticks, unfortunately, I wasn't able to find a free game for Linux that specifically used the drums. Instead, I found something arguably better: a free Linux drum kit program called Hydrogen that lets you create your own drum tracks and can interface with the keyboard or a MIDI device. Hydrogen was packaged for my distribution, or alternatively, you can download and build it from the official site. Unfortunately, the Wii drum kit doesn't act as a MIDI device, and Hydrogen isn't set up to accept input from a joystick. Hydrogen does allow you to use keys on the keyboard to activate different parts of the drum kit, so I had to figure out a way to map the joystick buttons to key events. Lucky for me, such an application already exists called joy2key. joy2key is a pretty basic program. You run the program on the command line and tell it which joystick to use and which keys to map to particular joystick buttons. Then, you can click on the application to bind it to, and it will send all joystick events to that particular window.

joy2key also already was packaged by my distribution, and after it installed, I simply had to choose to which keys to bind buttons. The first five drum types are activated in Hydrogen by the Z, S, X, C and D keys, respectively. So, first I launched Hydrogen, and then in a terminal, I typed:

```
joy2key -X -buttons d c s x z -dev /dev/input/js0
 ➥-thresh 0 0 0 0 0 0 0 0 0 0 0 0
```

In addition to the -buttons option, the -X option tells joy2key to send X events. The -dev option points it to your joystick device, and the -thresh option sets the low and high thresholds to trigger events for each button. If you don't specify -thresh, joy2key prompts you to set the values each time you run it, and as these buttons are either on or off, I just set them to zero. After you run this command, your mouse icon should turn into a cross. Click on the Hydrogen window, and then joy2key will start sending events to Hydrogen.

The order of drum sounds and how they correspond to keys is set in the Hydrogen pattern editor (Figure 5). There are any number of different ways to arrange the sounds and button mappings, but probably the easiest order to keep straight sets the pattern editor as though you played across the Wii drum kit starting at the foot pedal. By default, this probably won't be set correctly to suit the joy2key settings, so click a particular drum sound to highlight it, and then press the up/down arrows on the top of that column to rearrange its order. On the bottom, put Kick, then a Snare, then a Hi Hat (like Open HH), then a Tom, then a Cymbal (Crash). Once you have arranged these sounds, hit some of the drum pads on the drum kit, and you should hear their corresponding sounds on your computer. Go ahead, play a drum solo or two to get accustomed to the current pattern.

Hydrogen is a complicated enough program to warrant its own article, but here are some of the many things you can do now that the Wii drum set works with it. For one, Hydrogen includes a number of different drum set samples from which you can choose, and you even can create your own, so you can experiment with a lot of different sounds for your drums. In addition, you also can use your drum set when recording different beat patterns. Finally, if you want, you could just hook up your computer to a loud set of speakers and start playing. Hydrogen includes a mixer for each sound, so you can adjust the relative volumes.

Well, if you weren't already tempted to buy a set of *Rock Band* instruments just for your Wii, now you have another excuse...er, reason...why you need them. It's a testament to how far Linux has progressed that you can get random devices like these working on your computer with minimal effort. As for me, I'm going to switch up the drum patterns in Hydrogen so that they feature more cowbell.■

Kyle Rankin is a Senior Systems Administrator in the San Francisco Bay Area and the author of a number of books, including *Knoppix Hacks* and *Ubuntu Hacks* for O'Reilly Media. He is currently the president of the North Bay Linux Users' Group.

Resources

Audacity: **audacity.sourceforge.net**

Frets on Fire: **fretsonfire.sourceforge.net**

Hydrogen: **www.hydrogen-music.org**

software development | BEST PRACTICES

OCTOBER 27-30, 2008 | HYNES CONVENTION CENTER | BOSTON, MA

EARLY BIRD DISCOUNT
Register by SEPTEMBER 19
Save up to $400

Over 175 classes and tutorials

TRACKS INCLUDE:

- Advanced Algorithms
- Agile Processes & Methods
- C++
- Design & Architecture
- Languages & Implementation
- People, Projects & Teams
- Requirements & Analysis
- Testing & Quality
- Web Services/SOA

Plus

- Expo Floor
- Visionary Keynotes
- Case Studies
- Parties
- Birds-of-a-Feathers

And Much More!

FREE EXPO PASSES ARE AVAILABLE!

Register today at www.SDBestPractices.com

The World Electronics Expo and Robot, Gizmo & Gadget Show

Not a new product per se but a forum for many is the forthcoming dual-track event: The World Electronics Expo and the Robot, Gizmo & Gadget Show. This new event will feature the latest from the world of electronics. Categories will include gaming, audio, digital imaging, emerging technologies, home networking, home theater/audio, in-vehicle technology, wireless and the Robot, Gizmo and Gadget Show. Both members of the trade and the general public will experience close up the machines of the future and how they will affect our lives. The show will be held June 18–20, 2009, at the Las Vegas Convention Center in Las Vegas, Nevada.

www.theworldelectronicsexpo.com

Bernard Golden's *Open Source in the Enterprise* (O'Reilly)

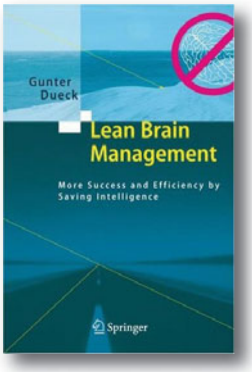
Is open source right for your company? A new resource from O'Reilly may help you decide. Dubbed *Open Source in the Enterprise* and written by Bernard Golden, this O'Reilly Radar Report is for CIOs, IT managers and business owners who want to make smart decisions about deploying open source. The study not only outlines open source from a business perspective, but it also presents three action plans to help companies effectively increase its use. Downloadable in PDF format, both individual and five-user site licenses are available.

www.oreilly.com

Gunter Dueck's *Lean Brain Management* (Springer)

"Intelligence is wasted on problems that themselves have been caused by an excess of intelligence", says author Gunter Dueck in his new book *Lean Brain Management* from Springer. This satirical book seeks to transform society to minimal intelligence everywhere possible. For example, after 30 minutes of "Googling", any human can talk intelligently on any topic. With this book, Dueck presents a radical suggestion for world improvement. The desire to laugh about the consistent economization of intelligence eventually will hopefully segue into a collective rude awakening. In keeping with the theme, the book is written in an easy-to-read fashion. It contains no self-doubt whatsoever.

www.springer.com



CherryPal C100 Computer

While your desktop PC is greedily gulping 114 Watts of power, the newly released CherryPal C100 cloud computer will do the same work with 98% less—only 2 Watts. CherryPal, Inc., says that its new creation has no moving parts, contains 80% fewer components, is highly secure and runs a customized version of Debian. No maintenance is required, as most information is processed and stored off-site in the so-called CherryPalCloud. The CherryPal also offers a new single software layer technology, which collapses the operating system and browser into one layer. The single layer makes the CherryPal exponentially faster, says its maker, and it virtually eliminates the risk of bugs or viruses for the user. The CherryPal sports Freescale's MPC5121e mobileGT processor, 256MB of DDR2 DRAM and a 4GB NAND Flash-based solid state drive.

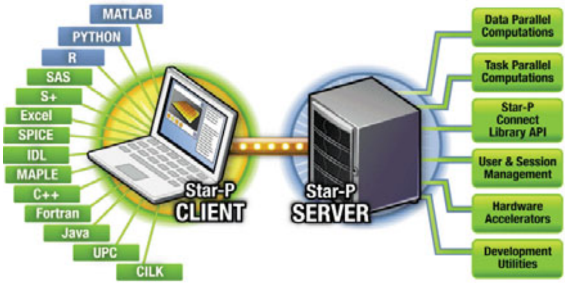
www.cherrypal.com

Vyatta Series 2500 Open Networking Appliances

The IT appliance sector is having its own Cambrian explosion, fueled in part by the newly released Vyatta Series 2500 Open Networking Appliance line. The line of networking appliances is "designed to meet the connectivity, security and protection demands of medium to large enterprises". The Vyatta 2501, the first appliance available in the series, offers tightly integrated routing and security features and broad interface support, up to 10Gbps. It further includes the Vyatta Community Edition 4 software (routing, firewall, VPN, VoIP QoS and so on), several LAN and WAN options, two onboard Gigabit Ethernet ports and one each PCI-X and PCIe expansion slots.

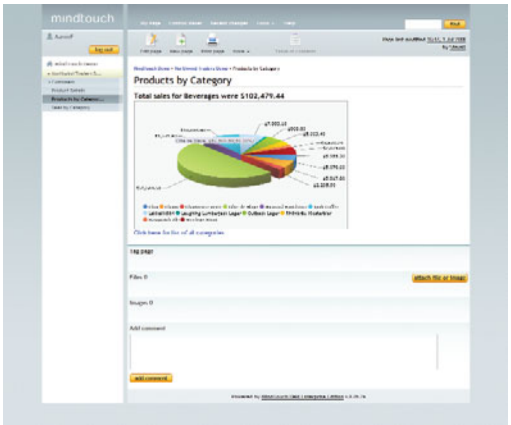
www.vyatta.com

Interactive Supercomputing's Star-P Software



In news for the HPC crowd, Interactive Supercomputing, Inc., recently upgraded Star-P, a software application for accelerating and managing HPC workloads across clusters and supercomputers. Star-P is an interactive parallel computing platform that allows scientists, engineers and analysts to create algorithms and models on their desktops using familiar mathematical tools, such as MATLAB and Python, and run them instantly and interactively on parallel computers with little or no modification. The new version adds support for the SGI Altix line of blade servers, Platform Computing's LSF workload management software and large scale-out workloads.

www.interactivesupercomputing.com



MindTouch's Deki "Kilen Woods"

MindTouch bills its Deki (formerly Deki Wiki) as an open-source enterprise collaboration and integration platform that helps information workers, IT professionals and developers collaborate and connect disparate enterprise systems and data sources. With Deki, businesses can connect and mash up the application and data silos that exist across an enterprise—including legacy systems, CRM and ERP applications, databases and Web 2.0 applications. Adapters are available for widely used IT and developer systems, such as SugarCRM, Salesforce, LinkedIn, MySQL, VisiFire, WordPress and more.

www.mindtouch.com

Software Workshop's ExtSQL

Enriching the vast MySQL ecosystem is the new ExtSQL, short for Extended usage statistics for SQL. The product is designed to provide DBAs with a simple tool for monitoring database activity by individual user, database, host or even connection. Software Workshop asserts that present SQL monitoring tools allow only gross monitoring at the server level. ExtSQL provides extra detail, including historical data, and makes it available from the command line. The ExtSQL server is designed to be a drop-in replacement for an existing mysqld or postmaster executable.

www.extsql.com

Please send information about releases of Linux-related products to newproducts@linuxjournal.com or New Products c/o *Linux Journal*, 1752 NW Market Street, #200, Seattle, WA 98107. Submissions are edited for length and content.

Fresh from the Labs

Droopy

(stackp.online.fr/droopy)

First up this month, we have Droopy, a miniature Web server. Now, if you're like me, and the combination of seeing the words Linux and Web server usually results in a sleep-induced coma, fear not. This actually is more useful for average Internet users. Its sole purpose is to allow other people to upload files to your PC by presenting them with a Web page interface, and its requirements are about as minimalist as I've come across.

Installation Thankfully, Droopy has only one real requirement—Python. As 99% of you already have that installed, we can jump right into this one. Droopy itself is merely a Python script, so all you need do is head to the project's Web site, and save the droopy file to your local hard disk. You will be running Droopy through the command line, mind you, so save it to a directory that will be easy to access via the command line. The Droopy Web site recommends making the directories ~/bin and ~/uploads, and saving the droopy file to ~/bin.

Once you've done this, it's time to run the script. If you made the uploads directory, open a terminal there before running the script. This isn't a requirement, but wherever you run the script, this is where any uploaded files you receive will go.

Usage The Droopy site and man page have an example command that inserts a greeting message and displays a picture as well:

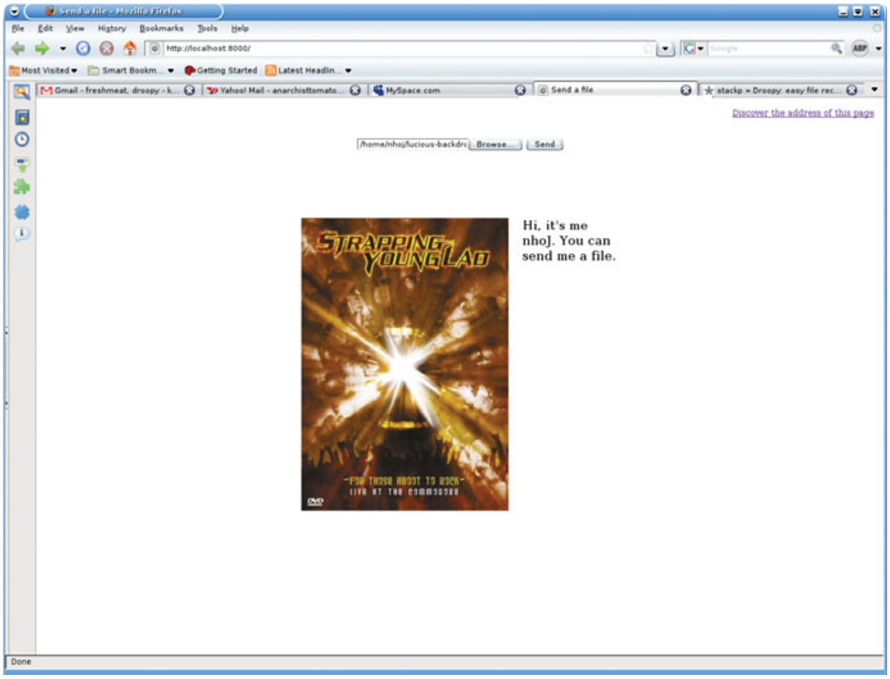
```
$ python ~/bin/droopy -m "Hi, it's
me Bob. You can
 ➥send me a file." -p ~/avatar.png
```

If you have Droopy installed somewhere other than ~/bin, change to path to wherever the droopy file is sitting now. If, like me, you're not called Bob, change the name (you also might want to use a less goofy message). The picture isn't a requirement, but it can help identify your page. It needn't be avatar.png either, any image file will do.

Once the script is running, you can visit a mini-Web site from any browser at http://localhost:8000/.

If all is well, you should have something that resembles the screenshot shown here. This is all well and good, but people need to upload to you. Clicking Discover the address of this page will give you a URL that you can then pass on to your friends, so they now can upload to you, provided the script is running. To upload to one of these pages, there's a rather obvious empty text field with Browse and Send buttons sitting next to it that will allow the people uploading to choose the file they want and send it to you. Once they have sent it, a notification should appear on your terminal output, and the new file will be sitting in your uploads directory.

Not being a security expert, I imagine there's probably some sort of vulnerability here (this most likely would be catastrophic on Windows), but I couldn't give any real advice in that regard. Personally, I don't have a mission-critical enterprise system, so I'm not exactly worried myself, but dig around if you're concerned. All in all, Droopy is a clever piece of scripting that is easy to install and fairly easy to use, provided that you're not scared of the command line. For those put off by transfer methods, such as IRC, MSN clones and the like (and not forgetting pesky e-mail size limits), this may be just what you're chasing.


Send a file - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:8000/
Most Visited Smart Bookm... Getting Started Latest Headlin...
Gmail - freshmeat, droopy - k...
Yahoo! Mail - anarchisttomato...
MySpace.com
Send a file
stackp » Droopy: easy file rec...
Discover the address of this page
/home/nhoj/lucious-backdro
Browse...
Send
STRAPPING YOUNG LAD
FOR THOSE ABOUT TO ROCK
LIVE AT THE COMMODORE
DVD
Done

Hi, it's me nhoJ. You can send me a file.

With Droopy, you can lose the limitations of annoying transfer programs with your very own mini-Web server.

safe-rm

(code.google.com/p/safe-rm)

With the advent of sudo and an increasing number of new Linux users, the possibility of users deleting mission-critical files by accident is becoming all the more real. To deal with this issue there is now safe-rm:

> safe-rm is intended to prevent the accidental deletion of important files by replacing /bin/rm with a wrapper that checks the given arguments against a configurable blacklist of files and directories that should never be removed. Users who attempt to delete one of these protected files or directories will not be able to do so and will be shown a warning message instead. Protected paths can be set both at the site and user levels.

Installation Installing safe-rm is a pretty rudimentary affair. You basically just copy one file to the right place. To begin, head to the Web site and grab the latest tarball. Extract it, and as root, copy the safe-rm file to /usr/local/bin, and rename it to rm.

Make sure the file is flagged as readable and executable for the rest of the system (as root or sudo):

```
# chmod a+rx rm
```

If this doesn't work, you may want to make a backup of the original rm in /usr/bin and then copy and rename safe-rm here. This will make your system use safe-rm in place of rm. Of course, you could leave the filename as is and enter safe-rm every time you want to delete a file, but who wants to do that?

As for usage, just use `rm` the same way you always have, but with the warm and fuzzy knowledge that you're not going to kill your system or accidentally cause nuclear war. Overall, safe-rm is a useful and clever modification on an age-old tool that hopefully will make its way into mainstream distros soon.

Freenukum

(launchpad.net/freenukum)

Ah, now for a bit of nostalgia. If your idea of vintage gaming is a Nintendo 64, you probably won't have a clue what I'm talking about. But, for those who are from the era of at least the 286, you no doubt will remember such classics as *Commander Keen*, *Jetpack* and, of course, *Duke Nukum*. If you're thinking *Duke Nukum 3D*, then think again. That was a remake of this! This was back in the days of the 2-D platformer, and when Commander Keen was king, this came along as a sort of Team America version—rude, crude and supposedly violent (but very tame by today's standards).

With these old classics fading into obscurity and requiring a lengthy explanation from wizened geeks like myself, enter *Freenukum*, a restorative Linux version on which to waste more office hours. An authentic reconstruction, *Freenukum* makes use of (and requires for the moment) the original level files to bring back the same feel of this classic platformer.

Installation The actual program installation is a very straightforward affair, with various binaries available or source code. The source is quite minimal, requiring only the usual:

```
$ ./configure
$ make
```

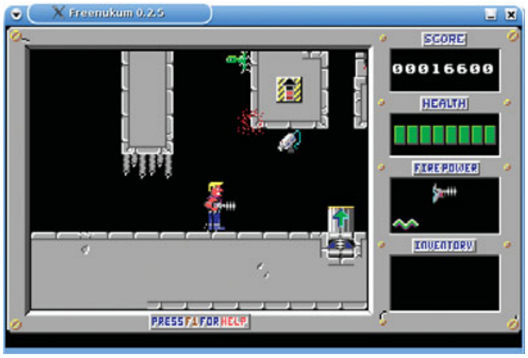
And, as root or sudo:

```
# make install
```

Compilation took only a few seconds on my system, and the configure script didn't give me any complaints.

With the compilation out of the way, you still have one more step before you can run the game. *Freenukum* currently requires the original level files to run, so you need to get a copy of the original from somewhere. Either the shareware version or the full version will work, so Google around and find a host that suits you. Of course, there are abandonware sites, but we aren't encouraging that sort of thing.

Once you have downloaded the original, copy the game's files into the directory ~/.freenukum/data (if you're a bit stuck here and using a graphical file manager, turn on Show Hidden Files). If it's not there, simply create the directories, and everything should be tickety-boo. If you're pedantic about keeping a tight system, a lot of those files aren't needed, but this game was made back in the day of the 286, so the game isn't exactly big. I just copied the whole game.



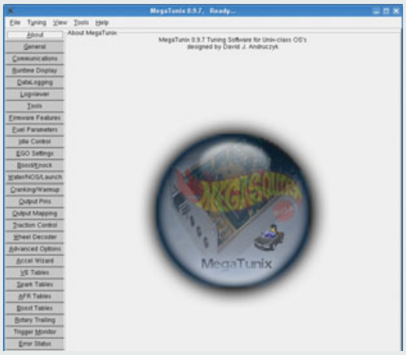
The now tame but classic *Duke Nukum* restored with *Freenukum*.

Projects at a Glance

I'm going on a petrol head stint this month and have picked up three cool looking projects for you fellow gas guzzlers.

MegaTunix
(**megatunix.sourceforge.net**)

For any ECU tweakers out there with Subaru-colored pajamas, Japanese Drift videos and a Colin McRae embroidered duvet, this is the program for you. Mega Tunix is "...the only tuning software for UNIX- (and now Win32-) class operating systems that supports all existing megasquirt firmwares". MegaSquirt is apparently "an open-source EFI controller for internal combustion engines, comprised of embedded software, tuning software and various build and deployment tools". For those readers who are still following me, the MegaTunix developers claim to have the most complete and accurate ECU interrogation of any project out there. The latest versions have been redesigned to be extensible further to support new firmware variants, and the GUI is broken down into lovely little tabs. Neat.

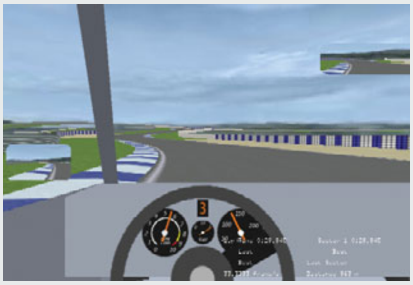


MegaTunix 0.9.7 Tuning Software for Unix-class OS's
designed by David J. Andruczyk

ECU trickery just got neater with lovely little tabs. And my goodness, there are a lot of them.

Vamos
(**vamos.sourceforge.net**)

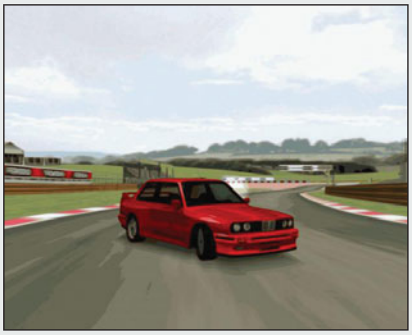
Vamos is a very young project concentrating on being "an automotive simulation framework with an emphasis on thorough physical modeling and good C++ design. Vamos includes a real-time, first-person, 3-D driving application". It also includes a number of cool real-world locations, with tracks such as Germany's Nurburgring and Japan's Suzuka Circuit, among others. However, this won't be a major draw card of authenticity just yet, as the graphics are still at a level comparable to a 286, and the cars resemble something more like what Postman Pat would drive. As a result, the project's author is inviting anyone to contribute to the effort. Still, it looks promising, especially as parts of its code are being borrowed from another project.



Vamos—Postman Pat shakes up the Laguna Seca speedway in his delivery van!

VDrift
(**vdrift.net**)

Powered by the just-mentioned Vamos engine, "VDrift is a cross-platform, open-source driving simulation made with drift racing in mind", and it's currently available for Linux, FreeBSD, Mac OS X and Windows (Cygwin). Although the game is in an early development stage, it is supposed to be very playable and quite feature-packed, with 19 tracks (including the Nordschleife track), 28 cars, AI players, "very realistic physics" and a (simple) multiplayer network mode. Initial screenshots look a little rudimentary at times, but seriously sweet at others. I look forward to playing this one and hope to have an in-depth view of both Vamos and VDrift over the coming months.



Tire squeal just got amplified ten times with VDrift!

Usage Once all that's out of the way, to run it, enter the following command:

```
$ freenukum
```

Once you're in the main menu, press the S key to start a new game. Left and right arrow keys control your directional movement, and the up arrow key is used to activate things such as platforms, switches and so on. The left Ctrl key is for jumping; the left Alt key is for shooting, and that's pretty much it—things were simple back in those days! Check the man page for further info on which items do what and further info on the game itself (type `man freenukum` at the console).

At its current state, some things aren't implemented in the menu yet, such as instructions or the high-score table, so you'll definitely need that man page. Even so, *Freenukum* still is in a pretty solid state, and it's very playable. Project author Wolfgang Silbermayr made me promise I'd mention that he's looking for some graphic and level designers to help make some original level files to include with the game by default. Once this happens, it'd be great to see *Freenukum* included in distro repositories.

A shareware download is available at **www.3drealms.com/duke1/index.html**.■

John Knight is a 24-year-old, drumming- and climbing-obsessed maniac from the world's most isolated city—Perth, Western Australia. He can usually be found either buried in an Audacity screen or thrashing a kick-drum beyond recognition.

Brewing something fresh, innovative or mind-bending? E-mail me at knight.john.a@gmail.com.

The Apache Software Foundation and Stone Circle Productions Presents

ApacheCon US 2008

The Official User Conference of The Apache Software Foundation

Leading the Wave of Open Source

November 3-7, 2008

Sheraton Hotel | New Orleans

Bringing Users, System Administrators, Enterprise Business Developers and Architects together with ASF Members and Committers to meet, geek and create the future of Open Source.

The Apache Open For Business Project
OFBiz.Apache.org

Announcing the co-location of OFBiz Symposium

The Apache Open For Business Symposium will offer an in-depth Training, along with two days of Business and Technical Sessions, covering innovations, developments, and opportunities in Open Source technology through the creation and collaboration of specialized applications.

ApacheCon and OFBiz Symposium, five days of Open Source at it's finest!

Internationally-Recognized Speakers, Presenters, and Instructors | In-Depth Trainings and Presentations "First-Looks" at Groundbreaking New Technologies | Apache from A to X | BOF Sessions. . . and more!

Special Events Include:

Hackathon, BarCampApache, Business Panel, Receptions, Hadoop Camp, Lightning Talks, PGP Keysignings, Fast Feather Tracks, and a "Voluntourism Project" to help rebuild New Orleans!

