Kinect | Backbone.js | Lessfs | Blender | OpenGoo | Mojolicious

LINUX JOURNAL™

Since 1994: The Original Magazine of the Linux Community

JULY 2011 | ISSUE 207 | www.linuxjournal.com

USE PUPPET FOR SYSTEMS MANAGEMENT

GATHERING FORENSICS DATA ON AN EXT4 FILESYSTEM

DATA DEDUPLICATION WITH LESSFS

COMBINE BACKBONE.JS AND RAILS FOR MORE POWERFUL APPS

COOL PROJECTS

Build a Natural User Interface for Linux with Kinect / Monitor Processes with Your Smartphone in Real Time / Create 3-D Games with the Blender Game Engine / Build an On-line Office with OpenGoo / Control Real-World Hardware with Linux / How-To: E-Publishing the Open-Source Way

$5.99US $5.99CAN

07

0 09281 03102 4

10 Gig On Board

Blazing Fast, Embedded 10Gb Ethernet

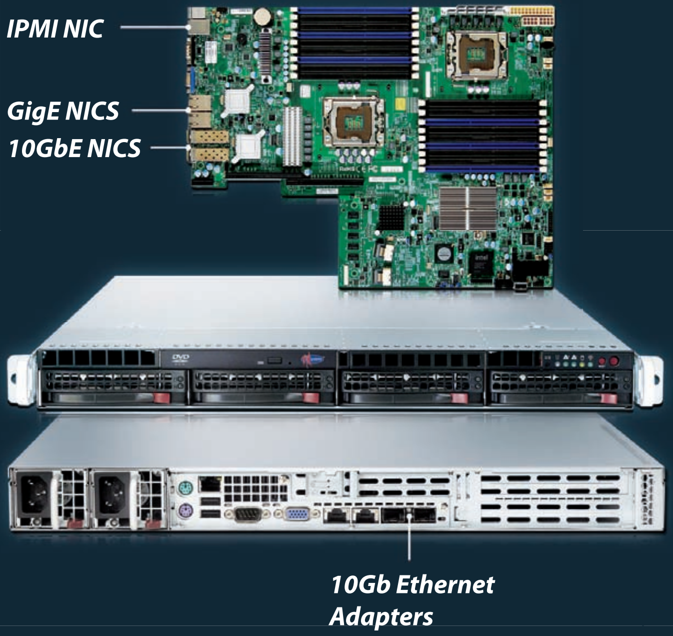
10G Rackmount Servers in the iX-Neutron server line feature the Intel® Xeon® Processor 5600/5500 Series, and come with 10GbE networking integrated onto the motherboard. This eliminates the need to purchase an additional expansion card, and leaves the existing PCI-E slots available for other expansion devices, such as RAID controllers, video cards, and SAS controllers.

For more information on the iX-1204-10G, or to request a quote, visit: **http://www.iXsystems.com/neutron**

30% cost savings/port over equivalent Dual-Port 10 GB PCI Express add-on card solution

KEY FEATURES:

- Supports Dual 64-Bit Six-Core, Quad-Core or Dual-Core, Intel® Xeon® Processor 5600/5500 Series
- 1U Form Factor with 4 Hot-Swap SAS/SATA 3.5" Drive Bays
- Intel® 5520 chipset with QuickPath Interconnect (QPI)
- Up to 192GB DDR3 1333/1066/800 SDRAM ECC Registered Memory (18 DIMM Slots)
- 2 (x8) PCI-E 2.0 slots + 1 (x4) PCI-E 2.0 (in x8 slot -Low-Profile - 5.5" depth)
- Dual Port Intel® 82599EB 10 Gigabit SFP+ - Dual Port Intel® 82576 Gigabit Ethernet Controller
- Matrox G200eW Graphics
- Remote Management - IPMI 2.0 + IP-KVM with Dedicated LAN
- Slim DVD
- 700W/750W Redundant AC-DC 93%+ High-Efficiency Power Supply



Call iXsystems toll free or visit our website today!

1-855-GREP-4-IX | www.iXsystems.com

RouterBOARD SXT

- Solid all-in-one design
- One hand enclosure access
- Quick and easy to mount
- 5GHz 802.11a/n wireless module
- 16dBi dual chain antenna built-in
- Signal strenght LED indicators
- One 10/100 Ethernet port
- USB 2.0 port
- FCC certified
- Voltage and temperature monitors
- Packaged with mounting bracket, attachment ring, power adapter and PoE injector
- **Only $89** for the complete kit

SXT 5HnD is a low cost, high speed wireless device which can be used for point to point links, or as a CPE for point to multipoint installations.

Dual chain 802.11n and TDMA technology help to achieve even 200Mbit real throughput. Complete with a ready to mount enclosure and built-in antenna, the package contains everything you need to make a point to point link, or connect to an AP.

The SXT is powered by RouterOS, the most advanced router, bandwidth controller and firewall.

MikroTik routerboard www.mikrotik.com

CONTENTS

JULY 2011
Issue 207

COOL PROJECTS

FEATURES

They say work smarter, not harder. They must be using our processor.

intel
Sponsors of Tomorrow.™

The next generation of intelligent server processors
The Intel® Xeon® processor 5600 series automatically regulates power consumption to combine industry-leading energy efficiency with intelligent performance that adapts to your workload. Check out the new intelligent features of the Xeon® 5600 at intel.com/itcenter.

intel Xeon® inside™

SEE WHAT INTELLIGENCE CAN DO

Servers from iXsystems feature the Intel® Xeon® processor 5600 series.

iXsystems®

Enterprise Servers
for Open Source
www.iXsystems.com
1-855-GREP-4-IX

Intel is not responsible for and has not verified any statements or computer system product-specific claims contained herein.

The **iX2216-10G** features dual on-board Intel® 82599EB 10 Gigabit SFP+ Ports, dual on-board Intel® 82576 Gigabit Ports, and 18 DIMM slots supporting up to 192GB of DDR3 ECC Registered memory. Ideal for HPC, Data Center, Virtualization, Clustering, and Cloud Computing applications.

The **iX1204-10G** features the latest Intel® CPUs based on the 32nm and 45nm next-generation microarchitecture. The next step in intelligent performance, automated energy efficiency, and flexible virtualization.

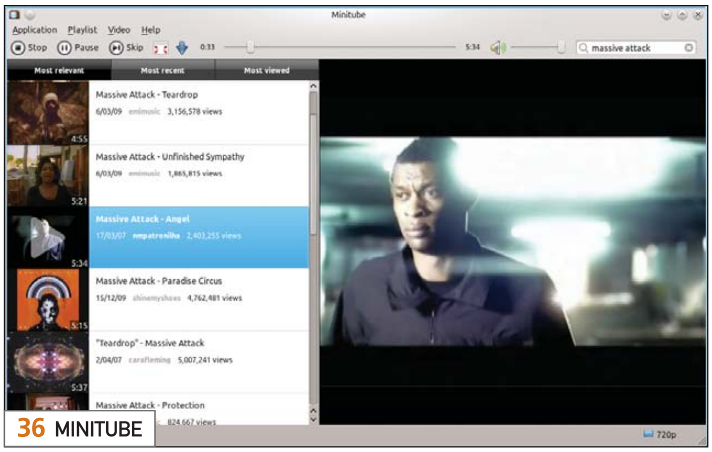
iX2216-10G

- Dual Intel® Xeon® 5600 Series Processors
- 2U Form Factor with sixteen 2.5" SAS/SATA Hot-Swap Drive Bays
- On-Board Dual Port Intel® 82599EB 10 Gigabit SFP+

iX1204-10G

- Dual Intel® Xeon® 5600 Series Processors
- 1U Form Factor with 4 Hot-Swap SAS/SATA Drive Bays
- On-Board Dual Port Intel® 82599EB 10 Gigabit SFP+

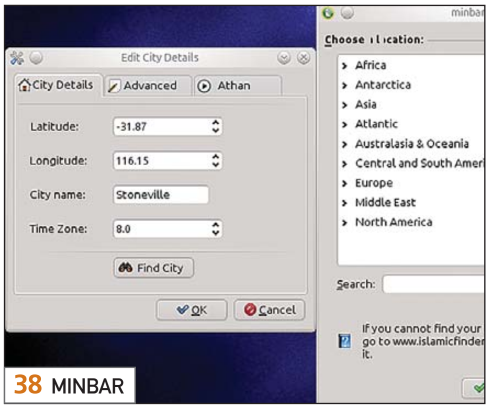
CONTENTS JULY 2011 Issue 207



Minitube
Application Playlist Video Help
Stop Pause Skip
massive attack
Most relevant Most recent Most viewed
720p

Massive Attack - Teardrop 6/03/09 emimusic 3,156,578 views
Massive Attack - Unfinished Sympathy 6/03/09 emimusic 1,865,815 views
Massive Attack - Angel 17/03/07 mmpatronilha 2,403,255 views
Massive Attack - Paradise Circus 15/12/09 shinemyshoes 4,762,481 views
"Teardrop" - Massive Attack 2/04/07 carafleming 5,007,241 views
Massive Attack - Protection

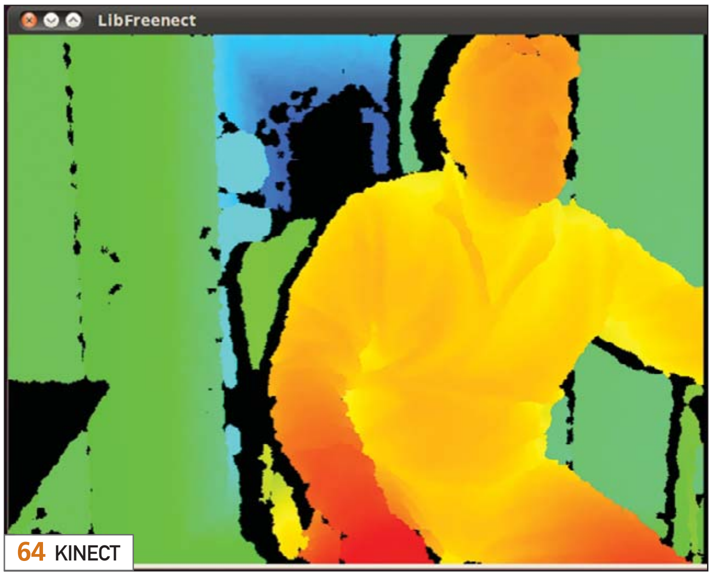
36 MINITUBE



Latitude: -31.87
Longitude: 116.15
City name: Stoneville
Time Zone: 8.0
If you cannot find your go to www.islamicfinder it.

38 MINBAR

58 MOJOLICIOUS



64 KINECT

USPS *LINUX JOURNAL* (ISSN 1075-3583) (USPS 12854) is published monthly by Belltown Media, Inc., 2121 Sage Road, Ste. 310, Houston, TX 77056 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 16476, North Hollywood, CA 91615. Subscriptions start with the next issue. Canada Post: Publications Mail Agreement #41549519. Canada Returns to be sent to Pitney Bowes, P.O. Box 25542, London, ON N6C 6B2

More TFLOPS, Fewer WATTS

Microway delivers the fastest and greenest floating point throughput in history

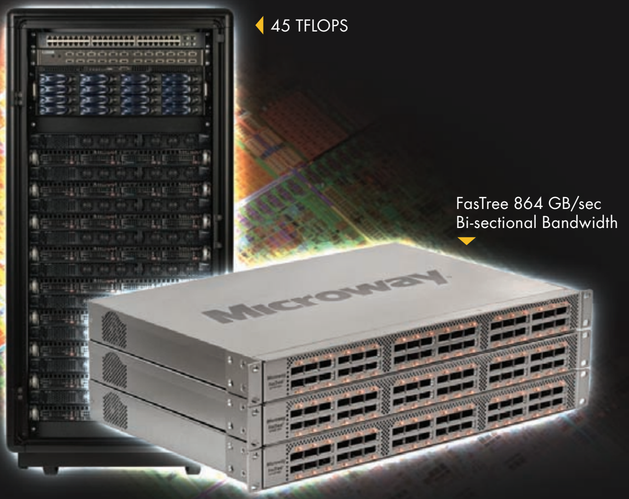
Enhanced GPU Computing with Tesla Fermi

- 480 Core NVIDIA® Tesla™ Fermi GPUs deliver 1.2 TFLOP single precision & 600 GFLOP double precision performance!
- New Tesla C2050 adds 3GB ECC protected memory
- New Tesla C2070 adds 6GB ECC protected memory
- Tesla Pre-Configured Clusters with S2070 4 GPU servers
- WhisperStation - PSC with up to 4 Fermi GPUs
- OctoPuter™ with up to 8 Fermi GPUs and 144GB memory

New Processors

- 12 Core AMD Opterons with quad channel DDR3 memory
- 8 Core Intel Xeons with quad channel DDR3 memory
- Superior bandwidth with faster, wider CPU memory busses
- Increased efficiency for memory-bound floating point algorithms

Configure your next Cluster today!
www.microway.com/quickquote
508-746-7341



FasTree™ QDR InfiniBand Switches and HCAs

- 36 Port, 40 Gb/s, Low Cost Fabrics
- Compact, Scalable, Modular Architecture
- Ideal for Building Expandable Clusters and Fabrics
- MPI Link-Checker™ and InfiniScope™ Network Diagnostics

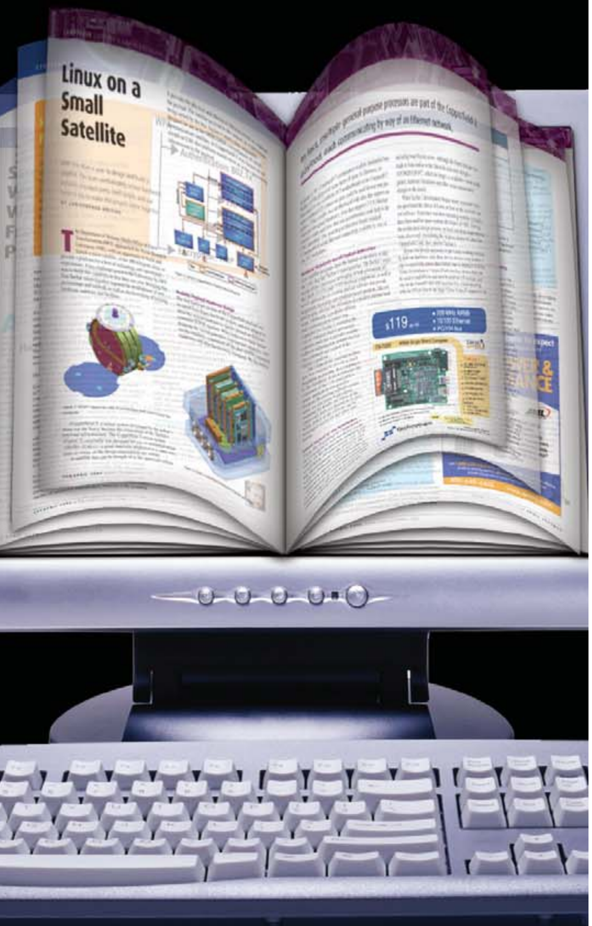
Achieve the Optimal Fabric Design for your Specific MPI Application with ProSim™ Fabric Simulator

Now you can observe the real time communication coherency of your algorithms. Use this information to evaluate whether your codes have the potential to suffer from congestion. Feeding observed data into our IB fabric queuing-theory simulator lets you examine latency and bi-sectional bandwidth tradeoffs in fabric topologies.

NVIDIA. TESLA with CUDA | intel Channel Partner Premier 2010 | AMD Fusion Partner Premier | GSA

GSA Schedule Contract Number: GS-35F-0431N

Microway®
Technology you can count on℠



LINUX JOURNAL

Executive Editor	Jill Franklin jill@linuxjournal.com
Senior Editor	Doc Searls doc@linuxjournal.com
Associate Editor	Shawn Powers shawn@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
Security Editor	Mick Bauer mick@visi.com
Hack Editor	Kyle Rankin lj@greenfly.net
Virtual Editor	Bill Childers bill.childers@linuxjournal.com

Contributing Editors
Ibrahim Haddad • Robert Love • Zack Brown • Dave Phillips • Marco Fioretti • Ludovic Marcotte
Paul Barry • Paul McKenney • Dave Taylor • Dirk Elmendorf • Justin Ryan

Proofreader	Geri Gale

Publisher	Carlie Fairchild publisher@linuxjournal.com
General Manager	Rebecca Cassity rebecca@linuxjournal.com
Senior Sales Manager	Joseph Krack joseph@linuxjournal.com
Associate Publisher	Mark Irgang mark@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

Editorial Advisory Panel
Brad Abram Baillio • Nick Baronian • Hari Boukis • Steve Case
Kalyana Krishna Chadalavada • Brian Conner • Caleb S. Cullen • Keir Davis
Michael Eager • Nick Faltys • Dennis Franklin Frey • Alicia Gibb
Victor Gregorio • Philip Jacob • Jay Kruizenga • David A. Lane
Steve Marquez • Dave McAllister • Carson McDonald • Craig Oda
Jeffrey D. Parent • Charnell Pugsley • Thomas Quinlan • Mike Roberts
Kristin Shoemaker • Chris D. Stark • Patrick Swartz • James Walker

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: +1 818-487-2089
FAX: +1 818-487-4550
TOLL-FREE: 1-888-66-LINUX
MAIL: PO Box 16476, North Hollywood, CA 91615-9911 USA
Please allow 4–6 weeks for processing address changes and orders
PRINTED IN USA

LINUX is a registered trademark of Linus Torvalds.

PRINTED WITH SOY INK®

COLLABORATION
EDUCATION
OPPORTUNITY
VANCOUVER
CANADA
AUGUST 17-19, 2011

LINUXCON

PRAGUE
CZECH REPUBLIC
OCTOBER 26-28, 2011
THE PREMIERE
LINUX CONFERENCE
HOSTED BY THE LINUX FOUNDATION

LEARN MORE AT EVENTS.LINUXFOUNDATION.ORG

SHAWN POWERS

All That Glimmers Isn't Hydrogen

When I was younger, my friend Pete and I would do cool projects pretty much every weekend. One of the most memorable was when we had a fish tank, a car battery, some carbon rods and a desire to create giant exploding balloons. The best we managed to accomplish was to fill Pete's basement with chlorine gas. His mom was not thrilled. This issue of *Linux Journal* is dedicated to cool projects. Every single one of them is cooler than anything Pete or I ever created—except for maybe that vacuum-cleaner-powered hoverboard...that was actually pretty cool.

Reuven M. Lerner starts us out describing how to combine two different MVC frameworks, Backbone.js and Rails. Having both client-side and server-side frameworks can make some powerful applications. Dave Taylor also has a cool column this month, as he shows how to determine the day of any historical date. Was I born on a Wednesday? I'm not sure, but after reading Dave's column, I can figure it out.

Kyle Rankin licks his wounds a bit this month and lets us all learn from his mistakes. Kyle's forensics articles and presentations are known far and wide in *Linux Journal* circles as "the guide" to examining compromised servers. He had to make an unexpected addendum to his procedure when it came to ext4 though, and we get the first-hand version of why. My suggestion for Kyle is that he just use good passwords, and he'd never have to worry about system compromises (tee-hee!).

Mike Diehl starts off the projects with the Blender Game Engine. Not to be confused with the "Will It Blend" guy, Mike shows how to create a 3-D game using all open-source tools. Blender might just be the Swiss Army knife of the Linux world, as it can do everything from 3-D rendering to video editing to game creation. Check out Mike's article if you want to blend yourself up a fancy game cocktail.

The office in my new home is pretty cool, but it's not nearly as cool as Hani Saigh's idea of a cloud-based office. Granted, he doesn't mean an actual office in the clouds, but having your office software accessible everywhere is still a nifty idea. Hani introduces OpenGoo, a Web-based workspace that allows for collaboration and calendar sharing. If you prefer your computing to be more down to earth, PJ Radcliffe's article on controlling real-world hardware might tickle your fancy. There's nothing more down to earth than soldering on a circuit board, and with Open-USB-IO, you can control those circuits with Linux—pretty cool stuff.

Many of us are system administrators, at least on some level, so monitoring systems is something we think about constantly. Jamie Popkin describes how to use Mojolicious, which allows you to monitor processes remotely with your smartphone. If that's not cool enough, perhaps you could combine his ideas with Rick Rogers' article on using the Microsoft Kinect with Linux. Granted, you might look silly waving your hands around in a restaurant just to check on a remote rendering project, but if you're willing to bring a Microsoft Kinect with you to a restaurant, I'm guessing "looking silly" isn't really a concern.

On top of all that, we have Petros Koutoupis' demonstration of data deduplication with Linux. Granted, storage is cheap, but efficiency is priceless. At my local school district, just this morning someone sent a 6MB photo to all 1,000 of my users. That one e-mail is taking up more than 6GB of space on my e-mail server! Data deduplication is an awesome concept, and it's one that I will be exploring this afternoon. To automate such processes, or any configuration process for that matter, Jes Fraser's article on Puppet is a must-read. Although I can manage individual configuration files for my dozen or so servers here at work, it's starting to become overwhelming. Up that number by an order of magnitude, and it's impossible to manage. That's where Puppet comes in. Jes explains how it works and walks through the process of setting it up.

And, of course, if you're just looking for some recreation this month, we have that for you too. If you want to browse YouTube without a browser, John Knight has you covered with his monthly dose of project announcements. If writing a book is more up your alley, but you need help with formatting for e-readers, check out Dan Sawyer's article on e-publishing. He'll walk you through creating and converting for e-readers.

The only example I recommend you don't follow this month is mine. Although creating hydrogen gas might sound like fun, it was about as smart as the time Pete and I tried to loosen the bolts on his truck's gas tank—with a blowtorch. I wish I were kidding. Enjoy the cool projects issue! ■

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.

KIWI PYCON WELLINGTON 2011

SIMPLICITY•FLEXIBILITY•BEAUTY

27–28 AUGUST

NZ.PYCON.ORG

TE WHAEA: NATIONAL DANCE AND DRAMA CENTRE
11 HUTCHISON ROAD, NEWTOWN, WELLINGTON, NEW ZEALAND

Node.JS | Untangle | Cfengine 3 | Armitage | Firewall Builder
LINUX JOURNAL
Linux HA Firewall Tutorial
Increase Security with Cfengine
SECURITY TESTING with Armitage and Metasploit
Secure Your Virtual Environment
Defend Against DNS Cache Poisoning
NEW COLUMN: Tales from the Server Room
REVIEWED:
The Google Cr-48 "Mario" Chrome OS Notebook
Untangle's Multi-Functional Firewall Software

Node Correction

I started reading Avi Deitcher's article titled "Simplicity and Performance—JavaScript on the Server" in the April 2011 issue. I just wanted to let you know that the command given to clone the Node repository from git:

```
git clone git://github.com/ry/node.git
```

should be changed to:

```
git clone git://github.com/joyent/node.git
```

at least as of March 21, 2011. As I couldn't get the first command to work, I did a little looking around and found the correct repository.

I'll try using this program (Node) to do some experimenting with JavaScript using Node and see how well this works. Thanks for the article, and keep on doing such a great job each and every month.

PS. I've been a subscriber since 1999 except for a few short lapses when I purchased my monthly copy from a local bookstore. Thanks.

--
Michael Soibelman

Avi Deitcher replies: *Ha, great! Ryan (the original inventor of Node) works for Joyent. At some point the company must have decided it owned the intellectual property and made him move it.*

The Origins of R

I was surprised to read in Joey Bernard's March 2011 UpFront article that the statistical analysis language R was "developed at Bell Labs". Although Bell Labs has given us much to be thankful for, R actually was developed by Ross Ihaka and Robert Gentleman from the University of Auckland in New Zealand. Of course, being an open-source (GPL) project, it has attracted contributions from all over the globe.

At the NZ Open Source Awards in 2010, Ross Ihaka was presented with the "Catalyst Lifetime Achievement in Open Source Award" for his work on R and for supporting the R community.

--
Grant McLean

Joey Bernard replies: Mea culpa. *This comes from not reading documentation closely enough. The language S was the one developed at Bell Labs, and R was developed at the University of Auckland, as you said. Thank you for correcting this and making sure that the hard work of Ross Ihaka and Robert Gentleman is recognized properly.*

Personal DNS Server

I've slowly been migrating my many sites and services from a well-known hosting provider using Plesk to self-hosted at Linode. One thing I've been procrastinating on is DNS. Kyle Rankin's article ["Your Own Personal Server: DNS", April 2011] shows how simple it really is, and it was the inspiration for me finally to migrate over. Many of the BIND/DNS books out there go very in-depth, but over-complicate configuration. Had I realized it was this simple, I would have pulled the trigger months ago. Kyle's article did a great job of covering the basics.

--
Ryan Duff

Net Neutrality

I've been reading *Linux Journal* for decades and appreciate the fine work you do.

I wanted to comment on Keith Reed's letter to the editor on Net Neutrality legislation in the April 2011 issue. You will doubtless be swamped by hundreds of other letters on this topic, but it is an important issue to to me, so I wanted to offer my two cents.

Discussions of Net Neutrality legislation frequently conflate three issues:

1. The ability of network providers to charge for the actual bandwidth and data delivery they provide.

2. The ability of network providers to shape traffic to meet Quality of Service (QoS) guidelines.

3. The ability of network providers to impose surcharges or to shape traffic based on destination.

In my opinion, the first two issues are red herrings raised by opponents of Net Neutrality legislation to distract from their actual goal, which is #3 above. I don't know of anyone seriously calling for restrictions related to issues 1 or 2. My concern, and the concern of many, is that my network provider not be allowed to filter or shape my traffic based on the destination of my packets.

The argument that the network providers built these networks and should be allowed to run them as they see fit, conveniently overlooks the fact that the networks largely were built as public/private partnerships. The network companies were granted local monopolies and obtained the easements for their cable plan through the government power of eminent domain. They were granted these extraordinary benefits because they promised to act as common carriers, providing a public good.

If the network companies want to create purely private networks, they are free to do so. They just have to give up their remaining local monopolies and pay fair value for the easements for their cable plant. Until they do so, I insist that they should operate as common carriers.

--
Charles E. Grant

I agree with you, but I think #2 is also key in their plan as well. By crippling things like VoIP, ISPs can offer their own services and have them "just work". Granted, that is related to #3, but the QoS part of the equation is worrisome for me as well. I know it can keep prices lower for the average consumer, but there needs to be an option for users like, well, me. I fear the future of the Internet without some form of Net Neutrality.—Ed.

Audible Compatibility with Linux

Audible says, "At this time Audible is not compatible with the Linux operating system. Audible is actively pursuing compatibility with Linux in all versions by pursuing support from the Open Source community that develops this platform." I joined Audible in 2002 and saw that exact message shortly after, and it has not changed to this date today.

Audible is one of the big reasons I have heard people say they will not switch to Linux. There already are ways to take out the protections and turn the audio books to MP3 files, but I have a very large library on Audible's site and need (as do many other people) for it to just work. "Just work" is what Ubuntu is all about. If you do not think it is serious, do a Google search for "Audible Linux", and you will end up with a different frame of mind.

Could someone please contact the Audible folks and tell them how much money they are losing? In the process, please offer to work with them to make Audible compatible with the Linux operating system. Ubuntu is the largest and should have a little bit of pull in whatever negotiations are needed. It would help if the other OSes would get on the wagon for the long haul.

--
Victor Jones

I think the recent release of an Android client is a huge step in the right direction. Hopefully, we'll see Audible continue down that road.—Ed.

D-Link's Boxee Box

How does a review of the D-Link Boxee Box [April 2011] qualify for inclusion in *Linux Journal*? It has nothing to do with GNU/Linux, and worse, the Boxee is HDCP-crippled.