Experience the future of Open Source

http://www.us.apachecon.com

Sign Up Today!

Receive 10% Off Full Conference and Single Day Passes

(To receive the 10% discount, use discount code "Community"; Training Sessions Excluded)

Sponsored by:

Google™ | hp® invent | HIPPO | HotWax Media | IONA | spring source | YAHOO!

CrowdVine Social Networks | Day | LINUX MAGAZINE | (mt) mediatemple | WSO2 Oxygenating The Web Service Platform

New Orleans: Open. To Just About Everything!

HARDWARE

Load Me Up, Load Me Down

The second-generation HP Media Vault. DAN SAWYER

The HP Media Vault 5150 is a Linux-based network-attached storage (NAS) device that aims to be the end-all-be-all for home and small-office network file management and media service. It boasts not only a large capacity (700GB or 1.4TB depending on how you allocate it), it also has a hardware RAID-1 option and USB ports for attaching additional storage. Its internal drive bays use SATA drives, and the internal capacity theoretically is upgradable to the limit of SATA drive technology, and it hooks into your network through Gigabit Ethernet. Running out of bandwidth, therefore, is not in the cards.

What It Does

The HP Media Vault runs an SMB server, serving up browsable shares to the network. Due to its large capacity, it's very useful for a number of purposes, and it comes outfitted with a number of helper applications that allow home users to maximize the benefits of having such a device around. These bundled applications allow users to run an iTunes server, share photos on-line with automatic gallery generation, expose selected directories to the Internet and stream media to properly enabled appliances that hook up to TVs and stereos. In other words, in addition to being an all-purpose backup server, this thing aims to be your TiVo, your jukebox, your photo server, your document server and your Web server, all rolled into one with an automated backup cherry on top.

All this functionality is administrable through a handy-dandy suite of programs bundled with the device that runs on any modern Windows box. It makes efficient use of open-source programs for nearly all its features, and it is generally a well-engineered little piece of technology. Certainly, home brewers who are looking to create their own NAS appliances could do worse than look at what HP has pulled off with this little gadget.

The Good

The Media Vault lives up to its hype rather handsomely. It's pretty easy to administer with the bundled software—easy enough that an average computer user should have very little difficulty getting up and running and secured. The documentation that ships with it is aimed entirely at novice users, walking them step by step through the self-explanatory configuration screens and leaving, as far as HP is concerned, nothing to chance.

The automatic backup function is a particularly nice touch—although underneath the hood, it's little more than an active cp script running in the background, the interface on it is slick and should make data protection miles easier for the average Joe. As someone who climbed out of the hell of doing sysadmin work in my younger years, I must confess that I think it's rather like giving condoms to teenagers—it's better that they have the ability to protect themselves, but most of them probably won't think of it when they're in the heat of the computing moment. Still, we can hope.

The UPnP/DLNA server option, which is what allows the Media Vault to act as a streaming server for set-top boxes, actually works only with a limited number of devices, as the standard is pretty new. But, it seems to work with those devices seamlessly. A number of programs also receive DLNA streams, most particularly VLC and MythTV, which means Linux-savvy home users can use the Media Vault as a streaming server all on its own instead of configuring a separate streaming server for their media automation systems.

Mounting It from Linux

Using the Media Vault from a Linux box on a heterogeneous network is dead easy, so long as you have the relevant Samba packages installed. You'll need SMBFS support and Samba client support if you want to set your Media Vault shares to mount to your filesystem at bootup. In order to pull this off, I had to do a little detective work to discover the share names to plug in to fstab.

I used `smbclient -L hpmediavault` to grab the following shares list from the Media Vault:

```
Domain=[HPMEDIAVAULT] OS=[Unix] Server=[Samba 3.0.25b]

        Sharename       Type      Comment
        ---------       ----      -------
        Photos          Disk      Default_Photos
        Music           Disk      Default_Music
        Videos          Disk      Default_Videos
        Backup          Disk      Default_Backup
        Documents       Disk      Default_Documents
        IPC$            IPC       IPC Service (HPMediaVault Server)
Domain=[HPMEDIAVAULT] OS=[Unix] Server=[Samba 3.0.25b]

        Server                 Comment
        ---------              -------

        Workgroup              Master
        ---------              -------
```

Because there isn't a default ubershare, you'll have to add one line to your fstab for each share. So long as you have the proper Samba support installed, from here on out it's very easy. For each share, add a line as follows:

```
//hpmediavault/sharename /your/mountpoint/here smbfs
 ➥username=username,password=password,user,defaults 0 0
```

Note the use of the user-mountable flag—this is important if you expect to be able to write to the share at all. Samba mounts are picky about who mounted the drive, and most systems won't let users write to a mounted smbfs share unless they mounted it themselves.

The Media Vault ships in a completely unsecured state—no password is required to log in or configure the device. To my mind, for a device aimed squarely at the average-Joe end of the market, this is the perfect default. I've seen people I otherwise care for very much turn into incomprehensible babbling masses when confronted with a factory-set admin password—they generally don't know enough to look for a sentence like "factory default login". Of course, this is a double-edged sword, as there's nothing actually compelling users to set a proper password or to take the additional available steps to secure the box, so there will doubtless be a number of unsecured servers coming online in the coming months as the Media Vault is adopted by its core audience.

Attaching external storage to increase the capacity (or to back up) the Media Vault is also dead easy. Simply plug in a hard drive, allocate it with the administration utility, and assign it a mountpoint. Once that's done, you're ready to rock and roll. The Media Vault supports ext3 and FAT32 filesystems natively, and it supports NTFS on a read-only basis.

Finally, a number of nice little options are available, such as control over hard drive spindown intervals and LED brightness—both of which are very nice if you decide to set up the device in your bedroom.

The Bad

That's not to say that all is wine and roses. There are a lot of niggling little problems with the HP Media Vault that keep it just on this side of perfect.

The first, and perhaps the most irritating, is that despite the easy kernel-level support for NFS, HP has chosen to strip this functionality from the Media Vault. The Media Vault *only* serves up files over Samba, and although Samba is nice, it requires extra tweaking and software installation for Linux and Mac clients compared to NFS. HP could have broadened its market at virtually zero expense simply by leaving NFS in the system.

HP also has, alas, not organized its documentation in a way that's particularly friendly to those of us who don't—or can't—use the included administration software. This is a shame, as administering all but the most advanced functions of the Media Vault is simple for anyone with a Web browser and an SSH connection. With a little digging around—and the help of the good folks at HP's Marketing department—I found the Web admin panel, enabled SSH, and got the server up and running. See the Configuration without Windows sidebar for instructions on how to configure your Media Vault if you want to do it the old-fashioned way.

To get full functionality out of the server, you have to use HP's bundled administration software, and this software doesn't play nice with most operating systems. More to the point, it plays nice only with Windows XP and Vista—it won't even install on Windows 2000 or older systems, and it doesn't work with Wine. This is a problem if you're wanting to use some of the more advanced newbie-friendly features, such as the iTunes server or the auto-generating photo albums and video playlists.

However, if you're willing to go without those things, most everything else can be accomplished from the Web admin panel. And, if you're a better hacker than I am, you can configure the iTunes server manually over SSH using the instructions on the Firefly home page (**www.fireflymediaserver.org**).

However, to my mind, the most egregious problem is that currently no firmware restore exits, nor any hardware reset, nor are there any operating system restore disks either bundled with the product or available for download. This means that if you screw up the system,

Configuration without Windows

For those of you who, like me, don't keep Vista or XP machines around, setting up the Media Vault is a bit more of an adventure. We simply don't have the option of using the bundled software without borrowing someone else's computer or breaking down and installing XP on machines that we've previously kept sacrosanct from MS Product Activation. This is how you set up all but the most advanced features of the HP Media Vault using SSH and a Web browser.

The first thing you need to do, after plugging the Media Vault in to your network and powering it on, is to find the IP address. As it comes set up to grab a DHCP lease automatically, the easiest way to do this is to log in to your router and find the most recent lease. Once you find the address, pull up that address in your Web browser. The browser screen is a fairly straightforward Webmin panel—it allows you to create users, access levels and directories, and to enable DLNA streaming on a per-directory basis.

In order to enable SSH access, you need to go into the System section, press Edit, and set your admin user name and password. The password you set becomes the root password for the box, and you now can log in via SSH. The user name and password you set also become the login info for the Web admin panel. The System section, by the way, is where you can set the LED brightness level and the hard disk spin-down interval. While you're setting up access levels, you'll want to add a user (or a few users) appropriate to your environment in the User screen. Everyone gets access to the basic pre-allocated folders, and each user can create his or her own folders that you can set as private or shared, both through the Web admin panel and through regular permissions management.

The other thing you'll want to do immediately is to allocate the disks on the Disks page. By default, the MV5150 comes with one of the 700GB disks allocated and the other unallocated. You have the choice here to allocate the second disk as a RAID-1 mirror or to allocate it as additional disk space, resulting in 1.4TB of total space. This page is also where you can add external USB disks of the appropriate filesystem types.

Once you plug it in, you can, with a bit of jiggerypokery, find the thing with an SMB browser. It's actually non-obvious in some SMB browsing software (including some versions of Windows), but direct access can be had at smb://hpmediavault. The easy way to deal with this, of course, is to set up your workgroup information in the Network tab of the Web admin panel.

The last thing to do to get the system up and running is to set up the Web server by enabling remote access in the Remote Access tab. Annoyingly, it doesn't seem to work without a domain registration (free for a year, costing money after that), but checking this off allows the folders whose permissions you have set as browsable to be browsed from the Internet through a handy PHP interface. Hacking this thing so it'll serve up your documents without going through the activation process is pretty simple: SSH into the box, create a symlink in the /usr/htdocs folder to the /share/1000/ folder. You then can serve up files at http://myserver'sipaddress/symlink/sharefoldername/filename.

The last tab you'll want to check out is the Backup tab at the far right. This allows you to hook up a USB drive and do a selective backup—direct copy, not compressed—of selected directories. This process will wipe the destination drive, but it's nice to have the easy redundancy option with the processing performed locally on the Media Vault rather than clogging the network by copying between one remote share and another.

you're screwed. And, as the root partition is writable, screwing this thing up while you're hacking it is easy. One misstep, and you've bricked the device, and there is no recourse short of shipping the item back to HP, and it's unclear whether the repair would be covered under warranty.

Conclusion

Despite my lengthy griping above, this is a seriously well-designed NAS. HP has done its homework and designed a box that will hit its target market right between the eyes. Unfortunately, it's not going to do more than that, so despite the fact that I've been really impressed by it, my buy recommendation is a tepid one.

For Linux users looking for a safely hackable NAS, it might be a bit much. The lack of any system restore means that this box is fragile and might not play nice if you prod it in the wrong place. It's likewise priced on the high side for what it delivers to someone who isn't using it from a Windows machine and doesn't need serious data redundancy.

For average home users who are big into Web 2.0 services, it likewise should be a very useful item, saving a lot of time and making it even easier for people to plug their lives in to the Internet or take the bother out of managing their media collections over the home network.

For the price (almost $700), the HP Media Vault 5150 is in the no-man's land between a great value and an overpriced toy. It's well-outfitted, physically robust, well-designed and has a lot of great little features that make it ideal for a small-office/home-office environment. Particularly impressive are its easy backup features and its extensibility. I personally have found it quite useful as a footage server, storing recordings and raw video for projects I'm working on in my studio and for streaming draft projects out to the screening room for previews.

A subset of the Linux market will find this box well worth the price. If it suits your needs, it should be an excellent addition to your network. But, if you're not in the position to take advantage of the Windows-only value-added features, and the data security that the RAID and scheduled backups afford you isn't worth paying the premium for, you may want to give this one a miss. Here's hoping HP continues to build great Linux-based devices, and in the future leaves them a little more open for those of us who like Linux on more than just our servers.■

Dan Sawyer is the founder of ArtisticWhispers Productions (www.artisticwhispers.com), a small audio/video studio in the San Francisco Bay Area. He has been an enthusiastic advocate for free and open-source software since the late 1990s, when he founded the Blenderwars filmmaking community (www.blenderwars.com). He currently is the host of "The Polyschizmatic Reprobates Hour", a cultural commentary podcast, and "Sculpting God", a science-fiction anthology podcast. Author contact information is available at www.jdsawyer.net.

Register Now!

JOIN US

ONTARIO LINUX FEST

OCTOBER 25, 2008
TORONTO, ONTARIO CANADA

Get all of the details at http://onlinux.ca

Phone: 877-TUX-FEST 866-883-1172 x5102 info@onlinux.ca

LINUX PRO MAGAZINE waterloodatafortress

hp netdirect open solutions Google LINUX JOURNAL Since 1994: The Original Magazine of the Linux Community

SOFTWARE

Review of Scalent's Virtual Operating Environment

Scalent's V/OE virtualizes the entire data center, including storage, network and server operating systems, enabling fluid deployment or repurposing of servers from physical to virtual and back again.

LOGAN G. HARBAUGH

As the use of Linux in the data center continues to expand, the need for management tools for deployment, version control and patch management becomes more critical. In addition, the loads on servers can vary dramatically during special events, bringing a need to be able to reconfigure servers quickly and dynamically from one operating group to another to provide temporary capacity expansion, and then repurpose them back to their original groups once the high levels of demand have passed.

The Virtual Operating Environment (V/OE) from Scalent Systems, Inc., offers a mix of management and deployment tools that provides a flexible and far-ranging system for deploying and managing Linux systems in both standard and virtual environments. Scalent is not simply a deployment management system—it also can manage switches, storage and boot images, enabling a server, for instance, to be repurposed from a Web server on the public network to an application server on the development network, with all necessary changes handled from a single console with a few clicks.

> We were able to create a VLAN that matched the lab network, connect to my network, log in to the server, download and install the agent, connect to the Scalent controller and manage that new server in about 15 minutes total.

Scalent can migrate a physical server to a virtual server, which is not unusual, but it also can migrate a virtual server back to a physical server easily, which is quite unusual. The way it does this is by integrating the Scalent software with both network and SAN hardware, using boot from SAN to allow a single boot image to be cloned and deployed to one or many servers easily, either physical or virtual. Once deployed, a server also can be migrated automatically in case of failure.

The SAN can be either Fibre Channel or iSCSI, and in the case of iSCSI SANs, Scalent has licensed emBoot, which allows systems to boot from an iSCSI target without requiring an expensive iSCSI-specific Ethernet controller.

The Scalent software can be integrated with many different storage systems and network hardware, allowing enterprises to use their existing hardware if desired. Scalent provides engineering support to integrate the software with your hardware and get everything up and running. For the purposes of my testing, I received a preconfigured rack of equipment that included five servers, one running the Scalent software, a Fibre Channel switch, Ethernet switch and IBM storage system. Scalent sent Field Engineer Steve Leung along with the equipment to help integrate the system into my test network and demo the software.

The first steps—integrating the Scalent software with the switches, storage system and the servers in the rack—already had been done, as they would be for any Scalent customer. In addition to the systems in the rack, we added two servers from my lab to the Scalent network—an HP Proliant ML370G5 and an HP Proliant DL360G3. This involved configuring the servers for PXE boot and setting up the Fibre Channel controllers to boot from the SAN, then connecting them via Ethernet and Fibre Channel.

Adding the new server from my lab to the pod Scalent brought was quick and easy. We were able to create a VLAN that matched the lab network, connect to my network, log in to the server, download and install the agent, connect to the Scalent controller and manage that new server in about 15 minutes total. Then, the Scalent appliance was able to deploy personalities to the VMware ESX 3.5 server on the ML370G5 in less than a minute.

Once a server is connected to the Scalent network, configured to PXE boot, and has boot from SAN enabled on its Fibre Channel adapter, it receives a mini-boot environment from the Scalent server that allows it to boot from SAN and be managed. Then, all that is necessary is to use the Scalent software to create a boot image for that system (which can be cloned from an existing image if desired), set up a LUN for that image, and point the server at that image. The Scalent V/OE system works with a large variety of switches and storage through APIs, and it also is able to talk with load balancers, such as F5's BigIP.

Creating a new OS image is simple—after creating a new LUN from which to boot the server, any OS is installed as if it were being installed to a local disk. Once that image is created, it can be cloned by the storage system and used to boot any other server. Most flavors of Linux are supported, as is Windows 2003 Server.

If a server needs to be repurposed, all that is necessary is to create a new image, point the server at the new image, and reboot it—no copying of files to the actual server is necessary,

because the server simply boots from the new LUN. Scalent does support a local boot option, where the boot image is copied to the local drive on the server as well.

Scalent installs an agent on each server instance to monitor server activity and enable failover to another physical or virtual instance if the server goes down. The lightweight agent can be downloaded from the Scalent controller to each server quickly and easily. It shows status, load, operating conditions, connectivity and so forth, giving an excellent overall view of network health from the Scalent controller.

In addition to creating and moving boot images for servers easily, the Scalent system makes it simple to create virtual LAN segments to isolate networks and to create SAN environments with the proper storage connected to each server. This means that moving a server instance from one logical group to another also can change network settings automatically to put it into a different VLAN, change SAN port settings so that the appropriate storage is available, performing all the tasks from a single console rather than having to log in to Fibre Channel and Ethernet switch consoles and the storage systems console separately to move things around.

In the case of large organizations where each of these tasks might be compartmentalized and performed by separate groups, the system supports multiple levels of users with specific, granular permissions.

The easy and quick support for virtual LAN and SAN segments makes it very simple to secure networks by keeping different groups of servers on different segments, but it removes the need to have special-purpose servers physically isolated on separate network switches.

From the fault-tolerance angle, creating failover servers for business-critical systems is quick, easy and flexible. Failover servers don't have to be identical—if a server fails, the system boots the same image on new hardware. The Scalent image creation utility does a full install with all drivers, so images should work on any hardware, although some Linux display drivers may not function without reconfiguration. There also can be some issues with moving from Intel to AMD or vice versa, as well as moving from 32-bit to 64-bit. But in general, the parameters for creating a backup server are much looser than most redundant systems.

The Scalent system can replicate the storage used for boot images to secondary remote storage, and it can bring up an entire server farm on new hardware at a new location in only the time required for bootup. Because all changes are reflected on the boot image in real time, servers are up to date with changes as of the time of failover. The gap in service is limited to the time it takes for the new servers to boot. As the switches, IP addresses, subnets and storage LUNs are all managed together, the new servers in the new location have the same IPs as the originals and continue operation as if there had been no change.

Given the increasing use of virtualization, Scalent's support for a single boot image for both physical and virtual servers is a big deal.

This entire process can be automated, so that an entire data center could be moved to another location automatically in case of failure. This level of functionality is easy to set up with the Scalent system, and without it, nearly impossible to achieve without a great deal of configuration and testing of some platform such as OpenView.

Given the increasing use of virtualization, Scalent's support for a single boot image for both physical and virtual servers is a big deal. This only works with VMware's ESX 3.5, because earlier versions of VMware don't support booting from a block device. Scalent also is partnered with XenSource to enable support for Xen and XenSource virtualization systems as well.

For migration of VMs from one ESX server to another, the Scalent server can handle all partitioning, access to storage, networking and so on. For physical-to-virtual migration or virtual-to-physical migration, the same boot image is used for both physical and virtual servers, so no translation or conversion is required. This enables migration from physical to virtual or virtual to physical with no conversion process or delay required. In contrast, other systems that support migration use a translation process, and although physical-to-virtual conversion works well, virtual-to-physical migration may be problematic.

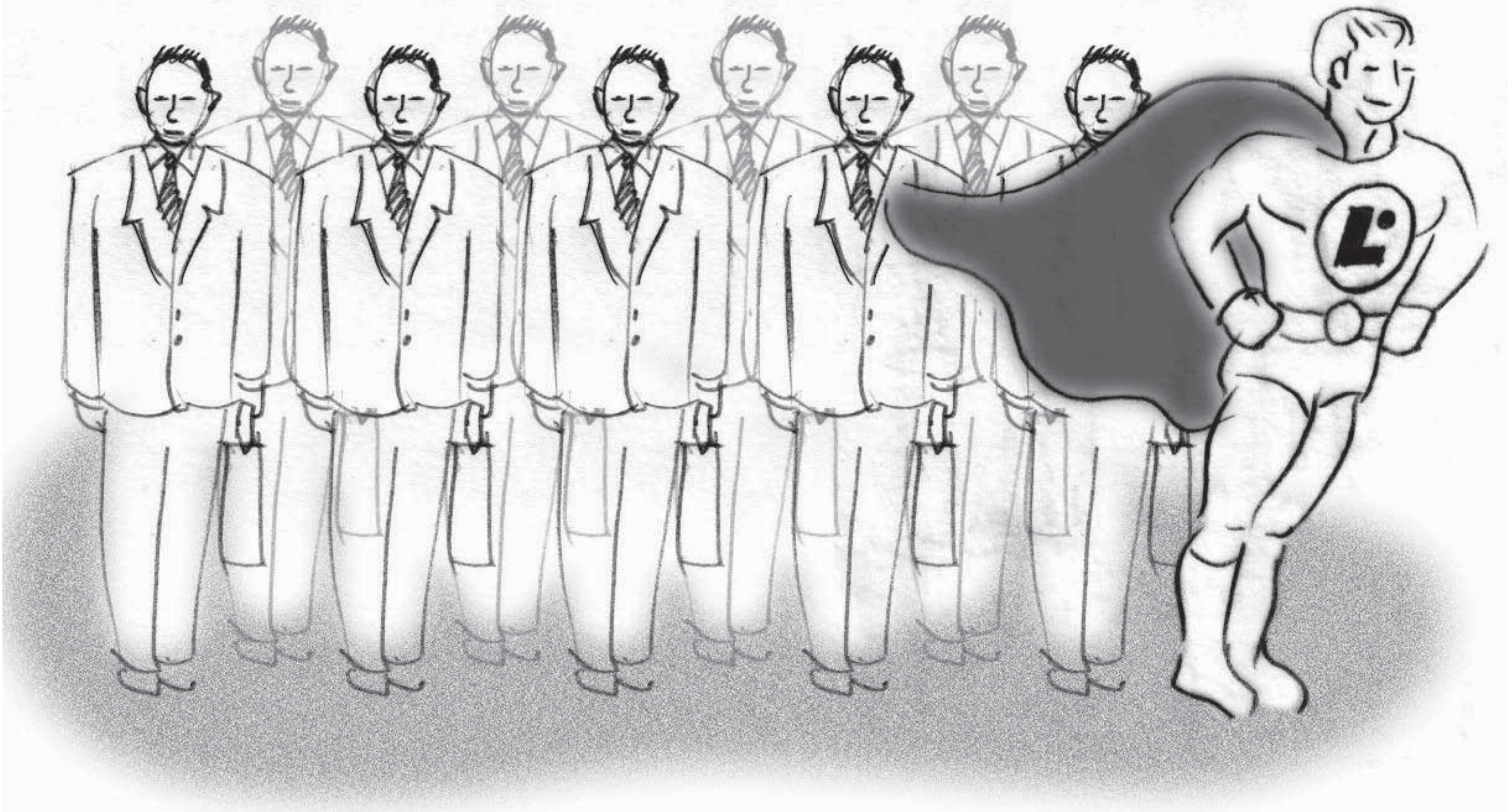
When using the boot from SAN with Fibre Channel adapters, Scalent supports both Emulex and QLogic HBAs, and it also supports Emulex's world-wide-name (WWN) aliases in BIOS, as well as at the driver level for QLogic. Normally, some back and forth is required to get things set up, as a WWN has to be assigned after a new LUN is created, the server masked to that name, then the image created, an alias WWN assigned by the Scalent controller, and then the WWN on the Fibre Channel HBA changed to match the alias. With the new functionality in Emulex controllers, an alias can be assigned during the initial configuration, which means that the process is simplified considerably.

The Scalent system also supports iSCSI boot from SAN using emBoot. This means that a specialized iSCSI Ethernet controller, also known as a TOE controller, is not required.

Scalent prices its system in packs per managed physical machine CPU socket. For example, 12 sockets could be six two-socket servers or three four-socket servers. Pricing is about $1,000 per physical socket managed. There is no limitation on the number of virtual systems or OS images managed.

Although $1,000 per system is not inexpensive, the ability to migrate systems from one server, network and SAN easily to another provides a degree of flexibility not available with any other system I've used, along with an ease of setup and management that is also unique. As data centers continue to grow, and the need for dynamic capacity management becomes more critical, the Scalent V/OE system starts to look like a real bargain.■

Logan G. Harbaugh is a freelance reviewer and IT consultant located in Redding, California. He has been working in IT for almost 20 years and has written two books on networking, as well as articles for most of the major computer publications.



Interview with Guido van Rossum

Despite some revolutionary new features, "**Python 3.0** will be the same language you've loved and used before, it's just been cleaned up a bit", says Python creator, Guido van Rossum.

JAMES GRAY

Python is the wildly popular, high-level programming language that was recently voted Favorite Scripting Language in the 2008 *Linux Journal* Readers' Choice Awards. In this interview, Python's creator Guido van Rossum shares his insights about the revolutionary new Python 3000, why the pain from backward incompatibility is worth it, what he foresees for the Python 2.6 fork, and what he's been up to lately at Google.

"You've probably heard that Python 3000 will introduce backward-incompatible changes. That alone probably is enough to get developers excited, or at least upset."

JG: By the time readers see this interview, Python 3000 (aka Py3K and Python 3.0) should be available. What is in the new version that will excite developers?

GVR: You've probably heard that Python 3000 will introduce backward-incompatible changes. That alone probably is enough to get developers excited, or at least upset. So let me emphasize first that, by and large, Python 3.0 will be the same language you've loved and used before, it's just been cleaned up a bit. You may want to contrast this with Perl 6 vs. Perl 4, where Perl 6 is a totally new language, with a completely different implementation. We're not doing anything remotely as drastic as that!

Many of the cleanups are pretty benign. For example, we're finally getting rid of string exceptions (all exceptions have to be defined as classes). There is a large class of cleanups like this, and I refer your readers to the python.org Web site for the (mostly) boring details.

Some changes *seem* controversial but actually are a big improvement, such as replacing the print statement with a print() function. The big advantage of making it a function is that we can use the familiar keyword=value syntax to specify behavioral variations like printing to a different file or suppressing the final newline. We also can add new keywords more easily. For example, in Py3k you can override the separator between items, and this makes future evolution much easier compared to evolution of a statement-based syntax. Using standard function syntax also makes it much easier to replace the built-in print function with a function of your own design. This is a common transformation over the lifetime of a program. What started out as simple print statements at some point have to become logging calls or at least redirectable to a different file, and all these changes are easier to make consistently with function calls.

> **"The 2to3 tool takes care of the syntactic changes, and the Py3k warnings in Python 2.6 handle those changes that a purely syntactic tool cannot handle easily."**

There is one group of changes that is (relatively speaking) revolutionary, and at the same time, it is probably responsible for the most conversion pain, *and* for the largest sigh of relief. We're adopting a fundamentally different attitude toward Unicode. A bit of history: Python 1 supported only eight-bit strings, which were used for text and binary data alike. Python 2 kept this dual use of eight-bit strings, but added Unicode strings. This was done so as to maintain backward compatibility with Python 1, but it created a new major ambiguity. There were two ways of representing text strings, either as eight-bit strings or as Unicode strings. Moreover, the meaning of eight-bit strings remained ambiguous, as these were used for text as well as binary data.

In Python 3, we're breaking with compatibility and drawing the line differently. There will be a bytes type to be used for binary data (and *encoded* text, like UTF-8 or UTF-16), and there will be an str type to be used for text only and capable of representing all Unicode characters. The implementation of the bytes type closely resembles that of the old eight-bit string type, and the implementation of the str type is copied from the old Unicode type. The big improvement over Python 2 is that both ambiguities I mentioned above are removed. There is now a 1:1 mapping between usage (data or text) and types (bytes or str). Reports from early adopters have shown that developers really appreciate this change and are happy to pay for it. Some third-party projects, such as Django, already have adopted a convention in Python 2 that essentially is the same. All text is stored in Unicode strings, and eight-bit strings store only binary data, but Python 2 doesn't help enforce this.

There also are some other changes related to Unicode. The default source encoding is now UTF-8, identifiers can contain non-ASCII letters, and the repr() function no longer will turn all non-ASCII characters into hex escapes (it still will escape control characters of course).

JG: In retrospect, do you regret any changes that made it through to the final version?

GVR: No, I'm very happy with the outcome. I think we've struck a phenomenal balance between changing too much and changing too little. It has really helped that toward the end of the Py3k development, we switched to a time-based release schedule, so we had a clear way to stop the never-ending stream of proposals for yet more language improvements.

JG: Python 3000 is currently slower than 2.5. Will it be as fast or faster once it is seriously tuned?

GVR: I expect that by the time 3.0 is released, we'll be close to the 2.5 speed. We'll probably keep tuning it well beyond that, and if past history is any measure of future performance, we'll see continued speed improvements as new releases come out.

JG: Python 3 breaks backward compatibility with version 2.6. This is a pretty bold step for a programming language in general and in particular for one with a user base the size of Python's. The only other time I remember somebody trying this was when Microsoft went from VB6 to VB.NET, a move that has a lot of VB6 programmers still miffed six years later. Do you have concerns regarding this move?

GVR: I think you may have forgotten about Perl 6.

My understanding is that VB.NET was actually fundamentally different from VB6, much more so than Python 3 differs from Python 2. Most of the differences in Python 3 are relatively close to the surface. In particular, we've made a conscious choice *not* to radically change the underlying implementation. If I understand correctly, VB.NET uses a completely different virtual machine (based on the new .NET technology) from VB6. This is not the case for Python 3. We started Py3k as a branch of the Python 2 VM and gradually modified it to support the new language. But, most implementation details are exactly the same, and up to this date, we routinely merge changes from the trunk (which will be released as Python 2.6) into the Py3k branch.

I certainly don't want to underestimate the cost for developers of the transition from Python 2 to Py3k. We have been thinking about this transition for at least two years now, and we have several parallel strategies in place to make developers comfortable with the change.

First of all, Python 2 will be fully supported for a long time in parallel with Python 3. My personal expectation is that there will be a period of at least three to five years where developers have complete freedom to choose between Python 2 or Python 3, getting the same level of support. There will be new releases of Python 2, starting with 2.6, in parallel with the Python 3 releases.

Second, we have designed a specific two-prong transition strategy. The first prong of this strategy is the release of Python 2.6 simultaneously with the 3.0 release. 2.6 will be backward compatible with 2.5, but it also will contain an *optional* set of warnings that alert you about a variety of issues in your program that will break if and when you port it to Py3k. These warnings are issued only when specifically requested via a command-line option, so that they are not an impediment toward upgrading from 2.4 or 2.5 to 2.6, regardless of whether you are planning to port your code over to 3.0. In addition, 2.6 also will contain some back-ported 3.0 features, which we hope will encourage people to start using 2.6 in a way that will reduce the pain when they are ready for 3.0.

The second prong of the transition strategy is a source code conversion tool that we call 2to3. This tool handles most of the small syntactic changes you encounter when converting Python 2 code to Py3k. For example, it automatically translates print statements into print() function calls, turns Unicode literals (such as `u"..."`) into regular string literals, strips the trailing L from long integer literals, and so on. It also does a decent (though not perfect) job of converting calls to popular dictionary methods like .keys() and .iterkeys() into their Py3k equivalent.

The two prongs complement each other nicely. The 2to3 tool takes care of the syntactic changes, and the Py3k warnings in Python 2.6 handle those changes that a purely syntactic tool cannot handle easily. Because Python is such a dynamic language, conversions that require information about the type of a variable or attribute generally cannot be automated. The 2to3 tool leaves these alone, but there is enough overlap between the 2.6 and 3.0 languages that, in general, it will be possible to change your source code in such a way that it still is compatible with Python 2.6 (and usually with older versions as well), produces no Py3k warnings, and can be translated safely to valid Python 3.0 source code using the 2to3 tool.

JG: Also, how complex do you think that the upgrade process to Python 3000 will be?

GVR: I think I've given a decent indication of the complexity in my answer to the previous question. The general work flow for a conversion could be as follows:

1. Start with code that works under Python 2.4 or 2.5 and has a good test suite.

2. Port to Python 2.6. This should be straightforward. Try to run the test suite under Python 2.6, resolve issues found, and repeat until all tests pass. Python developers have used this process for years with the transition to each Python version, and the expectation is that there won't be many changes to make.

3. Turn on Py3k warnings and run the test suite again. Resolve issues reported, and repeat until all tests pass without warnings.

4. Run the 2to3 tool over your source code, including your test suite, and run the converted test suite under Python 3.0. If there are issues, *don't fix them here*, but fix them in the 2.6 code base, and repeat starting from step 3.

In terms of revision control, you most likely will be maintaining two branches of your code long term: the 2.6 version and the 3.0 version. Changes to the 2.6 version should be merged to the 3.0

Another, quite unrelated, but also hugely exciting, trend is the activity in the PyPy Project.

version using the 2to3 tool.

JG: What kind of feedback have you gotten from the early adopters of Python 3000 thus far?
GVR: We've heard everything from pure excitement to extreme fear. Given the magnitude of the change, we can't expect everybody to be happy, but the general trend is one of cautious optimism. As expected, most developers are happy with most of the new features. Although almost everyone has a pet peeve or two, those appear to be mostly outliers, and there aren't any changes that stand out as unwanted by many.

JG: Have any large projects already been converted to Python 3000, and what have the results been?
GVR: It's too early to say. We've only just released the first betas of 2.6 and 3.0, and so far, the focus of third-party developers, especially of large packages, has been on 2.6 over 3.0.

JG: Is there a chance that there might be a rogue fork of the 2.x line, and would this bother you?
GVR: I don't expect any "rogue" forks to happen. The Python community tends to prefer consensus over conflict, at least in the long term.

JG: What was the process by which changes were accepted or rejected in the upgrade process?
GVR: We started out by setting some basic parameters for the upgrade, in PEP 3000: the goal was primarily to fix early design mistakes and clean up situations where two ways to do something had evolved out of a desire to improve the language while also maintaining backward compatibility (for example, new-style vs. classic classes). This was a powerful argument to keep many of the more radical change proposals out of the door.

The rest was a matter of long community discussions with the occasional tie-cutting by yours truly in case a consensus remained elusive. I have an incredibly subtle set of gut feelings for judging the most "Pythonic" solution to any one issue, keeping a precarious balance between pragmatics and principles. But, I have tried to use this only after ample discussion had clarified motivations and use cases for proposed changes.

JG: Were there any changes you wanted that were rejected, or any that you didn't want that were accepted?
GVR: That's hard to say. I certainly have proposed things that were rejected, but in the end, I always ended up agreeing with the rejection—and, ditto in the other direction.

JG: How are your synapses currently firing regarding Python 4000 and beyond?
GVR: I hope I'll be in retirement by then!

JG: Our Publisher Emeritus and your old friend Phil Hughes asked me to ask you, "Is Django [the high-level Python Web framework] as cool as it appears?"
GVR: Oh yes, it is. (And hi, Phil!) I like it because it strikes a very Pythonic balance between theory and practice, and because the organization of the project is very similar to that of Python itself. The Django developers run an excellent open-source project, listening carefully to their users and contributors, without being distracted by "feature-itis".

JG: KDE 4.x has abandoned the classic desktop for Plasma, which supports writing scripted add-ons, or applets, in a number of programming languages. Do you see a role for Python in this space?
GVR: This is the first I've heard of this, so I'd rather not make any rash comments. I hope that if Plasma becomes popular, its developer makes it scriptable using Python.

JG: What interesting trends have you seen lately in the development of the Python community?
GVR: I'm very happy with the influx of new developers in the past year or so. This has really enriched the community with new ideas and new areas of expertise, and removed the pressure from some of the old hands who have been keeping things running for many years.

Another, quite unrelated, but also hugely exciting, trend is the activity in the PyPy Project. As you may remember, PyPy started out as an attempt to write a portable Python interpreter in Python, made fast by the use of a Python-specific JIT. Most PyPy developers are in Europe, and with two years of EU (European Union) funding, the project has made tremendous progress. As agreed ahead of time, the EU funding ended after two years, but recently Google has started funding some specific PyPy activities, and I am excited that these will eventually make PyPy a viable alternative to CPython.

JG: You have been working for Google now for almost three years. Can you divulge what they've had you working on, or is it top secret? Also, is Python subject to Google's 80/20 rule—the one that allows employees to spend 20% of their time on personal projects that are potentially worthwhile to the business—or do you have a different arrangement?
GVR: It's no secret that my first Google project was Mondrian, an internal Web tool for collaborative code reviews using Perforce. Since last November, I've been working on Google App Engine, an exciting project that allows Web developers to run scalable Python Web applications on Google's powerful infrastructure. (In the future, other languages also will be supported.)

I have written an App Engine demo that reuses some components of Mondrian and refactors them into a code review tool for Subversion. With Google's permission, I have released this as open source. You can see it working at **codereview.appspot.com**, and you can find a link to the source code there as well.

I don't have a 20% project per se, but I have Google's agreement that I can spend 50% of my time on Python, with no strings attached, so I call this my "50% project".

JG: Thanks so much for your insights, Guido, and good luck with the new Python!■

James Gray is *Linux Journal* Products Editor and a graduate student in environmental sciences and management at Michigan State University. A Linux enthusiast since the mid-1990s, he currently resides in Lansing, Michigan, with his wife and cats.

A TALE OF TWO LANGUAGES

Inform version 6 and version 7 are about as different as two languages can be, so your choice of which one to use for your next great game boils down to personal preference.

DANIEL BARTHOLOMEW

As soon as the interactive fiction (or text adventure) genre got started with the arrival of Will Crowther's *Advent* in 1975, people have been working on ways to make creating it easier.

The original *Advent* was programmed in FORTRAN. The expanded version, created by Don Woods, used PL/1. Both of these languages are general-purpose computer languages well suited for many tasks, but not particularly suited for creating interactive fiction. Many of the early fans of *Advent* wanted to create their own version of the game, and using FORTRAN or PL/1 was difficult to impossible, depending on the computer platform to which they were porting it.

One thing many of these early implementers realized about interactive fiction was that most of it was just text, and that the operating system-specific parts of a game amounted to only about 10% or less of the overall size. The solution many arrived at was to put all the operating system-independent text and logic into a story file and then create an interpreter or virtual machine that could play or run the story file. These story files first were written in a custom language designed for creating adventure games and then compiled into the proper format. This method allowed *Zork*, the most popular and commercially successful *Advent*-like game, to be available for 23 different platforms, a testament to the power of "virtualization" decades before it became a hot buzzword.

Infocom, like all the commercial publishers, had its own language. Infocom's language was called ZIL (for Zork Implementation Language). It was not public and had to be compiled on a mainframe. When compiled, it would run inside the Z-Machine (Zork-Machine). Ports of the Z-Machine were built for every platform Infocom supported. The ZIL language was powerful and had lots of features, but its proprietary nature, and the fact that a mainframe was required for compilation, put it out of the reach of hobbyists, so the community created its own languages. Six/Fant, DDL (Dungeon Definition Language), ADL (Adventure Definition Language), TADS (Text Adventure Development System), AGT, Hugo and others were created, enabling anyone to write interactive fiction.

Throughout Infocom's short lifetime, several versions of the Z-Machine were created. These versions differed in the complexity of the story files that they would support. For example, version 3 of the Z-Machine supports up to 255 objects, 32 attributes and a maximum story file size of 128K. Version 5 of the Z-Machine could handle up to 65,535 objects, 48 attributes and a maximum story file size of 256K. Version 8 of the Z-Machine is identical to version 5, with the added feature of supporting story files up to 512K in size. Version 5 was the one used by Infocom for most of its advanced games.

By the early 1990s, all the commercial publishers had gone out of business, so if you wanted to play a new piece of interactive fiction, you either had to write it yourself or hang out on the newsgroups dedicated to interactive fiction. The classic games from Infocom and others still could be found on various re-releases and Collector's Editions. The interactive fiction genre wasn't dead, but many consider the period between 1989 and the mid-1990s to be the "dark ages".

Despite progress, the Infocom games still were seen as the high watermark in interactive fiction, and creators had a hard time approaching them with the various community-developed systems. They just weren't powerful enough or were missing important features or were lacking a decent parser or all of the above.

It wasn't until after Graham Nelson successfully reverse-engineered the Z-Machine, and created a language and compiler that would produce story files compatible with any interpreter, that supported version 5 (and 6 and 8) of the Z-Machine, that the second golden age of interactive fiction began. He called his creation Inform, and he based the design of the language on ZIL.

Speaking of ZIL, here is a simple example:

```
<OBJECT SERVER
        (LOC SERVER-ROOM)
        (DESC "an ancient server")
        (FLAGS NODESC)
        (SYNONYM SERVER COMPUTER MACHINE ANCIENT OLD)>
```

Basically, when creating something in ZIL, you have objects, and those objects have properties and attributes. This format proved itself to be very flexible for Infocom's games, and the language was modified and improved as new versions of the Z-Machine were developed.

Inform has followed a similar path. Like the Z-Machine with which it is designed to be compatible, the Inform language has gone through several refinements over the years. The current versions of Inform are 6 and 7. Of the two, version 6 of Inform is very similar to ZIL. Here is a ZIL example in Inform 6:

```
Object server "server" server_room
    with
        description "an ancient server",
        name 'server' 'machine' 'computer' 'ancient' 'old',
    has scenery;
```

Everything is still an object, and most of the differences between them (at least in the example above) are in matters of naming. FLAGS has been renamed to has, SYNONYM has been renamed to name, NODESC has been renamed to scenery and so on. That's not to say that there aren't substantial differences—there are. Probably the main difference at this point is that Inform 6 can create far larger and more complex games than any of the ones Infocom created with ZIL.

However, even with a powerful and refined language at their fingertips, many of the top interactive fiction writers—including Graham Nelson, Emily Short, Andrew Plotkin, Sonja Kesserich and others—felt that writing interactive fiction in Inform 6 was not natural enough. Writing in Inform 6 feels like programming; there's no way around it. The flippant retort to such a statement is, "Well, yeah. You're creating a computer game. What did you expect?" Their response was that creating interactive fiction should be more like creating regular "non-interactive" fiction.

How radically different is Inform 7 from Inform 6? Well, here's the same example from above in Inform 7:

```
The ancient server is scenery in the server room. Understand
"machine" and "computer" and "old" as the ancient server.
The description of the ancient server is "an ancient server".
```

So, instead of statements, we have natural language paragraphs. However, while Inform 7 was in the planning stages, the decision was made that even though it was going in a radically new direction, it should still be built on top of Inform 6. What Inform 7 does behind the scenes during compilation is to parse what you write and translate it into Inform 6. The translation is done by machine, so it doesn't look like what a human would write, but it is valid Inform 6 and can be compiled by the regular Inform compiler. This allowed developers to focus on the design of the language instead of the design of a brand-new compiler.

Listing 1. Describing a CD Tray in Inform 7

```
094  The CD Tray is part of the old server.
095      The description is "It's a CD tray."
```

Listing 2. Essential Code for Compiling Inform 6

```
001  !% -SD
002
003  !=======================================================
004  Constant Story "The Server Room";
005  Constant Headline
006      "^An Interactive Fiction by Daniel Bartholomew.^";
007  Release 1; Serial "080625"; !for keeping track of releases
.
.
172  !=======================================================
173  ! Entry point routines
174
175  [ Initialise;
176      location = break_room;
177      "^^^^It's Saturday, a nice one at that, and you've been
178          called to fix a server. Again.^^You've had it. This
179          server is going to run Linux starting today! The process
180          will be easy, just put the disc into the server go.
181          Now where is that Ubuntu CD?^";
182  ];
183
184  [ Deathmessage;
185      if (deadflag == 5) print "You have won";
186  ];
187
188  !=======================================================
189  ! Standard and Extended Grammar
190
191  Include "Grammar";
192
193  !=======================================================
```

To illustrate the differences between Inform 6 and Inform 7 I've created an example game in both languages. Due to space constraints, we can't print the full games here, but they are available on our FTP site at **ftp.linuxjournal.com/pub/lj/issue174/10130.tgz**.

The compiled version of each game also can be downloaded from the *Linux Journal* FTP site (available at the same address listed above). Even though the source code is very different, they both play the same.

As I refer to the two programs, I will use the Inform version (i6 or i7), followed by a colon (:) and the line number(s). So, if I refer to the Inform 7 version of the cd-tray object, I would write it as i7:094-095 (Listing 1).

The differences between the two languages are apparent from the first lines. When starting a story in Inform 6, there are some essential housekeeping duties that need to be done at the beginning (i6:001-007) and end (i6:172-193) in order

Listing 3. Initial Program Code for Inform 7

```
001  "The Server Room" by "Daniel Bartholomew"
002
003  Include GNU GPL v3 by Free Software Foundation.
004
005  The maximum score is 6.
006
007  When play begins, say "It's Saturday, a nice one at that,
008       and you've been called to fix a server. Again.
009       [paragraph break]You've had it. This server is going to
010       run Linux starting today! The process will be easy, just put
011       the disk in the server and go. Now where is that Ubuntu CD?"
```

Listing 4. Syntax for Describing a Room

```
Inform 6 Version:
114  description "The fans, the lights, the chill... yep, it's a
115       server room. Full of servers from a dozen vendors,
116       each with their own quirks.^^Your attention is
117       immediately drawn to a server 2/3 of the way up rack 7.
118       The indicator light is blinking red and beeping.",

Inform 7 Version:
082  The Server Room is a room. "The fans, the lights, the chill...
083       yep it's a server room. Full of servers from a dozen
084       vendors, each with their own quirks.[paragraph break]Your
085       attention is immediately drawn to a server 2/3 of the way up
086       rack 7.  The indicator light is blinking red and beeping."
```

Listing 5. Programming Game Logic

```
Inform 6 Version:
120  daemon [;
121       if (location ~= server_room) return;
122       beeping = random(7);
123       switch (beeping) {
124            1: "^The beeping is driving you crazy.";
125            2: "^It's hard to think, with all of the beeping.";
126            3: "^The monotony of the beeping is maddening.";
127            4: "^You can't stand the beeping.";
128            5: "^The beeping reminds you of your alarm clock.";
129            6: "^beep . . . beep . . . beep . . . beep . . . beep
130                 . . . beep . . .";
130            7: "^If you don't stop the beeping soon, you'll lose
132                 what little hair you have left.";
133       }
134  ],

Inform 7 Version:
097  Every turn while in the server room, say "[one of]The beeping is
098       driving you crazy.[or]It's hard to think, with all of the
099       beeping.[or]The monotony of the beeping is maddening.[or]You
100       can't stand the beeping.[or]The beeping reminds you of your
101       alarm clock.[or]beep . . . beep . . . beep . . . beep . . .
102       [or]If you don't stop the beeping soon, you'll lose what
103       little hair you have left.[purely at random]"
```

Listing 6. Syntax for Scoring

```
Inform 6 Version:
150       before [;
151            Open:
152                 if (openedtray == 1) {
153                      score = score + 2;
154                      openedtray = 2;
155                 }
156       ],

Inform 7 Version:
105  After opening the CD tray for the first time: award 2 points;
106       say "You press the button and the CD tray pops out."
```

for the story to compile (Listing 2).

In Inform 7, the only thing that is absolutely required is the first line (i7:001). To be fair, the initialization routine in Inform 6 also includes the message that is displayed at the beginning of the game. In Inform 7 this also is there (i7:007-011), but it's not required for successful compilation (Listing 3).

The defining of variables is similar in both versions—for instance, the maximum score variable (i6:009, i7:005). The main difference is that the Inform 6 version uses C-like syntax:

```
Constant MAX_SCORE = 6;
```

And, Inform 7 uses a sentence that reads similarly to how one would read the Inform 6 version out loud:

```
The maximum score is 6.
```

Some of the differences between versions are similarly minor. One example is the method of inserting paragraph breaks into long sections of text, such as in the description of the Server Room (i6:114-118, i7:082-086, Listing 4). In Inform 6, two carat symbols (^^) are used, and in Inform 7, the statement `paragraph break` enclosed in square brackets ([]) does the job.

One big mistake I made when I started was creating everything first in Inform 7, and then creating the equivalent Inform 6 code. This proved tricky on a couple occasions—most notably with the random beeping messages that appear at every turn while you are in the server room (until you fix the server and win the game). Programming such logic is very different in each version. In Inform 7, writing this took almost no thought at all (i7:097-103). Basically, I described what I wanted to happen every turn, and Inform 7 made it happen. The Inform 6 code to do the same thing is not as advanced (i6:120-134). It's basically a case statement, but it's significantly more difficult to write in comparison (Listing 5).

Scoring is another thing that Inform 7 simplifies. In Inform 6, you can give an object the scored attribute, but for custom cases, like awarding 2 points instead of the default 1 point, you need to track things yourself (i6:150-156). In Inform 7, the process is much easier (i7:105-106, Listing 6).

So, which version is best? If I had to choose my favorite of the two, it would be Inform 7, hands down.

I prefer Inform 7 not because I find Inform 6 difficult to use or because it's an unpleasant experience. On the contrary—Inform 6 is well designed and has excellent documentation for both beginners and advanced users. It also has some clear advantages—a main one being precision. Inform 6 will do exactly what you tell it to do. Inform 7, on the other hand, is less precise, and you sometimes need to fiddle with it to get it to do exactly what you want. This lack of precision is a result of how Inform 7 translates your code into Inform 6 prior to compilation—it has to guess at what you mean sometimes, and occasionally, it will get things wrong.

Despite some disadvantages, the reason that I prefer Inform 7 over Inform 6 is because I am more of a writer than a programmer. The closest I get to programming on a regular basis is bash scripting with a little PHP thrown in now and again. The natural sentence structure of writing something in Inform 7 is more intuitive to me than the C-like syntax of Inform 6. As a writer, I sometimes think to myself, "If when reading code I pronounce 'somevar=42' as 'somevar gets 42', why can't I write it that way?" I realize and accept that computer languages are as terse and precise as they are for many reasons, but I also think efforts like Inform 7 should be applauded and copied wherever and whenever possible.

The precision of Inform 6, as with other programming languages, is good in some instances, but bad in others. One misplaced comma or semicolon or bracket prevents your game from compiling, for example. Inform 7 is not exactly forgiving in this area either, but I find myself making fewer mistakes because of the flexible and natural way you're allowed to write.

Another big advantage of Inform 7 is the wonderful integrated IDE that comes with it, complete with excellent built-in documentation, a debugger and other tools. With Inform 6, compilation was a command-line affair that could be difficult to set up properly. I won't go into the Inform 7 IDE here, but if you are interested, see the two-part series "An Introduction to Gnome Inform 7" on the *Linux Journal* Web site.

With Inform 7, there's never been a better, or easier, time to create your own interactive masterpieces. So, find the key to the grate, and join me down underground. I'll be in a maze of twisty little passages, all alike, trying to find the bearded pirate who stole my gold—if the dwarf doesn't kill me first. XYZZY!■

Daniel Bartholomew has been a fan of interactive fiction since he first was exposed to it on his Apple IIe many, many years ago. He lives with his wife and children in North Carolina.