--
Ian Stirling

The Boxee Box runs Linux, thus its inclusion in the magazine.—Ed.

Re: Letters Complaining about Linux

In addition to the reasons you give, in your answer to Gary Dale's letter, about people switching back to Windows after trying Linux, I will suggest another one [see the April 2011 Letters section]. You can't just buy hardware at your local store and expect it will work under Linux. Some hardware vendors have a policy of ignoring Linux. Presumably, they can afford to, since they sell enough products to Windows users. And, it's anybody's guess whether Microsoft offers them financial incentives for noncooperation with Linux.

Then there is the attitude on the part of some Linux workers that anything but absolutely open-source software is immoral to use. I was having quite a bit of trouble with the wireless adapter in a new laptop computer, trying to use the open-source driver. Then someone mentioned to me that a Linux driver was available from the Broadcomm Web site; I downloaded that, and things have been fine ever since. Recently I bought a Brother scanner/printer. The SANE Web site does not mention this model; and all the other Brother machines they mention are listed as unsupported. But, I found easy-to-install SANE and CUPS drivers on the Brother Web site, and the machine works fine. I'm willing to thank Broadcomm and Brother for supporting their equipment under Linux, rather than denouncing them for not open-sourcing their drivers.

--
Jim Haynes

I agree with your positive attitude. Yes, I'd prefer everyone open-source their drivers, but supporting Linux users with binary-only drivers is a huge first step. This is similar to my views on things like Netflix. If Netflix released a binary-only player for Linux, I would give it tons of credit. As it is now, people can play Netflix under Linux (Roku, Boxee Box), but desktop users can't do the same. That's frustrating.—Ed.

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 +1-818-487-2089. E-mail us at subs@linuxjournal.com or reach us via postal mail, Linux Journal, PO Box 16476, North Hollywood, CA 91615-9911 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, PO Box 980985, Houston, TX 77098 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**.

diff -u

WHAT'S NEW IN KERNEL DEVELOPMENT

Harry Wei has volunteered to be the official maintainer for all **kernel docs in Chinese**. There were a few bumps along the way. **Tao Ma**, at one point, objected that Harry hadn't demonstrated his qualifications or translated large amounts of documentation. But, oftentimes maintainership is based more on a willingness to put in the time, than on any specific expertise or qualification. Harry said he promised to do his best, and that was that.

This is actually one of the first and greatest innovations **Linus Torvalds** achieved with Linux. While the GNU folks developed the idea of free software, they limited themselves by sticking with just a tight group of core developers, who largely would ignore suggestions and patches from outside. It was Linus' belief that everyone had something valuable to contribute that led to the style of open-source development that now dominates the world.

Miklos Szeredi has documented the **overlay filesystem**. The overlay is an example of a "union"-style filesystem, in which two distinct filesystems are silhouetted against each other, so as to appear as a single one. Where their directory structures overlap, files from both filesystems appear to the user to be in the same directories.

It's very cool, and very weird, because of the shenanigans that have to go on behind the scenes to deal with conflict resolution. What do you do if both filesystems have a file called /home/zbrown/todo.txt? When new files are created, on which filesystem are they stored? A variety of such cases have to be handled in ways that make the most intuitive sense to the user as well as the sysadmin maintaining the system.

Rob Landley has automated some **kernel documentation** at **kernel.org/doc**, which he hopes to continue to improve. The page now extracts much documentation from the kernel source tree itself, links to various relevant standards, and also links to other external documents, both about the kernel and about other related technologies.

He affirms that there's still a lot of work left to do before the doc pages are really great. In particular, the automated extraction process is dependent on there being no errors in the way docs are housed in the kernel source tree, which is not the case. But hopefully, the existence of the kernel.org/doc pages will serve as a kind of regression test that will help others find and fix all those errors. As they do, the doc pages naturally should improve by reflecting those fixes.

A number of kernel developers have decided to take on the issue of **bufferbloat**, and **John W. Linville** recently created the **git://git.infradead.org/debloat-testing.git tree**, specifically to house patches relating to that issue.

Bufferbloat is a problem that may or may not exist within the whole Internet, caused by hardware engineers at a variety of private companies who created the physical technology underlying the Internet. These engineers may have been tempted by the low cost of RAM to create huge throughput buffers that regulate Internet traffic under times of heavy load. Proponents of the bufferbloat idea also speculate that these engineers did not bother to design their hardware to behave well if these buffers became saturated, because they assumed the buffers would handle all foreseeable situations. So, as the Internet has become more and more saturated over time, with video and other high-data traffic, these invisible buffers have been filling up, causing the entire Internet to approach a state of general choppiness in which data no longer is transmitted smoothly from place to place.

The Linux folks are attempting a kind of Herculean challenge. These buffers could exist anywhere along a network connection. They have not been acknowledged by the companies that may have created them, and it is virtually impossible for one system to test for these buffers. But, the Linux folks believe there are algorithms to be discovered that can address bufferbloat directly and help return the Internet to a smooth-flowing state or prevent the further degradation of the global system.

—ZACK BROWN

How You Can Have Mosquito Vision

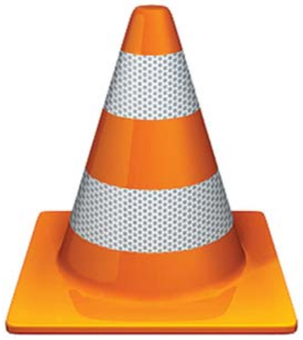
If you've ever been outside on a summer night, then come indoors to find you've been attacked by bloodthirsty mosquitoes, you know that those little buggers must be able to see in the dark. In fact, mosquitoes use infrared light to hone in on our body's heat in order to find our juicy bits.

Infrared light also is the same light that most television remote controls use for sending signals. If you've ever tried to troubleshoot a nonworking remote, you know it's frustrating that you can't see if the remote is "lighting up". Although it's difficult to convince a mosquito to tell you if your remote is working, it is possible to convince your cell phone, or any other digital camera, to do so.

Simply look at the infrared emitter at the business end of your remote through the view-screen of your favorite digital camera (or phone). If the remote is working, you'll see the light it's giving off very clearly. It works well and is much easier than training mosquitoes!

—SHAWN POWERS

NON-LINUX FOSS



As Linux users, it seems that we have little problem playing a wide variety of video formats. That is not always the case with other operating systems. Certainly with Windows 7, Microsoft has improved the number of video formats it supports, but it pales in comparison to VLC. If you want to play an AVI encoded with an obscure codec or watch a DVD, or if you have a hankering for multicast network video streaming, the VLC player from VideoLAN supports it all (**www.videolan.org/vlc**).

VLC isn't just for Windows either. OS X and Linux users alike can take advantage of its awesome compatibility. The best part? VLC is open source under the GPL! Enjoy video and freedom at the same time.—**SHAWN POWERS**

More Cool Projects from the LinuxJournal.com Archives

Cool Projects is one of our favorite topics at *Linux Journal*, and it's one we've explored quite a few times. There are so many cool projects in our archives, it's tough to know where to start.

Perhaps I could interest you in the programmable iRobot Create? The Create is an educational robot brought to you by the same folks who make the Roomba, so at the very least it'll be a great way to scare your pets and children! (See "Fun with the iRobot Create": **www.linuxjournal.com/article/10262**.)

Maybe you'd like to get super geeky and/or ferment some beer? Either would be a noble pursuit, so check out Kyle Rankin's article "Temper Temper" (**www.linuxjournal.com/article/10809**). My favorite quote says it all: "I mean, why wouldn't you power your fridge with Linux if you had the chance?"

And, if you haven't already, you must check out one of my all-time favorites: "Build Your Own Arcade Game Player and Relive the '80s!" by Shawn Powers (**www.linuxjournal.com/article/9732**). I dare you to read it and not be nostalgic for the 1980s. (Unless, of course, you don't remember the 1980s, and then get off my lawn! Only joking, I love you too.) Just looking at the pictures, I swear I can smell the rancid spilled Coke and hygienically challenged teenage boys in the mall arcade. Ah, the good-old days.—**KATHERINE DRUCKMAN**

CompTIA Linux+™ POWERED BY Linux Professional Institute

LPIC, for Those Needing Proof They're Geeks

The Linux Professional Institute has offered LPIC-1, LPIC-2 and LPIC-3 training for years. In these tough economic times, making yourself more employable is always a good plan. Unlike the Red Hat Certified Engineer certification process, the LPI certs are platform-agnostic. The requirements include a well-rounded list of objectives for certification. A cool bonus feature for the LPIC-1 exam is that it has the exact same requirements as the CompTIA Linux+ certification. So you can study once and get two certs! For more information, check out **www.comptia.org** and **www.lpi.org**.

—**SHAWN POWERS**

The Web Proxy Autodiscovery Protocol

WPAD certainly isn't new technology. In fact, it's been around for many years. However, it seems that many system administrators are unaware of its magic. Simply put, WPAD allows you to offer proxy information to users in your network without ever touching their computers. The feature is supported by most browsers, and in general, it "just works".

Although proxy information can be sent over DHCP, unfortunately, not all clients honor those settings. For maximum compatibility, it's best to have a local DNS record that points the domain "wpad" to a Web server. You put a configuration file named wpad.dat in the root level of that Web server, and clients get proxy information automatically, assuming they're configured to do so. (Most are by default; this is what your browser refers to as automatically detecting proxy settings.)

Here's a simple wpad.date file:

```
function FindProxyForURL(url, host)
{
   if (isPlainHostName(host) ||
       dnsDomainIs(host, "my.local.network.domain.org") ||
       (host=="127.0.0.1") )
       return "DIRECT";
   else
       return "PROXY my.proxy.server.address:8080";
}
```

For more detailed information on how to configure your custom wpad.dat file, check out **en.wikipedia.org/wiki/Proxy_auto-config**.

And, for more information on the Web Proxy Autodiscovery Protocol itself, see **en.wikipedia.org/wiki/Web_Proxy_Autodiscovery_Protocol**.—**SHAWN POWERS**

How-To: Release Stuck NFS Mounts without a Reboot

Computing environments may revolve around heavy usage of NFS infrastructure. Network areas are hosted and provided by storage file servers, with compute servers mounting the exported areas into their directory tree. Periodically, the mounts expire when not in use and are removed from the directory tree on local machines.

If NFS traffic between file servers and compute servers is disrupted, machines holding active mounts will not be able to release them. This could happen in several cases. A potential root cause for the problem might be a network outage; however, this issue will be resolved the moment network traffic is resumed. A far more serious scenario is when a file server stops its service due to malfunction or coordinated maintenance, including an EOL (End of Life) cycle. Any host holding active mounts of areas exported by a file server that is no longer reachable will not release them.

This situation leads to so-called stuck mounts. Any process running on an affected computer server waiting for NFS activity will remain in uninterruptible sleep state without any way to kill it. Up to date, the only sensible solution was to reboot affected hosts. However, critical computer servers, including infrastructure and interactive-use machines, cannot be rebooted without prior coordination. This leaves a time window until the next reboot, during which the affected hosts will continue running in an unstable state.

We provide a rebootless solution to this problem by bringing up a fake file server, a dedicated host assigned the same IP like an unreachable file server, which then rejects the NFS requests from compute servers, freeing them. The special host uses its own network architecture, allowing it to be moved to any VLAN (Virtual LAN) used by file servers. The IP change is done using a Web interface. (A VLAN is a group of hosts with a common set of requirements that communicate as if they were attached to the same broadcast domain, regardless of their physical location. A VLAN has the same attributes as a physical LAN, but it allows for end stations to be grouped together, even if they are not located on the same network switch.

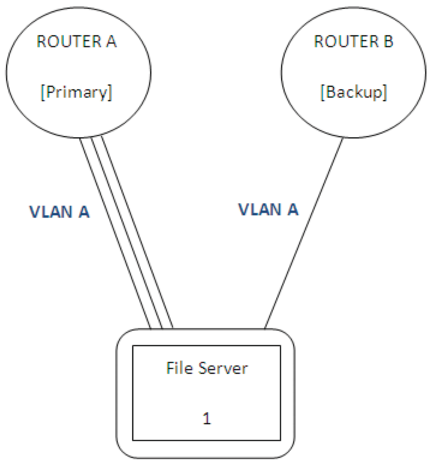


Figure 1. Connection to NFS Router

Network reconfiguration can be done through software instead of physically relocating devices.)

The stuck NFS mounts solution consists of several key components:

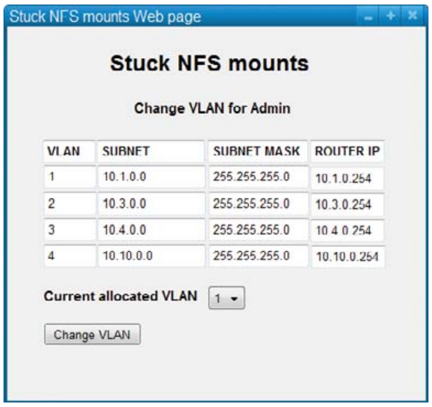
- A dedicated host used for faking the IP addresses—we use a virtual machine with minimal hardware requirements. The virtual machine is assigned a separate NIC in the virtualization server. This host is known as the Admin server.
- A network infrastructure that supports on-the-fly assignment of the "fake" file server (Admin) to the file server VLANs. Admin is connected directly to the NFS router using a single standard NFS connection, with a 3Gb channel to primary port and 1Gb channel to backup port (Figure 1).
- A Web interface for assigning Admin to file server VLANs.
- A mechanism to change the IP address on Admin, which can be automated (scripted) or manual.

In our site, Admin is a virtual machine hosted on a virtualization server. The host runs a standard enterprise Linux image; however, it is used only for IP changes and no other purposes, and is configured with minimal disk space and memory. The virtual machine is assigned its own NIC in the virtual server network configuration.

The NFS stuck mounts VLAN change is executed using a Web interface. The GUI allows system administrators with no networking knowledge or access permissions to change the VLAN allocation for the specific router port where the dedicated server is connected to the desired VLAN. The GUI has a static configuration file, with the server name, router name and dedicated port. Once called through the Web interface, it uses SNMP to collect the data from the router and present the table in the middle, the current status and the VLAN drop-down menu. Figure 2 shows a mockup of the Web interface.

On clicking the Change VLAN button, the script, using SNMP, changes the VLAN allocation for the specific port. Figure 3 shows the network flow.

Stuck mounts usually are identified by indirect symptoms exhibited by affected hosts, such as increased load due to a large number of processes in an uninterruptible sleep state or lsof not being able to complete its run. The first step is to open the Web interface and change the VLAN to that matching the unreachable file server. Once that's done, data-center operations technicians are instructed to change the IP address and default gateway on the Admin server to match that of the unreachable file server. Alternatively, the IP address and default gateway can be configured manually by logging in to the machine via the virtualization console and performing the change. The host IP address can be changed



VLAN	SUBNET	SUBNET MASK	ROUTER IP
1	10.1.0.0	255.255.255.0	10.1.0.254
2	10.3.0.0	255.255.255.0	10.3.0.254
3	10.4.0.0	255.255.255.0	10.4.0.254
4	10.10.0.0	255.255.255.0	10.10.0.254

Current allocated VLAN 1

Figure 2. Part of the Web Interface for VLAN Change

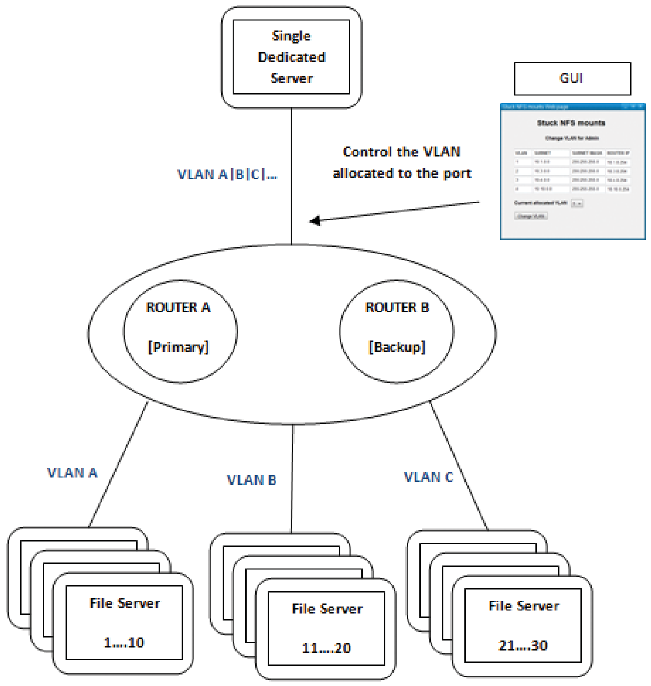


Figure 3. VLAN Change Diagram

under /etc/sysconfig/network/ifcfg-ethx or ifcfg-eth-idxxx; the default gateway is configured under /etc/sysconfig/network/routes.

Once those two steps are complete, the network service on Admin needs to be restarted with `/etc/init.d/network restart`. The "fake" file server will come up and start rejecting NFS requests from affected hosts, releasing the stuck processes. At this stage, system administrators also can try to umount the stuck mounts.

We encountered a number of challenges in setting up this solution. On the network side, file servers are connected to routers; whereas compute servers are connected to switches. This necessitated an innovative approach by the network operations team, which resulted in the dedicated connection to the NFS router.

Furthermore, this solution requires allocating site assets to be used only rarely. We chose the virtual machine because of reduced cost and higher flexibility in the setup rather than using a physical machine. Finally, the IP/VLAN change procedure requires discipline and some skill on behalf of system administrators to use it effectively.

We first tried the stuck NFS solution as a hack by manually changing the IP address of a standard computer server located on the same segment as the unreachable file server more than a year ago. The attempt proved successful, and we were able to save a number of important infrastructure machines with stuck mounts, preventing downtime and reboots.

Prompted by this early success, we implemented the full solution and have used it on a number of occasions. In July 2010, we prevented a number of Samba servers from being rebooted, alongside another 21 compute servers.

The stuck NFS mounts solution has a direct impact on our ability to recover from unexpected file server outages. We also are able to mitigate scheduled file server EOL leftovers without reboots. In fact, one might argue that the solution creates leniency that may lead to lesser discipline in using strict file server EOL procedures, which requires all mounts to be removed before the file servers are shut down. We believe that the stuck NFS mounts solution does not replace hard work and adherence to procedures and should be used only in an emergency.

The stuck NFS mounts solution is a valuable asset and a necessity in large and dynamic environments. It allows us to maintain high uptime of important machines, without unnecessary reboots due to mistakes or file server traffic disruptions.

Our solution is simple and effective, and it has proven its worth on several occasions in real-life circumstances. The implementation is transparent to users and allows system administrators with no network operations of NFS permissions or extensive knowledge to recover systems quickly and easily. We recommend its use as a standard in computing environments that rely on NFS infrastructure.

—IGOR LJUBUNCIC, DAVID ISRAEL, RAFI STEINBACH, YAAD HADAR and YAIR RAJWAN

They Said It

I believe that a scientist looking at nonscientific problems is just as dumb as the next guy.
—Richard Feynman

For a successful technology, reality must take precedence over public relations, for Nature cannot be fooled.
—Richard Feynman

There are 10^{11} stars in the galaxy. That used to be a huge number. But it's only a hundred billion. It's less than the national deficit! We used to call them astronomical numbers. Now we should call them economical numbers.
—Richard Feynman

The first principle is that you must not fool yourself—and you are the easiest person to fool.
—Richard Feynman

You can know the name of a bird in all the languages of the world, but when you're finished, you'll know absolutely nothing whatever about the bird....So let's look at the bird and see what it's doing—that's what counts. I learned very early the difference between knowing the name of something and knowing something.
—Richard Feynman

CALL FOR NOMINATIONS FOR THE 2011 *LJ* READERS' CHOICE AWARDS

2011 Readers' Choice LINUX JOURNAL

The 2011 Readers' Choice awards are just around the corner, and we know you're all eager to vote for your favorites, but first, we want to make sure they're all included. Nominate your top picks for this year's awards at **www.linuxjournal.com/rc2011** from July 6–20.

A Sage Experience

Several mathematics packages under the GPL are available on-line. You can find a package to help with almost any work you need to do, and in a way, you almost are spoiled for choice. But, if your work covers several areas of research, the one thing you likely are missing is a uniform interface to these tools. Sage fills that role (**www.sagemath.org**). Like most open-source software, Sage is available as packages for Linux, Windows, Mac OS X and Solaris. You also can download a live CD that boots up in a Linux environment based on Puppy Linux. And, of course, you always can download the source code and build it yourself. All of these options are available for download from the main site. The documentation also is very useful.

The basic idea behind Sage is to provide a unified experience when using the bundled packages. This is done through a Python framework wrapped around all of these packages. You can use Sage interactively in two different ways: a graphical interface or a terminal interface. When you start Sage, it starts up a Web server with a default port 8000. Once it has finished starting up (and it can take some time on some systems), simply point your browser at http://localhost:8000 (Figure 1). With this interface, you can create and manage your worksheets. The terminal interface (Figure 2) will be familiar to anyone who has used the command-line interface to programs like Maple, Mathematica, Maxima and so on. You also can use Sage in your own programs. You can link Sage into your own interpreted or compiled code, or you can write standalone Python scripts. This gives you access to all of the included libraries and packages, like GMP, PARI or Maxima.

If you've used other computer mathematics packages, Sage basics should be easy to pick up. The assignment operator is =. The usual comparison operators are available: ==, <=, =>, < and >. The operators +, * and - mean what you expect. With division, you need to be more careful when dealing with differences between reals and integers. The % operator gives you the remainder from a division operation. If you are interested in the

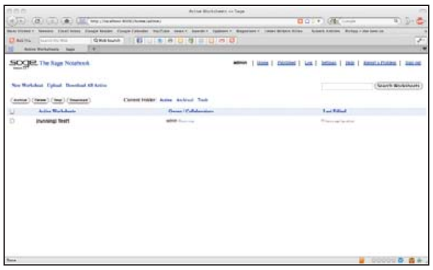
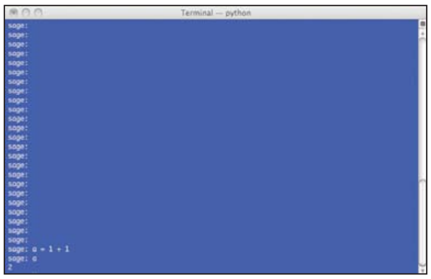


Figure 1. The starting point in the Web interface is a display of your worksheets, along with tools to manage those worksheets and make more.


```
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage:
sage: a = 1 + 1
sage: a
2
```
Figure 2. The console session allows you to interact with Sage in a form that may be more familiar to users of Maxima and Maple.

SILICON MECHANICS

visit us at www.siliconmechanics.com
or call us toll free at 866-352-1173

Silicon Mechanics is proud to announce their sponsorship of Wave Vidmar and the Ocean Row Solo North Atlantic Challenge 2011, a solo row from west-to-east across the North Atlantic.

Expedition

Departure:	July 2011, Cape Cod
Destination:	England
Duration:	140 days (estimated)

Philosophy
"It's not just one man rowing across an ocean, it's hundreds or thousands of people rowing across an ocean. It's a cooperative effort, that helps push the boundaries and limits, giving real-life examples to others in what they too can possibly achieve."

Equipment
"We only work with the best of the best. When you trust your life to equipment and/or services, you can't afford to have second-best. We only accept sponsors that have superior product, value, and policies."

Goals
Education and research: Wave's Science Education Adventure Program reaches hundreds of thousands of students around the world.

Boat
Liberty is 24 feet long, with a 6.5 foot beam, and weighs 1400 pounds fully laden. It is the most advanced, technologically equipped ocean row boat in the world. Shaped and designed using 3D modeling programs, the boat was tested under simulated extreme conditions long before it was ever constructed.

Mission Control
Silicon Mechanics and Intel have partnered to donate the hardware that will run mission control functions for the expedition. The Intel® Xeon® Processor E3-1200 Series (formerly codenamed "Sandy Bridge") is at the heart of his customized high-performance workstation.

For more information about our Intel Xeon Processor-based servers visit www.siliconmechanics.com/xeon

For more information about Wave Vidmar and Ocean Row Solo, visit www.oceanrowsolo.com. To track the expedition, visit www.siliconmechanics.com/ors.

integer quotient, you can get it by using //. Exponentiation can be achieved using **, or ^, which is a synonym. You also have access to several mathematical functions, such as functions like sin(), cos() and tan(), and basic functions, like sqrt() or exp().

One of Sage's strong points is the available documentation. From within the system, you can access lots of help. You can pull up a description of a function, along with examples, with the command `func?`. For example, if you want to know more about the sine function, type:

```
sage: sin?
Base Class:    <class 'sage.functions.trig.Function_sin'>
String Form:   sin
Namespace:     Interactive
File:           /Users/bernardj/Desktop/Sage-4.6.2-OSX-64bit-10.6.app/
               ➥Contents/Resources/sage/local/lib/python2.6/
               ➥site-packages/sage/functions/trig.py
Definition:    sin(self, *args, coerce=True, hold=False,
                   ➥dont_call_method_on_arg=False)
Docstring:
       The sine function.

       EXAMPLES:

          sage: sin(0)
          0
          sage: sin(x).subs(x==0)
          0
          sage: sin(2).n(100)
          0.90929742682568169539601986591
          sage: loads(dumps(sin))
          sin

       We can prevent evaluation using the ``hold`` parameter:

          sage: sin(0,hold=True)
          sin(0)

       To then evaluate again, we currently must use Maxima via
       ``sage.symbolic.expression.Expression.simplify()``:
```

If you don't know the full name of the function, you can try Sage's Tab completion. It behaves just like the Tab completion in bash, so it should be comfortable for users of most Linux distributions.

Creating your own functions is easy. You can define a function with the command `def`. Input variables simply need to be listed in the definition. You don't need to specify type for any variables. The body of your function follows Python coding rules. Unlike other programming languages, you don't use special characters (like parentheses) to define sections of code. Blocks of code are defined by indentation level. Lines of code at the same indentation level belong to the same block. As a really silly example, let's say you wanted a function to calculate the cube of a number. You could define a function called cube in this way:

When you partner with Silicon Mechanics, you get more than affordable, high-quality HPC — you get a team of Experts pulling right along with you.

Expert included.

```
sage: def cube(x):
        answer = x * x * x return answer
```

Then, you would be able to call this function just like you would with any other function:

```
sage: cube(3) 27
```

Within Sage, there is support for the common fundamental data types: integers, reals, strings and so on. There also is support for the compound data type, list. The list is opened with [and closed with]. Also, a list can contain any combination of data types. For example, a list could be defined as:

```
sage: a = [1, 5, "a string", 1/2, exp(10)]
```

You can access individual elements with an index. The index is 0-based. So, to pull out the third entry, you would use:

```
sage: s[2]
'a string'
```

Use the command `len` to find the length of your array. You can add or delete elements with the command `.append()` or `del`. As an example, if you want to delete the string and tack it onto the end of the array, you could use:

```
sage: b = a[2]
sage: del a[2]
sage: a.append(b)
```

A more structured compound data type is the dictionary or associative array. This is like a list, except each entry has a label along with a value. You could set up a mapping between letters and numbers with:

```
sage: mapping = {'a':1, 'b':2, 'c':3}
sage: mapping['b']
2
```

As you can see, you also define a dictionary with { and }, rather than [and]. Even more complex data types can be defined by creating a new class. A class has a couple standard functions you will want to override, __init__ and __repr__. The function __init__ gets called whenever a new instance of the class is created. You can place any initialization code here. The function __repr__ gets called whenever you want a representation of the object printed.