Resources

Example Games: **ftp.linuxjournal.com/pub/lj/issue174/10130.tgz**

An Introduction to Gnome Inform 7, Part 1: **www.linuxjournal.com/content/introduction-gnome-inform7-part-1**

An Introduction to Gnome Inform-7, Part 2: **www.linuxjournal.com/content/introduction-gnome-inform7-part-2**

Inform 6: **www.inform-fiction.org/inform6.html**

Inform 7: **www.inform-fiction.org/I7/Welcome.html**

Frotz: **frotz.homeunix.org/frotz**

Zoom: **www.logicalshift.co.uk/unix/zoom**

Fortran: **en.wikipedia.org/wiki/Fortran**

PL/1: **en.wikipedia.org/wiki/PL/I**

ADL: **adl.sourceforge.net**

Six/Fant: **www.graysage.com/cg/Compilers/SixFant**

TADS: **www.tads.org**

AGT: **www.markwelch.com/agt.htm**

Hugo: **www.generalcoffee.com/hugo.html**

The Interactive Fiction Archive: **www.ifarchive.org**

Baf's Guide to the IF Archive: **wurb.com/if**

A Beginner's Guide to Playing Interactive Fiction: **www.microheaven.com/IFGuide**

TECH TIP Check Your Computer's Temperature

You can check your computer's temperature using only standard tools, with the command:

```
$ cat /proc/acpi/thermal_zone/*/temperature
```

Depending on your hardware, you may have more than one sensor.

This method of checking your computer's temperate works only if ACPI thermal zones are supported on your computer. Most laptops and some desktop systems support them.

—MATTHEW MARTIN

Shell Scripting with a Distributed Twist: Using the Sleep Scripting Language

Learn a Perl-like language whose scripts move around your network.

Raphael Mudge

No one who isn't lazy writes scripts. Scripts save valuable system administrator time. In this article, I introduce the Sleep scripting language, which is a Perl-inspired language built on the Java platform. Although Java is sometimes a bad word in our community, Sleep can help you, because a Java-based language has several benefits. Scripts work on different platforms, data has the same form everywhere, and tools to solve any problem are available through the Java class library or open-source extensions.

With Sleep, you can save time on task automation and distributed computing. Sleep can help, whether you have one box or 10,000. Here, I introduce the language and its syntax, accessing the filesystem, talking to local and remote processes, and distributed computing with mobile agents.

Getting Started

You can use Sleep right away if you already have Java installed. Make sure the Java you use is the Sun Java. Any version 1.4.2 or later will do. Sleep does not run with the GNU Java that some Linux distributions use by default:

```
$ java -version
java version "1.5.0_13"
Java(TM) 2 Runtime Environment, Standard Edition
```

Installation is easy. Visit the home page (see Resources), and download the sleep.jar file. This file has everything you need to execute Sleep scripts:

```
$ wget http://sleep.dashnine.org/download/sleep.jar
```

You can execute a script on the command line with the following:

```
$ cat >tryit.sl
println("I am $__SCRIPT__ with " . @ARGV);
$ java -jar sleep.jar tryit.sl "hello icecream" 34
I am tryit.sl with @('hello icecream', '34')
```

Sleep scripts also are happy to exist as UNIX script files:

```
#!/path/to/java -jar /path/to/sleep.jar
println("Hello Icecream!");

$ chmod +x script
$ ./script
Hello Icecream!
```

Sleep Basics

Sleep and Perl have a lot in common. Variables are scalars, and scalars store strings, numbers, functions or Java objects:

```
# Set some variables
$w = "foo";
$x = 3.14 * 12;
$y = &someFunction;
$z = [java.awt.Color RED];
```

Like Perl, Sleep comments begin with a # and end with a newline.

Variable names inside double-quoted strings are replaced with their value at runtime. For example, `"this is a $x"` will use the current value of $x. To avoid this behavior, prefix a variable with a backslash. Double-quoted strings can format variables to a small degree. Use `"$[20]x"` to pad the value of $x with spaces until it is 20 characters wide. A negative number prefixes the value with spaces. The `$+` operator brings together the left and right values in a string. For example, `"a $+ b"` is `"ab"`.

Like Perl, Sleep has arrays and hashes. An array refers to values by a numerical index:

```
@a = @("a", "b");
@a[2] = "c";
push(@a, "d");
println(@a);

@('a', 'b', 'c', 'd')
```

Hashes store and get values with a string key. Think of these as a dictionary. The keys are not kept in order:

```
%b = %(a => "apple", b => "bat");
%b["c"] = 'cat';

println(%b);

%(a => 'apple', c => 'cat', b => 'bat')
```

Scripts can create hashes of hashes, arrays of hashes, arrays of arrays, and any other combination you can imagine. These data structures offer a flexible way for storing data. And, these structures are more than hashes and arrays. Scripts can use arrays as sets, stacks, queues and lists. Combinations of arrays and hashes can make finite-state machines, graphs and trees. You can make nearly any data structure you'll need.

Sleep provides a gamut of flow control options. The for loop, while loop and foreach loop are all here. If statements work as you would expect. Sleep differentiates strings and numbers for comparisons. Here, I use the Sleep console to show the difference:

```
$ java -jar sleep.jar
>> Welcome to the Sleep scripting language
> ? "3" eq 3.0
false
> ? "3" == 3.0
true
```

The assignment loop is found a lot in Sleep scripts. This loop evaluates a statement and assigns the result to a variable before executing the loop body. The loop keeps going while the result is not $null, which is the empty value—it is equal to an empty string, the number zero and a NULL reference all at once. Most functions return $null when they are finished. This script iterates over each line of a file:

```
$handle = openf("/etc/passwd");

while $entry (readln($handle))
{
   println($entry);
}
```

Sleep uses the same functions to work on files, processes and sockets. A scalar that holds a file, process or socket is a handle. The &readln function reads a line of text from a handle. The &println function prints a line of text. Likewise, &readb reads some bytes from a handle. And, &writeb writes bytes. The following is a Sleep version of the UNIX copy command:

```
global('$source $dest $handle');
```

```
($source, $dest) = @ARGV;

$handle = openf($source);
$data = readb($handle, -1);
closef($handle);

$handle = openf("> $+ $dest");
writeb($handle, $data);
closef($handle);

$ java -jar sleep.jar cp.sl a.txt b.txt
```

Notice the value @ARGV. This array holds the script's command-line arguments. The &closef function closes a handle.

Scripts declare named functions with the sub keyword. Arguments are available as $1 to $n:

```
sub foo
{
   println("$1 and $2");
}

foo("bar", "baz");

bar and baz
```

Sleep functions are first-class types. This means you can assign them to variables and pass them as arguments to functions. A script can refer to a named function with &functionName. Scripts also can use anonymous functions—anonymous functions? Yes. An anonymous function is a block of code enclosed in curly braces:

```
$var = { println("hi $1"); };

# call the function in $var
[$var: "mom"];

# call an anonymous function
[{ println("hi $1"); }: "dad"];

hi mom
hi dad
```

Sleep invokes functions and talks to Java through object expressions. An object expression encloses an object, an optional message and arguments in square brackets:

```
[$object message: arg1, arg2, ...];
```

The example below shows nested object expressions:

```
[[System out] println: "Hello World"];
```

which is equal to this Java statement:

```
System.out.println("Hello World");
```

When calling into Java, the message is the name of a method or field that belongs to the object. Arguments are converted to Java types as necessary, and some conversions are automatic. Nearly anything will convert to a string. However, a string will not convert to an int. Casting is possible, but I don't cover that topic here.

Now that you know a little about the Sleep language, it helps to see it in action. Next, I present several scenarios and Sleep-based solutions to them.

Filesystem Fun (the Biggest File)

My home directory has many files. I'm a digital pack rat, and I'm always low on disk space. I really have no idea what is on my disk. To help, I wrote a script to find the largest files within a directory and its subdirectories:

```
global('$size $file @files %sizes');

sub processFile
{
```

This script creates a data structure of files and their sizes, sorts it, and presents the results to the user. The &processFile function does most of the work, and it expects a file as an argument:

```
   if (-isDir $1)
      {
         filter(&processFile, ls($1));
      }
```

If the argument is a directory, the &ls function will provide the contents of the directory as an array. &filter expects a function and an array as arguments. &filter calls the function on each item in the array. I use &filter to call &processFile on the argument's subdirectories and files:

```
   else if (lof($1) > (1024 * 1024))
      {
         %sizes[$1] = lof($1);
      }
   }
```

The hash %sizes stores each filename and size. The key is the filename, and the size is the value. The &lof function returns the length of a file in bytes. I ignore files smaller than 1MB in size. I have so many files that this script exhausts the memory of Java before finishing. I could set Java to use a larger heap size with `java -Xmx1024M -jar sleep.jar`. Below, I chose to fix my script:

```
processFile(@ARGV[0]);
```

I call &processFile on the first command-line argument to kick off the script. When this function returns, the %sizes hash will contain an entry for each file in the specified directory and its subdirectories:

```
@files = sort({ return %sizes[$2] <=> %sizes[$1]; },
                                  keys(%sizes));
```

The &sort function processes the keys of %sizes and places them in order from largest to smallest size. Much like Perl, Sleep's &sort can use any criteria given by an anonymous function:

```
foreach $file (sublist(@files, 0, 50))
{
   $size = lof($file);
   println("$[20]size $file");
}
```

This script ends with a foreach loop to print out the 50 largest files.

And, lo and behold! I solved my problem. I found four copies of a Christmas movie I made on my Macintosh three years ago. Thanks to the script, I recovered several gigabytes of disk space.

Local Processes (PS. I Love You)

Recently, I had to watch this movie about a guy who sent letters to his wife after he passed away. I'm not really into the romantic-morbid genre; however, I thought I could show the people in my life how much I care about them. Here is a script that sends a random fortune to someone every 24 hours:

```
include("sendemail.sl");

while (1)
{
   sendemail($to => "rsmudge@gmail.com",
      $from => "raffi@hick.org",
      $subject => "P.S. I love you",
      $message => "This made me think of you:\n\n" .
                  join("\n", `fortune`)
   );

   # sleep for 24 hours
   sleep(24 * 60 * 60 * 1000);
}
```

I use `` `fortune` `` to execute the fortune command and collect its output into an array. Then, I combine this with the rest of the message body to make a thoughtful message. This script uses the $variable => value syntax to pass named arguments to &sendemail.

Backticks are one way to execute a process. I show the other way in the sendemail.sl code.

Sending E-Mail

I use the sendmail program to send e-mail. The sendemail.sl file contents are:

```
sub sendemail
{
   local('$handle');
   $handle = exec("/usr/sbin/sendmail -t $to");
```

Sleep executes processes with the &exec function. Scripts interact with processes as if they were files. As an aside, you can pass arguments with spaces to &exec. Use an array instead of a string. For example, `exec(@("/usr/sbin/sendmail", "-t", $to))` would work in this example:

```
   println($handle,
"TO: $to
FROM: $from
SUBJECT: $subject
$message");
```

Here, I send the e-mail message to the sendmail process over STDIN. Later in this article, I cover how to use Sleep for distributed tasks. *Don't* combine this e-mail example with that—I don't like spammers:

```
   closef($handle);
}
```

The last step is to close the handle. Having successfully automated my personal life, let's turn our attention to work matters.

Remote Processes (Automate SSH)

System administration is all about reaching out and touching everything. And, doing that requires automation. Sleep can automate SSH sessions with ease. Here is the &ssh_cmd function in action:

```
debug(7);

include("ssh.sl");

global('@output');

@output = ssh_cmd($user => "root",
                  $pass => "123456",
                  $host => "foo.example.com",
                  $command => "cat /etc/shadow");

printAll(@output);
```

This script authenticates to foo.example.com via SSH, executes `"cat /etc/shadow"`, and prints the result on the local machine. Before we go further, there is something you should know. Sleep doesn't have an &ssh_cmd function. We have to build it.

Adding SSH to Sleep

Perl has the CPAN for modules. Sleep scripts can take advantage of the Java class library to add functionality. Here, I walk you through the code for ssh.sl:

```
import com.trilead.ssh2.* from:
                          trilead-ssh2-build213.jar;
```

Sleep uses import to get access to classes in another package. Unlike Java, Sleep can import directly from a third-party Java archive file at runtime. This is useful for trying things out quickly. Here I use the Trilead SSH for Java library to add SSH to Sleep:

```
sub ssh_cmd
```

```
{
   local('$conn $sess $data $handle @data');

   # create a connection
   $conn = [new Connection: $host, 22];
   [$conn connect];
```

This code creates a new com.trilead.ssh2.Connection object. Next, I call the connect method on this object to set up an SSH connection:

```
   # authenticate
   [$conn authenticateWithPassword: $user, $pass];
```

Then, I call the authenticateWithPassword method on the connection. The Java library expects two string parameters. Sleep is smart enough to convert scalars to Java types as necessary:

```
   # execute the command
   $sess = [$conn openSession];
   [$sess execCommand: $command];
```

Here, I create an SSH session from the connection with the openSession method. This method returns a com.trilead.ssh2.Session object. Sleep places the object into a scalar variable. If you want to execute more than one command, create a session for each command as I've done here:

```
   # wire up a Sleep I/O handle for STDOUT
   $handle = [SleepUtils getIOHandle:
                  [$sess getStdout], $null];
```

The next thing to do is get the output from the session. Sleep has a class called SleepUtils with useful functionality. One of the methods constructs an I/O handle from Java input and output stream objects. Here, I made a readable I/O object from [$sess getStdout]. To write values, replace $null with the STDIN value for the session. This is available as [$sess getStdin]:

```
   # read output into an array
   @data = readAll($handle);
```

From this point, you can manipulate the remote process like any other handle. Below, I read the entire contents of the handle into the array @data:

```
   # close it all down
   closef($handle);
   [$sess close];
   [$conn close];

   return @data;
}
```

The last step is to close down the session and connection. The &ssh_cmd function returns the contents of @data.

Run This Example

To execute this code, create ssh.sl from the example above, download trilead-ssh2-build212.jar, and re-use the SSH automation code for your own purposes. Place all these files in the same directory. Then, type:

```
$ java -jar sleep.jar yourscript.sl
```

Distribute Tasks with Mobile Agents

Programs that move from computer to computer are mobile agents. Agent programming is a way of thinking about distributed computing. Some tasks fit very well into the mobile agent paradigm. For example, if you have to search all files in a network for some string, it makes no sense to download every single file and search it. It is much more efficient to move the search code to each computer and let the searching happen locally. Mobile agents make this possible.

Mobile agents also save you from the need to define a client and server protocol. You can place the entire interaction between two or more computers into a single function and let it start hopping around to complete the task.

So, what does a mobile agent look like? A mobile agent is a function that calls &move to relocate itself. Here is a syslog patrol agent. This agent patrols your network, checking the syslog dæmon on each box. If the dæmon is down, it tries to restart it. After each patrol, the agent starts over again:

```
debug(7);

include("agentlib.sl");
```

Before this script can do anything, I include the agent library file (I dissect this file in the next section):

```
sub syslog_patrol
{
   local('$host @computers @proc $handle');

   $handle = openf("computers.txt");
   @computers = readAll($handle);
   closef($handle);
```

The first task is to get a list of all computers. For this, I read in the contents of computers.txt. I assume each line has the hostname or IP address of a computer ready to receive my agents:

```
   $handle = $null;
```

When an agent moves, it takes its variables, call stack and program counter with it. Sleep has to serialize this data to move a function. Serialization is the process of converting data to bytes. Scripts cannot serialize I/O handles. To prevent a disaster, I set the handle to $null before moving:

```
   while (size(@computers) > 0)
      {
         $host = @computers[0];
```

The next task is to loop through each host. In this script, I use a list iteration approach. This approach removes the first item from @computers with each execution. @computers gets smaller and smaller until nothing is left. The item we want to work with always is at the front. I use list iteration here because foreach loops are not serializable:

```
      move($host);
```

This one function call is all it takes to relocate the agent. The statement after this function will execute from $host with its variables and state intact. In this example, I don't have any error handling. I assume the host is up and that the agent can move itself there. Error handling isn't hard to add, and the Sleep documentation provides more on this topic:

```
      @proc = filter({
              return iff("*syslogd" iswm $1);
          }, `ps ax`);
```

This code gets a list of all processes that match the wild card `"*syslogd*"`. &filter applies the anonymous function to each item in the array given by `` `ps ax` ``. And, &filter collects the non-$null return values of these operations and puts them into an array. This is Sleep's version of grep. I can use the size of the @proc array to check whether syslog is running:

```
      if (size(@proc) == 0)
      {
         chdir('/etc/rc.d/init.d');
         `./syslog start`;
      }
```

Here, I check whether syslog is running. To start it, I change directories, and execute the syslog dæmon:

```
      @computers = sublist(@computers, 1);
   }
```

The last step of the loop is to remove the first item from @computers. I use &ublist to do this:

```
   sendAgent($home, lambda($this, \$home));
}
```

At the end of the patrol, I send the agent back to the starting computer. I use &lambda to make a fresh copy of the agent function with no saved state. I pass the $home variable into the copy so it knows where to go when it restarts:

```
sendAgent(@ARGV[0], lambda(&syslog_patrol,
                             $home => @ARGV[0]));
```

This code launches the agent into the system. I assume @ARGV[0] is the hostname of the home system with the computers.txt file.

Adding Agent Support

It should be no surprise that Sleep doesn't have &move. Again, we have to build it. Isn't that half the fun? The agentlib.sl file has two functions: &move and &sendAgent:

```
inline move
{
   callcc lambda({
      sendAgent($host, $1);
   }, $host => $1);
}
```

&move is an inline function. An inline function executes with the parent's variable scope, and commands, such as return, callcc and yield affect the parent. They are useful for hiding flow control tricks made possible with callcc. callcc is like a goto. It pauses the current function and calls the specified anonymous function with the current function as an argument. A paused function resumes execution the next time a script calls it. So, why is this exciting to us? Sleep's paused functions are serializable. This means a script can write a paused function to a socket or a file:

```
sub sendAgent
{
   local('$handle');
   $handle = connect($1, 8888);
   writeObject($handle, $2);
   closef($handle);
}
```

For example, the &sendAgent function writes a paused function to a socket. This function expects a hostname and a function as arguments. It connects to the host with &connect, writes the function with &writeObject, and closes the handle. One piece of magic is missing. It makes no sense to send agents without receiving them.

Receiving Agents

Middleware is software that receives agents. It sits between the operating system and the agents. The following code makes up middleware.sl:

```
include("agentlib.sl");
```

The agent middleware must include the agentlib.sl file. This gives it and the agents it executes access to &sendAgent and &move:

```
while (1)
{
   local('$handle $agent');

   $handle = listen(8888, 0);
```

The middleware executes in an infinite loop listening for connections on port 8888. The &listen function waits for a new connection:

```
   $agent = readObject($handle);
   closef($handle);
```

The &readObject function reads an object in from a handle. Here, I assume I am reading a function from the handle:

```
   fork({ [$agent]; }, \$agent);
}
```

The last step is to execute the agent itself. &fork executes code in an isolated thread. I make the agent available in the thread by giving it to &fork. The code I use here executes the agent. When the thread starts, the agent resumes execution from where it left off.

Run This Example

To execute this example, place a copy of middleware.sl and agentlib.sl on each computer. Then, execute the middleware with:

```
$ java -jar sleep.jar middleware.sl
```

On the first computer, make a script with the &syslog_patrol agent. Create a computers.txt file that lists each IP address with the agent middleware. Then, run your script with:

```
$ java -jar sleep.jar syslog_agent.sl [local ip address]
```

Now you have a syslog agent patrolling your network. Don't you feel safe?

What's Next?

Sleep is a language for the Java platform built with the UNIX programming philosophy. Sleep allows you to use existing tools to create solutions to problems. I've shown you how to solve a few system administration problems with Sleep. These examples offer a starting point for you to use the language.

When evaluating a new language, I look for how easily I can bring in external functionality, solve a problem or two and process data. Sadly, I wasn't able to cover data parsing in this article. But, that's okay, Sleep supports all this stuff. You can read the documentation to get a feel for regular expressions, pack and unpack, and &parseDate.

To make the most of these examples, I recommend you run them. Links to the documentation and examples are available in the Resources section. Good luck, and enjoy the language.■

Raphael Mudge is an entrepreneur and computer scientist based out of Syracuse, New York. He also wrote Sleep. You can find links to his other work at www.hick.org/~raffi.

Resources

Examples from This Article: **sleep.dashnine.org/ljexamples.tgz**

The Sleep Home Page: **sleep.dashnine.org**

The Sleep 2.1 Manual: **www.amazon.com/dp/143822723X** or **sleep.dashnine.org/documentation.html**

Trilead SSH for Java: **www.trilead.com/Products/Trilead_SSH_for_Java**

THE FALCON PROGRAMMING LANGUAGE IN A NUTSHELL

Falcon is based on an open coding approach that seamlessly merges procedural, object-oriented, functional and message-oriented programming.

Giancarlo Niccolai

In late 2003, I had the problem of making business-critical decisions and performing maintenance actions in real time, analyzing data that was passing through the servers I was charged with controlling. Data throughput was on the order of thousands of messages per second, each of which was made of complex structures and possibly nested maps, whose size was measured in kilobytes.

The applications in charge of those controls were already almost complete, and they were heavily multithreaded by design. The only thing missing was the logic-processing engine. That would have been the perfect job for a scripting language, but the memory, CPU, threading, responsiveness and safety constraints seemed to be a hard match.

After testing the available solutions, I decided to try to solve the problem by writing a scripting language from the ground up, taking into consideration those design constraints. After the decision was made to move forward, useful items commonly found missing from other scripting languages were added to the design specification. So, Falcon mainly was designed from the beginning to meet the following requirements:

- Rapidly exchange (or use directly) complex data with C++.
- Play nice with applications (especially with MT applications) and provide them with ways to control the script execution dynamically.
- Provide several programming paradigms under the shroud of simple, common grammar.
- Provide native multilanguage (UTF) support.
- Provide a simple means to build script-driven applications, easily and efficiently integrated with third-party libraries.

As soon as I was able to script the applications that drove the initial development and meet these ambitious targets in terms of overall performance, I realized that Falcon may be something useful and interesting for others also, so I went open source.

The project is now reaching its final beta release phase, and Falcon has become both a standalone scripting language and a scripting engine that can drive even the most demanding applications.

The Falcon programming language now is included with many high-profile distributions, including Fedora, Ubuntu, Slackware, Gentoo and others. If your distribution doesn't include it yet, you can download it from **www.falconpl.org**, along with user and developer documentation.

Falcon currently is ported for Linux (32- and 64-bit), Win32 and Solaris (Intel). Older versions work on Mac OS X and FreeBSD. We are porting the newer version shortly, and a SPARC port also should be ready soon.

The Language

Falcon is an untyped language with EOL-separated statements and code structured into statement/end blocks. It supports integer math (64-bit) natively, including bit-field operators, floating-point math, string arrays, several types of dictionaries, lists and MemBuffers (shared memory areas), among other base types and system classes.

Morphologically, Falcon doesn't break established conventions, for example:

```
function sayHello()
  printl( "Hello world!")
end

// Main script:
sayHello()
```

You can run this script by saving it in a test file and feeding it into Falcon via stdin, or by launching it like this:

```
$ falcon <scriptname.fal> [parameters]
```

We place great emphasis on the multiparadigm model. Falcon is based on an open coding approach that seamlessly merges procedural, object-oriented, functional and message-oriented programming. We're also adding tabular programming, sort of a multilayer OOP, but we don't have the space to discuss that here. Each paradigm we support is generally a bit "personalized" to allow for more comfortable programming and easier mingling with other paradigms.

Falcon Procedural Programming

Falcon procedural programming is based on function declaration and variable parameters calls. For example:

```
function checkParameters( first, second, third )
  > "------ checkParameters -------"
  // ">" at line start is a short for printl
  if first
     > "First parameter: ", first
  end

  // ... and single line statements
  // can be shortened with ":"
  if second: > "Second parameter: ", second
  if third: > "Third parameter: ", third
  > "------------------------------"
end

// Main script:
checkParameters( "a" )
checkParameters( "b", 10 )
checkParameters( "c", 5.2, 0xFF )
```

You can use RTL functions to retrieve the actual parameters passed to functions (or methods). Values also can be passed by reference (or alias), and functions can have static blocks and variables:

```
function changer( param )
  // a static initialization block
  static
     > "Changer initialized."
     c = 0
  end

  c++
  param = "changed " + c.toString() + " times."
end

// Main script:
param = "original"
changer( param )
> param           // will be still original
changer( $param ) // "$" extracts a reference
> param           // will be changed
p = $param        // taking an alias...
changer( $param ) // and sending it
> p               // still referring "param"
```

Again, RTL functions can be used to determine whether a parameter was passed directly or by reference.

The strict directive forces the variables to be declared explicitly via the def keyword:

```
directive strict=on

def alpha = 10     // we really meant to declare alpha
test( alpha )      // call before declaration is allowed

function test( val )
  local = val * 2  // error: not declared with def!
  return local
end
```

Falcon has a powerful statement to traverse and modify sequences. The following example prints and modifies the values in a dictionary:

```
dict = [ "alpha" => 1,
  "beta" => 2,
  "gamma" => 3,
  "delta" => 4,
  "fi" => 5 ]

for key, value in dict
  // Before first, ">>" is a short for "print"
  forfirst: >> "The dictionary is: "

  // String expansion operator "@"
  >> @ "$key=$value"

  .= "touched"

  formiddle: >> ", "
  forlast: > "."
end

// see what's in the dictionary now:
inspect( dictionary )
```

Notice the string expansion operator in the above code. Falcon provides string expansion via naming variables and expressions and applying an explicit @ unary operator. String expansions can contain format specifiers, like `@ "$(varname:r5)"`, which right-justifies in five spaces, but a Format class also is provided to cache and use repeated formats.