A common task you likely will want to do is solve equations, either exactly or numerically. You can do this with the `solve` command. Define the variables you want to use with the `var` command. Here's a simple example:

```
sage: x = var('x')
sage: solve(x^2 + 3*x +2, x)
[x == -2, x == -1]
```

You also can solve equations with multiple variables:

```
sage: x, y = var('x', 'y')
sage: solve([x+y==6, x-y==4], x, y)
[[x == 5, y == 1]]
```

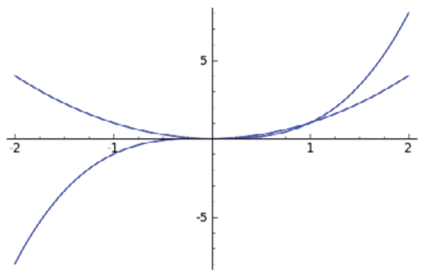


Figure 3. You can plot multiple functions on the same graph.

If you're more interested in calculus, you can do differentiation and integration—for example:

```
sage: x = var('x')
sage: diff(sin(x), x)
cos(x)

sage: y = var('y')
sage: integral(y, y)
1/2*y^2
```

Sage is capable of handling 2-D and 3-D graphs. You can draw shapes, such as circles or polygons, by using commands with parameters to define dimensions. You also can draw graphs of functions with the `plot` command:

```
sage: x = var('x')
sage: plot(x^2, (x, -2, 2))
```

This displays your plot immediately. If you want to hold off on the actual display, you can assign the plot to a variable with:

```
sage: p1 = plot(x^2, (x, -2, 2))
```

To plot this function, use `show(p1)`, which is useful if you want to plot more than one function on the same graph. For example, the following produces the graph shown in Figure 3:

```
sage: x= var('x')
sage: p1 = plot(x^2, (x, -2, 2))
sage: p2 = plot(x^3, (x, -2, 2))
sage: show(p1+p2)
```

You can write your own Sage scripts for more complex applications. These can be written in a simple text file. Once you have written one, you can read it in and run it within Sage with the `load` command. For example, to run a script named test1.sage, do:

```
sage: load "test1.sage"
```

Using `load`, sage reads in the file only once. If you are working on developing a script, use the `attach` command instead. This attaches the script to your Sage session and reloads it whenever it changes, which is useful while you develop your own procedures.

The last step you probably will want to do is generate some kind of documentation around a fabulous discovery you made. Sage includes LaTeX functionality to allow you to produce really pretty output. You also can access jsMath for pretty printing mathematical equations.

This article barely scratches the surface of what you can do with Sage. With nearly 100 packages included, going through the documentation on the Web site is a must.—**JOEY BERNARD**

20th USENIX SECURITY SYMPOSIUM

San Francisco, CA • August 8–12, 2011

Join us for a 5-day tutorial and refereed technical program for researchers, practitioners, system administrators, system programmers, and others interested in the latest advances in the security of computer systems and networks.

The Program Begins with 2 Days of Training Taught by Industry Leaders, Including:

- Richard Bejtlich on TCP/IP Weapons School 3.0 (2 Day Class)
- Rik Farrow on SELinux—Security Enhanced Linux
- Jim DelGrosso on an Overview of Threat Modeling

And Continues with a 3-Day Technical Program Including:

Keynote Address
Charles Stross, Hugo award-winning author, on "Network Security in the Medium Term: 2061–2561 AD"

Paper Presentations
35 refereed papers that present new research in a variety of subject areas, including securing smart phones, understanding the underground economy, and privacy- and freedom-enhancing technologies

Plus:

- Invited Talks
- Panel Discussions
- Rump Session
- Poster Session
- Birds-of-a-Feather Sessions (BoFs)

Co-Located Workshops Include:

EVT/WOTE '11: 2011 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections, August 8–9, 2011

CSET '11: 4th Workshop on Cyber Security Experimentation and Test, August 8, 2011

FOCI '11: USENIX Workshop on Free and Open Communication on the Internet, August 8, 2011

WOOT '11: 5th USENIX Workshop on Offensive Technologies, August 8, 2011

HealthSec '11: 2nd USENIX Workshop on Health Security and Privacy, August 9, 2011

HotSec '11: 6th USENIX Workshop on Hot Topics in Security, August 9, 2011

MetriCon 6.0: Sixth Workshop on Security Metrics, August 9, 2011

Stay Connected...

http://www.usenix.org/facebook

http://twitter.com/USENIXSecurity

Register by July 18 and Save! **www.usenix.org/sec11/lj**

WEB HOSTING. TWICE AS SECURE.

1&1 DUAL H

Double Security, Double Reputability:

No one can afford downtime of their website...
1&1 is now offering dual hosting for the ultimate security of your website. Your website is hosted in two different locations in our data center. If the first location is unexpectedly interrupted, your site will automatically continue running in the second location – without any data loss.

MEMBER OF united internet

OSTING

No other web host offers as much expertise, know-how and quality as 1&1:

1&1 combines over 20 years of web hosting experience with the latest technology in our high-speed and high-performance American data center. More than 1,000 IT professionals will continue to develop our top performance web solutions for years to come. **NEW:** 1&1 is pleased to offer double security for your website with 1&1 Dual Hosting! All at unbeatably low prices!

Double Security:
1&1 Dual Hosting

Fast Global Network Connectivity:
210 GBit/s Connection

Top Performance:
High-end Servers

Environmentally Responsible:
100% Renewable Energy

Solid Technical Foundation:
Over 1,000 In-house Developers

NEW!

1&1 DUAL UNLIMITED

- 3 FREE Domains
- FREE Private Domain Registration
- UNLIMITED Web Space
- UNLIMITED Traffic
- UNLIMITED FTP Accounts
- UNLIMITED E-mail Accounts (2 GB)
- UNLIMITED Mailing Lists
- 20 Microsoft® SQL Databases
- ASP, .NET, AJAX, LINQ, PHP, Perl, SSI
- GeoTrust® Dedicated SSL Certificate
- NEW! 1&1 SiteAnalytics
- 99.99% Uptime
- 24/7 Toll-free Customer Support

1&1 DUAL UNLIMITED

$9.99 per month* (36 month term)

$11.99/month	(24 month term)
$12.99/month	(12 month term)
$13.99/month	(3 month term)

Please see following page for more 1&1 DUAL HOSTING packages.

1&1®

1-877-GO-1AND1 www.1and1.com

1-855-CA-1AND1 www.1and1.ca

* Visit www.1and1.com for full promotional offer details. Program and pricing specifications and availability subject to change without notice. 1&1 and the 1&1 logo are trademarks of 1&1 Internet AG, all other trademarks are the property of their respective owners. © 2011 1&1 Internet, Inc. All rights reserved.

Backbone.js and Rails

How to combine two MVC frameworks—one on the client, one on the server—for even more powerful applications.

REUVEN M. LERNER

Last month, I started looking at Backbone.js, an increasingly popular framework for Web applications written in JavaScript. If you think that Web development frameworks exist only on the server, you're in for a surprise. A growing number of frameworks have sprung up on the browser, written in JavaScript and meant to execute completely within that environment. You can think of such frameworks as having less to do with the Web and more to do with the creation of a full-fledged application inside a browser, using a combination of HTML, CSS and (of course) JavaScript.

Many client-side frameworks use the established MVC (model-view-controller) paradigm that has existed for several decades and has proven to be both useful and powerful in developing both Web and desktop applications. By keeping the code organized in a standard way, the design and maintenance of the application are simplified to a great degree, allowing you to concentrate on your application's domain, rather than more general architectural decisions.

In MVC, one set of objects (the models) represents the data, sometimes known as the "business objects", and all of their associated logic. The "view" consists of what the user will see on the screen, either initially or after some manipulations have been performed. Finally, the "controller" objects receive requests from the user and route them to the appropriate models and views. Each implementation of MVC is slightly different, but the separation of authority into these three categories helps a great deal.

Last month, I built a very simple appointment calendar, allowing you to indicate with whom you are meeting, and when (which has been broken into two files, Listings 1 and 2). Aside from the numerous usability problems that were associated with that application (not surprising for a magazine tutorial), there was one clear issue with this appointment calendar. The moment you close your browser window, the entire calendar disappears from memory, never to return.

Listing 1. appointments.html

```
<!DOCTYPE html>
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/
➥1.4.4/jquery.min.js"></script>
<script src="http://ajax.cdnjs.com/ajax/libs/underscore.js/
➥1.1.4/underscore-min.js"></script>
<script src="http://ajax.cdnjs.com/ajax/libs/backbone.js/
➥0.3.3/backbone-min.js"></script>
<script src="/javascripts/appointments.js"></script>

<title>Appointments</title>
</head>
<body>
<h1>Appointments</h1>

<table>
<tr>
<th>Person</th>
<th>Date/time</th>
<th>Note</th>
</tr>
<tr id="new-appointment">
<td><input type="text" name="person" /></td>
<td><input type="text" name="starts_at" /></td>
<td><input type="text" name="note" /></td>
</tr>
<tr align="center">
<td colspan="3"><input type="button" id="add-appointment"
 ➥value="Add Appointment"/ ></td>
</tr>
</table>

<hr />

<p>Number of appointments: <span id="number-of-appointments">
➥</span></p>

<table id="appointments">
<tr>
<th>Person</th>
<th>Date/time</th>
<th>Note</th>
</tr>
</table>

</body>
</html>
```

For me, and I assume for many other Web developers, this is a new situation. For years, my Web applications largely have been a visual representation of what I had stored in the database. True, users always could change things without clicking a "submit" button, but AJAX reduced that risk. And, even if you lost one page's worth of data, it usually was the minority of what someone had entered. In the world of JavaScript applications, by contrast, you don't lose only a little bit, you lose the entire thing! This is clearly unacceptable.

For this reason, browser-based MVC applications increasingly are hooking together with server-side MVC applications. The JavaScript model in such an application saves itself not to disk or to a database, but via a RESTful URL on a server somewhere. That server is, in turn, running its own MVC application, and accepts the data, stores it and makes it available to other applications. I've started to call these applications "MVC-squared", because they involve two separate MVC applications—although clearly, neither is all that useful without the other. Different client-side frameworks tackle this in different ways, and my hope is to cover a few other JavaScript frameworks in the coming months, so you can compare the different techniques they use to accomplish similar goals.

Setting Up Your Server

To begin, you need to set up a Web application on your server. I use Ruby on Rails here, but so long as you're using a server application that understands and adheres to the REST conventions, you should be just fine. First, I create a Rails application called "appointments" that uses PostgreSQL as its back-end database:

```
$ rails new appointments -d postgresql
```

I create a PostgreSQL database for the development environment:

```
$ createuser -P appointments

Enter password for new role:
Enter it again:
Shall the new role be a superuser? (y/n) n
Shall the new role be allowed to create databases? (y/n) y
Shall the new role be allowed to create more new roles? (y/n) n

$ createdb -U appointments appointments_appointments_development
```

Given that I gave the "appointments" user a password (also "appointments"), I have go into config/database.yml and add a value in the "password" field for the "development" environment. I always like to test that I can connect to the database using the `rails db -p` command, which connects me to the "psql" client program using the password in the development environment. If this works, I know my database connection is working correctly.

Now, let's create a resource in the Rails application—an appointment. The most important thing here is to ensure that your data is aligned between the server and the client, in both name and type. In this example application, I'm storing each

1&1 DUAL HOSTING

SECURE & SAVE!

1&1 DUAL BASIC
- 1 FREE Domain
- 10 GB Web Space
- UNLIMITED Traffic
- NEW! 5 FTP Accounts
- NEW! 1&1 SiteAnalytics

$2.99 per month* (36 month term)
$3.99/month (24 month term)
$4.99/month (12 month term)
$6.99/month (3 month term)

1&1 DUAL ADVANCED
- 2 FREE Domains
- 150 GB Web Space
- UNLIMITED Traffic
- NEW! 50 FTP Accounts
- NEW! 1&1 SiteAnalytics

$4.99 per month* (36 month term)
$5.99/month (24 month term)
$6.99/month (12 month term)
$8.99/month (3 month term)

.com
Now only $7.99/first year*

.biz
Now only $3.99/first year*

1&1 DOMAINS
Starting at
$3.99 first year*
FREE Private Registration!

HURRY – OFFER ENDS 6/30/2011!

*Visit www.1and1.com for full promotional offer details. Program and pricing specifications and availability subject to change without notice.

More special offers available online.

www.1and1.com
www.1and1.ca

1&1®

1&1 and the 1&1 logo are trademarks of 1&1 Internet AG, all other trademarks are the property of their respective owners. © 2011 1&1 Internet, Inc. All rights reserved.

Listing 2. appointments.js

```
$(document).ready(function () {
    Appointment =
    Backbone.Model.extend({
    url : function() {
    var base = '/appointments/';
    if (this.isNew()) return base;
    return base + this.id + '.json';
    }

});

Appointments =
Backbone.Collection.extend({
    model: Appointment,
    url: "/appointments.json",

    update_appointment_counter: function() {
    $("#number-of-appointments").html(this.length);
    },

    initialize: function(models, options) {
    this.bind("add", options.view.add_appointment_row);
    this.bind("add", this.update_appointment_counter);
    }
});

AppController =
Backbone.Controller.extend({
    routes: {
    "" : "index"
    },

    index: function() {
    appview.appointments.fetch({
    success: function(response) {
    appview.appointments.each(function(item) {
    appview.add_appointment_row(item);
    }
    );
    $("#number-of-appointments").html(appview.appointments.length);
    },

    error: function(response) {
    alert("error loading!");
    }
    });
    }
    });



AppView =
Backbone.View.extend({
    el: $("body"),

    initialize: function() {
    this.appointments = new Appointments(null, {view:this});
    },

    events: {
    "click #add-appointment": "add_appointment"
    },

    add_appointment: function() {
    var person = $("#new-appointment td input[name=person]").val();
    var starts_at = $("#new-appointment td
                          ➥input[name=starts_at]").val();
    var note = $("#new-appointment td input[name=note]").val();

    var new_appointment =
    new Appointment({person: person, starts_at: starts_at,
                        ➥note: note});
    this.appointments.add(new_appointment);
    new_appointment.save();
    },

    add_appointment_row: function(model) {
    $("#appointments").append("<tr><td>" + model.get('person')
     ➥+ "</td>" +
    "<td>" + model.get('starts_at') + "</td>" +
    "<td>" + model.get('note') + "</td></tr>");
    }
    });

var appview = new AppView;
var myController = new AppController;
Backbone.history.start();

});
```

appointment as three fields: the name of the person with whom I'm meeting ("person"), the date and time of the meeting ("starts_at") and notes about the meeting ("notes"). The Backbone.js application uses these same names, but assumes that all are text strings; I define the starts_at column as being of type "timestamp".

Now, let's create the resource, along with a skeleton controller and views:

```
rails g scaffold appointment person:text starts_at:timestamp note:text
```

I should note that this is a highly non-normalized database definition, which means that it'll be highly inefficient, as well as inflexible. I'm a big fan of normalization, but right now, that's less important than seeing how the parts fit together.

The first and most important thing that this generator does is create a migration (in db/migrations). The migration looks like this:

```
class CreateAppointments < ActiveRecord::Migration
  def self.up
create_table :appointments do |t|
  t.text :person
  t.timestamp :starts_at
  t.text :note

  t.timestamps
end
  end

  def self.down
drop_table :appointments
  end
end
```

I always like to tighten up the definitions a bit, as well as provide some indexes that'll allow me to search through the database more efficiently:

```
class CreateAppointments < ActiveRecord::Migration
  def self.up
create_table :appointments do |t|
  t.text :person, :null => false
  t.timestamp :starts_at, :null => false
  t.text :note, :null => true

  t.timestamps
end

add_index :appointments, :person
add_index :appointments, :starts_at
add_index :appointments, :note
  end

  def self.down
drop_table :appointments
  end
end
```

Run the migration with `rake db:migrate`. Assuming that all went well, the database now contains an "appointments" table, which you can access via ActiveRecord.

The generator did a bit more than just create the migration, a basic controller and views; it also added the line:

```
resource :appointments
```

to the config/routes.rb file. This one line has a great deal more influence than you might think. It tells Rails that you want to expose the "appointments" object as a resource, using the seven standard controller actions (index, show, new, create, edit, update and delete) that correspond to HTTP request methods. By ensuring that your object is exposed using REST, you more easily can hook it up to your Backbone.js application.

You also will need to adjust your controller (Listing 3). Backbone.js communicates with the server by default using JSON, which means that in each scaffold controller action, you either can add a new line for the JSON format or replace it entirely. For example, after adding JSON to the "index" action, it looks like this:

```
def index
  @appointments = Appointment.all

  logger.warn "Found '#{@appointments.size}' appointments"

  respond_to do |format|
    format.html # index.html.erb
    format.xml  { render :xml => @appointments }
    format.json  { render :json => @appointments }
  end
end
```

Removing non-JSON entirely lets you rewrite it as:

```
def index
  Appointment.all.to_json
end
```

which is both shorter and more understandable.

Now, this example Rails application might be small, but it already works. You can start the server with `rails s` and visit it at port 3000 (the default). Because I didn't remove the original index page from the "public" directory, simply going to localhost:3000 is going to show just the default "Welcome to Rails" page. But, if you go to http://localhost:3000/appointments/, you see that the scaffolding is available, showing a list of the current appointments (that is, nothing), as well as a link to create a new appointment.

MVC, Please Meet MVC

You now have two distinct applications that share an underlying data model. On the browser is the Backbone.js application, whose "Appointment" model and associated collections allow you to create appointments, as well as view them. And there's the Rails application, which also has an "Appointment" model, as well as ways to view them. Now, let's connect them, such that the Backbone.js model will save and receive its data from the Rails application.

You can get this all to work by adding a "url" method to your model, such that it knows the URLs it

Listing 3. appointments_controller.rb

```
class AppointmentsController < ApplicationController
 def index
   render :json => Appointment.all
 end

 def show
   render :json => Appointment.find(params[:id])
 end

 def new
   @appointment = Appointment.new
 end

 def edit
   @appointment = Appointment.find(params[:id])
 end

 def create
   params.delete('action')
   params.delete('controller')
   @appointment = Appointment.new(params)

   if @appointment.save
     render :json => 'Appointment was successfully created.'
   else
     render :json => 'Error creating appointment.'
   end
 end

 def update
   @appointment = Appointment.find(params[:id])

   if @appointment.update_attributes(params[:appointment])
     render :json => 'Appointment was successfully updated.'
   else
     render :json => 'Error updating appointment.'
   end
 end

 def destroy
   @appointment = Appointment.find(params[:id])
   @appointment.destroy

   render :json => "Appointment destroyed."
 end
end
```

can use to retrieve and store data. Let's also add a controller, which will make it easier to organize the code, as well as accomplish various tasks with it.

It turns out that although Backbone.js is an MVC framework, it grew iteratively and over time. Controllers weren't originally part of the framework, which means that as you saw in last month's code example, you even can get away without a controller at all, using only views and models. That alone is quite different from Rails MVC, which wouldn't do much without a controller.

Another difference between Backbone.js controllers

Controllers, like models, collections and views, are objects defined by Backbone.js, which you extend in order to use.

and their Rails counterparts is that in Backbone.js, routes—the mappings between URLs and controller actions—are defined right inside the controller, alongside variables and functions. (By contrast, Rails puts such information in a separate configuration file, config/routes.rb.) I also should note that talking about "routes" in the context of a JavaScript application is a bit strange, because the whole point of such an application is that you stay within the same URL. Thus, routes in Backbone.js refer to the portion of the URL following the hash (#) character, originally intended to be used as a named anchor, but today used more by JavaScript.

This means that a route of `""` (that is, the empty string) will be run by default when your application is opened. A route of `abc` will be run when the URL ends with #abc (entered manually in the URL, or from within a link), and a route of `abc/:def` will be run when the URL ends with #abc/SOMETHING, where SOMETHING can be anything at all and is passed as a parameter to the function invoked by the controller.

Controllers, like models, collections and views, are objects defined by Backbone.js, which you extend in order to use. A simple controller might look like this:

```
AppController = Backbone.Controller.extend({
routes: {
     "" : "index",
     "say/:something" : "say"
},

index: function() {
  alert("Now invoked controller index");
},

say: function(something) {
  alert("You said " + something + "!");
}

});
```

This controller has two defined routes. If the hash character isn't present, it will invoke the "index" function. If the hash character is there, and if there is the word "say", a parameter and then some text following it, it will invoke the "say" function, printing the contents of the parameter.

Notice that the routes here are extremely flexible. You aren't restricted to the standard Rails-style RESTful routes. But of course, you aren't dealing with REST anymore, because this is the application itself, not a resource for other applications to use. If you insert the following HTML into your document:

```
<p>Internal link <a href="#say/something_at_all">here</a></p>
```

this adds a link that will result in the invocation of `say("something_at_all")`.

If you put the above into your Backbone.js application and reload, you'll soon see that...well, nothing happens. That's because you might have defined the controller object, but you haven't created it. Thus, you need to put the following line into your code to create a new AppController object:

```
var myController = new AppController;
```

Now if you try to reload, you'll find that...well, once again, things won't work. That's because controllers in Backbone.js are integrated with a history object (Backbone.History), which keeps track of the URLs you recently visited. If your browser supports the "onhashchange" event, Backbone.js will take advantage of it. But if it doesn't, Backbone.js will poll the browser 20 times each second, checking to see if the URL has changed, and if so, firing the appropriate function, based on the current URL.

So, you need to start the Backbone.History object:

```
Backbone.history.start();
```

With all of these in place—routes inside the controller object definition, an actual controller object and a running history object—Backbone.js springs into action. If you simply go to the URL for your page, the `""` (empty string) route will execute the "index" function.

In a real-world application, you don't want the index function to put up an alert box. Rather, you'll almost certainly want it to load any data that already might have been saved on the server. In

YMI RackMountPro.com

Storage. IT's our business. Solid storage, flexible solutions.

YMI 5U STORKING

The YMI 5U STORKING is exactly what you are looking for.

intel Xeon inside

Powerful. Intelligent.

STOP HERE

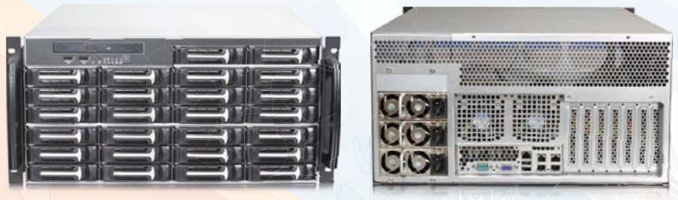
- **Modularized Chassis Design** for easy upgrade and maintenance
- **High Efficiency** 1150 watt 2+1 redundant power supply
- **High Quality 6Gb/s SAS Backplanes** and data cables
- **Intel® 5520** chipset server board
- **Dual Intel® Xeon® Processors 5600 Series**
- **12GB DDR3/1333 Memory** (upgradable to 192GB)
- **Dual Intel®** 82576 Gigabit Ethernet ports
- **Onboard VGA**
- **6Gb/s MegaRAID®** internal RAID controller*

LSI

*LSI MegaRAID® 9280-24i4e: 24 internal ports plus 4 external JBOD expansion ports for up to 240 SATA or SAS drives. Supports LSI™ MegaRAID® CacheCade™ software which allows use of SSDs as controller cache. Please contact us for more details.

5U 24 BAYs 72TB **$13,999**

24 x 3TB 7200rpm 64MB, RAID edition SATA 6.0Gb/s HDDs



5U 36 BAYs 108TB **$19,999**

36 x 3TB 7200rpm 64MB, RAID edition SATA 6.0Gb/s HDDs

LSI, the LSI logo, MegaRAID are trademarks or registered trademarks of LSI Corporation in the U.S. and other countries.
Intel, the Intel logo, Xeon, and Xeon Inside are trademarks or registered trademarks of Intel Corporation in the U.S. and other countries.

Yang Ming International Corp. (RackMountPro.com)

The Leading Server Builder in America. Enhancing Cloud Computing, The Optimized Technologies for Cloud

595 Yorbita Road, La Puente, CA 91744 ■ Tel: (800) 526-8650 ■ Fax: (626) 956-0098 ■ sales@rackmountpro.com

other words, after you have initialized all of your objects, you'll want to create the controller, which then will load objects from the remote server.

Saving and Loading

If you haven't quite grasped what we're doing here, consider this. In a typical server-side MVC application, the view is shown to the user, and the model is grabbed from a database. That hasn't changed here, except that the "user" for this Rails application is the Backbone.js application. The MVC of the Backbone.js application, by contrast, expects that its model will come from the server's view output.

How does this Backbone.js program know how to load information from the server? As mentioned above, you can add a "url" function to both your model and to your collection. For example, the start of your collection now can look like this:

```
Appointments = Backbone.Collection.extend({
    url: "/appointments.json",
```

The .json suffix tells Rails that communication will be done in the JSON format. When you receive output, it will be in JSON. Backbone.js can handle XML output as well, but all of the cool kids use JSON nowadays, so let's do that as well.

Once you tell the Appointments collection its URL, you then can ask it to retrieve data from that URL and populate itself with Appointment models. You simply can invoke the built-in fetch() method, which executes asynchronously, but even better, you can pass the fetch() method an object with "success" and "error" attributes, each of which contains a function that is executed based on the result from invoking fetch().

You invoke fetch(), as you might expect, from the controller, on the "index" route, which happens when the page is loaded without any hash tags. In other words, your initial visit to the page will invoke the controller's "index" action, which will tell the Appointments collection to load data from the server via the URL /appointments.json.

If all you wanted to do was retrieve the list of appointments, that would be fine. But you also want to display that list to the user. This means you need to iterate over each received appointment object and display it for the user. Fortunately, Backbone.js is built on top of the Underscore library for JavaScript, giving you access to all sorts of useful functions, including many for iteration. So when you retrieve the data, you then can set the following to be your "success" function, iterating over each item that you received from the server, and using a view method to add a new row to your HTML table:

```
success: function(response) {
    appview.appointments.each(function(item) {
                 appview.add_appointment_row(item);
             }
                );
    $("#number-of-appointments").html(appview.appointments.length);
}
```

Finally, you also will need to save any new appointment that you have created, using the HTML form, to the server. In order to do this, you must set the "url" attribute on the model (as opposed to the collection). This is a bit more complicated than the URL for the collection, because it must include the unique ID of the appointment you are saving, but only if you're updating an existing model. (Which, I should note, is impossible in the current version of the software, but it would be a welcome feature in future versions.) If you're creating a new appointment, you need not send any ID at all:

```
url : function() {
var base = '/appointments/';
if (this.isNew()) return base;
return base + this.id + '.json';
}
```

Now, when does a model get saved? Immediately after creating it, at least in this code. In the view function add_appointment, which executes (based on a jQuery callback) when someone clicks on the "add appointment" button, you first create a new appointment:

```
var new_appointment =
    new Appointment({person: person, starts_at: starts_at,
                ➥note: note});
```

You then add this new appointment to your collection:

```
this.appointments.add(new_appointment);
```

Finally, you also save the appointment to the server:

```
new_appointment.save();
```

Because your Appointment object is based on the Backbone.js model, it automatically knows how to save itself to the URL. And sure enough, when you add a new appointment, an AJAX query automatically fires in the background, sending a POST or PUT HTTP request to the server (as appropriate).

Conclusion

It can take a bit of time to get used to this MVC-squared (or whatever you wish to call it) paradigm. Once you get used to the fact that your server is being used for storage, data validation and providing a good data representation, but that the user interactions are taking place in the browser and in a separate application, things start to fall into place. Backbone.js is a great way to get started with this sort of application, especially if you're familiar with jQuery and want to use jQuery widgets in your application.■

Reuven M. Lerner is a longtime Web developer, architect and trainer. He is a PhD candidate in learning sciences at Northwestern University, researching the design and analysis of collaborative on-line communities. Reuven lives with his wife and three children in Modi'in, Israel.

Resources

The home page for Backbone.js is on GitHub, at **documentcloud.github.com/backbone**. This page not only points to the code, but also some tutorials and documentation.

In a step I hope many other authors will follow, the authors of Backbone.js put up a copy of the source code, thoroughly commented in a beautiful format, at **documentcloud.github.com/backbone/docs/backbone.html**. I encourage anyone interested in Backbone.js to read through the code and comments. I certainly learned some things about Backbone.js in particular and JavaScript in general from reading through this code.

A number of tutorials and blog postings describe how to do interesting things with Backbone.js. One that reviews how to use Rails 3 with Backbone.js, including some configuration adjustments that are necessary for it all to work, is **robertogds.com/post/3324511589/howto-backbone-js-using-rails-3**.

What Day Is That Date in the Past?

Parsing cal output.

DAVE TAYLOR

Last month, we started a script that worked backward from a day and month date and figured out the most recent year—including possibly the current year—that would match that date occurring on that particular day. For example, April 1st as a Friday was most recently in this year, 2011, but April 1st as a Tuesday? When did that last occur?

To make things interesting, our script is focused on tapping in to one of the unsung utilities of Linux, cal, and parsing its output to identify a day for a given date.

As is typical with a shell script, much of the work so far has been involved in normalizing the input data so that what we hand to the cal program will work and be understood by the program.

The bigger challenge, however, was to figure out whether a possible date could be in the current year. Since the program always is looking backward, it needs to know the current date to compare. That is, I'm writing this on April 3, 2011. If I check for the most recent April 1 being a Friday, it should say 2011, but if I check for the most recent May 1 being a Sunday, it should not suggest 2011. That's in the future and isn't a valid answer.

That's all shown in my previous column, so let's get on to something new: figuring out how to parse the cal output.

Parsing cal Calendars

For any given month and year, cal produces output similar to this:

```
    August 2008
Su Mo Tu We Th Fr Sa
                1  2
3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
31
```

Let's say we're looking for August 3rd. To search for it in this output, we need to specify that there should not be a digit before or after the date. This is doable with a simple regex:

```
$ cal aug 2008 | grep -e '[^0-9]3[^0-9]'
3  4  5  6  7  8  9
```

(As you'll learn later, this is insufficient as a regular expression. If you're really paying attention, you're already suspecting it's going to end up being a bit more complicated.)

Now, we need to figure out which digit matches.

awk to the Rescue

The basic approach we're going to use is to have awk step through each field on lines that match the pattern specified by using a for loop:

```
{ for (i=1;i<=NF;i++) if ($i~/regex/) print i}
```

We could use this with the grep statement above, but let's save a command by letting awk do the conditional test too:

```
$ cal aug 2008 | awk -e '/regex/ { for (i=1;i<=NF;i++)
  if ($i~/regex/ print i }'
```

To test this, let's use a regular expression that tests for the 5th day of the month:

```
[^0-9]5[^0-9]
```

This kind of works, but there's a problem. If we search for the 10th, because it appears at the very beginning of the line, it doesn't match the regular expression fragment `[^0-9]10`. The solution means our regex becomes more complicated, but here it is—one that works for the situation where it's possibly either the beginning of the line or the end of the line:

```
[^0-9]10[^0-9]|^10[^0-9]|[^0-9]10$
```

The | is a logical "or" statement, so it's now the earlier expression *or* one that has the pattern we seek followed by not-a-digit, but is at the beginning of the line (the ^ by itself) *or* is the pattern preceded by not-a-digit at the end of a line (the $ notation).

Fortunately, we're writing a script so we won't have to type this in more than once. Just as well!

There's another wrinkle in this output. We need to know not only in what field the matching number appears, but also how many fields total are on the matching line. Why? Otherwise, match 2 above occurring on a Monday would look exactly like the above, the 2nd occurring on a Saturday.

Here's our test script fragment, so far:

```
expr="[^0-9]${day}[^0-9]|^${day}[^0-9]|[^0-9]${day}\$"
```

```
cal aug 2008 | awk "/$expr/ { print \$0 }"
```

Notice that we need to use double quotes so that the variable $day is expended, and then $expr is also expanded, which means that we also need to escape the $0 in this test.

That's not what we want though. The awk statement needs to be more sophisticated, because we want to know the matching field number (for example, day of week 1–7) along with the total number of fields in the matching line. Ready?

```
expr="[^0-9]${day}[^0-9]|^${day}[^0-9]|[^0-9]${day}\$"
cal aug 2008 | awk "/$expr/ { for (i=1;i<=NF;i++) {
     if (\$i~/${day}/) { print \"i=\"i\", NF=\"NF }}}"
```

The double quotes add a tiny bit of complication, but really, this is just a complicated script.

The output, against our August 2008 calendar, looks like this:

```
$ sh match.sh 2
i=2, NF=2
$ sh match.sh 10
i=1, NF=7
$ sh match.sh 19
i=3, NF=7
```

That all makes sense. The next challenge is to figure out what day of the week we've matched for a given day and number of days in the week. Remember, day #1 on a three-day week is Thursday, while day #1 in a seven-day week is Sunday. Confusing, eh?

Day Of Week as an Array

The fast way to calculate this is to, well, pre-calculate it by creating a bunch of arrays. Like this:

```
if NF=1 days=[Sat]
if NF=2 days=[Fri,Sat]
if NF=3 days=[Thu,Fri,Sat]
```

and so on. There's a formula at play here, but more important, there's a pattern: (7-NF)-i is consistent. So day #1 on a three-day week is (7-3)+1 = 5 = Thursday, while day #1 on a 7-day week is (7-7)+1 = Sunday.

Let's double-check: in Aug 2008, Aug 1 is (7-2)+1 = 6 = Saturday, and Aug 4 = (7-7)+2 = Monday and Aug 31 = (7-1)+1 = 7 = Saturday.

Uh-oh, that last one's wrong, showing that we need to differentiate between the first week of the month, in which situation the days are right-aligned (as it were!), but in the last week of the month, they're left-aligned.

Ah, another nuance. Crikey, this is a rather tricky to write, isn't it?

Next month, we'll continue to build the script. Meanwhile, experiment with awk and regular expressions and see if you can find a more streamlined solution.■

Dave Taylor has been hacking shell scripts for a really long time, 30 years. He's the author of the popular *Wicked Cool Shell Scripts* and can be found on Twitter as @DaveTaylor and more generally at www.DaveTaylorOnline.com.

GENSTOR

Linux - FreeBSD -OpenSolaris - etc.

Proven Technology.
Proven Reliability.

When you can't afford to take chances with your business data or productivity, rely on a Genstor server customized to your specifications.

POWER
PERFORMANCE

Fly into the Cloud with Genstor Systems

- Up to 48 cores in a 1U.
- AMD Opteron 6100 series.
- Single high-efficiency power supply.
- Up to 512GB DDR3 memory.
- Ideal as front end processing servers.

- Up to 12 cores in a 2U
- Dual redundant high efficiency power.
- Up to 96GB DDR3 memory.
- Server Power Capping via Intel Intelligent Power Node Manager.
- Ideal as front end processing and/or storage.

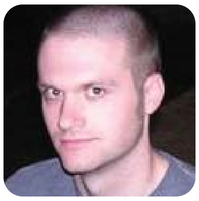
- Up to 4 GPU cards.
- Dual redundant high efficiency power.
- Up to 192GB DDR3 memory
- Up 24 cores using 2 CPUs.
- Up to 8 3.5" disks.

Genstor Systems, Inc.

intel Xeon inside™
Powerful. Intelligent.

1501 Space Park Drive
Santa Clara, CA 95054
www.genstor.com
E-mail: sales@genstor.com
Phone: 877-25 SERVER
408-980-0121

Intel®, the Intel® logo, Intel® Xeon®, and Xeon® Inside® are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.



Forensics with Ext4

Learn from my mistakes as I figure out how to gather forensics data on an ext4 filesystem.

KYLE RANKIN

One great thing about writing technical articles is that you have a nice collection of documentation you can turn to. I tell people that I probably reference my books and articles more than anyone else, because although I may not always remember specific steps to perform a task, I do always remember whether I wrote about how to do it.

One article I find myself referring to now and then is the "Introduction to Forensics" article I wrote in *Linux Journal* back in the January 2008 issue (my first feature article in *Linux Journal*). In that article, I walk through how to use Autopsy, a front end to Sleuthkit, to perform your own forensics investigation on a server that has been hacked. Recently, I had to perform an investigation on a server that fell victim to an SSH brute-force attack (use SSH keys!) and discovered that my personal documentation no longer worked. In this column, I walk through the symptoms of this problem and ultimately how I was able to work around it.

The Victim

As I mentioned, recently I investigated a server (let's call it alvin to protect the innocent) that had been compromised by a brute-force attack. I followed the procedure I document in my forensics article to a T. I pulled the plug from the server the moment I detected it was compromised, immediately created an image of the entire drive, created a second copy of the image on a separate drive that I would work from, and once my evidence was secure, I re-installed the affected server with a clean OS.

The first big difference about this system I ran into compared to past investigations was the sheer size of the data. It's one thing to image a 10–20GB disk and quite another when the drive is hundreds of gigabytes and every action—creating the initial image, image duplication and md5sums—takes hours. This was a slow process to say the least. Once I had my working image, I fired up Autopsy and started my investigation. Initially, everything was fine—Autopsy saw the image and was able to detect its partitions, and when I started to browse the filesystem, I could see the contents of the root directory.

It's one thing to image a 10–20GB disk and quite another when the drive is hundreds of gigabytes and every action—creating the initial image, image duplication and md5sums—takes hours.

The Problem

The first problem showed up when I tried to navigate through the filesystem with Autopsy—all of the directories below the root directory appeared to be empty. I knew that couldn't be right, but I wasn't sure what the problem was. At first, I thought I might just have a corrupted copy of the image, but since the md5sum seemed to take almost as long as copying the image, I started the image copy again from my gold master just to be sure. When that still didn't work, I tried to use Sleuthkit from the command line and even tried tools from The Coroner's Toolkit, yet I got the same result. I thought perhaps Autopsy couldn't support such a large disk image. Before I did too much more research, I decided to try to mount the image loopback to see if it was a corrupt image. After all, if it were, I'd need to create a fresh image before I went any further.

The challenge here was that I had created an image of the entire disk, not just individual partitions. Normally, when you mount something loopback, you mount an image of a partition, and the syntax is something like:

```
$ sudo mount -o loop /path/to/image /mnt/image
```

Then, you can browse /mnt/image like any other mounted filesystem. In this case, since the image was of a full partition, I had to figure out how to skip ahead in the image and mount only the partition. Mount has an option called offset that allows you to specify how far ahead in the file to skip before it mounts it as a filesystem, but the trick is knowing what that offset should be set to. It turns out that if you have parted installed, it's relatively simple to find the offset. First, I run parted against alvin's full disk image and tell it to print out the full partition table in bytes:

```
$ sudo parted /media/Forensics2/images/alvin-sda
GNU Parted 2.2
Using /media/Forensics2/images/alvin-sda
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) unit
Unit?  [compact]? B
(parted) print
Model:  (file)
Disk /media/Forensics2/images/alvin-sda: 251059544064B
```

```
Sector size (logical/physical): 512B/512B
Partition Table: msdos

Number Start         End           Size          Type      ...
 1     32256B        242242721279B 242242689024B primary   ...
 2     242242752512B 249999160319B 7756407808B   extended  ...
 5     242242753536B 249999160319B 7756406784B   logical   ...

(parted) quit
```

In this case, I'm interested in the first partition, and from this output, I can see that the partition starts at byte 32256. Now I can use that offset to mount the partition loopback:

```
$ sudo mount -o loop,ro,offset=32256 alvin-sda /mnt
```

Note that I mounted the filesystem read-only here. I didn't want to risk writing any new data to this partition! From this point, I found that I could browse the filesystem under /mnt just fine—the problem somehow had to be with Autopsy and Sleuthkit. I did research, but I couldn't really find any evidence that there was an upper limit to the image size that I was close to, but all the same, I decided to try to create an image of just the initial partition. A few hours later, Autopsy still couldn't use the new image, but strangely enough, I could mount the partition loopback on my filesystem just fine. After trying countless other things, I started to realize that it couldn't be the size of my partition that was a problem. It had to be something else, and it was then that I noticed that this filesystem was ext4.

Like Ext3 Plus One More

To be honest, apart from the fact that ext4 touted faster speeds with large numbers of files while claiming backward compatibility with ext3, I hadn't given the filesystem much thought. I had used it on a few new systems (it is the default on modern Ubuntu installs), and it seemed to work fine. Because it was supposed to be backward-compatible, I initially wrote it off as being a cause of my problem. After more research though, I discovered not only did other users complain about lack of support for ext4 in Sleuthkit, but that the backward compatibility isn't as compatible as you might think. Although it's true that you can mount ext2 and ext3 filesystems as ext4, you can mount ext4 filesystems as ext3 only if the filesystem does *not* use extents (which is a major new feature of ext4, so it's commonly turned on). Because I couldn't treat this ext4 filesystem as ext3, that meant neither could Sleuthkit.

So if I could mount the partition loopback, what is the big deal if Sleuthkit and Autopsy didn't work? Although being able to browse through the filesystem is a useful feature of forensics tools, another incredibly valuable feature is the ability to create a filesystem timeline. A filesystem timeline organizes all the files into a giant text database where they are ordered according to their MAC times (when the file was last Modified or Accessed or Changed its metadata). With a filesystem timeline, if you have a good idea when the attackers might have been on the system, you can start to track their virtual footprints as they execute programs, browse directories and untar their scripts onto the system. Normally, I would have Autopsy generate this file for me. Luckily, it turned out that a tool from The Coroner's Toolkit called mactime was able to generate the timeline either from an image or from a mounted filesystem. Here's the command I used to create a timeline from the filesystem mounted at /mnt:

```
$ sudo mactime -d /mnt -R -g /mnt/etc/group -p
 ➥/mnt/etc/passwd 1/2/1970 > timeline.txt
```

The -d option specifies the directory where the filesystem is mounted, and -R tells mactime to scan recursively through that directory. The -g and -p options tell mactime to get group and user ID information from the group and passwd files in my mounted filesystem, and finally, the date specifies the date range to start from. The date must be after 1/1/1970 (I'll leave why that is as an exercise for the reader), so I chose the next day.

Although I didn't have the same user experience I was used to with Autopsy, once I could browse the filesystem and view the timeline, I could use the same investigation techniques. Ultimately, I was able to piece together the timeline when the attacker compromised the machine via a weak password, changed the password so no one else could break in the same way, and then downloaded and launched SSH brute-force attack scripts to spread onto other servers.

I admit I was really surprised to see that some of my favorite forensics tools no longer worked. Although it is great that Linux filesystems continue to improve, one unintended downside (or possibly an upside if you are an attacker) is that forensics tools must be upgraded continually to work on modern filesystems. Although I was able to mount the partition and create a timeline, I don't know that I would have been able to recover any deleted files from the partition—another valuable use of forensics tools. That said, at least if I (or you) need to analyze an ext4 image again, I now have all of the steps documented.■

Kyle Rankin is a Sr. Systems Administrator in the San Francisco Bay Area and the author of a number of books, including *The Official Ubuntu Server Book*, *Knoppix Hacks* and *Ubuntu Hacks*. He is currently the president of the North Bay Linux Users' Group.

The Tails Project's The Amnesic Incognito Live System (Tails)

Making delirious dictators worldwide quake in their boots, the Tails Project recently announced numerous improvements to its anonymity-obsessed Linux distro, The Amnesic Incognito Live System, aka Tails. Now in version 0.7, Tails is the spiritual successor of the well-known Incognito Live System and is developed with the support of the Tor Project, an onion routing project originally developed to protect US military communications. The live distro, which runs on any PC powerful enough to run Windows XP, is based on Debian Live and runs directly from CD and/or USB Flash memory. No trace is left after using Tails, thanks to many features, such as independent operation of all software and all hardware drivers from the PC's operating system, no permanent data storage and all the channeling of all Internet connections through the Tor anonymization network. "With Tails", say the distro developers, "we provide a tongue and a pen protected by state-of-the-art cryptography to guarantee...basic human rights and allow journalists worldwide to work and communicate freely and without fear of reprisal." The journalists of these pages humbly salute the valiant effort.

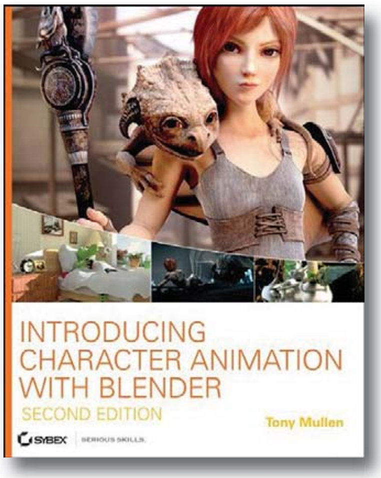
tails.boum.org

Rectiphy's ActiveImage Protector

From the ticker at the storage desk comes news from Rectify, announcing the new version 3.0 of the company's ActiveImage Protector (AIP) live data management backup solution for physical machines and virtual environments. Key new features in AIP V3.0 include disk-to-disk copy, bad-sector skipping and the ability to restore image files to new virtual or physical locations. AIP's Smart Sector Technology enhancements now enable the backup of Linux partitions (in Ext2/Ext3/Ext4 formats) from Windows. AIP Linux Edition now also supports hot imaging of a Linux server hard disk or volume. Rectify's ActiveImage Protector Family of backup solutions includes Server, Desktop, Hyper-V with ReZoom, Linux, Virtual Environment and IT Pro editions.

www.rectiphy.com

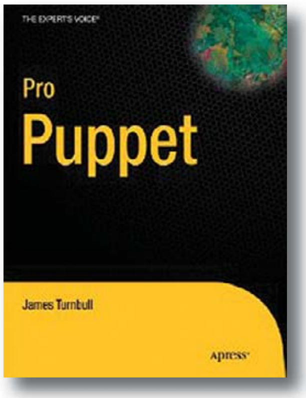
Tony Mullen's *Introducing Character Animation with Blender*, 2nd Edition (Sybex)



Blender is one of the slickest, most powerful, open-source 3-D graphics programs out there, and if you're ready to scale the learning curve toward mastery of animation, get your hands on Tony Mullen's *Introducing Character Animation with Blender*, now in its 2nd edition from Sybex. Written in a friendly but professional tone, with clear descriptions and numerous illustrative screenshots, the book's tutorials focus on how to accomplish actual animation goals, while illustrating the necessary technical methods along the way. These are reinforced by clear descriptions of how each specific aspect of Blender works and fits together with the rest of the package. By following all the tutorials, readers will gain all the skills necessary to build and animate well-modeled, fully rigged characters of their own. A full-color gallery includes sample characters and frames from animations by many of the Blender community's most talented artists, which help to illustrate the impressive potential of the software.

www.sybex.com

James Turnbull and Jeffrey McCune's *Pro Puppet* (Apress)



If you've mastered the basics of the popular, open-source Puppet configuration management tool, you might be inclined to delve into James Turnbull and Jeffrey McCune's *Pro Puppet*, an in-depth guide to installing, using and developing Puppet. The book, published by Apress, is a comprehensive follow-up to the publisher's previous and more introductory title *Pulling Strings with Puppet*. Puppet provides a way to automate everything from user management to server configuration. Besides exploring the essentials, readers will learn how to create Puppet recipes, extend Puppet, automate tasks, learn insider tricks and use Facter to gather configuration data from servers.

www.apress.com

Eaton Corporation’s Eaton 5PX UPS

Eaton Corporation recently unveiled its new Eaton 5PX UPS, an uninterruptible power supply (UPS) that allows for energy metering down to the outlet segment level, a feature that the company calls an industry first. The feature enables IT managers not only to control individual power outlets or groups of outlets across the network, but also to track and reduce the power consumption of the protected servers and other IT equipment. In addition, the 5PX provides 28% more wattage than a traditional UPS, allowing IT managers to protect a greater number of devices. Virtualization strategies are supported via seamless integration with Eaton’s Intelligent Power Software Suite. The device is targeted at small- to medium-business data centers.

www.eaton.com/5px

Zorin PC

After waiting like Job for the pre-installed Linux machines we deserved, we’ve arrived at a literal Garden of Eden full of worthy choices. A compelling new offering is the Zorin PC, a new mini-laptop that runs its own Linux distro, Zorin OS. On the hardware side, the Zorin PC’s sexy selling point is its rotatable touchscreen display, which allows one to use the device as a regular laptop, a tablet device, an e-reader or a media center. The display rotates 180°, allowing one to share pictures or videos with friends. Folding it down as a tablet makes it feel like an Amazon Kindle. The touchscreen uses digitizer technology, which lets one take handwritten notes. On the software side, one can order a Windows 7/Zorin OS dual-boot option, the latter of which comes in Home, Educational and Business editions. A software repository is included for installing tons of additional software programs.

www.zorinpc.com

Component for OpenERP

Not only are maps cool and intuitive, they are—thanks to Camptocamp—a new geographic business intelligence tool in the OpenERP suite of business applications. Camptocamp’s Geolocation Component for OpenERP allows users to place simple marks, graphs and colors by geographical area, calculate distances and surfaces, analyze sales figures by specified geographic area, optimize and visualize logistical flows, manage fleets and provide map data for other components of OpenERP.

www.camptocamp.com

Opsview Enterprise

The list of reasons to eschew proprietary monitoring applications just grew longer with the 3.12 release of Opsview Enterprise, an Nagios-based enterprise IT monitoring platform. The system helps organizations monitor their physical and virtual infrastructure, both on-premise and in the cloud. Opsview’s automated configuration reduces deployment times, and its improved graphical interface provides organizations with a better single view of their entire IT environment. The upgraded edition of Opsview includes features such as improved cluster monitoring, Rest API, SUSE Linux support, SNMP aggregation and greater visibility and faster access.

www.opsview.com

Please send information about releases of Linux-related products to newproducts@linuxjournal.com or New Products c/o *Linux Journal*, PO Box 980985, Houston, TX 77098. Submissions are edited for length and content.

Fresh from the Labs

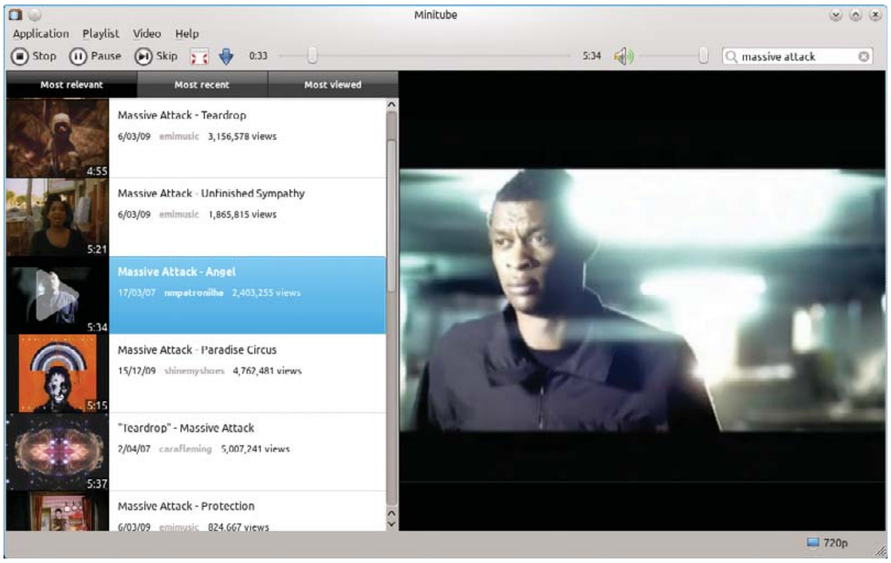
Minitube—Independent YouTube Viewing

flavio.tordini.org/minitube

One of the most annoying burdens facing Linux users is Adobe's proprietary Flash and its lack of general hardware acceleration. Watch a YouTube video with one of your Windows-using mates around, and you can be guaranteed of visual tearing and some pretty low-par performance—not cool. However, you need not fear. Minitube has come to the rescue! To quote the Web site:

> Minitube is a YouTube desktop application. With it, you can watch YouTube videos in a new way: you type a keyword, and Minitube gives you an endless video stream. Minitube is not about cloning the original YouTube Web interface; it aims to create a new TV-like experience.
>
> **Light on your computer:** by consuming less CPU, Minitube preserves battery life and keeps your laptop cool. That's because Minitube does not use the Flash player.
>
> **High definition:** Minitube plays HD videos up to 1080p. Go full-screen and watch them play fluidly.
>
> **One-click downloads:** download your favorite clips to your computer and put them on your portable device. Downloaded files are in MPEG4 format, which is compatible with most devices, including Apple ones.

YouTube in Glorious HD-Accelerated Full-Screen



Massive Attack - Teardrop
6/03/09 emimusic 3,156,578 views

Massive Attack - Unfinished Sympathy
6/03/09 emimusic 1,865,815 views

Massive Attack - Angel
17/03/07 nmpetronilha 2,403,255 views

Massive Attack - Paradise Circus
15/12/09 shinemyshoes 4,762,481 views

"Teardrop" - Massive Attack
2/04/07 carafleming 5,007,241 views

Massive Attack - Protection
6/03/09 emimusic 824,667 views

Minitube also provides its own playlist editor, allowing you to queue whichever videos you like, in any order.

Installation 32-bit x86 users have it easy, as a binary tarball is provided on the Web site's main page. Binary packages are available in various forms, some in repositories but outdated, others in personal archives. See the setup instructions page if you want to try your luck. For the rest of us (including me, a 64-bit user), there's the source.

Grab the latest tarball, extract it, and open a terminal in the new window. Before you can start compiling, you need to install a number of preliminary libraries. Luckily, the Web site provided the following command for users of Ubuntu and other Debian derivatives for installing dependencies:

```
sudo apt-get install libqtgui4 libqt4-xml libqt4-network
➥libqt4-dbus phonon-backend-gstreamer gstreamer0.10-ffmpeg
➥gstreamer0.10-plugins-bad
```

I can't give information for packages under other distributions, but hopefully the package names above should give you a clue as to what you need.

Once you have installed the dependencies, installing Minitube is actually pretty easy. Just enter the commands:

```
$ qmake
$ make
```

The documentation warns that this step is for packagers and not end users. I'm not sure what the concern is, but if you're as equally unworried as myself, enter these commands.