Both user-defined collections and language sequences provide iterators that can be used to access the list traditionally. Functional operators such as map, filter and reduce also are provided.

Falcon Object-Oriented Programming

A Falcon script can define classes and instantiate objects from them, create singleton objects (with or without base classes) and apply transversal attributes to the instances. The provides keyword checks for properties being exposed by the instances:

```
// A class
class Something( initval1, initval2 )
  // Simple initialization can be done directly
  prop1 = initval1
  prop2 = nil

  // init takes the parameters of the class
  // and performs more complex initialization
  init
     self.prop2 = initval
     > "Initializer of class Something"
  end

  function showMe()
     > "Something says: ", self.prop1, "; ", self.prop2
  end
end

// A singleton instance.
object Alone
  function whoAmI()
     > "I am alone"
  end
end

// an instance
instance = Something( "one", "two" )
instance.showMe()

//"Alone" is already an instance
if Alone provides whoAmI
  Alone.whoAmI()
end
```

Falcon has a Basic Object Model (BOM), which is available in all the items. Objects and classes can override some methods. For example, passing an item to the > print operator causes its toString BOM method to be called, and that can be overridden as follows:

```
object different
  function toString()
     return "is different..."
  end
end

> "the object... ", different
```

Falcon supports multiple inheritance, but it disambiguates it by forcing inheritance initialization and priority, depending on the order of the inheritance declarations.

Classes also support static members that can be shared between objects of the same class and methods with static blocks that can work as class-wide initializers. Methods can be retrieved and also called directly from classes when they don't need to access the self object, providing the semantic of C++/Java/C# static methods.

It is possible to merge normal procedures with methods by assigning procedures to properties:

```
function call_me()
  if self and self provides my_name
     > self.my_name
  else
     > "Sorry, you didn't call me right."
  end
end

object test
  prop1 = nil
  my_name = "I am a test!"

  function hello()
     > "Hello world from ", self.my_name
  end
end

// normal calls
call_me()

// using the procedure as a method
test.prop1 = call_me
test.prop1()

// or a method as a procedure
proc = test.hello
test.my_name = "a renamed thing"

// see: proc will dynamically use the right "self"
proc()
```

Attributes

Attributes are binary properties that can be either present or not present for a specific instance or object, regardless of its class.

Attributes have a great expressive power, and in Falcon, they indicate what an object *is*, what it *has* and what it *belongs to*, depending on the context. For example, we can define a ready attribute that indicates the objects ready for elaboration:

```
// declaring an attribute "ready"
```

```
attributes: ready

class Data( name )
  name = name

  function process()
     > "Processing ", self.name, "..."
  end
end

// create 10 processors
processors = []
for i in [0:10]
  processors += Data(i)
  if i > 5: give ready to processors[i]
end

// work with the ready ones
for d in ready
  d.process()
end
```

RTL provides several functions to manipulate attributes.

The has and hasnt operators check for the presence of an attribute. For example:

```
attributes: ready
class SomeClass
  //... other class data ...
  // born ready!
  has ready
end

item = SomeClass()
if item has ready
  > "Item was born ready!"
end
```

Functional Programming

The base construct of Falcon functional programming is the callable sequence, also known as Sigma. At the moment, the only sequence supported is the array, but other types of sequences (such as lists) should be supported soon.

Basically, a Sigma is a delayed call that can work like this:

```
function test( a, b, c )
  > "Parameters:"
  > a
  > b
  > c
end

// direct
test( "one", "two", "three" )

// indirect
cached = [ test, "four", "five", "six" ]
cached()
```

The call respects the procedural paradigm (variable parameters), and the array is still a normal vector that can be accessed and modified through the standard language operators and RTL functions.

This delayed call is still not a full "functional context evaluation". The proper functional evaluation process is called Sigma reduction. It recursively resolves Sigmas from inner to outer and left to right when they are at the same level, substituting them with their return value.

Special functions known by the VM as Etas start and control functional evaluation; the simplest Eta function is eval(), which initializes and performs a basic Sigma reduction.

For example, the expression "(a+b) * (c+d)" can be written in a Lisp-like sequence:

```
function add( a, b ): return a+b
function mul( a, b ): return a*b

> "(2+3)*(4+5)= ", eval(.[mul .[add 2 3] .[add 4 5]])
```

The .[] notation is shorthand for array declarations whose elements are separated by white space instead of an explicit ",".

Falcon RTL comes with a rich set of Etas, such as iff (functional if), cascade (which joins more standard calls in a single sequence), floop and times (different styles of functional loops), map, filter, reduce and many others.

Functional sequences can be parameterized through closure and references. For example, the above example can be made parametric in this way:

```
// add and mul as before...
function evaluator( a, b, c, d )
  return .[eval .[mul .[add a b] .[add c d]]]
end

tor = evaluator( 2,3,4,5 )
> "(2+3)*(4+5)= ", tor()
```

Traditional functional operators, such as map, filter and reduce, are supported, but the out-of-band item system expands their functionality.

Out-of-band items are items marked with a special flag through the oob() function. Although they are normal items in every other aspect, this special mark indicates that they hold unexpected, special or somehow extraordinary value traveling through functional sequences. Although this is not a direct support for monadic calculus, monads can be implemented at the script (or binary module) level through this mechanism.

Falcon also supports Lambda expressions and nested functions.

We currently are working on some extensions to make Sigmas even more configurable—for example, parameter naming (similar to Lisp field naming) and access from the outside to the unbound variables used in the sequence.

Falcon functional programming merges with OOP, as Sigmas can be set as object properties, and object methods can be used as Kappas (Sigma-callable header symbols):

```
object SomeObj
  a_property = 10
```

```
   function myProp( value )
      return self.a_property * value
   end
end

> "5*10=", eval( .[SomeObj.myProp 5] )
```

Message-Oriented Programming

Because attributes are a very flexible means of declaring dynamic Boolean properties and a set of "similar" objects, we have used them as the main driver for message-oriented programming.

Basically, objects and instances with a certain attribute can receive messages built for that attribute's holders. The target objects will receive messages through a method named after the attribute.

The rest of the message-oriented programming support is built on this basic mechanism—message priority queues, automatic event dispatching, inter-agent messaging services and so on.

As a minimally meaningful sample would require 50–100 lines (messages are among many agents), we'll skip it here, and try to explain what's nice about message-oriented programming.

The main point is that you can summon remote execution in unknown objects willing to participate in the message without direct knowledge of them. Messages can carry anything, including methods or whole Sigma sequences for remote execution in foreign objects.

Messages don't even need to be point to point. The message receivers cooperatively can form a reply by adding something to the forming return value. For example, a central arbiter can send a "register" message, and every object willing to register can add itself to a queue of items willing to register in a queue traveling with the message. The queue even can contain target register procedures to be invoked by the arbiter once the register message processing is complete.

An example that easily displays the power of this paradigm is the implementation of an assert/retract/query mechanism.

A central object registering assertion listens for messages of these three types. Any part of the program then can send an assertion, a name bound with executable code, which can be anything, including code generated dynamically or loaded from plugins. Items in need of some algorithm can then query the system (sending a query message) asking for it to be provided.

If available, the code is returned, and it can be invoked by the agents in need of it.

You also can do this through a global dictionary, where code is associated with algorithm names, but that approach requires all users of the code to know the central dictionary and to interact with it. Asking a smoke cloud to take care of arbitrating the code repository is easier, simpler, more modular, more flexible and allows for central checking and managing. When that comes at no additional performance cost because of the language integration, it's an obvious advantage.

Some Things We Didn't Say

Stuffing all the things that Falcon can do for you into a short article is not easy, mainly because the things some will find useful may be useless for others. We didn't discuss co-routines, the indirect operator, the upcoming tabular programming, the reflexive compiler, the Falcon Template Document system (our active server pages), the multithreading module or many other things we've done and are doing to make Falcon the best language we can.

A DBI module already is available for interacting directly with MySQL, Postgre and SQLite3, and ODBC and Firebird will be ready soon too. A module for SDL is standing, and we're starting to work on a generic binding system to provide full support for Qt, GTK, GD2 and many other libraries.

We are still a small group, and the language specifications are still open. So, if this project interests you, and you want to add some binding or test some paradigm/language idea, we welcome you.■

The main point is that you can summon remote execution in unknown objects willing to participate in the message without direct knowledge of them.

Giancarlo Niccolai was born in Bologna, Italy, and he graduated in 1992 in IT at Pistoia. He currently works as IT designer and consultant for software providers of the most important financial institutions on the continent. He previously has worked with many open-source projects and consistently participates in the xHarbour (XBase compiler) Project. He has expertise in several programming languages and deep interests in natural languages and linguistic/physiology sciences.

TECH TIP Reset a Messed-Up Terminal

Ever perform a cat command on a binary file at the command line? Usually, you get a screen full of bizarre characters, and sometimes you end up with a terminal that's unusable. Rather than having to close the terminal and re-open it, just issue a reset command:

```
$ reset
```

And, all should go back to normal. Sometimes the carriage return may stop working in your terminal, in which case, you need to do the following to make reset work:

```
$ <LF>reset<LF>
```

<LF> is the line-feed character, normally Ctrl-J.

—FRED RICHARDS

THE INTERNET INDUSTRY EVENT

ISPCON

FALL 2008

GET REAL

SIGN UP ONLINE TODAY!

ISPCON is where the service provider industry goes to **GET REAL** about the future of their businesses. Isn't it time you:

- increase your ARPU?
- optimize your operations?
- pump up profits?
- discover what's next?

Whatever your plans, **ISPCON** will help you **GET REAL** about them. This is the industry's only forum where peers learn from peers in real one-on-one discussions about what works, what doesn't and what's next.

NOV. 11-13, 2008

McEnery San Jose Convention Center • San Jose, California

GET REAL

GET TO ISPCON!

WWW.ISPCON.COM

Get event updates and special offers via email! www.ispcon.com/optin

State of the Art: Linux Audio 2008, Part II

Evaluating the condition of sound and music production software.

DAVE PHILLIPS

In this second part of my survey of Linux audio development, I focus on the application side of things. I would have liked to have included many other tools and applications, but time and space always are in short supply. So, my apologies if your favorite program isn't listed; feel free to let me know what you think I'm missing.

Music Production

People coming to Linux from the Windows/Mac world of commercial sound and music software might think they've stepped backward in time. Linux audio and MIDI production software usually is not as visually attractive as the rainbow of products advertised in the major music magazines, but most musicians will agree that the sound is the thing. In that regard, Linux can stand tall and even can claim some colorful packages of its own.

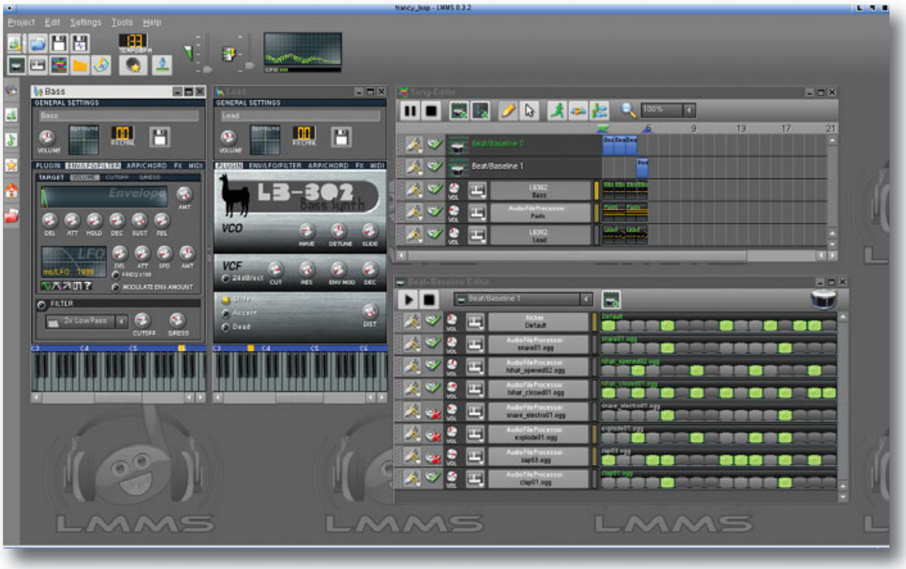


Figure 1. LMMS in Action

Simple Production

ALSA supplies command-line utilities for simple recording and playback of audio and MIDI. These tools (arecord/aplay and arecordmidi/aplaymidi) are useful for quick uncomplicated purposes, and most distributions provide GUIs to ease their use. At the next level, LMMS (Linux MultiMedia Studio; Figure 1) and Jokosher are good examples of desktop music production software designed in the manner of Apple's popular Garage Band. They engage the user quickly with colorful uncomplicated GUIs, but they are quite powerful within their design constraints. Both programs are in current development and have active communities of users and developers. Wouter Boeke's AMUC (Amsterdam Music Composer) is another less-weighty program that includes many attractions for the desktop composer, including an integrated synthesizer, notation capability and very light resource requirements.

Complex Production

Ardour dominates the professional-grade category of serious recording tools for Linux. Paul Davis continues to lead Ardour's programming team, and the project remains one of the finest examples of Linux audio software development. Ardour 2.5 is a mature application, and the developing Ardour 3.0 promises to bring the program to a new level, thanks especially to Dave Robillard's work on its new MIDI recording and editing capabilities. No strict timetable exists for Ardour's releases, and I certainly can't predict when 3.0 will make its public debut. However, Ardour's development track record is well defined, with a consistent series of releases, so I hope we may see it before year's end. Of course, SVN sources are available to anyone who wants to test the cutting edge while waiting for the public release.

Smaller but still powerful alternatives are available. Rui Nuno Capels' QTractor is a multitrack/multichannel DAW (digital audio workstation) with a design similar to the portable studios in the digital audio hardware world. QTractor also distinguishes itself by its support for natively compiled Linux VST plugins, along with the usual complement of LADSPA and DSSI plugins. Remon Sijrier's Traverso employs a highly efficient interface, is very easy to use and provides a complete production system, from recording your first tracks to burning an audio CD.

Kai Vehmanen's Ecasound occupies a unique position in the Linux audio software world. Ecasound is a command-line DAW, a complete audio recording and processing solution that requires no graphics displays. It runs in an interactive mode or can be driven by user-composed scripts; it is fully JACK-aware; it records in multichannel modes—the list of Ecasound's capabilities stretches on and on. Ecasound is a long-lived project, and I'm happy to report it's still developed and maintained by its original author.

Fervent Software's Rosegarden is another venerable Linux music application with a long and healthy development track. Rosegarden always has supported common-practice notation as a composer's interface, and its developers now plan to strengthen that interface further. Given its JACK support, there's little need for Rosegarden to repeat all the duties of a DAW, and it's a win for notation-based composers to have their notation-based GUI JACK-sync'd to the DAW of their choice.

Developer Werner Schweer has moved his MusE audio/MIDI sequencer in the opposite direction—he has removed its notation interface and refocused that code into the MuseScore program (see below). Meanwhile, MusE continues to evolve as a dedicated audio/MIDI sequencer, and version 1.0 is currently in alpha release.

Production Helpers

Consider the common studio scenario of a MIDI sequencer driving two or three softsynths whose output is directed into Ardour. When your work is done, you can save each application to its current state, but there's no easy way to recall every component to its session state upon re-opening the project and its parts. The LASH software provides an elegant solution to that problem, but its adoption has been slow. Client applications must include direct support for LASH, and so far, developers have been focused on other problems. Nevertheless, the project remains active, the client list grows, and I hope to see wider adoption of LASH throughout the Linux audio development community.

Mastering is a process normally associated with the post-production stage of a recording project. When mastering a project, track levels are balanced and the final touches of compression and EQ are applied to add that touch of audio perfection before burning the master disc. Fortunately, Linux can claim

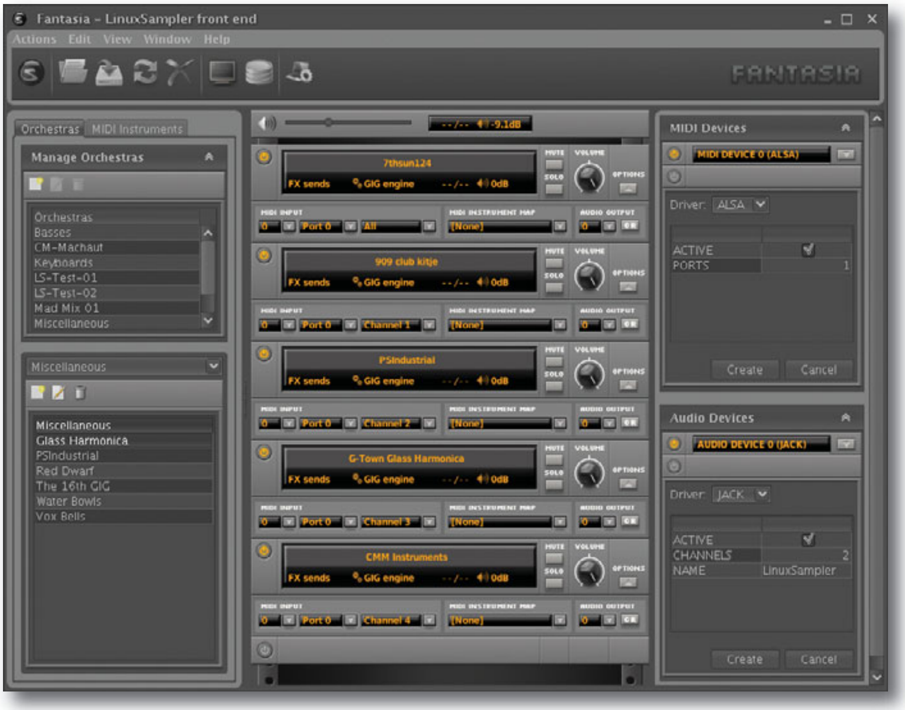


Figure 2. LinuxSampler Fantasia GUI

an excellent mastering utility, the JAMin program designed by Steve Harris and developed with help from a talented crew of Linux audio programmers. JAMin's last major release (0.95.0) dates from 2005, but the project already is mature and continues to show intermittent CVS activity.

Synthesizers and Samplers

Many older softsynth projects (amSynth, ALSA Modular Synth and ZynAddSubFX) are unmaintained and in need of attention. The synths mentioned above sound great, but they could all benefit from amenities, such as current compiler optimizations, LASH support, JACK support and so on.

Significant synths in current development include Ingen (LADSPA/LV2/DSSI plugin-based synth), QSynth (soundfont2 synthesizer) and FMS (modular synthesis). Recently, a new crop has appeared with some very unusual approaches to synthesis methods and GUI design. Malte Steiner's Minicomputer is a powerful subtractive synthesizer with eight monophonic "pages". Justin Smith's Synth Of Noise is a glitchmeister's dream synth, and Juan Pedro Bolivar Puente's Psychosynth presents a unique 3-D interface for creating basic (and not so basic) synthesis networks.

Samplers are represented by Specimen and the LinuxSampler Projects. These applications differ in some significant ways: LinuxSampler utilizes files in the GIG format made popular by Tascam's GigaSampler, and Specimen is happier with soundfile formats supported by libsndfile. LinuxSampler (Figure 2) is a client/server architecture with at least two GUIs and a command-line interface. Specimen is a standalone GTK-based application. LinuxSampler and Specimen both support JACK, but Specimen also supports ALSA and is a LASH-savvy application. LinuxSampler has more features associated with the GigaSampler model and is the more consistently maintained program, but both samplers are useful in the complete Linux music-maker's studio.

I also must mention Tapeutape, Florent Berthaut's MIDI-controllable "virtual sampler". Tapeutape has a rich set of features (including LASH support) and is designed especially for live performance, with or without a GUI. The latest version of the program is 0.0.5 from April 2007, but the author has indicated that he's still working on it, and an update should be released by the time this article is published.

Drum Machines

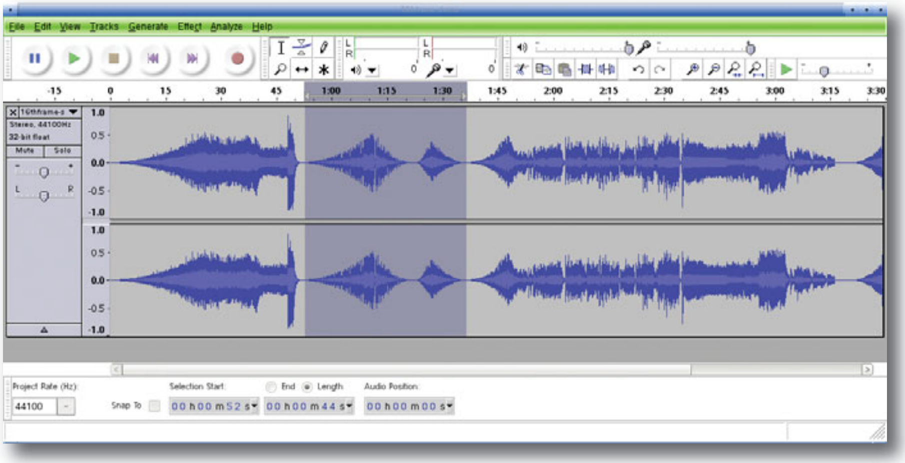
Hydrogen holds its position as the premier Linux drum machine/rhythm programmer. Its development track slowed for a while—version 0.9.3, the current stable release, dates from early 2006—but work proceeds on the SVN sources, and community support is active and strong. Version 0.9.4 promises great improvements—thanks especially to the new stewardship of Sebastian Moors and his development crew.

Samplers and soundfont players function nicely as drum sound sources in a MIDI sequencing environment, and drum loops have become a common method of composing rhythm tracks in the modern DAW. Given these factors, it's not surprising that few virtual drum boxes are created or maintained these days. However, the orDrumbox program has a number of interesting musical features and could be a worthy contender for Hydrogen, though it will need JACK support first.

Personal DSP/Guitar FX

Until recently, JACK Rack was the preferred standalone signal processing system for Linux audio production. That program has many features to recommend it, including access to the full range of LADSPA plugins and parameter control with MIDI continuous controllers. Alas, project development is slow, averaging two releases per year, and no release has been made yet in 2008.

Linux-based guitarists now have a very fine effects processing system with Rakarrack, a new system based on effects algorithms culled mainly from the ZynAddSubFX synthesizer. Version 0.2.0 is available now, and Rakarrack is in heavy development. Future releases will give Linux guitarists a more comprehensive instrument-specific effects system, including cabinet simulations and more effects.



Project Rate (Hz): 44100

Snap To □

Selection Start: 00 h 00 m 52 s

○ End ● Length: 00 h 00 m 44 s

Audio Position: 00 h 00 m 00 s

Figure 3. Audacity Soundfile Editor

Soundfile Editors

Many projects in this domain have strong development tracks. Bill Schottstaedt's great Snd continues to grow nicely, with many enhancements and fixes from its wide community of users and developers. Younger projects, such as Audacity (Figure 3), mhWaveEdit and Sweep, show current development, but unfortunately, the much-anticipated update for ReZound has yet to materialize, and we still await better JACK integration with Audacity and Sweep.

Audio Plugins

LADSPA, the Linux Audio Developer's Simple Plugin API, is an excellent resource for audio plugin developers, and users now can enjoy many fine plugins created with the LADSPA API. Standout sets include Tim Goetze's CAPS suite, Steve Harris' indispensable SWH package and Tom Szilagyi's TAPS collection, but many other LADSPA gems are available. The overall collection continues to expand, albeit slowly.

The intentional simplicity of the LADSPA API necessarily restricted plugin designs primarily to effects and dynamics processing. The emerging LV2 specification takes LADSPA to the next level, particularly with regard to instrument plugins. LV2 competes with the DSSI (Disposable SoftSynth Interface), but the developers of both projects are working toward the common goal of providing Linux with something like the famous VST/VSTi plugin architecture for Windows.

Direct support for VST/VSTi plugins currently exists in two forms. Bridges, such as FST (FreeVST) and the dssi-vst utility, can run some native Windows VST/VSTi plugins directly under Linux, while Lucio Asnaghi's JOST Project works at porting open-source VST plugins to native Linux versions. Applications with support for VST/VSTi plugins (Windows or native Linux) include Ardour, Rosegarden, LMMS and QTractor. However, Ardour's support requires a special build procedure, and the resulting binary may not be redistributed.

The terms of the Steinberg API forbid the free redistribution of the VST SDK, so a mature LV2 is likely to be an attractive alternative for plugin developers. Time will tell, and although the specification is already a worthy contender, users need plugins. A few projects already address that need (see the list at **lv2plug.in**), but more would be better.

The developers of LMMS have resolved the issue in another way by coding a drop-in replacement for the needed VST SDK, making it possible to provide direct VST support without the Steinberg code. This development is recent, and it remains to be seen whether Linux audio developers will incorporate that solution into their own programs.

Music Notation Software

This domain can be divided between programs that function primarily as a composer's workspace and programs that function as music typesetting software. The magnificent LilyPond Project dominates the music typesetting category, and NtEd and Canorus are the best currently maintained notation-based composition interfaces. However, Werner Schweer's MuseScore rapidly is evolving into a superb WYSIWYG graphic interface for music composition, but it requires a cutting-edge installation of Qt and its other dependencies.