If your distro uses sudo:

```
$ sudo make install
```

If your distro uses root:

```
$ su
# make install
```

Once that's finished, you can run the program with:

```
$ minitube
```

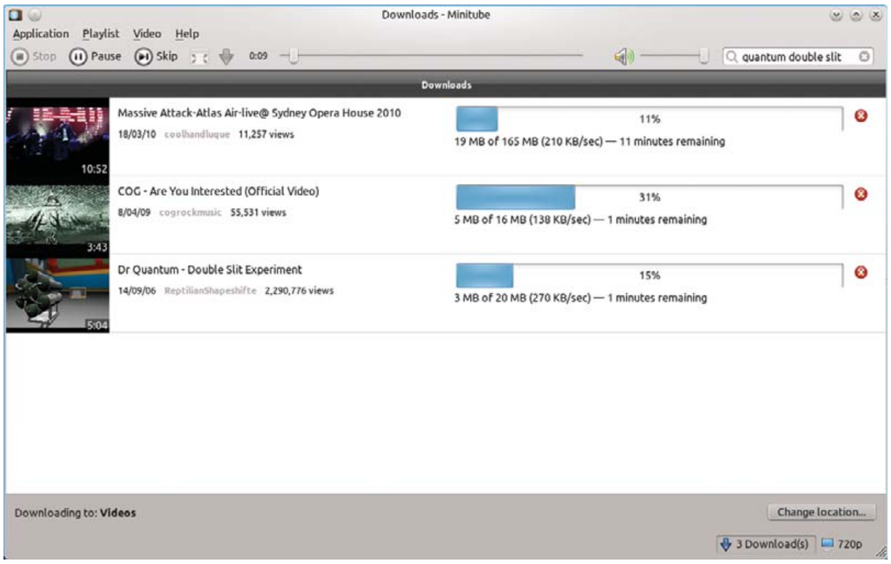
However, if you share the developer's aversion to this installation method, you can run the program with this command:

```
$ ./build/target/minitube
```

Usage Actually using Minitube is a pretty easy and intuitive affair. In the middle of the screen as well as on the top right are search bars where you enter keywords, the same as you would on YouTube. Enter your search, and a list of results will appear in the next screen, sorted by groups of ten.

At the time of this writing, when I tried it, the first result automatically started playing. I'm not sure I like that, but I imagine the interface for playing and the playlist will be refined over time. This is software in development, after all.

Scroll down the pane on the left, left-click on any video to play it, and the video plays on the right. It's all pretty standard so



Massive Attack-Atlas Air-live@ Sydney Opera House 2010 — 18/03/10 coolhandluque 11,257 views — 10:52 — 11% — 19 MB of 165 MB (210 KB/sec) — 11 minutes remaining

COG - Are You Interested (Official Video) — 8/04/09 cogrockmusic 55,531 views — 3:43 — 31% — 5 MB of 16 MB (138 KB/sec) — 1 minutes remaining

Dr Quantum - Double Slit Experiment — 14/09/06 ReptilianShapeshifte 2,290,776 views — 5:04 — 15% — 3 MB of 20 MB (270 KB/sec) — 1 minutes remaining

Minitube's Built-in Downloader—Practical

Powerful: Rhino

Rhino M6500/E6510
- Dell Precision M6500 w/ Core i7 Quad (8 core)
- Dell Latitude E6510 w/ 2.53-2.8 GHz Core i5/i7
- Up to 17" WUXGA LCD w/ X@1920x1200
- NVidia Quadro FX 3800M
- 250-750 GB hard drive
- Up to 32 GB RAM (1333 MHz)
- DVD±RW or Blu-ray
- 802.11a/b/g/n
- Starts at $1385

- High performance NVidia 3-D on a WUXGA RGB/LED
- High performance Core i7 Quad CPUs, 32 GB RAM
- Ultimate configurability — choose your laptop's features
- One year Linux tech support — phone and email
- Three year manufacturer's on-site warranty
- Choice of pre-installed Linux distribution:

Tablet: Raven

Raven X201 Tablet
- ThinkPad X201 tablet by Lenovo
- 12.1" WXGA w/ X@1280x800
- 2.0-2.13 GHz Core i7
- Up to 8 GB RAM
- 250-500 GB hard drive / 160 GB SSD
- Pen/stylus input to screen
- Dynamic screen rotation
- Starts at $1940

Rugged: Tarantula

Tarantula CF-31
- Panasonic Toughbook CF-31
- Fully rugged MIL-SPEC-810G tested: drops, dust, moisture & more
- 13.1" XGA TouchScreen
- 2.4-2.53 GHz Core i5
- Up to 8 GB RAM
- 160-750 GB hard drive / 256 GB SSD
- Call for quote

EmperorLinux

...where Linux & laptops converge

www.EmperorLinux.com

1-888-651-6686

Model specifications and availability may vary.

far, but double-click and *voilà*, full-screen playback with hardware acceleration—the way YouTube was always meant to be!

Exploring the program further, you can edit the playlist on the left by clicking and dragging videos up and down. You also can refine searches by using the three tabs: Most relevant, Most recent and Most viewed. Thankfully, the controls everyone takes for granted also are included, such as a sliding volume control, stop, pause, skip and the all-important seeking bar.

However, this program's biggest hook is its Download option. Obviously, this allows you to download the video you're watching at the time, but what isn't so obvious is that Minitube also has a download manager, where you not only can download videos, but also download multiple streams simultaneously. It's clever too; try to quit while it's in the middle of a download, and Minitube warns you and prompts you with a dialog asking whether to quit or to wait until the downloading has finished.

If you still need to do things the normal way, the Video menu has options for opening the actual YouTube page (presumably so you can read all of the obligatory snide comments below the video). Other clever features include copying the YouTube URL, as well as the actual streaming video URL, for the advanced users out there. And for the rest of us, at the bottom right is a setting for maximum video definition, 360, 720 and 1080p—very handy.

At the end of the day, Minitube provides a sleek solution to one of the most annoying problems Linux users face: bad YouTube playback. It's not Linux's fault, but proprietary solutions such as Flash always will burden us with these annoyances. Of course, it's not just YouTube that uses Flash and suffers from these performance woes, but at least it's a start (I'm sure Minitube could be modified to work with other popular sites, such Hulu and so on).

Although Minitube has some interface oddities, it still is a slick program nonetheless. And, when watching these videos in a simple and comfortable interface, released from the clunky shackles of a Web browser, I can't help but feel that this is what YouTube was meant to be like in the first place.

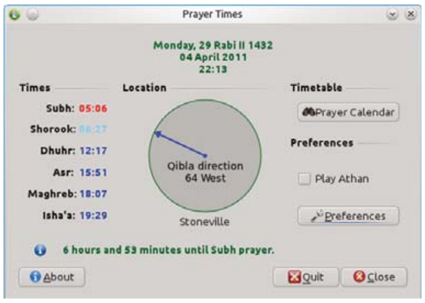
Minbar and the Islamic Tools and Libraries

projects.arabeyes.org/project.php?proj=ITL

I've been meaning to explore applications available for our Muslim readers for some time now. My main focus here is on the Minbar Project, but I also explore some of the highlights of the Islamic Tools and Libraries (ITL).

As a side note, I'm not a Muslim myself, so for our Islamic readers, I beg your patience for any silly mistakes I might make (I hope my research will be accurate, nevertheless).

To quote the man page: "Minbar is a GNOME Islamic prayer times application. At first start, you have to configure it with your location, time zone and madhhab details."



Monday, 29 Rabi II 1432
04 April 2011
22:13

Subh: 05:06
Shorook: 06:27
Dhuhr: 12:17
Asr: 15:51
Maghreb: 18:07
Isha'a: 19:29

Qibla direction
64 West
Stoneville

☐ Play Athan

6 hours and 53 minutes until Subh prayer.

Minbar provides Islamic prayer times and direction in a convenient app that lives conveniently in your taskbar.

Installation Unfortunately, Minbar's Web site was down when I wrote this, so no source tarball was available. However, Minbar seems to be a common package in distro repositories, so you should be able to install it that way.

Once installed, Minbar should be in your system menu (Utilities→Minbar Prayer Times on my Kubuntu machine), or you can run it with the following command:

```
$ minbar
```

Usage You'll be greeted by a fairly simple window, with Qibla direction in the center and prayer times on the left. However, you need to do some setting up before you really can use Minbar.

As the man page states, you first have to configure your location, time zone and Madhhab details. Click on Preferences, and you'll be presented with your city details. Here you define your latitude, longitude, city name and time zone. If you live in a major city, click on Find City, and there's a good chance your city will be on the list, so you won't need to enter these details manually.

I'm not entirely sure to which part of the GUI the Madhhab preference is referring—whether it's in the Calculation Method or perhaps more likely, your choice of Athan.

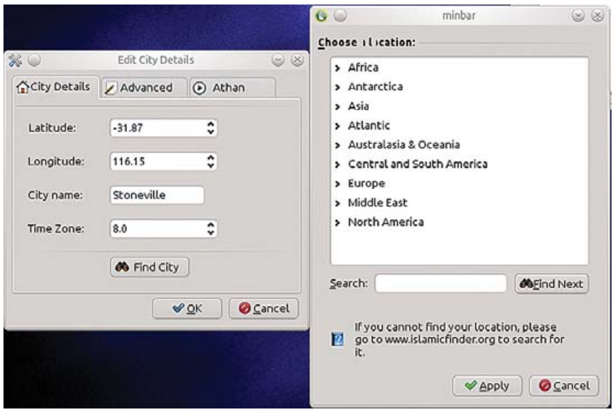
For choosing the Calculation Method, this is under the Advanced tab. I chose "University of Islamic Sciences, Karachi (Shaf'i)", just because one of my favorite foods is made in Karachi. Athan has its own tab.

With my installation, there weren't any Athan files included, but you can browse for your own through your filesystem and test it with the Play and Stop buttons. Separate file choices are available for the Subh Athan and the Normal Athan.

Once your configuration is out of the way, click OK, and you'll go back to the main screen. Here you will find that all of the values are updated and running in real time, including the newly aligned Qibla direction and prayer times on the left.

A handy feature below that is a green text field giving you a real-time countdown until the next prayer time, which at this moment says, "5 hours and 55 minutes until Subh prayer".

If you look to the right, there's a check box where you can choose whether or not to play the Athan. An invaluable timetable also is on the right, with the button marked Prayer Calendar. This was quite educational to me as a non-Muslim, as I didn't realize there were different prayer times at different points on the calendar. For instance, the month I wrote this, while Dhuhr pretty



Latitude: -31.87
Longitude: 116.15
City name: Stoneville
Time Zone: 8.0

If you cannot find your location, please go to www.islamicfinder.org to search for it.

Minbar makes it as easy as possible to enter your world location, finding the Qibla direction automatically.

much stayed put, Shorook was at 6:52 on the 12th and 7:02 on the 26th.

In the end, Minbar is a clever little application, and once it's configured, it isn't the least bit daunting to use, with very simple GUI elements. Minbar also integrates into the desktop nicely like any other widget, living in the taskbar with small status updates when the mouse pointer hovers over it and calling up the main window when clicked on.

Ultimately, Minbar is an excellent use of modern technology for making daily life more convenient while remaining unobtrusive to the rest of the desktop. This excellent desktop integration, in turn, hopefully should allow Muslims to integrate Minbar into their daily work PCs and have prayer time reminders live seamlessly alongside other working applications.

The Islamic Tools and Libraries Let's briefly explore some more components of ITL. To quote the ArabEyes.org Web site: "The Islamic Tools and Libraries (ITL) is a project to provide a plethora of useful Islamic tools and applications as well as a comprehensive feature-full Islam-centric library. The ITL project currently includes Hijri date, Muslim prayer times and Qibla."

Included are two packages (of which I also found in my distro repository): libitl and itools.

About the libitl package, the documentation says, "This library allows applications to convert between Hijri/Gregorian dates and compute Muslim prayer times and Qibla direction based on multiple methods of calculation."

The itools documentation says:

> The itools is a collection of command-line tools that mimics the development of the underlying ITL library (libitl) and is meant to always give the end user simple means to access its functions. The available tools are:
>
> - ical: display a Hijri calendar.
> - idate: multi-method Hijri/Gregorian date converter.
> - ipraytime: prayer times and Qibla calculator and schedule table generator.
> - ireminder: prayer time reminder Perl script.

If you check out the Projects page at ArabEyes, you'll see that volunteers are working on a large number of major projects to achieve better Arabic support, such as Firefox, GNOME, KDE, Drupal and so on.

Hopefully projects like these will make GNU Linux the easy option in the Islamic and/or Arabic worlds (non-Muslim Arabs at least can benefit from the better localization ITL will provide).■

John Knight is a 26-year-old, drumming- and bass-obsessed maniac, studying Psychology at Edith Cowan University in Western Australia. He usually can be found playing a kick-drum far too much.

Brewing something fresh, innovative or mind-bending? Send e-mail to newprojects@linuxjournal.com.

VESA-Mountable NVIDIA® ION2 System
Fully configured with HDD, RAM. Features GeForce® Graphics, flexible storage options, and Wi-Fi.

Fanless, Rugged Intel® Core™ Platform
No fans, no moving parts. Just quiet, reliable performance. Full featured; no compromises.

Genuine expertise.

www.logicsupply.com/linux

FUN AND MAYHEM WITH THE BLENDER GAME ENGINE

Create 3-D games using the keyboard or mouse as controllers. **MIKE DIEHL**

I've been working with Blender 3-D for several years now, but I started playing with the game engine only recently. I've had a lot of fun with it, and I'm sure you will as well. With the Blender Game Engine (BGE), you can create 3-D games using the keyboard or mouse as controllers. Your game can trigger events when objects collide with each other or when they get within a certain distance from each other. There is a built-in state engine, so that objects in your game can change their behavior as required. Although there is a powerful and well-documented Python API, we won't be using it today. In fact, we won't be writing a single line of code!

In the April 2009 issue of *Linux Journal*, I wrote an article demonstrating the Irrlicht 3-D engine by creating a basic 3-D environment in less than 100 lines of C. In this article, I demonstrate the BGE by creating a functional video game, complete with realistic physics, and all I'm going to do is connect dots.

One of the biggest attractions of the BGE is the built-in Bullet Physics Engine. The physics engine allows users to assign various physical characteristics, such as mass, to objects in their blender model and configure how they respond to collisions and movement. When done properly, users end up with blocks that fall when dropped and tumble when they collide. Balls roll down hill, and dominoes knock each other down.

In order to demonstrate the important aspects of the BGE and the Bullet Physics Engine, I've devised a little game that I affectionately call *The Super Mega Demolition Derby*. The premise of the game is simple; you build walls and towers and such with various types of building materials. Then, you drive a car through them and watch them crumble to pieces! Okay, so I don't know how you'd actually go about keeping score in a game like this, but games are supposed to be fun, and based on how much time I've spent destroying block walls, I think this is a fun game.

Figure 1 shows an action shot from one of the test animations I made while playing the game. As you can see, the modeling and texturing are very simplistic (this isn't meant to be a modeling tutorial). I'm sure you can see that this is the classic "drive a car over a ramp and through a wall" stunt. To set up this stunt, I created three stacks of blocks, stacked four high. I put the purple ball on top. Then, I positioned a ramp in front of the wall and put the car a bit further out. Once the lighting and camera angles were set up, I started the game and drove the car up the ramp.

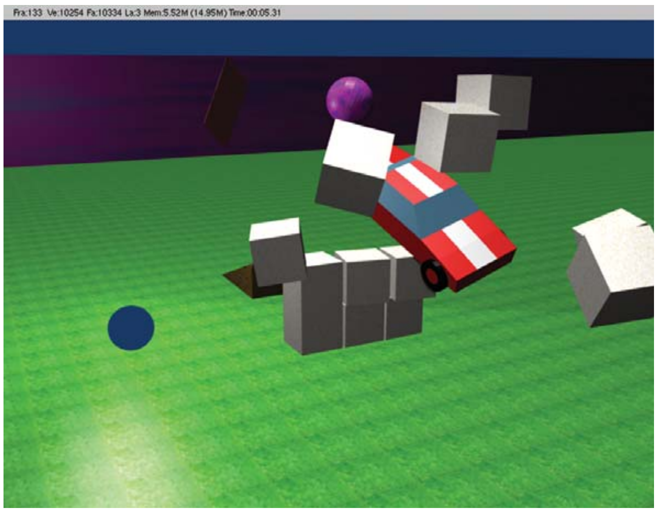

Fra:133 Ve:10254 Fa:10334 La:3 Mem:5.52M (14.95M) Time:00:05.31

Figure 1. Action Shot from a Test Animation

The first step in creating the game is to create a model for all of the various objects. The arena is simply a 100x100 plane. I then extruded the edges to create a fence to prevent the car from driving off the edge of the arena. The car was created from a cube that was scaled and extruded to form a very basic car shape. The rest of the game objects are fairly trivial.

Once the game objects are finished, it's time to start the game logic. Because the car has both physical attributes and keyboard controls, it's the most complicated, and this is where we'll start. Selecting the car and pressing F4 brings up the Logic menu. The configuration I used for the car is shown in Figure 2. Here you see three columns of controls. The first column is where you configure the object's physical attributes, which I discuss later. For now, let's take a look at the Sensors, Controllers and Actuators columns.

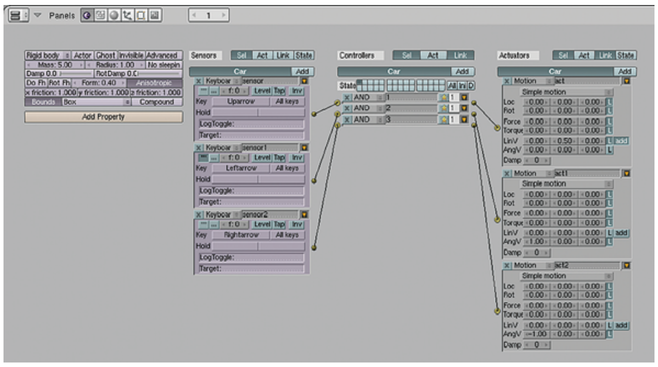


Figure 2. Logic Panel

The Sensors column is where you select to which events you are interested in having your game object respond. In this case, you can see that the car responds to the up, left and right arrow keys. The keyboard sensor allows you to configure various parameters, such as repeat rates and keyboard modifiers. Game objects also can respond to mouse events, collisions, proximity to other objects, timers and messages sent from other objects.

Sensors are connected to controllers, which function essentially as filters and determine when a given sensor is of interest. As you can see, I selected the boolean logic equivalent to "always" in determining when the car should respond to keyboard events. I could have referenced Python code or any of the 30 different game states if I had chosen to, but that just introduces more complexity than the game requires. The state engine could have been used to introduce traps, puzzles and other hidden elements to the game. Implementing actual intelligence probably would have required some Python code.

The Actuators column is where the action comes from. Although I used "simple motion" for the car's keyboard events, the possibilities are a bit daunting at first. I had the option of changing the object's location and rotation along the X, Y or Z axis. I also had the option of applying a force or torque. Finally, I had the option of changing the object's linear and angular velocities. By selecting the L at the right, I told the BGE to use the object's Local orientation to apply the motion changes, as opposed to using global coordinates.

Getting the car's handling to "feel" right took a bit of tuning, and as it is, it's still a bit more like driving in slick mud than driving in a grassy arena. More tuning needs to be done before it's right. On the other hand, I was able to use a material setting to add random bumps to the driving surface that caused the car to bounce around a bit.

The BGE allows you to do more than just move objects around. You can use actuators to make sounds, send messages to other objects, create and destroy objects, change game state, change cameras or even start an animation sequence. It is hard to come up with something that can't be done with the actuators that the BGE provides. For example, I've considered the possibility of creating a series of actuators that destroy a box object and replace it with an identical-looking box that has the "Explode" modifier applied to it. This would allow me to create a box that explodes upon impact, but I've not tried it.

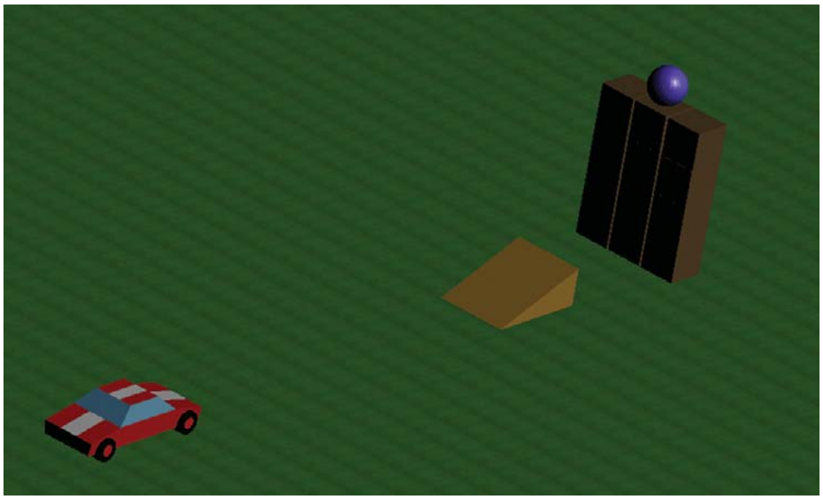


Figure 3. Screenshot of the Game

Now that you have a car you can drive around, you need to make it and the other game objects behave according to the laws of physics. This is done using the controls in the first column of the Logic panel in Figure 2. The first thing you have to do is decide what type of object you want, which essentially determines "how many" of the physical laws apply to the object. You have several types from which to choose, including static, dynamic, rigid body, soft body and sensor.

A static object responds to collisions, but it's not affected by gravity. I configured the ramp as a static object, for example. This way, the ramp would respond realistically if a car ran into it from behind, but I could also "stack" ramps in otherwise impossible configurations in order to build more complex obstacle courses.

You can use actuators to make sounds, send messages to other objects, create and destroy objects, change game state, change cameras or even start an animation sequence.

Dynamic objects are just like static objects except that they're affected by gravity. They are not subject to rolling physics. When I initially set the car as dynamic, I found it didn't drive up the ramp as I expected. It resembled more of an escalator ride to the top of the ramp, because the car didn't tilt as it crossed the ramp.

When I configured my car to be a rigid body, I had everything. The car would fly off the ramp, fall to the ground and roll if it hit too hard or hit something on its way down. Most of the objects in the game are configured as rigid bodies.

The soft body type was fun to play with. By tweaking various parameters, I was able to come up with a giant beach ball that bounced and deformed when dropped. By loosening things up a bit more, I ended up with a giant ball of Jell-O, and who hasn't dreamed of driving a car through a giant ball of lime Jell-O? I know I have. This is the type to use on objects you want to deform upon impact.

A sensor is simply an object that can be used to trigger events but doesn't necessarily need to have the entire laws of physics applied to it. For example, a sensor could be used to spring a trap, release an enemy or simply change the state of the game upon collision. In this game example, a sensor object could be placed in the middle of a platform to trigger an elevator to rise when the car is positioned properly

Once you've determined what type of object you want, it's time to tune various parameters. For example, you can determine how much mass the object has, which will determine if it falls like a feather or a proverbial ton of lead, as well as how it behaves when it collides with other objects. By clicking on the Advanced button, you also can set limits on how fast the object can move, among other things. One of the most important things to tweak is the Bounds configuration. This configuration determines how collision detection is done. I found that if I used a spherical bounds for my car, things just didn't work well—the car ended up rolling down the path to the ramp instead of driving along.

You also can use the physics panel to assign various properties to an object. For example, you might assign a "damage" property to the car and assign an initial value of 0 to it. Then, for each collision, you could create an actuator that increased this value. Finally, you can use the property as a sensor and trigger an action when it exceeds a certain value.

So, what do we have so far? Figure 3 is a screenshot of the game from another camera angle—it's a car that can be driven around in an environment that includes several different obstacles. This car (as well as the obstacles in the game) has physical properties, such as linear and angular momentum, mass and elasticity. All of these objects behave as you would expect them to behave in a real-life situation.

From this, I'm expecting to create even more interesting objects for the game. For example, I imagine that a very low mass object like a playing card could be created easily. From that, a house of cards could be constructed and subsequently destroyed. Dominoes, although fun to watch, are so trivial to create, they're almost uninteresting. In my opinion, beach balls and giant globs of Jell-O certainly warrant further investigation.

By parenting the camera to the car, I've been able to devise a cockpit view or a third-person view. Using this view, I could drive around the arena just like any other driving simulation. The only drawback was that once I drove the car through a wall, I wasn't able to watch it tumble to the ground unless I quickly turned the car around. It might be nice to set up several cameras, strategically place them in the game and create keyboard triggers to select them. Still, this is very easy to do with the BGE.

As you can see, the BGE is very powerful and doesn't require any programing to get started. However, if you know or are willing to learn Python, you can do a lot more interesting things. Tutorials on the Web demonstrate how to use Python code to create projectile weapons, for example. I've already alluded to the idea of using Python code to endow game objects with artificial intelligence. The BGE Python API also is very well documented. In fact, the BGE is the sole reason I've set out to learn Python.

As cool as it is, the BGE does have some limitations. Like Blender itself, the BGE is extremely powerful, and although they're both well documented, getting started can be a bit intimidating. The water and cloth simulations don't function in the game engine. I also note that the objects in this game don't cast shadows. I've not been able to figure out how to fix that, yet. The biggest challenge I faced with the BGE was the fact that object textures don't always seem to work the same in the BGE as they do in the render engine. Obviously, because the BGE is generating screen images in real time, some compromises have to be made. I eventually got satisfactory results, but sometimes I had to resort to UV mapping the texture onto the object to get it to work properly. Maybe I'm missing something.

As you can see, the BGE and Bullet Physics Engine are extremely powerful and configurable ways to create life-like game environments quickly. My initial plan for this article was to use a simple flight simulation as a demonstration. Once I started writing, I decided that a flight simulation just didn't fully demonstrate the power of the BGE.

Other types of games are well suited to writing with the BGE. For example, a billiards game is an obvious choice. Perhaps an off-road or monster-truck-driving simulation would be fun. Not having to write code in order to get game objects to behave intuitively means you can concentrate on creating interesting games with realistic content, and that is really cool.■

Mike Diehl runs an IP telecommunications company and lives in Albuquerque, New Mexico, with his wife and three sons. He can be reached at mdiehl@diehlnet.com.

Resources

Blender: **www.blender.com**

"3-D Graphics Programming with Irrlicht" by Mike Diehl, *LJ*, April 2009: **www.linuxjournal.com/article/10294**

SILICON MECHANICS
RACKMOUNT SERVERS • STORAGE • HPC

For more information visit
www.siliconmechanics.com/R350,
www.siliconmechanics.com/A350,
or call us toll free at **866-352-1173**.

There's a lot of heavy lifting going on these days in the world of data processing. Just ask Jason, a Silicon Mechanics Sales Expert who fields questions every day about the best way to address a vast range of computing workloads. One solution finding enthusiastic adoption is hybrid CPU / GPU computing, such as with the Silicon Mechanics Rackform iServ R350 and the Rackform nServ A350.

We start with your choice of two state-of-the-art processors, for fast, reliable, energy-efficient processing. Then we add two NVIDIA® Tesla GPUs, to dramatically accelerate parallel processing for applications like ray tracing and finite element analysis. Load it up with DDR3 memory and you have herculean capabilities and an 80 PLUS Gold Certified power supply, all in the space of a 1U server.

When you partner with Silicon Mechanics, you get more than breakout technologies that will carry the weight of your workload—you get an expert like Jason.

Expert included.

Put Your Office in the Clouds with OpenGoo

Create multiple private workspaces for clients, friends, family and coworkers, and share documents, calendars and tasks easily. / **HANI SAIGH**

Google is a great company that delivers world-class solutions, yet like governments, if you rely on them or any other entity to keep your best interests at heart, you may be in need of some Freedom Technology. I am a regular user of Google's Gmail, and I was using its Calendar and Sites daily in my work. We stored all relevant assets for Web projects, like database login information, Web site content and images, logos and banners in Google Sites. Being able to access all this information from anywhere at anytime, while keeping it private, proved to be a valuable tool. One fine morning in February 2009, I tried to log in to Google Sites to begin a new project and was met with "Your Account Has Been Disabled".

After being directed to a short form and submitting several requests for help, it took three weeks for Google to re-instate access to the account. I learned that the affiliate links we stored in Google Sites in our "Code Snippets" repository were considered to be in violation of Google's Terms of Service. Our sites were used as a private intranet, and the information stored there was used only to build Web sites, not display them, so it wasn't a violation at all. Yet, I had to re-create the information for roughly 25 different clients and projects, and I felt robbed of my data—or at least strong-armed away from it.

As soon as access was restored, I downloaded all of the remaining data, deleted it and vowed never to use Google or any other SaaS account for mission-critical data again. From now on, it's my data, and I'm taking back control. This set me on a search for a Google Apps/Sites replacement that I could run on my own server. I made a list of requirements for the ideal solution:

- Store information (mainly documents, graphics and notes).
- Allow me to share it with users, designers, developers and clients selectively.
- Act as a repository of tasks, deadlines and milestones.
- Easily keep everyone involved in the project(s) in the loop.
- Share tasks and calendar events.
- Be easy to back up.
- Be open source/GPL'd.
- Run on *my* server.

I discovered OpenGoo (now called FengOffice) on SourceForge less than a month later. Fast-forward two years, and my system never has experienced a hiccup, let alone data loss, and at least 25 different people I work with use it every day. My data is backed up securely, automatically every night, and it's accessible 24/7 from any Web browser or mobile device. OpenGoo uses the ExtJs framework and is designed to run completely in the Web browser, so it's platform-independent. The data is stored in open, non-proprietary formats, so I can access it from any device out there and transfer it easily between applications. It's pure Geek Zen. It's Freedom Technology.

Zen and the Art of Installation

I'm going to walk you through setting up and running your own "office in the clouds" that will give you both peace of mind and total control. Once installed on your server, you will have 24/7 access to your Web-based documents (including graphics), spreadsheets (coming soon), presentations, task lists (with time tracker), e-mail, calendars, Web links, contacts and reports.

The prerequisites for running OpenGoo are the following:

- Linux server.
- Apache >= 2.0.
- MySQL >= 5.0 (5.2 recommended).
- MySQL >= 4.1 with InnoDB support.

Note: the default memory limit for PHP is 8MB. A new OpenGoo install consumes 10MB, so you may get a message similar to "Allowed memory size of 8388608 bytes exhausted". Solve this by setting `memory_limit=32` in php.ini.

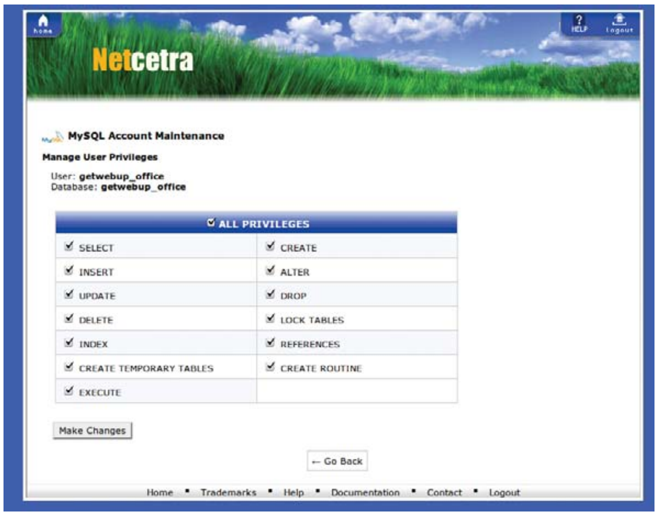
Let's launch the on-line office at http://getwebup.com/office. Here are the steps:

1. Download the latest version of OpenGoo at **sourceforge.net/projects/opengoo/files/fengoffice**.

2. Unpack the files and upload (via FTP) everything under feng_community to your desired location (for this example, I created and used /home/getwebup/public_html/office).

3. Log in to your server CPANEL, create the database and user/password (I used getwebup_office for both). You are going to have a great place to store this information soon, so go ahead and use the password generator to create a difficult password.

4. Add the user getwebup_office to the database: getwebup_office, and be sure to assign all privileges.



MySQL Account Maintenance

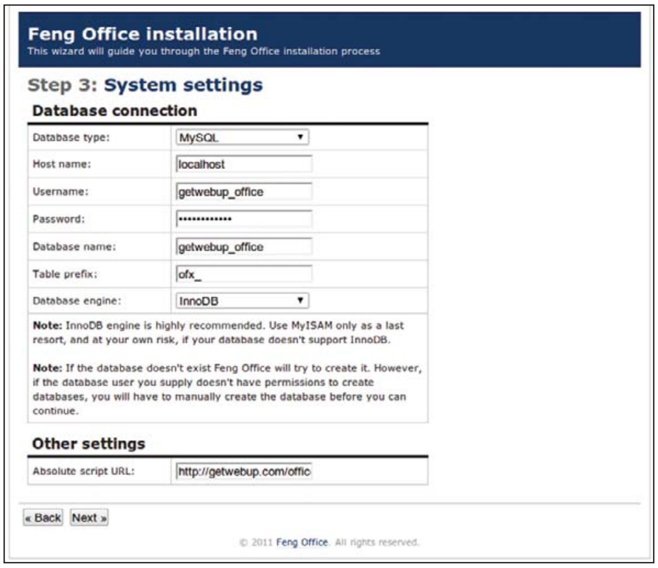
Manage User Privileges

User: getwebup_office
Database: getwebup_office

☑ ALL PRIVILEGES	
☑ SELECT	☑ CREATE
☑ INSERT	☑ ALTER
☑ UPDATE	☑ DROP
☑ DELETE	☑ LOCK TABLES
☑ INDEX	☑ REFERENCES
☑ CREATE TEMPORARY TABLES	☑ CREATE ROUTINE
☑ EXECUTE	

Figure 1. Adding the User to the Database in CPANEL

Now you can exit CPANEL, as the rest of the installation is done from within OpenGoo's installer. In your browser, go to **getwebup.com/office/feng/public/install**.



Feng Office installation

This wizard will guide you through the Feng Office installation process

Step 3: System settings

Database connection

Database type:	MySQL
Host name:	localhost
Username:	getwebup_office
Password:	••••••••••••
Database name:	getwebup_office
Table prefix:	ofx_
Database engine:	InnoDB

Note: InnoDB engine is highly recommended. Use MyISAM only as a last resort, and at your own risk, if your database doesn't support InnoDB.

Note: If the database doesn't exist Feng Office will try to create it. However, if the database user you supply doesn't have permissions to create databases, you will have to manually create the database before you can continue.

Other settings

Absolute script URL:	http://getwebup.com/offic

© 2011 Feng Office. All rights reserved.

Figure 2. Installation Made Easy

The OpenGoo installation will prompt you for your database and server information, after which you create your Administrative login. Once completed, you are ready to go, and you can access your new on-line office at http://.getwebup.com/office.

Setting Up the Ultimate On-line Office

Let's get to work setting up OpenGoo as your private digital workspace. I use it every day, and it's been an incredible asset. My clients tell me it's very intuitive, and they enjoy being able to keep abreast of the development process. They can schedule appointments, comment on projects, add tasks or notes and contribute documents and images easily. The developers and I don't have to search our e-mail for attachments that inevitably get lost like that "other" sock in the wash.

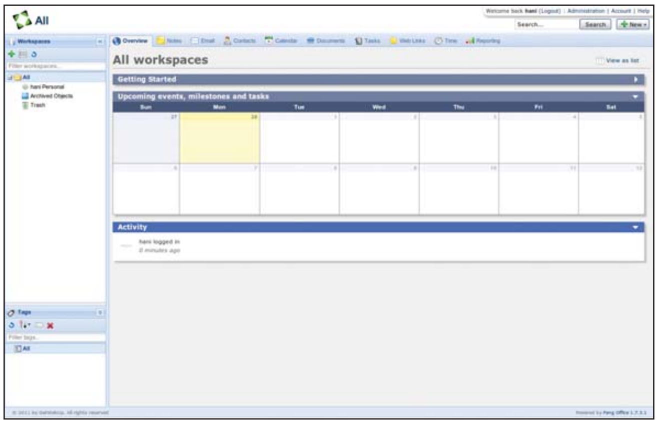


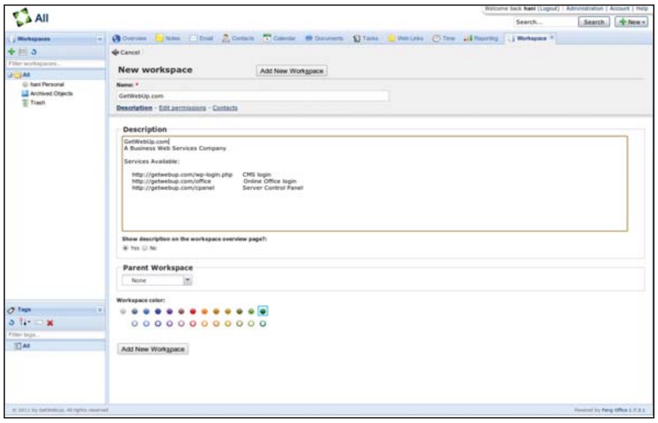
Figure 3. Here's a screenshot from our new office installation.

In the left-side window pane, you have Workspaces, and dominating the center and right of the screen is the All Workspaces

Activity window and a tabbed interface with these sections: Overview, Notes, Email, Contacts, Calendar, Documents, Tasks, Web Links, Time and Reporting.

Creating Workspaces

Workspaces are where all the action happens. Let's assume you are using this office for a company called GetWebUp, which provides business Web services (using only open source and open standards as the delivery vehicle). You'll note that a "userPersonal" workspace already is created. You can use this for personal login information for Web sites or even to store to-do lists, recipes and journals. This workspace will not be visible to anyone but you.



GetWebUp.com
A Business Web Services Company

Services Available:

http://getwebup.com/wp-login.php   CMS login
http://getwebup.com/office   Online Office login
http://getwebup.com/cpanel   Server Control Panel

Figure 4. Adding Workspaces in OpenGoo

Create a new workspace by clicking on the green plus sign just below the Workspaces header. This opens the New Workspace page. Let's give this one the name "GetWebUp.com". Click the Description link, and type some basic information for the project:

```
GetWebUp.com
A Business Web Services Company
Services Available:
http://getwebup.com/wp-login.php    CMS login
http://getwebup.com/office          On-line Office login
http://getwebup.com/cpanel          Server Control Panel
```

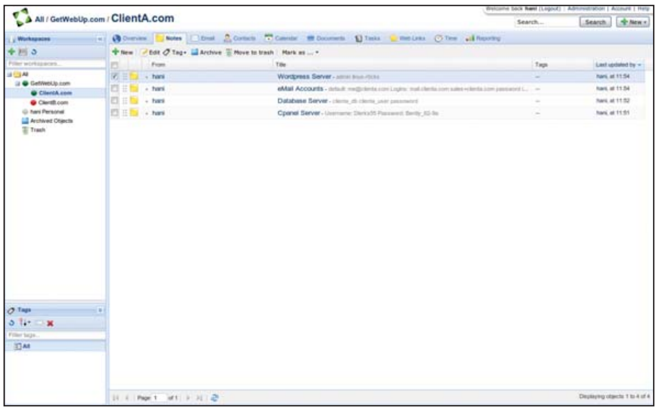
Select Yes in the Show Description option, assign a Parent Workspace (None), and give the workspace a solid-color green (for Active). Click Add New Workspace, and it's created and ready to use for organizing your GetWebUp office data.

Creating Sub-Workspaces

Select the newly created workspace, click Add Workspace to create sub-workspaces under it, and add two client Workspaces. Name one Client A.com, and use a solid-color green (for Active). Name the other Client B.com, and use solid-color red (for InActive). The GetWebUp Parent Workspace is selected by default, and these two Workspaces now will appear as Sub-Workspaces, indicated by the plus sign to the left of the Workspace name. Now you've got a basic structure in place, so let's create some data.

Putting Everything in Its Place

When we bring in a new client, I create and store several logins, like control panels, WordPress logins and database servers. I use the Notes tab for this. Click on the ClientA.com Workspace, and then click Notes + New. Here you have the ability to assign multiple workspaces, tags and link objects to the Note. For now, let's just type in a title and some dummy data to serve as a placeholder. Add one note each for: "Cpanel Server", "Database Server", "eMail Accounts" and "Wordpress Server".



From	Title	Tags	Last updated by
hani	Wordpress Server	--	hani, at 11:54
hani	eMail Accounts	--	hani, at 11:54
hani	Database Server	--	hani, at 11:52
hani	Cpanel Server	--	hani, at 11:51

Figure 5. Using Notes to Organize Logins

Now that you have a place to store ClientA's login information, let's visit the Documents tab and create placeholders for the Web site content. Click Documents + New, select Document from the drop-down menu and create the following by saving blank documents: "Home Page Information", "Article 1", "Article 2", "Article 3", "Testimonials" and "Contact Us Info".

You also can upload images that will be used for the project. In the Documents tab, click + New, and select Upload: "ClientA.com-logo.png", "ClientA.com-Banner300x300.png".

Now you have a place to store and organize the information you get from the client before development takes place, so the content is ready when the Web site is. It also provides great version tracking with document check-outs, as well as a commenting system on each piece of content in the system. When the client has a Web site strategy, branding guidelines and marketing campaigns, those also can be uploaded or created here. For the time being, the data is assigned (and visible) only to the user hani (Admin).

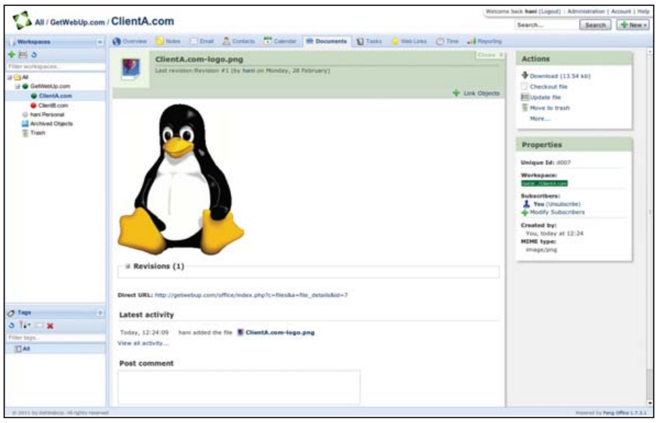


Figure 6. Storing Images in the Documents Section

Sharing and Collaboration, Open-Source Style

You've got your Workspace set up and data in the system, so let's create some users to collaborate with. Go to the Administration panel via the link at the top right of the screen next to "Welcome back hani". Click on + Add Company under the Client companies icon. Create Client Company. Add ClientA.com, and fill in the details as desired. Add ClientB.com, and fill in the details as desired.

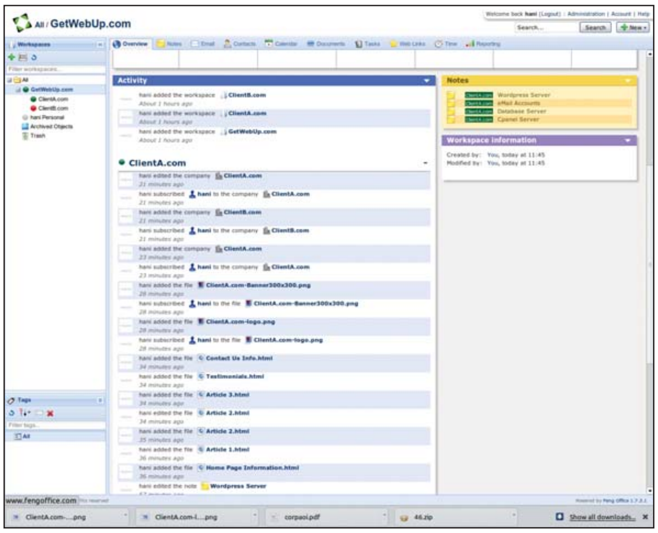
Now click on + Add User under the Users icon. Create users Dave, Dirk and LJReader, and assign them to ClientA.com. Click on the Permissions link, and drill down to select the ClientA.com Workspace. All permissions for the objects in that Workspace are pre-selected. Now assign a display name, and either let the system generate a password or specify a password for the user, and check Send email notification to have it sent. Click Add User, and OpenGoo adds the user to the office under the ClientA.com Workspace.

Repeat the steps for the rest of ClientA.com's users, and do the same for these users of ClientB.com: Jill, Kyle and Shawn.

The users are ready to use the office, so let's get back to the GetWebUp.com Workspace.

The Workspace Overview Page

When you return to the GetWebUp.com Workspace, you will have a default view of the Overview tab. This gives you the 42,000-foot view of the action in any given Workspace. User activity, new



hani added the workspace ClientB.com
About 1 hours ago
hani added the workspace ClientA.com
About 1 hours ago
hani added the workspace GetWebUp.com
About 1 hours ago

ClientA.com

hani edited the company ClientA.com
21 minutes ago
hani subscribed hani to the company ClientA.com
21 minutes ago
hani added the company ClientB.com
21 minutes ago
hani subscribed hani to the company ClientB.com
21 minutes ago
hani added the company ClientA.com
23 minutes ago
hani subscribed hani to the company ClientA.com
23 minutes ago
hani added the file ClientA.com-Banner300x300.png
28 minutes ago
hani subscribed hani to the file ClientA.com-Banner300x300.png
28 minutes ago
hani added the file ClientA.com-logo.png
28 minutes ago
hani subscribed hani to the file ClientA.com-logo.png
28 minutes ago
hani added the file Contact Us Info.html
34 minutes ago
hani added the file Testimonials.html
34 minutes ago
hani added the file Article 3.html
34 minutes ago
hani edited the file Article 2.html
34 minutes ago
hani added the file Article 2.html
35 minutes ago
hani added the file Article 1.html
36 minutes ago
hani added the file Home Page Information.html
36 minutes ago
hani edited the note Wordpress Server

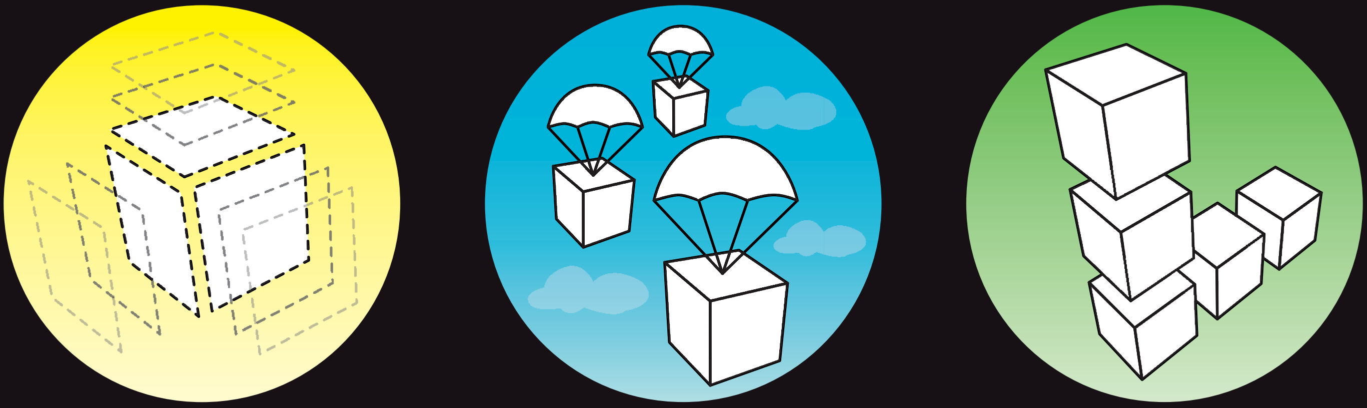
Notes
ClientA.com Wordpress Server
ClientA.com eMail Accounts
ClientA.com Database Server
ClientA.com Cpanel Server

Workspace information
Created by: You, today at 11:45
Modified by: You, today at 11:45

Figure 7. Workspace Activity Feed

comments, notes, documents and tasks appear here. We have generated a lot of activity ourselves, and a news feed of what we have been doing appears here, as well as handy hyperlinks to the users/data we worked on.

Click on the Notes tab, and select all the notes you created



Develop. Deploy. Scale.

Full root access on your own virtual server for as little as $19.95/mo

Multiple Linux distributions to choose from • Web-based deployment • Five geographically diverse data centers • Dedicated IP address • Premium bandwidth providers • 4 core SMP Xen instances • Out of band console access • Private back-end network for clustering • IP fail-over support for high availability • Easily upgrade or add additional Linodes • Free managed DNS

For more information visit **www.linode.com** or call us at **609-593-7103**

linode.com

previously (one by one) and click Edit. Here, you can assign custom properties, tags and turn commenting on or off. For now, let's just add Subscribers. This is the equivalent of sharing the data with only the specified users, so click on that link. Select hani from GetWebUp and Dave, Dirk and LJReader from the ClientA.com company. These users now will have access to and be notified of comments and changes to these notes. Click Apply. Use this same method to share all the documents you created earlier.

Managing Time in Your Digital Office

The calendar in OpenGoo is very well implemented. It will import from other systems and allow you to create events quickly, or you can click Edit Event Details to delve into alarms, recurrence and subscribers with ease. Your views for Month, Weekly, Daily and Agenda are present with a large +Add Event button above them. Click on the day directly in the Month view and the hour specifically in the Day view to set up appointments and meetings in that slot.

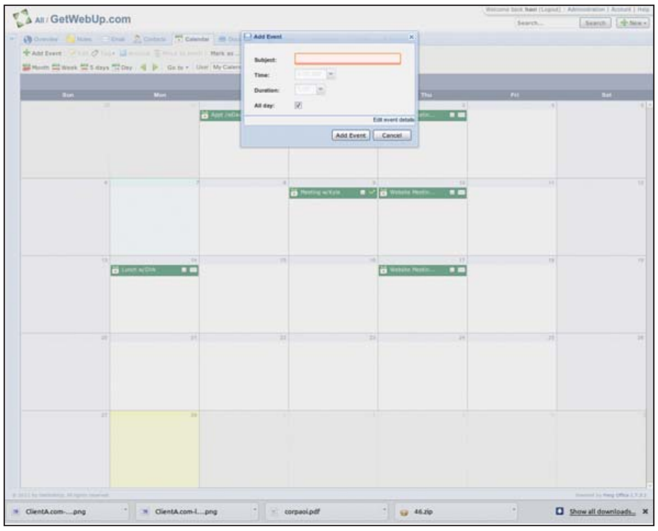


Figure 8. Monthly Calendar View with Workspace Pane Collapsed

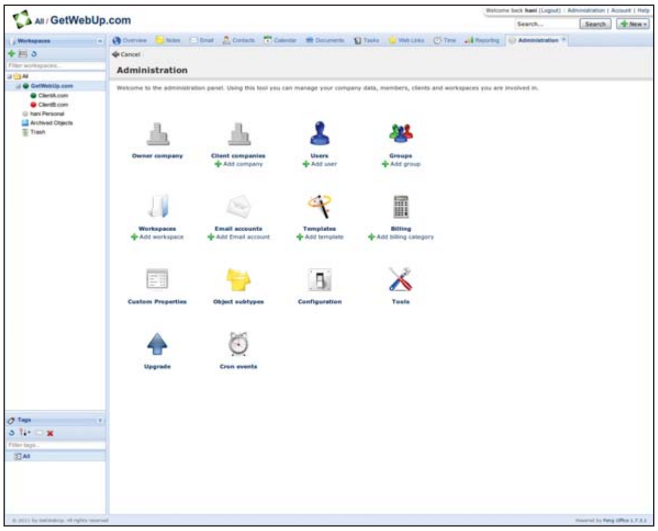
In the Tasks tab, you can set Milestones for your projects and assign Tasks to the users involved in the project. These are treated like regular documents and allow users to comment, collaborate and share data directly within the Task's screen. This screen also includes two very useful buttons: Start Work, which is a work-timer feature, and Add Work, which allows you to add tasks within tasks. Both will give you data to crunch in the Reporting tab when you perform project time/cost analysis.

Weblinks allows you to store URLs that link to project-related sites. We use this section to store quick links to login pages, development servers and control panels. Each link can have its own Subscriber/Workspace settings, which makes it useful for a project manager to create a central repository of private and public URLS that both developers and clients can access in one place.

Once the system is in use on a regular basis, the Reporting tab will give you an inexhaustible number of ways of looking at your data. From simple time-card generation to detailed workspace usage and milestone reports, this section lets you create virtually any type of report required from the office.

Rolling Up Your Sleeves

Once you have your on-line office running and users on the system, you can perform regular admin tasks by visiting the Administration panel. An upgrade tool and cron jobs are available here, as are data and user tools. OpenGoo even provides the ability to create templates, user groups and perform billing functions. Let's look at the settings you can modify.



Welcome to the administration panel. Using this tool you can manage your company data, members, clients and workspaces you are involved in.

Figure 9. OpenGoo Administration Panel

In the Configuration section, you can access the General section, which allows you to specify the File storage system—either database or filesystem. I opt for the filesystem, then use rsync to send the entire office off-site for backups, restoration and portability. The database option allows you to manage your data MySQL-style from the command line. Another useful option to set here is to "use the client's logo as the application logo when they log in". This is a nice touch, as it gives each company a sense of ownership over its office/workspace.

The Modules screen under Configuration allows you to enable or disable each section (represented by tabs) in the office. We use Gmail in lieu of the Contacts and Email sections, so I disable those modules here to save screen real estate and provide a cleaner interface. You can use the Custom Properties section to add additional fields of data to any object type. For example, you can add an expiration date property to the Note object type. Object subtypes allows you to add subtypes to task objects like "Goal 1" and "Goal 2" and so on.

Profile and Workspace Personalization

The Account link is to the right of Administration, and here you can update your profile to include an avatar, specify an e-mail address, time zone and title. In the Edit Preferences screen, you can access the General, Dashboard, Task, Calendar and Email Options. Use the options under General to set a default workspace, localization and workday start time. You can add or remove Workspace widgets and set the Activity Widget size (number of items in the Activity stream) in the Dashboard options screen. The Calendar options page gives you the ability to start the week on Monday and show week numbers.

One final step I like to perform, although this is a Chrome browser feature rather than an OpenGoo setting, is to click on the wrench icon in Google Chrome, then Tools→Create Application Shortcuts to create Icons on my desktop and menus to access my office quickly. Chrome gives the application a non-decorated window, which gives it the look and feel of a desktop application. In Ubuntu, I always have this window running in its own workspace for fast access.

Office in the Clouds

Now you've got your office up 24/7 and out of reach of the general public, while remaining completely platform-independent and fully featured on any device. There is talk of iPhone and Android Apps for OpenGoo on the horizon, but for now, it's best on Galaxy Tab and iPad-sized screens unless you like pinch zooming. The application itself is open source and now hosted on your own server. The data is stored in an open and non-proprietary format, so you've taken ownership of your own office in the clouds and increased the versatility of your raw data in the process. You can export your data from OpenGoo and import it into any standard application. Although once you get used to having access to your data at any time and being in total control of it, why go back to running your business on someone else's proprietary platform?■

Hani Saigh is the Founder and CEO of Netgetup | Freedom Technology, a Linux/open-source and open-standards advocate and singer/guitarist/songwriter in the band Freedom Blvd. He loves astronomy and reads too much Carl Sagan, so not many people like to talk to him, which gives him plenty of time to write songs and develop cutting-edge business Web solutions built on Linux. He can be reached at hani@netgetup.com.