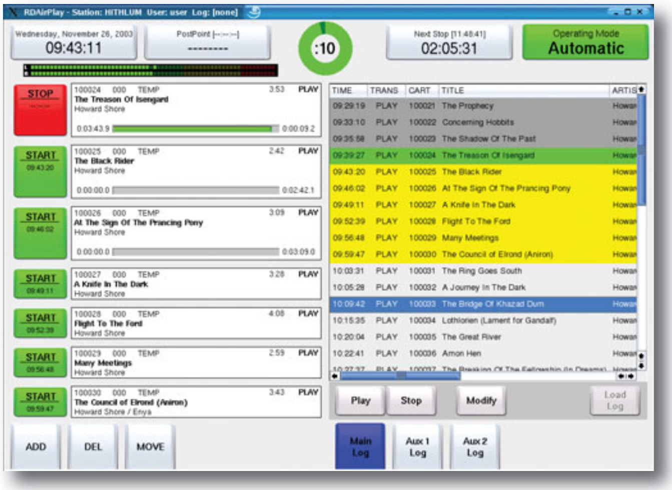
The Virtual DJ/VJ

The Linux digital DJ can choose between two professional-grade mixers, UltraMixer and Mixxx, both of which are beyond their 1.0 releases and continue to display strong development tracks. Alexander Koenig's great "virtual scratcher" terminatorX has not been developed since 2004, but at version 3.82, it's safe to refer to it as mature.

The digital video jockey (VJ) is well served by the current crop of video mixers for Linux. Outstanding packages include FLxER, FreeJ, Gephex and Veejay, all of which work with video files and streams in ways analogous to the actions of audio disc jockeys. Video input can be scratched, stuttered, processed with special effects, and mixed with other video (and other media). Common laptops now are powerful enough to handle the audio and video resource demands of this evolving art form, especially if they're running Linux.

Broadcasting Software

The Rivendell Project rules this domain. Rivendell (Figure 4) provides a complete solution for radio broadcasters (air-wave or network-based) who want to automate all or any part of their operations. The suite is an impressive achievement, with a fully professional set of features "...for the acquisition, management, scheduling and playout of audio content", according to its Web site. The latest public release is version 1.0, and the project development status is current and ongoing.



100024 000 TEMP 3:53 PLAY
The Treason Of Isengard
Howard Shore
0:03:43.9 0:00:09.2

09:43:20
100025 000 TEMP 2:42 PLAY
The Black Rider
Howard Shore
0:00:00.0 0:02:42.1

09:46:02
100026 000 TEMP 3:09 PLAY
At The Sign Of The Prancing Pony
Howard Shore
0:00:00.0 0:03:09.0

09:49:11
100027 000 TEMP 3:28 PLAY
A Knife In The Dark
Howard Shore

09:52:39
100028 000 TEMP 4:08 PLAY
Flight To The Ford
Howard Shore

09:56:48
100029 000 TEMP 2:59 PLAY
Many Meetings
Howard Shore

09:59:47
100030 000 TEMP 3:43 PLAY
The Council of Elrond (Aniron)
Howard Shore / Enya

TIME	TRANS	CART	TITLE	ARTIS
09:29:19	PLAY	100021	The Prophecy	Howar
09:33:10	PLAY	100022	Concerning Hobbits	Howar
09:35:58	PLAY	100023	The Shadow Of The Past	Howar
09:39:27	PLAY	100024	The Treason Of Isengard	Howar
09:43:20	PLAY	100025	The Black Rider	Howar
09:46:02	PLAY	100026	At The Sign Of The Prancing Pony	Howar
09:49:11	PLAY	100027	A Knife In The Dark	Howar
09:52:39	PLAY	100028	Flight To The Ford	Howar
09:56:48	PLAY	100029	Many Meetings	Howar
09:59:47	PLAY	100030	The Council of Elrond (Aniron)	Howar
10:03:31	PLAY	100031	The Ring Goes South	Howar
10:05:28	PLAY	100032	A Journey In The Dark	Howar
10:09:42	PLAY	100033	The Bridge Of Khazad Dum	Howar
10:15:35	PLAY	100034	Lothlorien (Lament for Gandalf)	Howar
10:20:04	PLAY	100035	The Great River	Howar
10:22:41	PLAY	100036	Amon Hen	Howar
10:27:37	PLAY	100037	The Breaking Of The Fellowship (In Dreams)	Howar

Figure 4. Rivendell's Air Play/Main Log Panel

Language-Based Software Sound Synthesis

Traditional software sound synthesis (SWSS) languages have flourished in Linux, and the platform continues to attract developers of such systems.

Csound enjoys the attentions of a wide community of users and a core development group of very talented programmers. The latest release, Csound 5.08, is a true powerhouse, with an amazing number of synthesis and processing opcodes, integrated GUI widgets, more complete JACK support and many other compelling features. The development of the Csound API has provided a mighty engine for programmers who want to leverage Csound's capabilities into their own software without having to rewrite its routines. Jean-Pierre Lemoine's AVSynthesis, Steven Yi's blue and Rory Walsh's Cabbage Project all depend on the Csound API for their audio processing functions.

Paul Lansky's venerable Cmix enjoyed continued development in the form of Dave Topper's superb RTCmix, but it seems that development has stalled since 2006. RTCmix definitely is worth getting into, and I hope that its development track will pick up again in the near future.

Bill Schottstaedt's Common Lisp Music (CLM) is another SWSS system derived ultimately from Max Mathew's legendary Music V. In fact, Bill recently incorporated Music V into CLM, but that's a trivial task for such a formidable developer. CLM has been in constant evolution for probably as long as Csound, and it enjoys the special attention of its own talented development crew. New releases are frequent and significant, typically adding new synthesis and processing functions along with such amenities as an amazing collection of bird-call synthesis routines and the aforementioned Music V. I also must mention Bill Schottstaedt's Common Music Notation (a Lisp-based music notation language) and his great Snd soundfile editor. All of his software is high quality and consistently maintained, and we are fortunate to have him and his work in the Linux audio camp.

Notable recent SWSS systems include ChucK, SuperCollider3 and the awesome Pure Data (Pd). Their modern characteristics include a more contemporary syntax and support for modern programming techniques, and in some cases, the language includes an integral (but not mandatory) GUI. ChucK and SuperCollider3 do not include integrated graphics primitives, but GUIs have been created for the language or for certain aspects of the language (for example, TAPESTREA, a fascinating tool for composition that requires ChucK's signal analysis and synthesis capabilities).

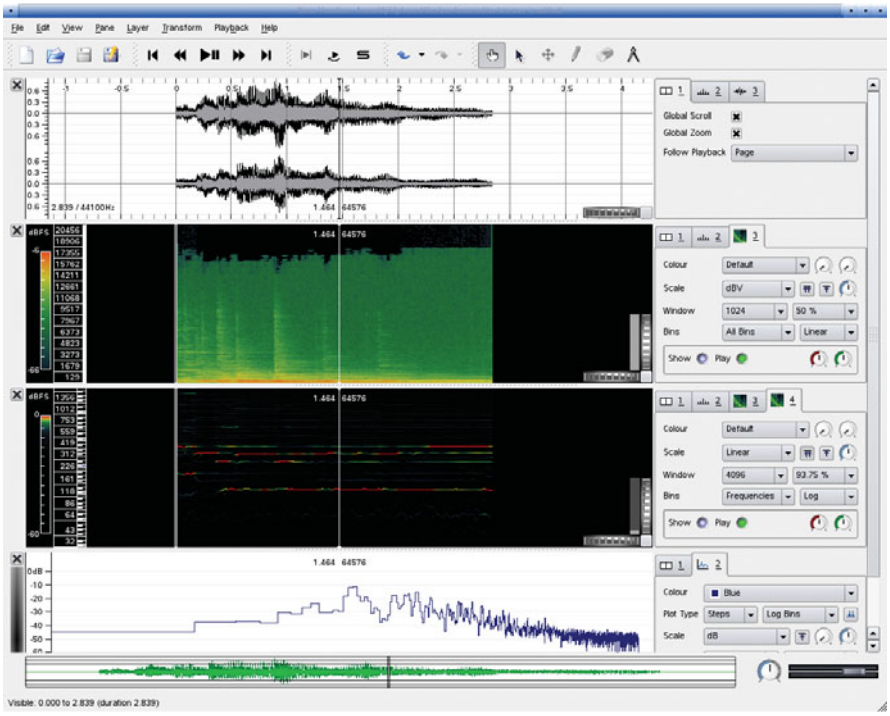
Pure Data deserves some further remarks. The systems I've mentioned here enjoy wide community support from users and developers, but Pd comes close to being a religion. It is mightily persuasive, with a variety of functions and routines that rival Csound, including a fantastic interface for working with OpenGL via the GEM library. Thanks to its vast resources (and excellent documentation), Pd can be pressed into virtually any audio, MIDI or video service.

Signal Processing/ Analysis/Resynthesis

In Ye Olden Times, the software found under this rubric would have included only language-based tools, but the scene has changed profoundly. The GUI is now the sound analyst's favored tool, and we can enjoy some wonderful software as a result of this focus on the user interface.

The award-winning CLAM Project continues along its innovative path, thanks to Pau Arumi and the development team at UPF in Barcelona. CLAM is the "C/C++ Library for Audio and Music", designed for rapid development of sound and music applications. The system includes unique tools and utilities for audio analysis, synthesis and signal processing, complete with graphic controls and displays.

GRAME's FAUST is both a language for real-time audio signal processing and a development environment for DSP programmers writing plugins or complete applications. FAUST is indeed fascinating software, with a strong development team and an excellent collection of tools and utilities. I plan to review FAUST in a future article for the



Global Scroll ☒
Global Zoom ☒
Follow Playback: Page

Colour: Default
Scale: dBV
Window: 1024 | 50 %
Bins: All Bins | Linear
Show ● Play ●

Colour: Default
Scale: Linear
Window: 4096 | 93.75 %
Bins: Frequencies | Log
Show ● Play ●

Colour: Blue
Plot Type: Steps | Log Bins
Scale: dB

Visible: 0.000 to 2.839 (duration 2.839)

Figure 5. Sonic Visualiser Analyzing a Musical Fragment

Linux Journal Web site.

Chris Cannam's Sonic Visualiser (Figure 5) is a program for "viewing and analyzing the contents of music audio files", but that description reveals little about the program itself. The project intends to provide the best audio visualisation software for viewing waveform and spectrographic data representations in forms that can be utilized and comprehended by anyone, not only audio processing professionals. However, Sonic Visualiser is no mere eye-candy maker; it is, indeed, a serious tool for studying music and sound.

Albert Graef's Pure (formerly Q) is not a DSP environment per se, but it is obvious from its examples that audio and MIDI applications are certainly among its major focus points. Additionally, Pure/Q includes some very cool methods for interfacing with the FAUST and Pd audio synthesis and processing environments.

Music Composition

Linux can claim one of the finest composition environments available to computer-based musicians, Rick Taube's Common Music. Professor Taube has maintained Common Music consistently for many years, and most recently, he has begun work on an entirely GUI-based environment (GraceCL) for the system. IRCAM's OpenMusic is another composition-centric program that will run under Linux, but unfortunately, it is maintained only sporadically.

Dedicated Distributions

Linux distributions with an emphasis on multimedia support have flourished in the past few years. Planet CCRMA, 64 Studio, JAD, Dynebolic and Musix have reduced the agonies that attend the configuration of a low-latency high-performance system. Some of those distributions include live disc images for "trying without crying", and other systems, such as Gentoo and Ubuntu, offer specialized versions of themselves optimized for audio work.

Remaining Difficulties

Despite the many advances in the Linux audio world, some irritating difficulties remain. The mainstream distributions have not yet agreed upon a common sound server, and they may never do so. Hardware support still is disappointing, especially in the pro-audio domain, and licensing issues continue to plague some projects. Nevertheless, many difficulties have been ameliorated or done away with entirely, as developers continue to work toward greater usability on the Linux desktop.

Conclusions

I began working with Linux in 1995, when only a few dozen decent audio/MIDI applications existed for Linux. I'm happy that we now have such a cornucopia of programs, despite their varying quality, and I see good signs indicating continuance of many of those programs. Obvious

Resources

This list includes only the programs referenced in the article text. More Linux sound and MIDI applications are listed in the Linuxaudio.org index of applications at **apps.linuxaudio.org**.

MUSIC PRODUCTION

ALSA: **www.alsa-project.org**
LMMS (Linux MultiMedia Studio): **lmms.sourceforge.net**
Ardour: **ardour.org**
QTractor: **qtractor.sourceforge.net/qtractor-index.html**
Traverso: **traverso-daw.org**
Ecasound: **www.eca.cx/ecasound**
Rosegarden: **www.rosegardenmusic.com**
MusE: **muse-sequencer.org**
LASH: **lash.nongnu.org**
JAMin: **jamin.sourceforge.net**

SYNTHESIZERS AND SAMPLERS

amSynth: **amsynthe.sourceforge.net**
ALSA Modular Synth: **alsamodular.sourceforge.net**
ZynAddSubFX: **zynaddsubfx.sourceforge.net**
Ingen: **wiki.drobilla.net/Ingen**
QSynth: **qsynth.sourceforge.net/qsynth-index.html**
FMS: **fmsynth.sourceforge.net**
Minicomputer: **minicomputer.sourceforge.net**
Synth Of Noise: **code.google.com/p/noisesmith-linux-audio**
Psychosynth: **www.psychosynth.com/doku.php**
Specimen: **zhevny.com/specimen**
LinuxSampler: **www.linuxsampler.org**
Tapeutape: **www.tardigrade-inc.com/Tapeutape**

DRUM MACHINES

Hydrogen: **www.hydrogen-music.org**
orDrumbox: **ordrumbox.sourceforge.net**

PERSONAL DSP/GUITAR FX

JACK Rack: **jack-rack.sourceforge.net**
Rakarrack: **rakarrack.sourceforge.net**

SOUNDFILE EDITORS

Snd: **www-ccrma.stanford.edu/software/snd**
Audacity: **audacity.sourceforge.net**
mhWaveEdit: **https://gna.org/projects/mhwaveedit**
Sweep: **www.metadecks.org/software/sweep/index.html**
Rezound: **rezound.sourceforge.net**

AUDIO PLUGINS

LADSPA: **www.ladspa.org**
LV2: **lv2plug.in**
DSSI (Disposable SoftSynth Interface): **dssi.sourceforge.net**
FST (FreeVST): **joebutton.co.uk/fst**
JOST: **www.anticore.org/jucetice**

MUSIC NOTATION SOFTWARE

LilyPond: **www.lilypond.org**
NtEd: **vsr.informatik.tu-chemnitz.de/staff/jan/nted/nted.xhtml**
Canorus: **canorus.berlios.de**
MuseScore: **mscore.sourceforge.net**

THE VIRTUAL DJ/VJ

UltraMixer: **www.ultramixer.com**
Mixxx: **www.mixxx.org**
terminatorX: **terminatorx.org**
FLxER: **www.flxer.net/software**
FreeJ: **freej.dyne.org**
Gephex: **www.gephex.org**
Veejay: **www.veejayhq.net**

BROADCASTING SOFTWARE

Rivendell: **www.rivendellaudio.org**

LANGUAGE-BASED SOFTWARE SOUND SYNTHESIS

Csound: **www.csounds.com**
RTCmix: **rtcmix.org**
Common Lisp Music (CLM): **ccrma-www.stanford.edu/CCRMA/Software/clm/clm.html**
ChucK: **chuck.cs.princeton.edu**
SuperCollider3: **supercollider.sourceforge.net**
Pure Data (Pd): **puredata.info**

SIGNAL PROCESSING/ANALYSIS/RESYNTHESIS

CLAM: **clam.iua.upf.edu**
FAUST: **faust.grame.fr**
Sonic Visualiser: **www.sonicvisualiser.org**
Pure: **pure-lang.sourceforge.net**

MUSIC COMPOSITION

Common Music: **www-ccrma.stanford.edu/software/cm/doc/cm.html**
and OpenMusic: **freesoftware.ircam.fr**

DEDICATED DISTRIBUTIONS

Planet CCRMA: **ccrma.stanford.edu/planetccrma/software**
64 Studio: **64studio.com**
JAD: **jacklab.net**
Dynebolic: **dynebolic.org**
Musix: **www.musix.org.ar**
Gentoo: **proaudio.tuxfamily.org/wiki**
Ubuntu Studio: **ubuntustudio.org**

targets for improvement include more pervasive support for JACK and the LASH session handler, standardization of the preferred sound server for normal users, and more direct driver support from hardware manufacturers. Some changes will come easily, and some will be troublesome, but it's in the nature of Linux to confront and conquer such difficulties. Meanwhile, I'm using Linux to produce my own media creations and enjoy them along with the works (commercial and otherwise) of others. Good things are happening around me now, and I see more good things coming down the road. Whatever they may be, I'll be sure to let you know about them here in the pages of *Linux Journal* and on LinuxJournal.com.■

Dave Phillips is a professional musician and writer living in Findlay, Ohio. He's been using Linux since the mid-1990s and was one of the original founders of the Linux Audio Developers group. He is the author of *The Book of Linux Music & Sound* (No Starch Press, 2000) and has written many articles on Linux music and sound issues for various journals and on-line news sites. When he isn't playing with light and sound, he enjoys reading Latin literature, practicing t'ai chi, chasing shar-pei puppies and spending time with his beloved Ivy.

release date: february 2008
LINUX JOURNAL
ARCHIVE 1994-2007
Includes issues 1–164 of *Linux Journal*

The Well-Tempered PHP Developer

Eclipse, with some plugins added to the mix, provides a full environment for PHP developers.

FEDERICO KEREKI

Almost 300 years ago, in 1722, Johann Sebastian Bach wrote a book (actually, two volumes) of preludes and fugues in all major and minor keys, "for the profit and use of musical youth desirous of learning, and especially for the pastime of those already skilled in this study". This work, *The Well-Tempered Clavier*, was intended to demonstrate the ability of a single instrument to play in all keys. In the same vein, in this article, we explore a set of tools, widely available on Linux systems, for a well-rounded PHP developer.

Since its inception in 1995, PHP has grown a lot. The current stable version (as of May 2008) is 5.2.6, and version 6 is in the works. You confidently can say that PHP is currently used for millions of Web sites, on millions of servers, and it's probably the most popular Apache module, outdistancing all other Web scripting languages. Apart from being used to generate dynamic Web pages, it also can be used for command-line work (I have used PHP for text file processing, in order to upload data to a database) or for server-side scripting, providing Web services and other functions. So, it's a safe bet that you can use PHP for practically anything you might need.

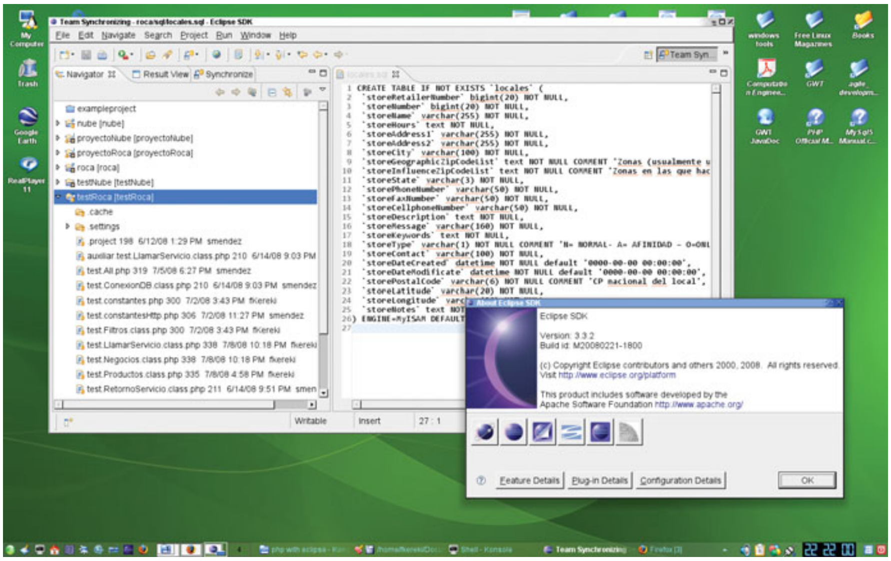
However, most PHP developers use only a few tools for development. In true code-hacking style, they employ a text editor, usually vi or emacs, and the barest programming and debugging aids. It should be no surprise that there are several (plenty) available tools that can help produce better tested and debugged code, whether you're working on your own or as a part of a team. In this article, we examine such a setup, based on Eclipse and several interesting plugins. Of course, this shouldn't be taken as the only way of doing things, and if you look around, you'll easily find other IDEs (Integrated Development Environments) and tools. This article is intended to be a nudge in one direction, rather than a mandated road to follow.

What Is Eclipse?

Eclipse is an integrated, extensible, development platform or environment. Originally, it was called VisualAge and was created for Java development (mostly written in Java itself), but it was renamed and then extended with additional plugins, so it can be used with many more programming languages and development tools—UML diagram creation and DB management are just two examples.

Although originally an IBM project, since 2003, Eclipse has been governed by the Eclipse Foundation, which adds several well-known technology companies as strategic members. The future of Eclipse doesn't depend on a single company. Eclipse is available under an open-source software license (not the GPL, but similar), and it eventually might use GPL version 3. The current version of Eclipse (3.4, also known as Ganymede) reportedly includes more than 18 million lines of code.

Thanks to its Java origins, Eclipse runs not only on Linux, but also on other operating systems, which is good for developers who target more than a single machine. Internationalization aspects are taken care of, and there are translations for several languages. Finally, the integration aspect of Eclipse is very important. You can do all your development (including not only code writing, but also testing, debugging, documentation writing, version control management and more)



```
exampleproject
nube [nube]
proyectoNube [proyectoNube]
proyectoRoca [proyectoRoca]
roca [roca]
testNube [testNube]
testRoca [testRoca]
  .cache
  .settings
  .project 198  6/12/08 1:29 PM  smendez
  auxiliar.test.LlamarServicio.class.php 210  6/14/08 9:03 PM
  test.All.php 319  7/5/08 6:27 PM  smendez
  test.ConexionDB.class.php 210  6/14/08 9:03 PM  smendez
  test.constantes.php 300  7/2/08 3:43 PM  fKereki
  test.constantesHttp.php 306  7/2/08 11:27 PM  smendez
  test.Filtros.class.php 300  7/2/08 3:43 PM  fKereki
  test.LlamarServicio.class.php 338  7/8/08 10:18 PM  fkereki
  test.Negocios.class.php 338  7/8/08 10:18 PM  fkereki
  test.Productos.class.php 335  7/8/08 4:58 PM  fkereki
  test.RetornoServicio.class.php 211  6/14/08 9:51 PM  smen
```

```
CREATE TABLE IF NOT EXISTS `locales` (
 `storeRetailerNumber` bigint(20) NOT NULL,
 `storeNumber` bigint(20) NOT NULL,
 `storeName` varchar(255) NOT NULL,
 `storeHours` text NOT NULL,
 `storeAddress1` varchar(255) NOT NULL,
 `storeAddress2` varchar(255) NOT NULL,
 `storeCity` varchar(100) NOT NULL,
 `storeGeographicZipCodeList` text NOT NULL COMMENT 'Zonas (usualmente u
 `storeInfluenceZipCodeList` text NOT NULL COMMENT 'Zonas en las que hac
 `storeState` varchar(3) NOT NULL,
 `storePhoneNumber` varchar(50) NOT NULL,
 `storeFaxNumber` varchar(50) NOT NULL,
 `storeCellphoneNumber` varchar(50) NOT NULL,
 `storeDescription` text NOT NULL,
 `storeMessage` varchar(160) NOT NULL,
 `storeKeywords` text NOT NULL,
 `storeType` varchar(1) NOT NULL COMMENT 'N= NORMAL- A= AFINIDAD - O=ONL
 `storeContact` varchar(100) NOT NULL,
 `storeDateCreated` datetime NOT NULL default '0000-00-00 00:00:00',
 `storeDateModificate` datetime NOT NULL default '0000-00-00 00:00:00',
 `storePostalCode` varchar(6) NOT NULL COMMENT 'CP nacional del local',
 `storeLatitude` varchar(20) NOT NULL,
 `storeLongitude` varc
 `storeNotes` text NOT
) ENGINE=MyISAM DEFAULT
```

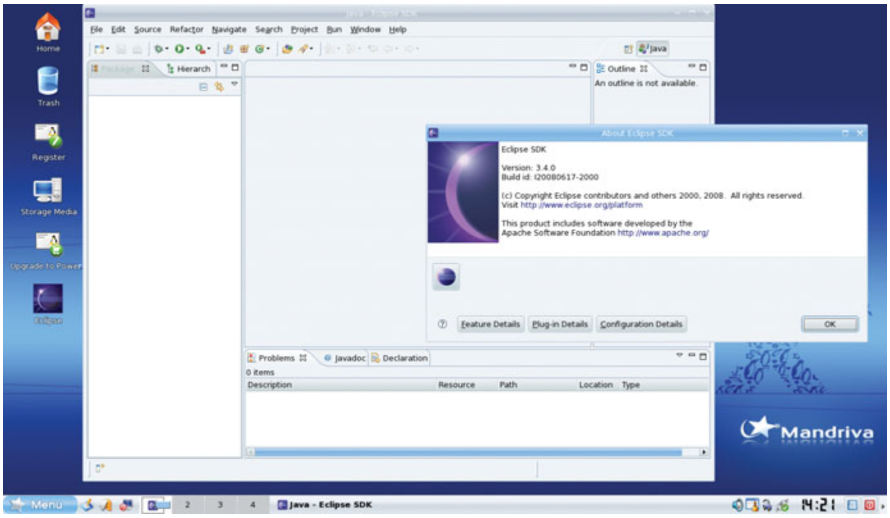
Eclipse SDK

Version: 3.3.2
Build id: M20080221-1800

(c) Copyright Eclipse contributors and others 2000, 2008. All rights reserved.
Visit http://www.eclipse.org/platform

This product includes software developed by the
Apache Software Foundation http://www.apache.org/

Figure 1. Eclipse Europa on OpenSUSE 10.3—all the plugins mentioned in this article were tested in this environment.