Resources

OpenGoo Installation: **www.fengoffice.com/web/wiki/doku.php/installation**

The following open-source libraries and applications work with OpenGoo:

ActiveCollab 0.7.1: **www.activecollab.com**

ExtJs: **www.extjs.com**

Reece Calendar: **sourceforge.net/projects/reececalendar**

Swift Mailer: **www.swiftmailer.org**

Open Flash Chart: **teethgrinder.co.uk/open-flash-chart**

Slimey: **slimey.sourceforge.net**

FCKEditor: **www.fckeditor.net**

JSSoundKit: **jssoundkit.sourceforge.net**

PEAR: **pear.php.net**

MONOSPACE
Boston - July 23rd to 25th
http://monospace.us
Microsoft
New England Research & Development Center
be part of the conversation
meet the mono engineers
create an ecosystem
© E. Huybrechts

Use Linux to Control Real-World Hardware

With Open-USB-IO, your Linux system can drive motors and LEDs, read switches, talk to RS-232 and more. You can add your own code to the ATMEGA32 microprocessor on the board and debug it with a symbolic debugger.

PJ RADCLIFFE

Linux is wonderful for many things, such as manipulating data and working with the Web. But, pull your gaze away from your computer's screen for an instant, and you'll notice that Linux (and any other operating system) can't do much in the real world. What might you want to do in the real world using Linux? How about controlling motors, solenoids and lights? Add to this the ability to sense switches, light levels and any analog voltage. What could you do with all this? That really depends on your imagination. How about an automated beer-brewing controller or automatic blinds? How about controlling a robot arm to throw a paper dart or creating a grey-water controller? After you have proved it works under Linux control, if need be, you can move the program into a micro-controller board so your Linux box is free to do other things.

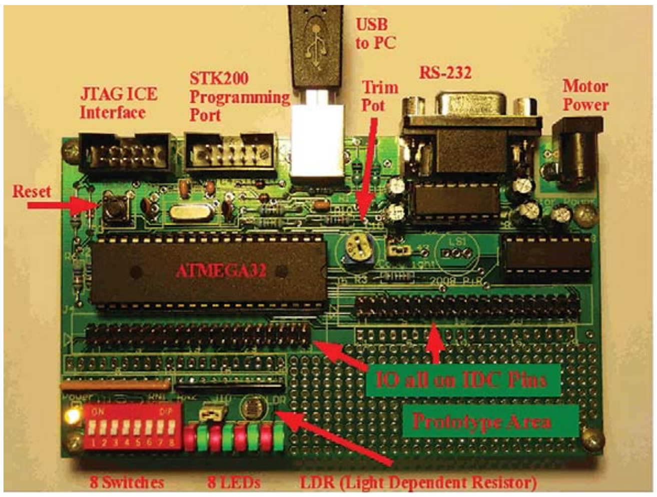


Figure 1. Open-USB-IO Board

If this appeals to you, the FOSS Open-USB-IO board shown in Figure 1 may be exactly what you need. There also is a live DVD that has a detailed manual, example projects and the source code for all the software. The live DVD is the easiest way to get results quickly. It also has a huge range of Linux development tools and helpful documentation. The manual can be downloaded free from my Web site listed at the end of this article.

Historically, hobbyists used the parallel printer port or a serial port to drive hardware. You still can find a lot of projects on the Web that use this approach. These legacy ports started disappearing from laptops long ago and are beginning to disappear from desktops. Hardware control via USB rapidly is becoming the only viable approach.

Open-USB-IO Features

Key hardware on the board includes digital and analog inputs and outputs, eight switches, eight LEDs, a light-dependent resistor, seven motor control lines that can take up to 50v and 500ma each, an RS-232 port and three Pulse Width Modulators, which are great for motor-speed control. All the connections come to plugs, which can be connected to external hardware with an old IDE cable (Figure 2).

There are three key bits of software. This first is a command-line program that drives the USB interface on the PC. It doesn't need any special drivers and runs like any other command-line program. The second is the USB client program that comes programmed into the Open-USB-IO board. This interfaces with the

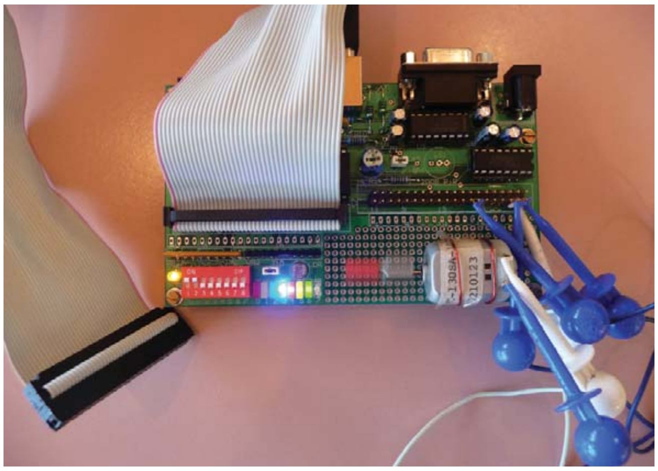


Figure 2. Cable Connection to External Hardware

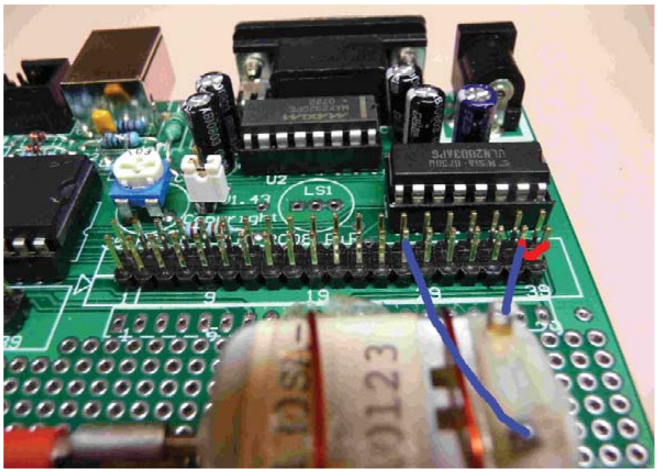


Figure 3. DC Motor Connections

USB and provides a whole range of useful commands to control the hardware. The third software program is also on the board: a USB bootloader that allows you to download your own code into the ATMEGA32 microprocessor using just the USB cable.

Control Where?

The Open-USB-IO hardware can be controlled from several different levels. You can use:

- GUI (coming soon or make your own).
- Command line.
- Bash script on the PC.
- C/C++ or almost any other language on the PC.
- C or assembler code that runs on the microprocessor on the board. User code can run at the same time as the USB interface, which includes a powerful symbolic debugger.

One neat approach is to write in C code on the PC and play around until you get what you want. Next, move this C code to a shell ATMEGA32 project, and make the code run on the board, which then does not need a PC connection.

Playing with the Command Line

Open-USB-IO comes with a command-line program called ousb that can be downloaded from my Web site (see Resources). Put this in the path, typically /usr/local/bin, and ensure that it's executable. Next, plug in a USB cable to the board and you're ready to go.

To see all the commands, just type `ousb`. To turn on all eight lights, open a terminal window and type the command:

```
$ ousb io portb 255
```

To turn on alternate lights and then turn them off, type:

```
$ ousb io portb 85
$ ousb io portb 0
```

If you want the state of the switches, type:

```
$ ousb -b io pinc
```

The `-b` will make the response in binary so you can see each bit clearly.

Let's control a motor. A Pulse Width Modulator (PWM) puts out a square wave where the on period is adjustable from 0% to 100%. Type the following commands:

```
$ ousb pwm-freq 1 700
$ ousb pwm 1 50
```

This turns on PWM 1 at a frequency of about 700Hz and a duty cycle (on period) of 50%. You should see LED3 glow at half intensity. If you have a small DC motor that can run off five volts, connect it as shown in Figure 3. Connect the motor to pins 39 and 27 of plug J5 (blue wires), and connect pin 39 to 37 (red wire). The motor should start turning slowly. To make it run at full speed, 100% duty cycle, type:

```
$ ousb pwm 1 100
```

What's the light level on the board? Try typing:

```
$ ousb adc 6
```

Try holding your finger over the LDR just above the LEDs, and try again.

The Open-USB-IO manual has a complete list of all the commands you can use and lots of examples.

Script Programming on the PC

Bash script programs are great for small jobs and are quick and relatively easy to develop. For our purposes, they really are just ousb command lines plus any control flow required. Below is one example from the live DVD that turns the LEDs into a light chaser and prints out the value of the switches:

```
#!/bin/bash
```

```
PATTERN=1
until  [ 0 != 0 ]
do
    ousb io PORTB $PATTERN        # write to the LEDs
    READ=$(ousb io PORTC)         # read the switches.
    echo " LED Output= $PATTERN, Input on PORTC= $READ."
    sleep 0.3
    let "PATTERN = PATTERN + PATTERN"
    if [ $PATTERN == 256 ]        # Check for roll over
        then PATTERN=1
    fi
done
```

C, C++ and Other Languages on the PC

Script programs are great, but they have a few problems. The syntax is a little arcane at times—for example, white space matters in some places. There is no real error detection. If a line has an error, it may not be detected until the line executes. If you use the `set -u` command, you sometimes can catch a misspelled variable.

For these and other reasons, larger applications are written in high-level languages, such as C, C++ and Python. Open-USB-IO is easy to control from these high-level languages, providing that the language has some way to invoke a command line and read the response. When programming Open-USB-IO applications on the PC, I usually program in C, because it's then a small step to moving the code into the microprocessor on the board.

Here is a simple C code fragment that runs on the PC and will read and write from the board:

```
//--- function to read and write Open-USB-IO.
int do_ousb_command(char* command)
{
  char   line[100];
  FILE*  fpipe;
  if ( !(fpipe = popen(command,"r")) ) {
      printf("pipe error\n");
      exit(1);
  }
  fgets(line, sizeof line, fpipe);
  pclose(fpipe);
  return(atoi(line)) ;
}

...

//--- how to use the function, a write then a read-
do_ousb_command("ousb -r io portb 0x55") ;
printf("Value of PORTB is now %i\n",
       do_ousb_command("ousb -r io portb")
```

As you will see later, to move this code to the ATMEGA32 microprocessor on the Open-USB-IO board, you throw away the function do_ousb_command. Each use of `do_ousb_command()` must be changed into a native IO port reference. In this case:

```
PORTB = 0x55;
portb_read_value = PORTB;
```

Faster and Faster

The examples above all invoke the ousb program once for each command. On most Linux boxes, this limits the speed to about 25 commands per second. This is fast enough for most applications, but sometimes more speed is required. For example, the motor drive lines can be used to drive a stepper motor, such as those found in old floppy drives. A speed of 25 commands a second means it takes some eight seconds to do a full rotation. This is where the ousb `-file` and `-multi` command options become useful and allow 200–250 commands per second. The `-file` option makes ousb take its commands from a file. The `-multi` option allows ousb to stay in memory and take its commands from a pipe connected to an application program. The live DVD has examples of how to use these options. Your stepper motor now takes only a second for a full rotation! To go even faster, you can write C code for the ATMEGA32 microprocessor.

Cutting Free

So now you have got the hardware working well from your C code on a Linux box. It's easy to move the same code into the microprocessor on the board and have the board operate in standalone mode. The manual and the example projects on the live DVD show you exactly how to do this. Basically, compile your code with the avr-gcc compiler then download it with the USB bootloader already programmed onto the board. This is called Co-USB, where your code cohabitates with the USB code on the ATMEGA32.

Now, unplug the USB cable, plug in a regulated 5-volt wall wart, and the board will happily work by itself!

Perhaps even better, the Co-USB concept allows you to add your code to the existing USB interface code and built-in debugger. The USB code is hidden away as interrupt-driven code and has little effect on the user code. With the USB link active, your application running on the Open-USB-IO board can talk to another program you write on the PC. This can be really useful—for example, for a data logger. The Open-USB-IO board can collect data in a standalone manner. You can come along with your laptop, plug in the USB cable, and then suck up all the data and post-process it.

Let's look at a simple example to show how Co-USB works. This is taken from the Co-USB folder available on the live DVD. A make file handles all the details of compiling and downloading code; type `make help` to see all the make options:

```
//====== user constants and variables.
volatile uint8_t  counter;
volatile float    fb[] = { 56789.1, 0.34};

//====== SYSTEM hooks.
void user_system_500us_interrupt()
{}
void user_command(uint8_t*  get_ctrl,
                  uint16_t* get_addr,
                  uint16_t* val)
{}

//====== Executive Loop.
void user_forever_loop()
{
    for ( ; ; ) {
        _delay_ms(200);    // delay 200ms
         PORTB ^= 0x01;    // toggle the PB0 LED.
```

```
            ++counter;          // increment once per loop.
            BREAKPOINT(3);      // breakpoint
        }
}
```

To compile the program, open a terminal and type `make all`. Download using the USB bootloader with `make usbprog`. The LED PB0 now should start flashing. The following commands come from a debugging session with a real Open-USB-IO board (text after the # are added comments):

```
$ ousb symr counter     # read the variable counter.
counter (type 'uc' at 0xc2) length 1 is:  55
$ ousb symw counter 0   # write zero to counter.
$ ousb symr -f fb 2     # read the array.
fb (type 'f' at 0x68) length 2 is:  56789.1  0.34
$ ousb bp 3             # Set breakpoint 3, code stops
$ ousb bp cont          # Continue,  stops on next loop
$ ousb bp -3            # Disable the breakpoint
$ ousb cont             # Continue, no breakpoints now
```

Note the user_command() function in the code above. From the PC, the command `ousb user` sends up to three integers to this function. You can put your own code in the function to do whatever you want. You also can use the function `user_system_500us_interrupt()` to execute your own code. As its name implies, it is called every 500 microseconds. If you don't need it, just leave it empty.

There are many useful options for displaying and writing different data types, such as strings and characters. All the read and write commands also can be applied to the EEPROM memory in the ATMEGA32, which keeps its value when power is removed.

Up, Up and Away

It takes very little knowledge of electronics to get a lot done with Open-USB-IO. Anyone who has done a college or secondary-school option in electronics, or who is a hobbyist, can connect up switches, motors and LEDs to do all manner of things. There are lots of electronics cookbooks and good resources on the Web to help you with ideas and circuits.

I have seen some very exciting projects done by people with little more than basic electronics skills. For example, how about controlling banks of floodlights using Open-USB-IO and optical relays? How about a complete pump controller that handles a dam, tank or grey-water, and has an RF link to a control panel? Or, consider making a white-line tracking model car. In all these projects, the Co-USB concept is very important: program the ATMEGA32 microprocessor with user code and link in the USB interface code with the built-in debugger. Develop and debug the user code on the ATMEGA32 until it works. Next, pull out the USB cable and leave the Open-USB-IO board running. If the program goes wrong, plug in the USB cable and debug what's happening.

Conclusion

Linux by itself is great, but it can be so much more if you can control hardware. Open-USB-IO has a unique combination of hardware and software features that offers a lot to casual users or experienced experts. You can control hardware from the PC command line or your own PC programs. You can program the ATMEGA32 microprocessor on the board, and even keep the USB interface code and its powerful symbolic debugger. It takes only a little knowledge to get real hardware working, and if you have more experience, you are limited only by your imagination.■

Dr PJ Radcliffe (pjr@rmit.edu.au) is a senior lecturer at RMIT University in Australia. He was once an ardent Windows programmer but then discovered Linux, which he now teaches along with the control of hardware using Linux. Of all the programming he's done, writing code to control hardware is certainly the most fun.

Resources

See **pjradcliffe.wordpress.com** to download the manual (yes do read the manual first) and for plenty of free, open-source software for Open-USB-IO.

See **interestingbytes.wordpress.com** for the live DVD that contains all the Open-USB-IO software properly installed, source code and more example programs. The DVD also contains many Linux development tools. This site retails the Open-USB-IO board as a kit or fully assembled, tested and programmed.

Open Frame Panel PC

PPC-E7+

- Fanless ARM9 400MHz CPU
- 4 Serial Ports & SPI
- Open Frame Design
- 10/100 BaseT Ethernet
- SSC-I2S Audio Interface
- 2 High Speed USB 2.0 Ports
- SD/MMC Flash Card Interface
- Battery Backed Real Time Clock
- Up to 1 GB Flash & 256 MB RAM
- Linux with Eclipse IDE or WinCE 6.0
- JTAG for Debuging with Real-Time Trace
- WVGA (800 x 480) Resolution with 2D Accelerated Video
- Wide Input Voltage from +12 Vdc to +26 Vdc

Microsoft Windows CE

2.6 KERNEL

Setting up a Panel PC can be a puzzling experience. However, the PPC-E7+ Compact Panel PC comes ready to run with the Operating System installed on Flash Disk. Apply power and watch either the Linux X Windows or the Windows CE User Interface appear on the vivid color LCD. Interact with the PPC-E7+ using the responsive integrated touch-screen. Everything works out of the box, allowing you to concentrate on your application, rather than building and configuring device drivers. Just Write-It and Run-It.

For more info visit: www.emacinc.com/panel_pc/ppc_e7.htm

Since 1985 OVER 25 YEARS OF SINGLE BOARD SOLUTIONS

EMAC, inc. EQUIPMENT MONITOR AND CONTROL

Phone: (618) 529-4525 • Fax: (618) 457-0110 • www.emacinc.com

So, You Want to E-Publish?

E-publishing for e-readers involves a bit more than merely printing to a PDF—at least if you want a professional-looking result. As always though, open source can help you get from here to there.

DAN SAWYER

With Sony e-readers, Kindles, Kobos, Nooks, iPads, smartphones and the new Nintendo handhelds all giving you the ability to read on the go without having to cart around a stack of paperbacks, the e-book market is growing like an alien embryo in a doberman. All the major stores allow independents to list with them, and the result is, not surprisingly, a lot of books that look like they weren't edited or typeset at all. Ever seen a blog with no theme, where the text is blocky with no indents or proper spacing? A lot of these books look like that.

All these devices have created a demand for content, which means that, as with HTML in the early days of the Web, there's a demand for people who can make it look good. There also are independent-minded folks who are determined to do their own typesetting rather than contracting out the work, even if it means shelling out a lot of money for programs with which to do it.

But, this is open source, that wonderful corner of the world where we don't need no stinking proprietary programs. When it comes to e-books though, every e-reader device has its own proprietary format. EPUB is the available open standard, and being an open standard, it's pretty easy to migrate from EPUB to other formats for other readers (some retailers even will do that dirty work for you). First though, you have to make an EPUB, which looks from the outside like a nontrivial task.

It turns out there are about a dozen different ways to make an e-book on Linux, from OpenOffice.org templates and extensions to HTML-to-EPUB conversion utilities. You can use Scribus to lay out your e-book, export to PDF, convert to HTML and convert from there to EPUB. You even can hand-code the thing.

If you're not familiar with the difficulties and advantages of each approach, the decision can mean making a choice blindly. Covering all the options in detail would take a lot more space than I have in this article, so I concentrate on a method that exposes the underlying structure of the e-book for fine tinkering while using a GUI that streamlines the process.

The Theory

E-book authoring essentially is a layout and design process that uses HTML and CSS to create the final output, which is a limited subset of XHTML and XML. This final output is all stuffed into a single zip file and is designed to be widely readable. Because of this, working with it can be just different enough from hand-coding vanilla HTML to become maddening. Nonetheless, to create a proper e-book, you need, at minimum, the following:

1. Hierarchical arrangement of content.

2. Clickable tables of contents.

3. Any graphics should be anchored inline with flowing text.

4. The ability to use tables.

5. The ability to use indents. (It may seem like a strange item to be on the list, but one of the most obvious markers of an amateurish e-book is lack of indentation.)

The Work Flow

After a lot of tinkering, I settled on a work flow that seems to work well for most purposes—an exception may be textbooks and other nonfiction works that have highly complex layout requirements. Although I suspect that this work flow may accommodate such projects, I can't say with confidence that it will scale that far. There is, of course, one way to find out.

STEP 1: the Manuscript

First things first, laying out your e-book is not the same thing as writing it. Word processors and writing programs like CeltX have tools designed to make certain types of writing easier to manage—take advantage of them.

If you want your e-book to look good, start with a solid, clean manuscript that won't require editing when you get to the other side. Forget headers and footers—they don't translate. There is a way to do them in EPUB, but it's an optional part of the standard that most e-readers haven't implemented yet, so it's rarely worth the trouble.

As I'm writing this, the internal structure of EPUB files is difficult to manage in OpenOffice.org and other word processors, particularly with longer documents. So, although OpenOffice.org has some extensions to let it export EPUB, it's much quicker and cleaner to do the layout in a different program.

NOTE:

The definitive resource for all manner of programs and plugins for creating e-books and converting between e-book formats, including the OpenOffice.org extensions, is the MobileRead Wiki: **wiki.mobileread.com/wiki/E-book_conversion**.

For our purposes here, I've chosen Sigil, a purpose-built e-book editor that works very much like Quanta and other Web-authoring suites. Although it's still in its early days, it seems fairly solid, isn't crash-prone, and every tool included in the stable release works as advertised.

Once you have a clean manuscript in your word processor, export it to HTML.

STEP 2: Sigil

To follow along with the rest of this how-to guide, you need Sigil. The program plays nice with newer distros in its binary form, and it's a breeze to build from source. You can find everything you need (including detailed install and build instructions) at **code.google.com/p/sigil**.

Once installed, run the program. It should work without a hitch, but if it fails to start, chances are you don't have a current-enough version of libstdc++. If this is the case, you need to go back and install from source. The rest of us will wait here and mock your name for not reading the dependencies list while you catch up.

Sigil will open up looking something like Figure 1. Begin by importing the HTML of your book using the option under the file menu. Once done, the fun begins.

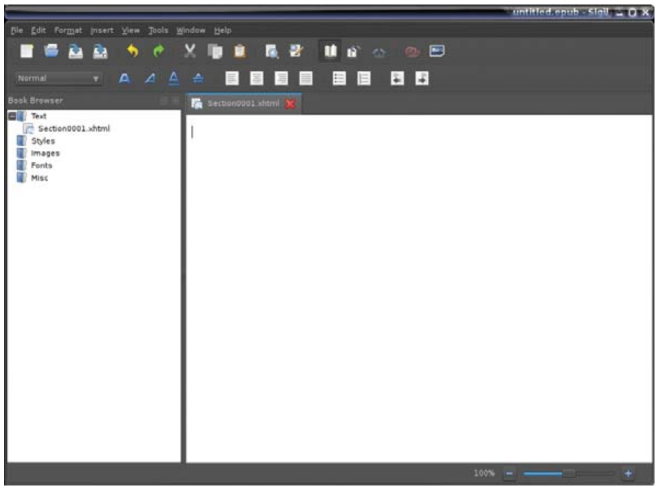


Figure 1. Sigil Opening Screen

The program has a few basic formatting tools. You can control paragraph spacing, chapters and so on very simply with the GUI tools, and I explain how to use them and the structural tools in a bit.

Like most WYSIWYG HTML editors, Sigil lets you edit the preview of your document directly. If you press the icon with the book and the brackets (Figure 2), it gives you split-window access to the underlying markup and CSS. (And, for purposes of brevity in this article, I'm forced to assume that you know the basics of HTML and CSS. If you need to get the basics, see **www.w3schools.com** or any one of a hundred other good, free resources on the Web.)

You'll notice that your imported HTML looks blocky—you've probably lost all your indents, although it's preserved

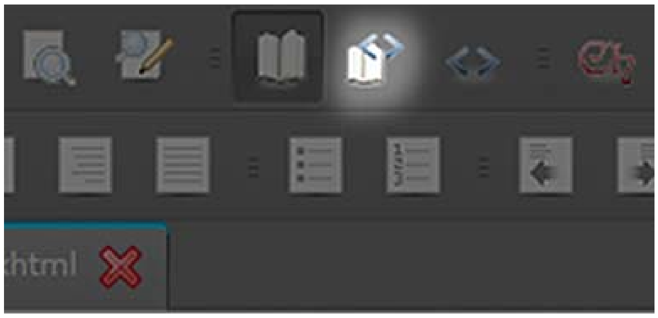


Figure 2. Toolbar Button to Show Underlying HTML and CSS

your bolds and italics—and it just looks and feels boring. Well, we're going to fix all that.

STEP 3: Basic Layout Conventions

The experience of reading on an e-reader is a mid-point between a paperback and a Web page, but the appeal of an e-book is that it delivers a more paperback-like experience than reading on an LCD without the bulk of an actual paperback.

There are a few layout conventions that differentiate a book from a Web site, and ignoring them gets you, at best, an e-book that looks like a vanilla text file with better fonts. (For a resource on good book layout conventions, see **www.members.shaw.ca/nathanieldesign/book_layout.htm**).

And, now that I've mentioned fonts, let's start there. The standards for the Web are Arial and Helvetica. They're clear, sans serif and easy to read even on poorly designed Web sites and dim screens. Aesthetically ennobling, they're not. Bare functionality is the name of their game, and they do it remarkably well. For e-books, however, you want a font that mimics the aesthetic of a paper book, so you'll want to use a proportional serif font like Garamond, Times New Roman or New Century Schoolbook. To set the font, edit the CSS like you would for a Web site to specify the font you need.

You also want to make sure your paragraphs are properly indented. If you're converting an OpenOffice.org .odt file to HTML, you're going to lose your tab-based indentations. The reason? In HTML, a tab means precisely nothing. The only way to get indentations in your e-book properly is to build them into the CSS. Using margin controls to do it in OpenOffice.org will put the CSS into your exported HTML, but you also can use the following code to indent a paragraph:

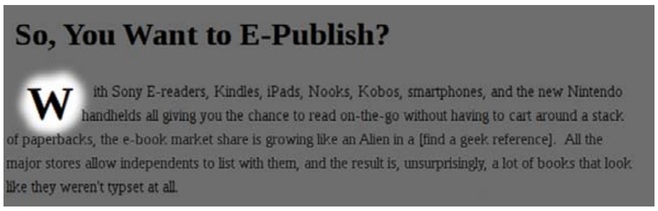
```
<p class="sgc-2"> paragraph </p>
```

Another little touch often used, particularly in novels, is the drop cap—the large, often stylized capital letter that begins the first paragraph of a chapter. You can do this too, in HTML, by applying the following to your CSS:

```
.dropcap {
     float: left;
     font-size: 3em;
     line-height: 1;
     font-weight: bold;
     margin-right: 0.2em;
}
```

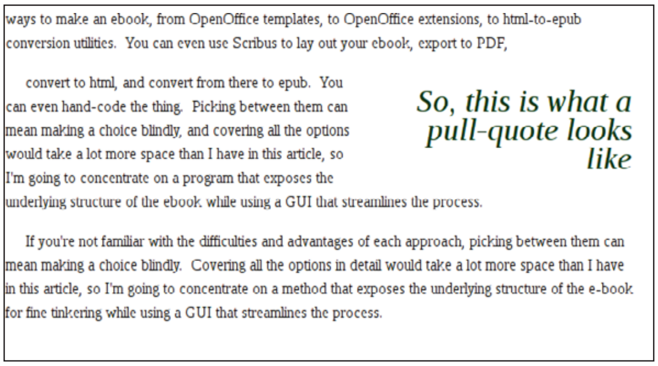
Once applied, you can use a <span class="dropcap"> tag to implement your drop cap wherever you see fit (Figure 3).

Finally, there's the thorny issue of inline images. Like HTML,


So, You Want to E-Publish?

With Sony E-readers, Kindles, iPads, Nooks, Kobos, smartphones, and the new Nintendo handhelds all giving you the chance to read on-the-go without having to cart around a stack of paperbacks, the e-book market share is growing like an Alien in a [find a geek reference]. All the major stores allow independents to list with them, and the result is, unsurprisingly, a lot of books that look like they weren't typset at all.