About Eclipse SDK

Eclipse SDK

Version: 3.4.0
Build id: I20080617-2000

(c) Copyright Eclipse contributors and others 2000, 2008. All rights reserved.
Visit http://www.eclipse.org/platform

This product includes software developed by the
Apache Software Foundation http://www.apache.org/

Feature Details | Plug-in Details | Configuration Details | OK

Figure 2. The just-released Ganymede on Mandriva 2008—be sure to do some tests before switching over to it.

from within a single program, with a common interface and style.

Starting in 2006, there has been a Simultaneous Release each year, covering not only the base Eclipse package, but also many other Eclipse-related projects. This is provided as a convenience, and it certainly helps avoid compatibility problems. The packages are named after the moons of Jupiter. In 2006, it was called Callisto. The 2007 version was Europa. And, in June 2008, as I'm writing this article, Ganymede has just been released. In this article, we use both Europa (Figure 1) and Ganymede (Figure 2) with an emphasis on the former.

I won't cover how to install PHP, Apache or related tools, but I do cover how to install Eclipse. Because of Eclipse's Java origins, first you need to get the Java Runtime Environment (JRE), although it's quite likely you already have it. I used the Sun 1.6.0 version, which already was installed. You could try using the IcedTea 1.7.0 version, but I cannot attest to its Eclipse (or other plugins) suitability. According to the Eclipse documentation, Java 1.5 should be good enough.

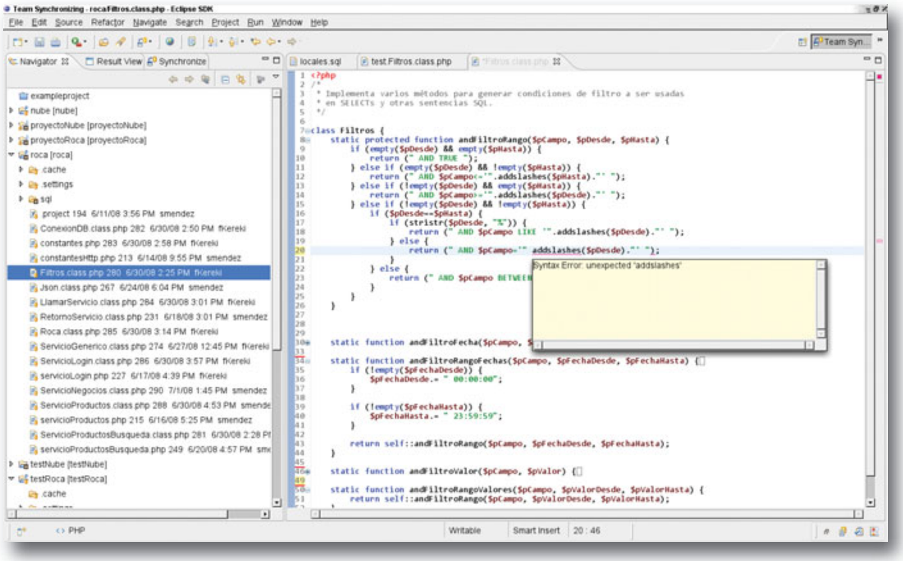
Getting Eclipse isn't difficult. Most distributions already include it, and you don't even need to visit the Eclipse Web site to download it, but it's likely you won't have the latest release. Go the Eclipse download site, choose the Eclipse Classic Project (version 3.4), and because the whole package weighs in at more than 150MB, select a close mirror. After the process is done, go to the directory where you downloaded the file, and do a `tar zxf eclipse-SDK-3.4-linux-gtk.tar.gz`. An eclipse directory will be created, and if you move to it and type `./eclipse`, Eclipse will be up and running.

PHPEclipse

The first plugin you will need for serious PHP work is named, appropriately enough, PHPEclipse. PHPEclipse has been around since 2002, and although the current stable version (1.1.8) is from 2006, there is work currently on version 1.2, and there has been a steady flow of updates, so the project still is quite alive. Getting PHPEclipse is easy; simply use the Eclipse update method, and add a new remote site (see Resources).

PHPEclipse provides not only basic editing facilities, but it also adds syntax coloring and bracket matching for easier reading; code folding, so you can hide a block or function; parameter hints and tooltips—for example, if you don't remember the parameters for the stristr() function, a little pop-up will remind you; and syntax checks (if you make a syntax error, you will get a wavy red underline at the place of the error and pop-up help, Figure 3). PHPEclipse also offers debugging (with either XDebug or DBG) and version control (CSV or SVN)—more on this below.

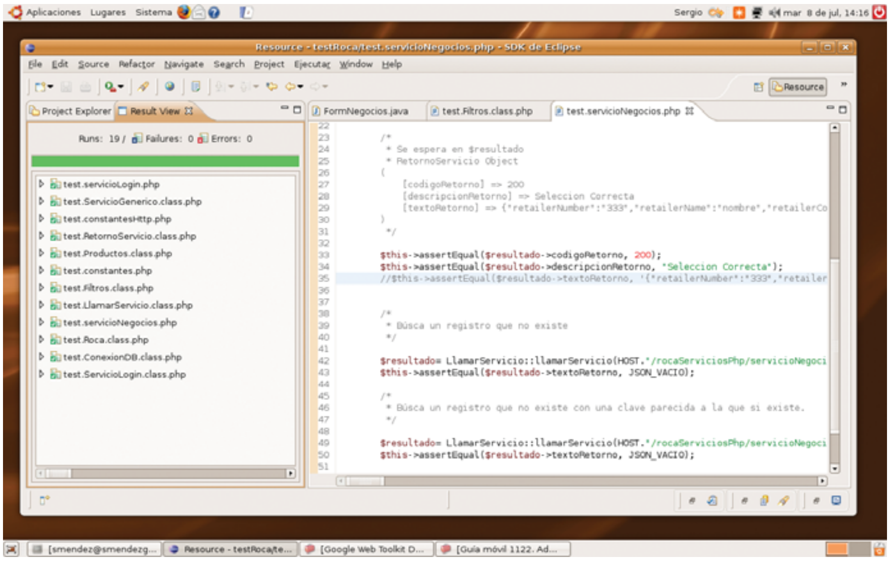
When you edit a PHP source file, several shortcuts and functions can speed you along. Need to find the declaration for a certain function or variable? Right-click on any reference



```
<?php
/*
 * Implementa varios métodos para generar condiciones de filtro a ser usadas
 * en SELECTs y otras sentencias SQL.
 */

class Filtros {
    static protected function andFiltroRango($pCampo, $pDesde, $pHasta) {
        if (empty($pDesde) && empty($pHasta)) {
            return (" AND TRUE ");
        } else if (empty($pDesde) && !empty($pHasta)) {
            return (" AND $pCampo<='".addslashes($pHasta)."' ");
        } else if (!empty($pDesde) && empty($pHasta)) {
            return (" AND $pCampo>='".addslashes($pDesde)."' ");
        } else if (!empty($pDesde) && !empty($pHasta)) {
            if ($pDesde==$pHasta) {
                if (stristr($pDesde, "%")) {
                    return (" AND $pCampo LIKE '".addslashes($pDesde)."' ");
                } else {
                    return (" AND $pCampo='" addslashes($pDesde)."' ");
                }
            } else {
                return (" AND $pCampo BETWEEN
            }
        }
    }



    static function andFiltroFecha($pCampo, $

    static function andFiltroRangoFechas($pCampo, $pFechaDesde, $pFechaHasta) {
        if (!empty($pFechaDesde)) {
            $pFechaDesde.= " 00:00:00";
        }

        if (!empty($pFechaHasta)) {
            $pFechaHasta.= " 23:59:59";
        }

        return self::andFiltroRango($pCampo, $pFechaDesde, $pFechaHasta);
    }

    static function andFiltroValor($pCampo, $pValor) {

    static function andFiltroRangoValores($pCampo, $pValorDesde, $pValorHasta) {
        return self::andFiltroRango($pCampo, $pValorDesde, $pValorHasta);
```

Syntax Error: unexpected 'addslashes'

Figure 3. Errors are highlighted immediately. Folding routines can help you see only the relevant code on-screen and hide the rest.

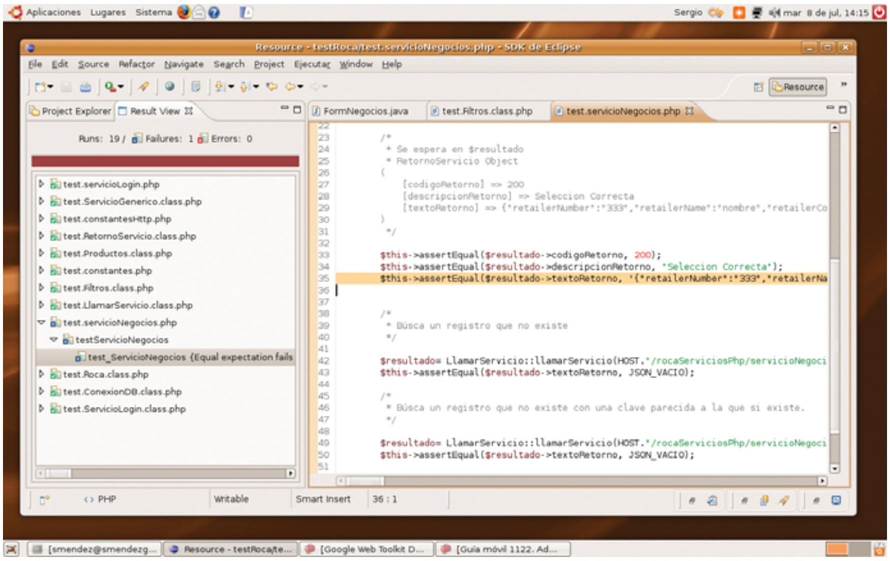

Aplicaciones Lugares Sistema
Sergio mar 8 de jul, 14:16
Resource - testRoca/test.servicioNegocios.php - SDK de Eclipse
File Edit Source Refactor Navigate Search Project Ejecutar Window Help
Resource
Project Explorer
Result View
FormNegocios.java
test.Filtros.class.php
test.servicioNegocios.php
[smendez@smendezg...
Resource - testRoca/te...
[Google Web Toolkit D...
[Guía móvil 1122. Ad...

Runs: 19 / Failures: 0 Errors: 0

- test.servicioLogin.php
- test.ServicioGenerico.class.php
- test.constantesHttp.php
- test.RetornoServicio.class.php
- test.Productos.class.php
- test.constantes.php
- test.Filtros.class.php
- test.LlamarServicio.class.php
- test.servicioNegocios.php
- test.Roca.class.php
- test.ConexionDB.class.php
- test.ServicioLogin.class.php

```

        /*
         * Se espera en $resultado
         * RetornoServicio Object
        (
            [codigoRetorno] => 200
            [descripcionRetorno] => Seleccion Correcta
            [textoRetorno] => {"retailerNumber":"333","retailerName":"nombre","retailerCo
        )
         */

        $this->assertEqual($resultado->codigoRetorno, 200);
        $this->assertEqual($resultado->descripcionRetorno, "Seleccion Correcta");
        //$this->assertEqual($resultado->textoRetorno, '{"retailerNumber":"333","retailer


        /*
         * Búsca un registro que no existe
         */

        $resultado= LlamarServicio::llamarServicio(HOST."/rocaServiciosPhp/servicioNegoci
        $this->assertEqual($resultado->textoRetorno, JSON_VACIO);

        /*
         * Búsca un registro que no existe con una clave parecida a la que si existe.
         */

        $resultado= LlamarServicio::llamarServicio(HOST."/rocaServiciosPhp/servicioNegoci
        $this->assertEqual($resultado->textoRetorno, JSON_VACIO);

```

Figure 4. A green bar shows all tests ran as expected.


Aplicaciones Lugares Sistema
Sergio mar 8 de jul, 14:15
Resource - testRoca/test.servicioNegocios.php - SDK de Eclipse
File Edit Source Refactor Navigate Search Project Ejecutar Window Help
Resource
Project Explorer
Result View
FormNegocios.java
test.Filtros.class.php
test.servicioNegocios.php
PHP
Writable
Smart Insert
36 : 1
[smendez@smendezg...
Resource - testRoca/te...
[Google Web Toolkit D...
[Guía móvil 1122. Ad...

Runs: 19 / Failures: 1 Errors: 0

- test.servicioLogin.php
- test.ServicioGenerico.class.php
- test.constantesHttp.php
- test.RetornoServicio.class.php
- test.Productos.class.php
- test.constantes.php
- test.Filtros.class.php
- test.LlamarServicio.class.php
- test.servicioNegocios.php
  - testServicioNegocios
    - test_ServicioNegocios {Equal expectation fails
- test.Roca.class.php
- test.ConexionDB.class.php
- test.ServicioLogin.class.php

```

        /*
         * Se espera en $resultado
         * RetornoServicio Object
        (
            [codigoRetorno] => 200
            [descripcionRetorno] => Seleccion Correcta
            [textoRetorno] => {"retailerNumber":"333","retailerName":"nombre","retailerCo
        )
         */

        $this->assertEqual($resultado->codigoRetorno, 200);
        $this->assertEqual($resultado->descripcionRetorno, "Seleccion Correcta");
        $this->assertEqual($resultado->textoRetorno, '{"retailerNumber":"333","retailerNa


        /*
         * Búsca un registro que no existe
         */

        $resultado= LlamarServicio::llamarServicio(HOST."/rocaServiciosPhp/servicioNegoci
        $this->assertEqual($resultado->textoRetorno, JSON_VACIO);

        /*
         * Búsca un registro que no existe con una clave parecida a la que si existe.
         */

        $resultado= LlamarServicio::llamarServicio(HOST."/rocaServiciosPhp/servicioNegoci
        $this->assertEqual($resultado->textoRetorno, JSON_VACIO);

```

Figure 5. A red bar means something's wrong; clicking on the problematic test takes you to the offending code.

to it (or press F3), and you will be taken there. If you are unsure about a certain PHP function, pressing Shift-F2 produces a manual—although you usually can get by with hovering the mouse over the function name.

For more prolix coding, there are several formatting functions. You can re-indent any portion of code simply by selecting it, then right-clicking and choosing Format, or by pressing Ctrl-Shift-F. You can turn lines into comments (and vice versa) by right-clicking and choosing Source→Toggle Comment, and all selected lines will get // added in front. (From now on, I skip the shortcuts; unless you are a die-hard Ctrl and Shift fanatic, you probably will use the mouse menus all the time.) Adding or removing larger comments (for example, ones like `/* ... */`) also is simple with a right-click, then selecting Source→Add Block Comment or Remove Block Comment. A Refactor function can help you change a variable or function name globally; there's no excuse for shoddy names anymore.

PHPEclipse is fully configurable. On the main menu, go to Window→Preferences, and select PHP. You can set your own specific preferences for most of the features I've covered (and even more that I didn't touch on here), so you can set up project standards. If more people are working on the project though, make sure everybody uses the same set of parameters. It's no fun having to reformat other people's code just because of a tabbing configuration difference. Collaboration and version control plugins are discussed below.

When your code is ready, you get Run As..., Debug As... and Profile As... commands. You can create profiles (including runtime parameters, environment variables, directories and more) and use them later with a single click. The results of the run will appear on a console, integrated within Eclipse.

Testing

When do you test your code? After everything is done? How quaint and old fashioned! Modern development methodologies suggest an iterative way of working, which combine developing automated test cases even before the actual programming is done. Having the tests available before actual development starts ensures quick feedback after any change, and it also provides design-level documentation, for each test serves as an example of what the code should do. Even more important, putting all the tests together in a test suite provides for regression testing—before any new code is committed, all pre-existing tests should pass. If a programmer makes any mistakes, changing the way a function should have worked, a well-designed test will catch the problem and alert you.

This way of programming has been named test-driven development (TDD) and is a part of many modern agile development techniques. The basic idea is simply preparing an automated test (automated means it can be run on its own, without users having to do anything, which implies that running the same tests several times a day is no chore) that exercises your code and tests the results it produces by checking assertions that are either true or false. If any assertion fails, some piece of code isn't doing its expected thing. Writing (or at least planning) the tests before writing code, makes the developer pay attention to code requirements and modularity—two important quality factors.

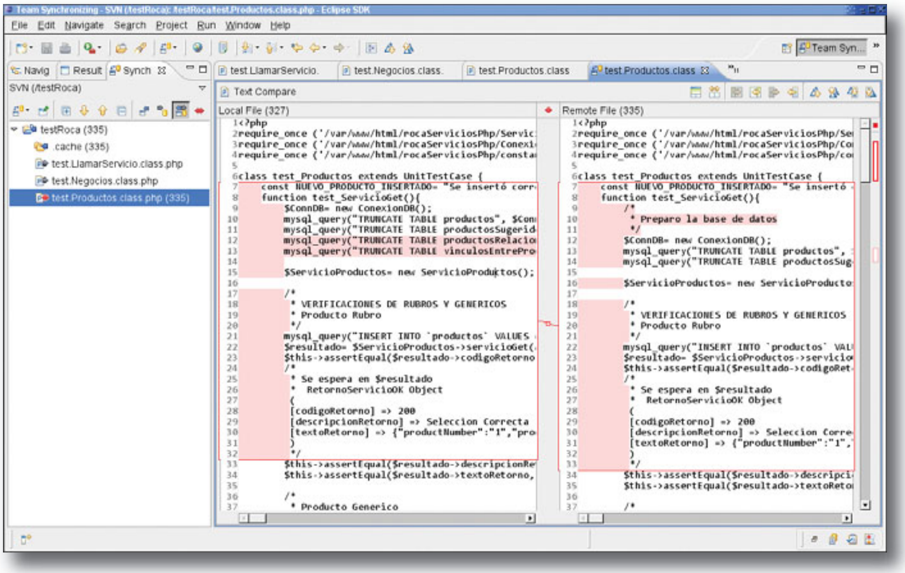
There are several tools for testing, generically named xUnit—for example, JUnit is used for Java development, cppunit for C++, PHPUnit for PHP and so on. Although the specific details logically differ between tools, they actually are quite similar. For our purposes, we work with SimpleTest, which is a plugin that provides PHPUnit tests within Eclipse.

SimpleTest is available as open-source code, and its latest version is 1.0.1 (from April 2008). You can download it from SourceForge (see Resources) and install it with Eclipse. After you install and configure it, a new option will be added to the Run As... menu, allowing you to execute PHP unit tests. You can run tests on their own, so you can test only a single routine, or you can build more complex test suites, so you can run lots of tests at the same time. You probably will use individual tests while coding and suite tests before uploading any code.

When you edit a PHP source file, several shortcuts and functions can speed you along.

When testing, a simple console with a colored bar will show up. Red means some test failed (your code doesn't do what was expected), and green means your code passed all tests (Figure 4). If you get a red bar, you can click on the offending test name, and you will be taken directly to the problematic test code (Figure 5).

Due to space constraints, I won't go into how to write tests or use mock objects; check the documentation for more information. In any case, for each project you work on, you should create a second, parallel, test project. (Of course, use version control for it, as well.) Getting used to automated testing takes some time, but the rewards are high, and you may even become, as it has been said, "test-addicted".



```
<?php
require_once ('/var/www/html/rocaServiciosPhp/Servic
require_once ('/var/www/html/rocaServiciosPhp/Conexi
require_once ('/var/www/html/rocaServiciosPhp/consta

class test_Productos extends UnitTestCase {
    const NUEVO_PRODUCTO_INSERTADO= "Se insertó corr
    function test_ServicioGet(){
        $ConnDB= new ConexionDB();
        mysql_query("TRUNCATE TABLE productos", $Con
        mysql_query("TRUNCATE TABLE productosSugerid
        mysql_query("TRUNCATE TABLE productosRelacio
        mysql_query("TRUNCATE TABLE vinculosEntrePro

        $ServicioProductos= new ServicioProduktos();

        /*
         * VERIFICACIONES DE RUBROS Y GENERICOS
         * Producto Rubro
         */
        mysql_query("INSERT INTO `productos` VALUES
        $resultado= $ServicioProductos->servicioGet(
        $this->assertEqual($resultado->codigoRetorno
        /*
         * Se espera en $resultado
         *  RetornoServicioOK Object
         (
         [codigoRetorno] => 200
         [descripcionRetorno] => Seleccion Correcta
         [textoRetorno] => {"productNumber":"1","pro
         )
         */
        $this->assertEqual($resultado->descripcionRe
        $this->assertEqual($resultado->textoRetorno,

        /*
         * Producto Generico
```

```
<?php
require_once ('/var/www/html/rocaServiciosPhp/Se
require_once ('/var/www/html/rocaServiciosPhp/Co
require_once ('/var/www/html/rocaServiciosPhp/co

class test_Productos extends UnitTestCase {
    const NUEVO_PRODUCTO_INSERTADO= "Se insertó
    function test_ServicioGet(){
        /*
         * Preparo la base de datos
         */
        $ConnDB= new ConexionDB();
        mysql_query("TRUNCATE TABLE productos",
        mysql_query("TRUNCATE TABLE productosSug

        $ServicioProductos= new ServicioProducto

        /*
         * VERIFICACIONES DE RUBROS Y GENERICOS
         * Producto Rubro
         */
        mysql_query("INSERT INTO `productos` VAL
        $resultado= $ServicioProductos->servicio
        $this->assertEqual($resultado->codigoRet
        /*
         * Se espera en $resultado
         *  RetornoServicioOK Object
         (
         [codigoRetorno] => 200
         [descripcionRetorno] => Seleccion Corre
         [textoRetorno] => {"productNumber":"1",
         )
         */
        $this->assertEqual($resultado->descripci
        $this->assertEqual($resultado->textoReto

        /*
```

Figure 6. With a synchronization conflict, you need to analyze the differences between versions and decide what to do.

Debugging

Can there be any errors if the testing techniques mentioned above are applied? Unfortunately, there are well-known theorems showing that no amount of testing can ensure program correctness, so now and then, you still will find yourself trying to figure out what went wrong. Classically, PHP programmers use print statements—usually, a `die(...)` instruction—but that's a cumbersome way of doing it. Furthermore, changing a program in order to see what happens (even if the change is an innocuous printing command) is not a good idea; you can make things even worse accidentally.

Although some languages (notably Java and Smalltalk) always have had quite good debugging environments, allowing you even to trace the code on a sentence-by-sentence basis, setting breakpoints, examining variables and so on, PHP programmers too often have found themselves with the short end of the stick.

There are basically two options: XDebug and DBG. XDebug is up to version 2.0.3 (from April 2008), and it's fully open source. On the other hand, DBG has two versions: a free one (at version 2.15.5) and a commercial one (at version 3.1.11). XDebug supports PHP 5.3, and DBG works only up to 5.2. For both programs though, the main sticking point is configuration, which is far too long to include here (see Resources). After you get the debugger to run, you will be able to debug your code easily; it's a pity that the installation procedure is such a chore.

Version Control (VC)

Version control (also known as revision control or source code control) is a must for large-scale, multi-developer projects, but it also offers significant advantages even for standalone work. The first time you thrash your code and manage to restore it or find what you changed thanks to your VC system, you will fully appreciate version control.

Basically, all VC systems allow you to store documents and record the changes made to them. VC systems allow you to inspect not only the latest version of any document, but also to go back to previous ones and work out the differences between any two versions. Explaining version control in all its details is beyond the scope of this article.

PHPEclipse can connect to SVN repositories by using either Subversive or Subclipse.

Many different version control programs are available: BitKeeper, Git, Mercurial, CVS and SVN, but not all of them have Eclipse plugins. At the very least, both CVS and SVN can be used with PHPEclipse, but I prefer the latter, because it allows for moving files; CVS doesn't.

PHPEclipse can connect to SVN repositories by using either Subversive or Subclipse. Note that at the time of this writing, both plugins are assured to work only with Eclipse 3.3 (Europa) and not with 3.4 (Ganymede). After installing one of those plugins, you will be able to download a working copy or synchronize your work with the repository, simply by right-clicking on the project and selecting the Team option. The results of a synchronization operation will show in a separate console and usually will consist of files that you should download (others have modified them and you are not up to date), files you should upload (only you have modified them) and conflict files. Clicking on a conflicting file will bring up a file comparison window (Figure 6), highlighting the differences between your code and the already uploaded code. How to merge that code is up to you.

Conclusion

Eclipse can provide a great environment for PHP development, with all the necessary tools for modern, agile development. Take the time to learn all the existing functions, and you will find yourself creating good quality code in a faster, surer and easier way.■

Federico Kereki is an Uruguayan Systems Engineer, with more than 20 years' experience teaching at universities, doing development and consulting work, and writing articles and course material. He has been using Linux for many years now, having installed it at several different companies. He is particularly interested in the better security and performance of Linux boxes.

Resources

PHP Home Page: **www.php.net**

Eclipse Home Page: **www.eclipse.org**

Eclipse Download Page: **www.eclipse.org/downloads**

IBM's Description of Ganymede's New Features: **www-128.ibm.com/developerworks/library/os-eclipse-ganymede**

The IcedTea Project—a Free Implementation of Java: **icedtea.classpath.org**

Sun Developer Network Page: **java.sun.com**

PHPEclipse: **www.phpeclipse.de**

PHPEclipse Latest Version Update: **update.phpeclipse.net/update/nightly**

PHPEclipse Full Documentation: **docs.schuetzengau-freising.de/modules/xdocman/index.php?doc=xo-002**

The Last Craft (PHP Testing): **www.lastcraft.com**

SimpleTest: **simpletest.sourceforge.net**

SimpleTest for the Eclipse Update: **simpletest.org/eclipse**

PHP Debugging: **www.ibm.com/developerworks/library/os-debug**

XDebug: **xdebug.org**

Configuring XDebug for PHPEclipse: **dev.phpeclipse.com/wiki/XDebug_and_PHPEclipse**

DBG: **dd.cron.ru/dbg**

DBG Installation Guide: **dd.cron.ru/dbg/installation.php**

PHPEclipse DGB Configuration: **docs.schuetzengau-freising.de/modules/xdocman/manual.php?doc=xo-002&id=sec.install_dbg&file=ch01s05.html**

The Subversive Plugin for Using Subversion: **www.eclipse.org/subversive**

Subclipse, an Alternative for Accessing SVN Repositories: **subclipse.tigris.org**

Enlightenment—the Next Generation of Linux Desktops

The soon-to-be-released version of Enlightenment, E17, offers a lightweight, yet stunning, alternative to KDE and GNOME. JAY KRUIZENGA

Do you remember the first time you saw the phenomenally successful "Get a Mac" ad campaign? The American ads feature actor Justin Long as the friendly, calm and casual Mac, paired with funny-man John Hodgman as the uptight, insecure and nerdy PC. And, the ads always begin the same way: "Hi...I'm a Mac, and I'm a PC." The obvious intent of each personification is to show that the Mac resembles a more youthful Steve Jobs, and the PC closely resembles Bill Gates. It's brilliant marketing. The gist of the ads is this: PCs are prone to malware of all types and are difficult to use, and the Mac is not only easy to use, but it's also safe and secure. For those who switch to a Mac, all their problems will disappear. The target audience for this campaign is not the avid PC user but rather those who use a PC because they are unaware of other options. And, this message has been extremely effective, with Mac sales increasing a whopping 12% at the end of fiscal year 2006—that's a total of 1.3 million new Mac users.

So, why is the "Get a Mac" campaign so successful? Because the ads utilize a technique known as framing, where the viewer's perception is manipulated through selective information. In this case, the ads support a framed dualism where the viewer's presented choices are only PC or Mac. No other choices (although obviously they exist) are mentioned. This leads the viewer to think the Mac is better than the PC for a multitude of reasons, each highlighted by the various ads. And, who wouldn't want to be more like the hip Justin Long?

We Linux users are thrust into an unspoken dualism of our own. Through the various flame wars pitting the KDE desktop over GNOME, the major distributions choosing sides and Linux founder Linus Torvalds throwing his weight behind KDE, it may appear to newbie Linux users or prospective users that Linux is a dualistic system. You choose either KDE or GNOME. Unlike the dualism shown in the Mac ads, both KDE and GNOME have good qualities. No one desktop reigns supreme. They both utilize the same Linux kernel, and both are equally successful.

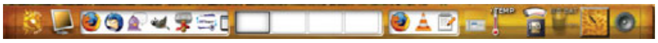


Figure 1. E's Very Useful Task Bar—an Essential Part of E17

Lost in the smokescreen of the desktop wars are the lesser-known desktops and window managers of which the lightweight Xfce desktop and the Enlightenment window manager are a part. This article focuses on Enlightenment, primarily the new and improved E17 (formerly known as DR17, because it's a developer release still in beta). Created in 1997, Enlightenment, hereafter referred to as E, originally was based on the FVWM window manager. Since then, it has forked out on its own and no longer shares borrowed code from FVWM or any other window manager or desktop. This lays precedent to the claim of E's developers that E17 is at the forefront of the next generation of desktops. However, the word desktop conjures up thoughts of KDE and GNOME, but that is not what is meant by "next generation". Rather, E is a desktop shell.

Desktop shell means an entity that sits somewhere between a minimal window manager and a full-featured desktop experience (like KDE or GNOME). For this reason, E's developers state that E is not intended to compete with either of those desktops. Instead, E is a desktop shell, combining a window manager with a file manager and configuration utilities. This new structure "will provide integration between files and your environment in a seamless manner while encompassing a graphically rich and flexible architecture".

E is possible because of the exclusive EFL (Enlightenment Foundation Libraries) written on behalf of E17. Parts of EFL are stable—like the newly updated Eet, a data encoding, decoding and storage library, which has been granted a 1.0 status. However, most of the coding is not yet complete, which places E17 in beta, rendering the system not completely stable as a desktop. Still, many users are choosing E17, thanks to its amazing ability to resurrect older PCs and bring systems with as little as 100MHz CPUs and 64MB of RAM to life again. Plus, E17 provides much-needed eye candy, with dazzling 2-D effects, to these older PCs—effects that would use a large amount of system resources through Compiz Fusion. No special 3-D graphics cards are needed for these effects on E. It's all in the EFL code.

In addition, EFL enables the potential for animated themes, animated boot screens, virtual desktops (up to 24) with separate animated backgrounds and more. Menus and borders are equally animated—or they can be if the theme allows—making E17 a unique experience.

In fact, that very uniqueness could be its potential downfall. Because E is not like anything else, users probably will encounter a short learning curve when using the desktop—figuring out where things are placed, how to summon the menu and how to configure various desktop elements. At first, one of the most disturbing features for me was calling up the menu by right-clicking my mouse on the desktop canvas. It takes a little getting used to, but after a while, it becomes second nature. It's the little things like this, the eccentricities of E, that seem awkward at first.

E17 noticeably lacks a stable file manager. As I mentioned earlier, the E developers melded the window manager with the file manager and configuration utilities, resulting in the next generation of desktop shells. Without the file manager, which is under heavy development, E is nothing more than a window manager. So, those distributions using E17 are integrating alternate file managers atop E to bridge this hole. Once the E file manager (EFM) is stable enough for everyday usage, it too will be configurable with eye candy equivalent in style to the rest of E. You will be able to search your files like any other file manager, with visual thumbnails that open into the application of your choice.

Other elements of E still on the plate include engage, the Mac OS X look-alike task bar (usable); entice, an image viewer; express, E's instant-messaging client; elation, a DVD-player GUI; embrace, an e-mail checker; elinguish, a BitTorent client; and several other components.

Almost everything about E is configurable. E includes a configuration panel allowing you to change many features, such as the wallpaper, theme, fonts, screen resolution, power settings, mouse and keyboard settings and more. This is nothing exceptional. I'm merely pointing out that E resembles a desktop with configuration options like KDE and GNOME. Clearly E is intended to be more than a simple window manager resting above a desktop foundation like frosting on a cake. E is both cake and frosting, but the cake still is being whipped together.

Another useful configuration option for E17 is the ability to change the language on the fly. Twenty languages currently are supported, including English, French, Russian, Korean, Chinese and Japanese. And, there is no need to restart the X server to switch between languages. It's instant.

E also includes the ability to add or remove little applications called modules. E's modules are similar to KDE's SuperKaramba or the Mac dashboard, adding functionalities like weather, calendars, volume control, temperature monitor, CPU frequency widget, battery monitor (for laptops), clock and more. The sky's the limit for future development of additional modules. And, selected modules appear in real time. There is no need to restart X or press a special combination of buttons to view them. It will be interesting to see the many modules that develop once E17 is officially released as a 1.0.

The question remains, is E17 ready for a standalone desktop? Probably not for business purposes, but it can be quite useful personally. Although E can crash, most crashes are not system-related, so whenever an application crashes, it

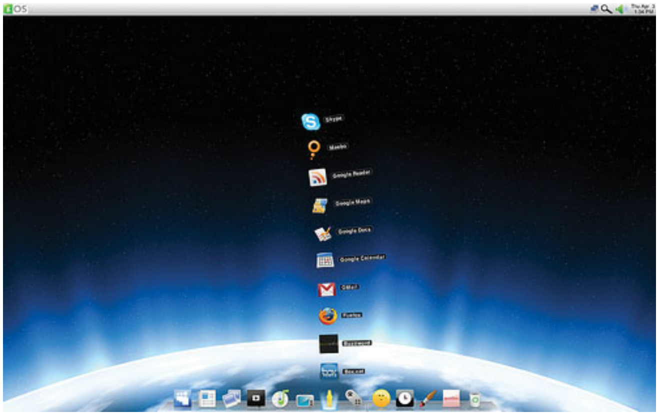


Figure 2. A Look at gOS

simply can be closed down and restarted. This can and does happen occasionally, and these minor inconveniences should be worked out in later releases.

There are a few simple ways to try E17. If you are running Ubuntu, there is a method from the user forums where you can install E to be one of the choices available at boot. However, post-installation, you will be missing pertinent files that enable every feature to work properly. For that reason, you might want to try a distribution from a live CD with everything tweaked to work. Using E17 as a window manager above either GNOME or KDE does not provide the full extent of E's power. Besides, this sort of defeats the purpose of resurrecting older equipment. If you install E as a window manager, you lose its power and speed. Yes, E is very fast—think Xfce on steroids.

Tutorials for installing E17 exist for Ubuntu, Fedora, Gentoo and Arch Linux. If you are interested in running E17 as a window manager, refer to the user forums for these distributions for directions. Instructions for Ubuntu are at **ubuntuforums.org/showthread.php?t=97199&highlight=E17+cvs**, and instructions for Fedora and Mandrake users are at **sps.nus.edu.sg/~didierbe**.

As mentioned previously, the best way to try E17 is by choosing a live CD with E pre-installed. There are a few from which to choose, and I briefly highlight each here. The following desktop experiences range from a lesser extreme, where E is moderately used, to a full extreme, where E is used exclusively.

gOS Space 2.9—the Lesser Extreme

Hardware requirements are 700MHz CPU, 384MB of RAM, 8GB disk space, graphics card capable of 1024x768 resolution, sound card and Internet connection.

There was a lot of hype over gOS when it was still being discussed in forums. It was thought that Google was creating a Linux distribution of its own. But, this turned out to be in error. gOS is a polished distribution that utilizes certain elements of Enlightenment for its beautiful special effects. It also uses the GNOME desktop and Compiz—thus, the slightly more modern hardware requirements. Space 2.9 is geared toward the 100,000,000 MySpace users. The revolutionary space dock used by gOS closely resembles the Mac OS X dock with stacks that open and swerve to reveal further options beneath.

gOS is an excellent system for the modern digital life. It includes everything users ever would need in an Internet system. However, gOS falls short in its full usage of E17. There are too many other elements in play where E is neither seen nor heard. For instance, E's Engage dock is replaced with a gOS creation. Plus, E's eye candy has been overridden by Compiz Fusion. So, where is E? In my opinion, gOS is a Mac copycat, and that's not a bad thing. In fact, I think it's a welcome twist to the numerous Windows lookalikes in the Linux community. So, if you're looking for a fast, fun to use and Mac-like distribution, try gOS.

Elive—the Further Extreme (Where Debian Meets Enlightenment)

Hardware requirements are 300MHz CPU and 128MB of RAM.

Elive is an attempt at a pure E17 desktop experience. It also includes the former E16 stable release; both are available at boot. I really enjoyed the E17 experience using Elive. It's small enough that it can be run comfortably from the live CD without installing it. Although some features, such as playing DVDs, were not enabled. The Elive CD is the fastest live CD I have tried to date, which must be due to the inherent speed of E.

Figure 3. A Glimpse at the Elive CD

Elive has a very polished look and offers two themes: night or day. I did not experience any crashes while using the system, although don't expect everything to work without problems, as E17 still is under development. If stability is what you prefer, you can try the E16 desktop, but E16 is not as pretty.

Elive contains its own configuration panel, called Epanel, which enables users to control the entire E system—adding and removing packages, configuring hardware (Elive has great hardware support by the way) and customizing the overall look and feel of the system.

If you want a true E experience, try Elive. My only issue with Elive is that it requires users to pay a minimal fee before downloading. The default is $15 US, though this can be dropped to $5. And yes, it is possible to download it free of charge, but to do so, you must send the developers an e-mail asking for an invitation code.

My only only concern with Elive is that Enlightenment is not ready as a full-featured desktop experience—some features seem unfinished. But, Elive is a wonderful awe-inspiring walk down the path to Enlightenment.

So, if you want to try E17 exclusively, with no added components from other window managers/desktops, don't hesitate to download Elive. After all, $5 will aid Elive's developers to continue their noble work.

OpenGEU (Formerly Geubuntu)—Somewhere in the Middle

I first should mention that OpenGEU is not an official Ubuntu derivative. It is based on Ubuntu and shares its repositories, but it's not Ubuntu. OpenGEU's subtitle explains the philosophy behind this newer distribution: "when a GNOME reaches Enlightenment". OpenGEU's ambition is to fill in the missing parts of E17 with the working parts of the GNOME desktop or Xfce. And, it does this very well. This hybrid system is a fully functional Enlightenment desktop with the power of Ubuntu's GNOME desktop melded with the effects of E17. For example, the file manager missing from E17 is filled with the Xfce Thunar file manager, and it works without a hitch.

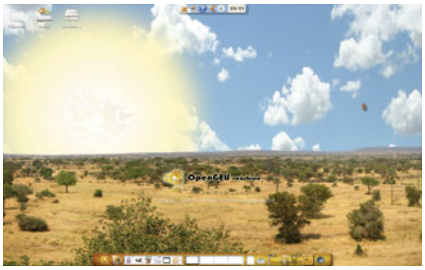


Figure 4. OpenGEU

OpenGEU includes two themes: sunshine and moonlight. Both are exquisitely beautiful with animated elements—typical E style. In the sunshine theme, the sunbeams appear to shine forth at certain times, and under the moonlight theme, the Enlightenment E logo apparent on the moon reflects within the ripples of an ocean of water at regular intervals. Users can change between themes at the press of a button. Other themes are included, and users can download additional themes from get-E.org.

OpenGEU not only borrows Xfce's Thunar file manager, but it also borrows its panel. And, the bar across the top of the screen is from GNOME. But, hidden beneath the scenes is E. I am delighted with the mix. The distribution is not without its bugs, but E's performance does not appear to be altered in the least through the addition of various GNOME and Xfce components. OpenGEU is a glimpse of what we can expect from the 1.0 E release.

There was one strange "bug" that I discovered when clicking on a file from my desktop. Instead of defaulting to the Thunar file manager, E's own file manager opened, and it left a bad taste in my mouth. I was humored by the wiggling icons, but the total experience is not finalized. It lacks a certain appeal—that look of completeness. I can understand why Thunar was chosen in its place. Perhaps EFM should be removed from OpenGEU entirely.

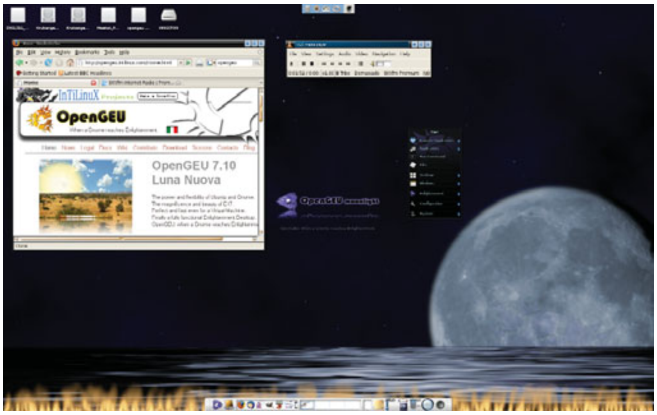


Figure 5. OpenGEU's Moonlight Theme

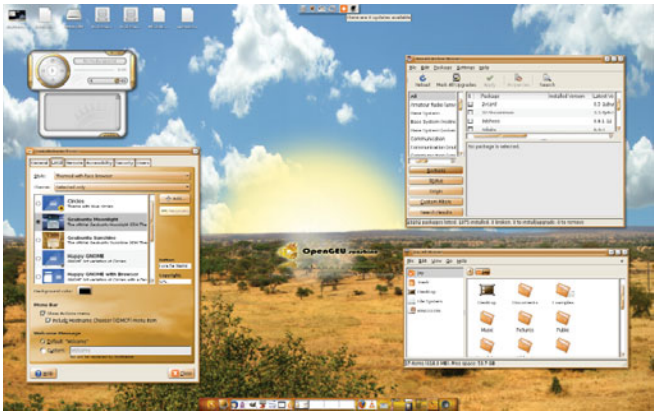


Figure 6. OpenGEU's Sunshine Theme

OpenGEU is different enough to be noticed by family and friends. It's easy to use, simple to install and fanatically fun. You can expect E's total functionality with animations, fading and shadows. I used OpenGEU for quite some time for the purpose of this review, and it is the most pleasant E experience I encountered. This is one distribution I'll definitely be watching, and it's the distribution from which I am writing this review. If you are looking for the ultimate E experience, try OpenGEU. You won't be disappointed.

Conclusion

E17 is under heavy development and probably not useful for business purposes. However, it's ready for personal use, especially for those with older PCs that could stand to be revived. System requirements for E are extremely low, with dazzling 2-D effects rivaling the best of Compiz Fusion without the need for an up-to-date graphics card. Completely rewritten using EFL, E is not like any other window manager or desktop in existence. It's intended to be the next generation of desktops—a desktop shell that sits somewhere between window manager and full-fledged desktop. E is not for everyone, but for most users, E should be a pleasant experience. And, it's lightning fast to boot. I hope that Linux users, old and new alike, will come to recognize that there's more to Linux than just KDE and GNOME.■

Jay Kruizenga resides in Grand Rapids, Michigan. A small-business owner, Linux advocate and freelance writer, Jay spends most of his free time reading, writing or creating projects.

Why We Need Hackers to Fix Health Care

Some of the most dangerous closed and proprietary systems are the ones you trust to save your life.

DOC SEARLS

My mother died five years ago of a stroke following an endoscopic procedure to remove a gallstone. The procedure perforated her duodenum, and digestive fluids leaked into her abdomen. She spent the next week in the Intensive Care Unit, fighting for her life. She was tough and lived through it, but the stroke got her a few days later.

The stroke probably was due to a blood clot, which probably formed because she was off her blood thinners. That was a medical error that might have been prevented had her gastroenterologist and her cardiologist been communicating with each other. My sister and I blame ourselves for not making sure those guys were talking. But, I also blame the hospital's IT system, which failed to keep both doctors in their shared patient's loop.

I also should have suspected the IT system of suckiness, because I brought it down myself one day while visiting Mom by using a browser on one of the nursing workstations there. I was surfing for about ten seconds when every screen in sight went blue. Shocked and concerned, I asked a nurse if this happened often. "Happens all the time", she said. "It's a new system." Of course, it ran on Windows.

This year, I had my own encounter with sucky systems. It started in April after I had a pulmonary embolism (a blood clot) in my right lung. While looking for the clot's possible sources, a CAT scan showed a cystic lesion on my pancreas. My gastroenterologist ordered an MRI, which showed more cysts. Radiologists said it wasn't clear if one of the cysts was communicating with the pancreatic duct, so my gastroenterologist recommended an endoscopic procedure to look up the duct and see what was going on—the same procedure that put Mom in the hospital.

The doctor told me before the procedure that there was only a 5% chance of getting pancreatitis from it. I said okay, and we went ahead with it. The next morning pancreatitis struck, and I spent the next nine days in the hospital taking no food or water while massive quantities of fluids were dripped into my veins. Pain was addressed with enough Demerol, Morphine and Dilaudid to satiate a junkie.

As I write this, I'm still recovering—and still in a state of mystery about my pancreas. The procedure did not see a cyst communicating with the duct. Neither did a second team of radiologists that viewed the same MRI. That team said I didn't need the procedure. But the word came too late, when I was already in the hospital.

One reason we couldn't get the MRI CD to the second team earlier was that we couldn't find a machine to read it. It wouldn't load on my gastroenterologist's Windows machine or on either of my Linux or my Mac machines. All I could see was a pile of Windows binaries and files.

So, although it was our error to hasten a procedure I didn't need, I also blame a system in which too much tech doesn't work, doesn't communicate with other tech, or doesn't use standard image and text file formats that any machine can read.

Among the many doctors I met in the hospital, one stood out, because he alone addressed the problems of bad data and bad communications. He said that the whole medical system is corrupted by collusion between equipment makers, software suppliers and institutional customers. The result is many closed systems, all lousy at communicating with each other. He said we need open systems, with data built around patients rather than locked inside closed silos. He liked Google Health, because at least it was trying to solve the problem from the patient's side, by making the patient the point of integration for health-care data from many different sources. (Microsoft also seems to be doing something similar with HealthVault.)

The whole matter of Personal Health Records (PHRs) is a complicated one. There are many open-oriented efforts going on there, and I hope one or more of them succeeds. Meanwhile, countless thousands of people die every year in the US alone from bad data and poor communications among health-care providers. This problem cannot be fixed from the top down, no matter how open its code.

It has to be fixed from the bottom up—by hackers and patients. Hackers need to build (or help health-care software companies build) new systems using free software and open-source code, so those systems can be improved and made more compatible on an ongoing basis. Plenty of money can be made selling systems and servicing them. You don't need closed code for that. Patients need to become platforms. Each of us needs to be able to gather, control and share our own health-care data, on our own terms—quickly, easily and securely. So services can be based on what makes each of us unique.

When I suggested this in a post on the *Linux Journal* Web site, some skeptical comments followed, especially from veterans of The System. But, their arguments were the same kind I heard 30 years ago against personal computing and open networking—that they were a cool idea, but that the Big Boys would never let it happen.

We know how that story turned out. I'd like the health-care story to turn out the same way. We need open-source hackers to make that happen. Preferably while I'm still alive.■

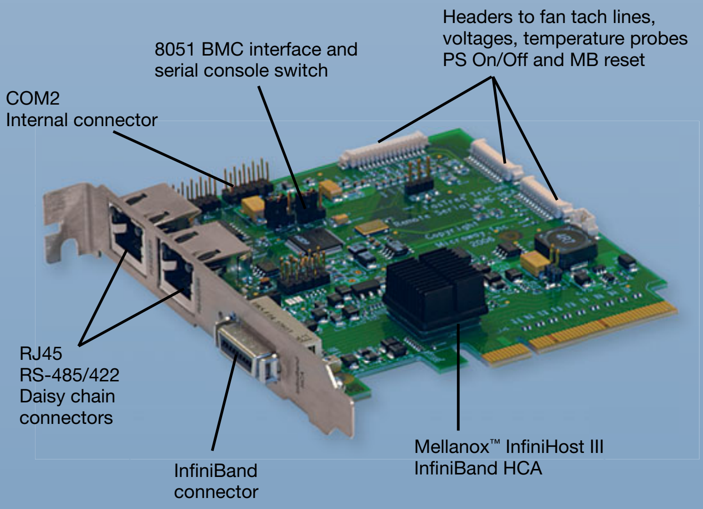
Doc Searls is Senior Editor of *Linux Journal* and a fellow with both Berkman Center for Internet and Society at Harvard University and the Center for Information Technology and Society at the University of California, Santa Barbara.

Affordable InfiniBand Solutions

4 Great Reasons to Call Microway NOW!

TriCom™

- DDR/SDR InfiniBand HCA
- "Switchless" serial console
- NodeWatch web enabled remote monitor and control

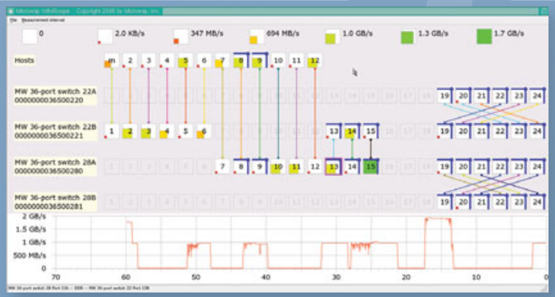

Headers to fan tach lines, voltages, temperature probes PS On/Off and MB reset
8051 BMC interface and serial console switch
COM2 Internal connector
RJ45 RS-485/422 Daisy chain connectors
InfiniBand connector
Mellanox™ InfiniHost III InfiniBand HCA

FasTree™

- DDR InfiniBand switches
- Low latency, modular design
- 24, 36 and 48 port building blocks

InfiniScope™

- Monitors ports on HCA's and switches
- Provides real time BW diagnostics
- Finds switch and cable faults
- Lane 15 interface
- Logs all IB errors


0
2.0 KB/s
347 MB/s
694 MB/s
1.0 GB/s
1.3 GB/s
1.7 GB/s
Hosts
MW 36-port switch 22A 0000000036500220
MW 36-port switch 22B 0000000036500221
MW 36-port switch 28A 0000000036500280
MW 36-port switch 28B 0000000036500281
2 GB/s
1.5 GB/s
1 GB/s
500 MB/s
0
70
60
50
40
30
20
10
0

ServaStor™

- Extensible IB based storage building blocks
- Redundant and scalable
- Parallel file systems
- Open source software
- On-line capacity expansion
- RAID 0,1,1E, 3, 5, 6, 10, 50

Upgrade your current cluster, or let us design your next one using Microway InfiniBand Solutions.

To speak to an HPC expert call ***508 746-7341*** *and ask for technical sales or email* ***sales@microway.com***
www.microway.com

Visit Microway at SC08 in Austin-Booth 1945

Microway®
Technology you can count on℠