Figure 3. Drop Cap CSS in Action


ways to make an ebook, from OpenOffice templates, to OpenOffice extensions, to html-to-epub conversion utilities. You can even use Scribus to lay out your ebook, export to PDF,

convert to html, and convert from there to epub. You can even hand-code the thing. Picking between them can mean making a choice blindly, and covering all the options would take a lot more space than I have in this article, so I'm going to concentrate on a program that exposes the underlying structure of the ebook while using a GUI that streamlines the process.

So, this is what a pull-quote looks like

If you're not familiar with the difficulties and advantages of each approach, picking between them can mean making a choice blindly. Covering all the options in detail would take a lot more space than I have in this article, so I'm going to concentrate on a method that exposes the underlying structure of the e-book for fine tinkering while using a GUI that streamlines the process.

Figure 4. Pull-Quotes

EPUB is a flowing-text format rather than a fixed-text format. And, like HTML, it's designed that way to accommodate a variety of screen sizes, so that the customer doesn't get tied to a single device. Anchoring images, thus, can become a bit of a problem if you use the wrong method.

The <img> tag tells the interpreter where to place your graphic depending on the align option—if you don't tell it, it just drops the image in as an inline element as if it were another character. Normally on a Web site, you can use top, bottom, left, right and center as your alignment options. You still can use those here, although top, bottom and center tend to look pretty ugly when used in an e-book, so I highly recommend using only `align="left"` and `align="right"` if you want the project to look good.

For reference, the EPUB standard supports PNG, JPG, GIF and SVG. Because it's a vector format, SVG is the only one of these that scales well, so I recommend using it for diagrams whenever possible. When sizing PNG, JPG and GIF, bear in mind that most e-book readers are comparatively low resolution, so you can use them at slightly more than Web-appropriate sizes without too much worry.

Finally, there are pull-quotes (Figure 4). To pull these off, you'll need a little CSS and a little HTML. The CSS defines the pull-quote class, and it goes something like this:

```
.pquote {
     float: left;
     width: 8em;
     background: #ddf;
     font-weight: bold;
     padding: 1em;
     margin: 0 0.5em 0.5em 0;
}
```

Then, to use the pull-quote inline, use this HTML:

```
<blockquote class="pquote"> pull quote text </blockquote>
```

Tables, the last thing in on the layout checklist, are the easiest to manage—simply put them into the code screen the way you would any HTML tables, and they will render beautifully inline.

STEP 4: Creating the Structure

The other major defining characteristic of a professional-feeling e-book is the internal structure. One of the major tools authors use to control the timing and flow of their prose is structure—page breaks, chapter breaks and so on. In e-books, which are generally a flowing-text format, you use internal data structure to reproduce similar effects. A clickable table of contents, hard breaks between the chapters and end notes where applicable are the bare minimum.

The complicated procedures one must go through to create this structure when exporting from OpenOffice.org via its available extensions (and this experience is typical of most free authoring solutions) is the reason that the commercial products on the market tend to sell for upward of $100–$200. This is where Sigil shines. It makes short work of this otherwise most annoying part of the process.

The first thing you'll want to appear in your e-book is the artwork. Move your cursor to the beginning of your text, and from the Insert drop-down menu, select Image. For optimal results across readers, you want cover art that is 590 wide x 750 high, 72ppi or higher resolution, saved in the RGB colorspace. JPG, GIF and PNG are all supported in the standard for cover art.

Next, to tag the art as your cover art, you need to go to the left sidebar, find the cover art image in the Images folder for the project and right-click on it. A menu will pop up. From the menu, select Add Semantics and click on the submenu option Cover art. This will set the tags in the book properly. Now, back in the main editing pane, position your cursor directly after the art, and press Ctrl-Enter. This creates a chapter break so that the image will display as a single-pane page on reader devices.

With the cover art done, it's time to create the rest of the internal structure. At every chapter break (or anywhere you want a hard break in the book), create another chapter in the e-book with Ctrl-Enter. You'll notice that, as happened with the artwork, each Ctrl-Enter creates a new tab, numbered serially.

Once you have broken all your chapters, it's time to create a table of contents. Start with your chapter headings (for example, "Chapter 1"). Highlight the text for your heading, then, in the Select Heading list box, select Heading 1. Do this for every chapter heading. For more complex documents, use Heading 2 and onward down the hierarchy—this will create an outline when you autogenerate the table of contents.

To autogenerate, go to the Tools menu, and select TOC Editor. Here you can decide which chapter breaks to include in the clickable table of contents, and you can edit the chapter titles as they will appear in that table of contents. Once you're satisfied with the way it looks, the last thing you need to do is create the metadata to display on your e-book reader's index.

Under Tools, select Meta Editor. In the window that pops up, fill in the title, author and language. Use the Add Basic and Add Advanced buttons to add other relevant fields, such as the ISBN, publication dates, genres and so forth. And, believe it or not, you're done.

This is where Sigil shines. It makes short work of this otherwise most annoying part of the process.

Wrapping It Up

Because of the way e-book authoring works, defining the style that suits your particular book is far easier than it looks at first glance. A passing familiarity with CSS and HTML should be enough to allow you to create eye-catching layouts with transitional artwork and all the trimmings. Several enterprising folks, such as the group at EPUB Zen Garden (**epubzengarden.com/contribute**) have released professional book templates for noncommercial use, and a number of others are selling good layout templates. Of course, with a good eye, a bit of patience and the tools described in this article, you can give your e-books their own unique look for only the cost of your own labor and expertise.■

Dan Sawyer is the founder of ArtisticWhispers Productions, a small audio/video studio in the San Francisco Bay Area where he produces full-cast audiobooks and radio dramas. He is the author of *The Clarke Lantham Mysteries*, the Parsec-nominated *The Antithesis Progression*, *Down From Ten* and several short stories. You can find out more about him and his various flavors of insanity at www.jdsawyer.net.

TS-WIFIBOX-2

A Complete Solution for 802.11g WiFi Applications

qty 1 $185

Powered by a
250 MHz ARM9 CPU

- Low power (3.2 watts), fanless
- Power via 5-12 VDC, USB, PoE (opt.)
- 64MB DDR-RAM
- 256MB ultra-reliable XNAND drive
- Micro-SD Card slot
- RS-232, RS-485, CAN, RTC
- Header with SPI and 11 DIO
- 480Mbit/s USB, Ethernet, PoE option
- Boots Linux 2.6.24 in < 3 seconds
- Un-brickable, boots from SD or flash
- Customizable FPGA - 5K LUT
- Optional DIN mountable enclosure

Ideal for gateway or firewall, protocol converter, web server, WiFi audio, and unattended remote applications

- Over 25 years in business
- Never discontinued a product
- Engineers on Tech Support
- Open Source Vision
- Custom configurations and designs w/ excellent pricing and turn-around time
- Most products ship next day

Technologic SYSTEMS

We use our stuff.

visit our TS-7800 powered website at

www.embeddedARM.com

(480) 837-5200

Watch Your Processes Remotely with Mojolicious and a Smartphone

How to create an app to monitor processes in real time.

JAMIE POPKIN

When was the last time you wanted to know the status of a command, script or process when away from the computer? Wouldn't it be great to free yourself from the office chair even though you are supposed to monitor something? The key to making this happen lies in modern Web technology.

A Web application is just another term for an interactive Web page. Mojolicious provides an excellent platform to launch a Web application. Many smartphones are equipped with Web browsers capable of consuming Web applications. Linux provides the best medium to bring both together.

Mojolicious is a shiny new Perl Web framework adapted from the technology previously known as Catalyst. Mojolicious is an HTML5-compliant tool that contains all the bells and whistles you would expect in a modern Web framework. I love Perl, so naturally I was on it like a nerd on *Star Trek* paraphernalia (which I also probably would be on).

I won't go too far into the advantages of Mojolicious and Perl, but I will mention that Perl has been known as the "duct tape of the Internet". It is hard to go wrong with the tremendous amount of modules, free code and support available.

I wanted something very basic to start this article. Luckily, Mojolicious comes with template tools and standalone Web servers. This allows for a very quick startup—no mucking around with Apache or anything like that, until you are ready to scale up, that is.

Setup

First, you need to have Perl installed on your system. It should be available in most repositories if it isn't installed by default.

Installing Mojolicious is as easy as typing the following:

```
curl -L cpanmin.us | perl - Mojolicious
```

This needs to be done as root. On Debian-based systems, like Ubuntu, you would type it as follows:

```
sudo -s 'curl -L cpanmin.us | perl - Mojolicious'
```

That's all you need. Let's get started.

Up and Running

Some nice template tools are available in Mojolicious. For the sake of this article, however, let's leave that for later. First, let's create the simplest Mojolicious application possible by typing the following into a file called gobble-first.pl:

```
#!/usr/bin/perl
use Mojolicious::Lite;

get '/' => {text => 'This is just the beginning!'};
app->start;
```

The first two lines source the Perl executable and Mojolicious module. The third line prints a simple message when a browser hits the root URL. The last line tells Mojolicious to start running.

Make the file executable by typing:

```
chmod 744 gobble-first.pl
```

Now, execute the file with:

```
./gobble-first daemon
```

The `daemon` option tells Mojolicious to start the lightweight HTTP server that comes packaged with the framework.

The command output that follows tells you Mojolicious is running on port 3000. Open a Web browser and point it to http://localhost:3000. *Voilà!* You now have a Web app running with four lines of code.

Having a Little Fun First

Now, let's create a simple app that exhibits the power of Mojo. You can use the built-in template tools to get started with as little typing as possible. Enter the following on the command line:

```
mojo generate lite_app gobble-light.pl
```

This creates the file gobble-light.pl, containing the following:

```
#!/usr/bin/env perl

use Mojolicious::Lite;

get '/welcome' => sub {
    my $self = shift;
    $self->render('index');
};
```

```
app->start;
__DATA__

@@ index.html.ep
% layout 'default';
% title 'Welcome';
Welcome to Mojolicious!

@@ layouts/default.html.ep
<!doctype html><html>
    <head>
        <title><%= title %></title>
        <%= base_tag %>
    </head>
    <body><%= content %></body>
</html>
```

There is some new stuff here. First is a `get` statement just before the `app->start;` line. It instructs the server to accept requests at the http://server-ip:3000/welcome URL. It also directs the users to a document called index. The index happens to be within the same file under the `__DATA__` string. You can think of this section as containing embedded files. In this case, there is an index.html file and layout file called default.html. The .ep after each filename is just a Perl naming convention for templates.

The important thing to note is you can insert Perl code into your templates and layout with special strings like this:

```
% perl code
<% perl code %>
<%= perl code %>
```

If you were to execute this file and point your browser to http://localhost/welcome, you would see the "Welcome to Mojolicious!" message.

This seems a little boring. Let's spice it up with a couple changes.

First, change the line:

```
get '/welcome' => sub {
```

to:

```
get '/(.me)' => sub {
```

Then, add the following embedded Perl variable just after the "Welcome to Mojolicious" text in the index.html.ep template:

```
<%= $me %>
```

The full index template, just below the `__DATA__` line, now should look like this:

```
@@ index.html.ep
% layout 'default';
% title 'Welcome';
Welcome to Mojolicious <%= $me %>!
```

Run the new app the same way as before:

```
./gobble-light.pl daemon
```

This time, point the browser on a phone to your Linux box. Add any text after the path:

```
http://server-ip:3000/jamie
http://server-ip:3000/foobar
http://server-ip:3000/blah
```

The brand-new dynamic page adjusts according to the URL.

Why is this important? You're about to use this functionality in a very useful way.

It is possible to have a full Web application in just one Perl script. I love this aspect of Mojolicious. I will keep expanding on the same original file for the remainder of this article.

Some Mojo for Monitoring Applications

For sending command output to the Web application, it is necessary to redirect all output to your Web service. This would be easy to do on its own. However, it makes most sense to see the output both on the originating terminal and your phone. The super-handy `screen` command is perfect for this.

Screen is a session management tool. The most useful feature, in regard to this application, is logging. A command can be run inside screen. All output to the terminal seems normal. Yet, you can send the output to a log file that can be accessed simultaneously by another application. The other application in this example will be a second thread of the script.

Screen writes to the log file every ten seconds by default. I changed this to every second. You can do that by adding the following line to the .screenrc file in your home directory:

```
logfile flush 1
```

I also added a line to customize

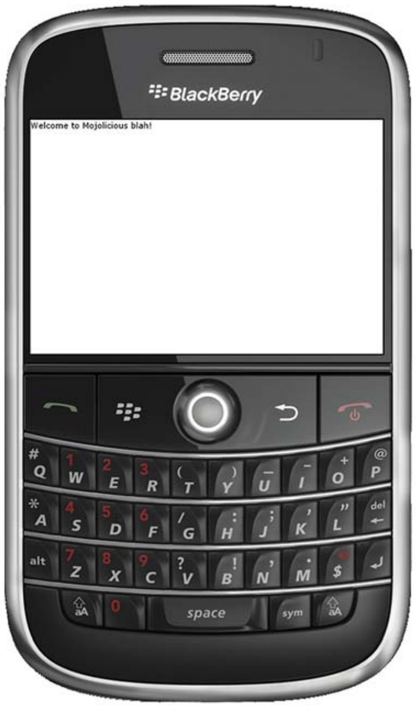

Welcome to Mojolicious blah!

Figure 1. Pointing a BlackBerry Bold to My Server at http://192.168.1.106:3000/blah

the name of the log file:

```
logfile html_gobble.log
```

There also should be a control file in /etc/ if you don't or choose not to have one in your home directory.

Listing 1 shows all the code required to send command output to Mojolicious and display it as a Web application.

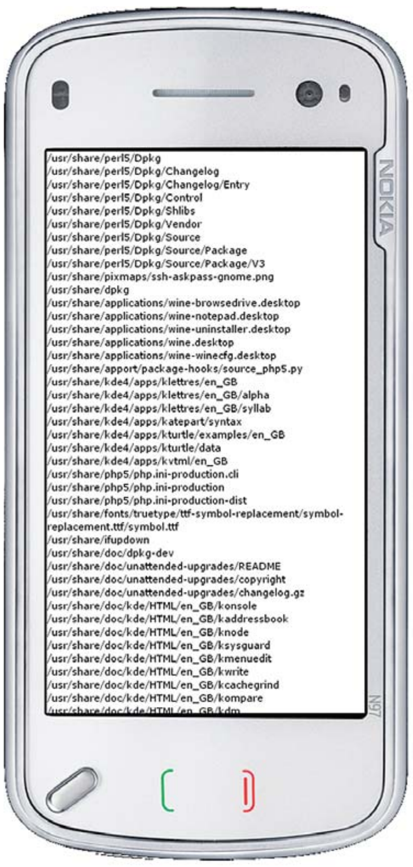
Executing the program shown in Listing 1 would look something like this:

```
./gobble-simple.pl here-i-am "sudo find / -mtime -66 -mtime +59"
```

Here I am searching the entire filesystem for files that were modified between 59 and 66 days ago. Notice that the `daemon` command-line option is no longer required, because it was added to the `app->start()` call. The output will go to: http://myserver-ip/here-i-am.

Listing 1. Code to Send Command Output to Mojolicious and Display It as a Web App

```
#!/usr/bin/env perl

use Mojolicious::Lite;

# Grab the command line variables
# by accessing the @ARGV array
my $my_url = $ARGV[0]; # The first is the url parameter
my $my_command = $ARGV[1]; # The second is the command

# We need two threads: one for screen and the other
# for Mojolicious and our web page.
my $childpid = fork();

if ($childpid) { # If this is the child thread

   # Send the command to the shell as a parameter of screen
   `screen -L $my_command`;

   # I want mojo to continue running after the child
   # is done. So just give a little reminder to kill it.
   print "Done...\n";
   print "Don't forget to kill the process 'kill -9 $childpid'\n";

} else { # If not, this is the parent thread

   get $my_url => sub { # Set the url to $my_url
       my $self = shift; # Grab the current instance

       # Open the log file and store it in <FILE>
       open(FILE,"html_gobble.log");

       # Create an empty array that will hold all our content
       my @new_file = ();

       # In web land, you need a <br> tag to end a line
       # or encase the line in <div> tags. Let's do the latter.
       # There are tag generating tools in Mojolicious, but let's
       # do it the old-fashioned way.
       #
       # Use perl's "magic open" to open and run through
       # each line of the file.
       while (<FILE>) {
           # The $_ variable holds the current line
           chomp $_; # Remove the newline character
           $_ =~ s/^/<div>/; # add <div> to the beginning
           $_ =~ s/$/<\/div>/; # add </div> to the end
           push(@new_file, $_); # add the line to our array
       }

       # Flatten the array into one long string with no spaces or
       # newline characters.
       my $one_string = join('',@new_file);

       # Tell Mojo to render the index template below
       # and pass our data "$one_string" as a new
       # variable called "$log_data"
       $self->render('index', log_data => $one_string);
   };

   # Start Mojolicious with the lightweight web server
   app->start('daemon');
}

__DATA__

@@ index.html.ep
% layout 'default';
% title 'Test';
<%== $log_data %>

@@ layouts/default.html.ep
 <!doctype html><html>
  <head>
   <title><%= title %></title>

   %# This tells phone browsers not to scale out
   %# when first loaded
   <meta name="viewport" content="initial-scale=1.0"/>

   <%= base_tag %>
  </head>
  <body>
   <%= content %>
  </body>
 </html>
```



```
/usr/share/perl5/Dpkg
/usr/share/perl5/Dpkg/Changelog
/usr/share/perl5/Dpkg/Changelog/Entry
/usr/share/perl5/Dpkg/Control
/usr/share/perl5/Dpkg/Shlibs
/usr/share/perl5/Dpkg/Vendor
/usr/share/perl5/Dpkg/Source
/usr/share/perl5/Dpkg/Source/Package
/usr/share/perl5/Dpkg/Source/Package/V3
/usr/share/pixmaps/ssh-askpass-gnome.png
/usr/share/dpkg
/usr/share/applications/wine-browsedrive.desktop
/usr/share/applications/wine-notepad.desktop
/usr/share/applications/wine-uninstaller.desktop
/usr/share/applications/wine.desktop
/usr/share/applications/wine-winecfg.desktop
/usr/share/apport/package-hooks/source_php5.py
/usr/share/kde4/apps/klettres/en_GB
/usr/share/kde4/apps/klettres/en_GB/alpha
/usr/share/kde4/apps/klettres/en_GB/syllab
/usr/share/kde4/apps/katepart/syntax
/usr/share/kde4/apps/kturtle/examples/en_GB
/usr/share/kde4/apps/kturtle/data
/usr/share/kde4/apps/kvtml/en_GB
/usr/share/php5/php.ini-production.cli
/usr/share/php5/php.ini-production
/usr/share/php5/php.ini-production-dist
/usr/share/fonts/truetype/ttf-symbol-replacement/symbol-
replacement.ttf/symbol.ttf
/usr/share/ifupdown
/usr/share/doc/dpkg-dev
/usr/share/doc/unattended-upgrades/README
/usr/share/doc/unattended-upgrades/copyright
/usr/share/doc/unattended-upgrades/changelog.gz
/usr/share/doc/kde/HTML/en_GB/konsole
/usr/share/doc/kde/HTML/en_GB/kaddressbook
/usr/share/doc/kde/HTML/en_GB/knode
/usr/share/doc/kde/HTML/en_GB/ksysguard
/usr/share/doc/kde/HTML/en_GB/kmenuedit
/usr/share/doc/kde/HTML/en_GB/kwrite
/usr/share/doc/kde/HTML/en_GB/kcachegrind
/usr/share/doc/kde/HTML/en_GB/kompare
/usr/share/doc/kde/HTML/en_GB/kdm
```

Figure 2. Output of a find Command on the Nokia N97

Here is what is happening. The script begins by accepting two arguments. The first is the URL to which the command will output. The second is the actual command, which should be wrapped in parentheses.

You need two threads, one for screen at your terminal and the other for Mojolicious and your Web app. When the command finishes, the first thread is essentially done. I could kill the Web app at this point; however, I like to browse the output afterward. So instead, I print a little message to the terminal so I remember to kill the process when I return.

The second thread opens the log file and formats each line into HTML. Then, it stuffs all lines into a variable, which is passed to the template below.

There are a couple things I don't like about this setup. First, you have to press the refresh button on your phone's browser each time you want to update the page. Second, each time the page refreshes, the window starts at the top of the page. This isn't ideal if you have lots of output to scroll through.

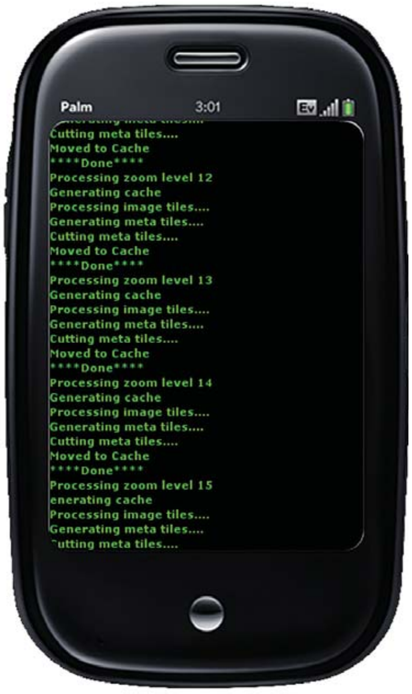
Adding JavaScript

To get some nice behavior from this little app, you need to add some functionality on the client side. All updates here will go into the template section as JavaScript. The following was added inside the <head> tag, just below the title:

```
<style type="text/css">
  body {
    background-color: #000000;
    color: #00ff00;
    font-weight: bolder;
  }
</style>

<script type="text/javascript" src="https://ajax.googleapis.com/
➥ajax/libs/jquery/1.5.1/jquery.min.js"></script>

<script type="text/javascript">
  $(document).ready(function () {
    %# Scroll down to the bottom
    $('html, body').animate({scrollTop: $(document).height()},
    ➥'slow');

    %# Schedule the first update in five seconds
    setTimeout("updatePage()",5000);
  });

  %# This function will update the page
  function updatePage () {

    $('#command-content').load(window.location.href +
    ➥' #command-content>div');
    $('html, body').animate({scrollTop: $(document).height()},
    ➥'slow');

    %# Schedule the next update in five seconds
    setTimeout("updatePage()",5000);
  }

</script>
```

JavaScript is a language available in just about any browser. It is the essence of Internet client-side scripting. I chose to use a fairly popular helper library called jQuery to make things easier. The first script tag imports this from Google's handy repository.

The second script tag contains the behavior code. jQuery is utilized to scroll to the bottom of the page in a smooth, animated fashion. Next, I call a function `updatePage` to be executed in 5,000 milliseconds (five seconds). This function first performs an AJAX call on the Web application. It then replaces the current view with everything it found. If there is no new content, the page would seem unchanged. If there is new content, the



```
Cutting meta tiles....
Moved to Cache
****Done****
Processing zoom level 12
Generating cache
Processing image tiles....
Generating meta tiles....
Cutting meta tiles....
Moved to Cache
****Done****
Processing zoom level 13
Generating cache
Processing image tiles....
Generating meta tiles....
Cutting meta tiles....
Moved to Cache
****Done****
Processing zoom level 14
Generating cache
Processing image tiles....
Generating meta tiles....
Cutting meta tiles....
Moved to Cache
****Done****
Processing zoom level 15
enerating cache
Processing image tiles....
Generating meta tiles....
utting meta tiles....
```

Figure 3. I often create large image caches for use with on-line maps. Here is the output on a Palm Pre getting updated automatically with AJAX.

The app is so efficient now that I increased the update cycle to every second.

page gets another nice animated scroll from the current position to the bottom. Then, the same function is called in five minutes. The browser keeps executing until it is closed or redirected.

I felt it also was prudent to improve how the interface looked. Just above the JavaScript code is some CSS styling rules that give things some flavor—lime green on black is much easier to look at.

Going Modern with WebSockets

At this point, the tool is pretty much behaving how I want it. I could, however, make a couple major improvements. You may have noticed that this is using a lot of unnecessary bandwidth. Every five seconds, it grabs the entire contents of the page. This can really add up—especially if you happen to forget the application running away in your pocket. My short-term memory isn't that good, and neither is my cell-phone plan.

The second flaw is the phone is doing most of the work. It really should be the other way around. The server should be keeping track of time and sending output without being requested. In fact, why should the entire document be sent if the phone needs only one or two lines?

Luckily, the modern Web has provided the fantastic new technology called WebSockets. There is an initial handshake between the client and server during connection, and then the connection stays open until one of the two decides to close. This allows the phone to sit there, listen and preserve battery power. New lines of content are sent when available. No traffic is sent when the process completes. Therefore, you don't need to worry about forgetting things in your pocket.

Start implementing this by setting up the WebSocket server. Add a Mojo looping object at the start of the script:

```
my $loop = Mojo::IOLoop->singleton;
```

Then, add the following right before the `app->start('daemon')` string:

```
# Set the socket to be the same url...
# but with a "-ws" at the end.
websocket $my_url . '-ws' => sub {
  my $self = shift;

  my $send_data;
  $send_data = sub {
    # Grab new lines from our function
    my $new_lines = updatePage();

    # If new lines are available, $new_lines will exist.
    if ($new_lines) {
      # Send content to the client
      $self->send_message($new_lines);

      # Do this all again in 1 second
      $loop->timer(1, $send_data);
    }
  };
  # We need this to start our loop for the first time
  $send_data->();
};
```

The app is so efficient now that I increased the update cycle to every second. This is absolutely fantastic to witness in person. You really get the feeling of monitoring output in real time.

The above code opens a WebSocket on the server. Next, an infinite loop is started that sends output from the function called `updatePage()`. The function looks for new lines in the log file then formats them for HTML. Here's what it looks like:

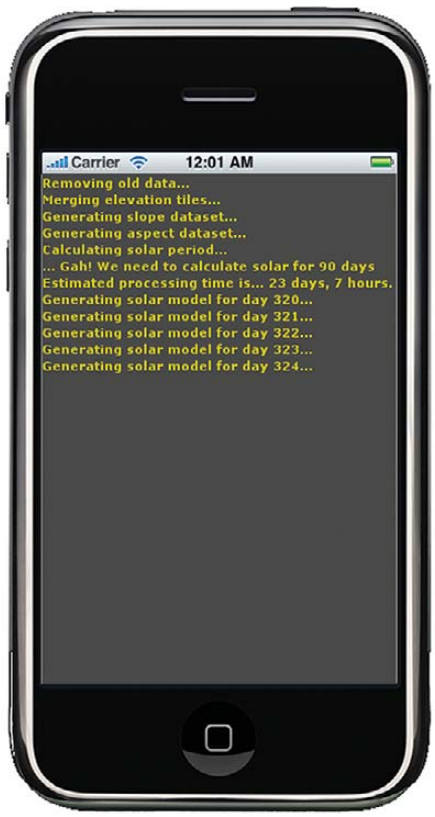
```
sub updatePage () {
  open(FILE, "html_gobble.log");

  my $iteration_count = 0; # zero the counter
  my $new_content = ''; # Initialize the new content variable

  while (<FILE>) {
    ++$iteration_count; # Increment
    # If there is a new line(s)
    if ($iteration_count > $line_count) {
      # We need to keep track of lines already added.
      ++$line_count;

      # Here we are adding the current line to the new
      # content variable with our html markup around it.
      $new_content = $new_content . "<div>" . $_ . "</div>";
    }
  }

  # Close the file handle.
  close (FILE);

  # Return new content
  return ($new_content);
}
```

On the client side, you no longer need the AJAX and looping logic. You do need WebSocket connection code though. The contents of the second <script> tag from the previous example is replaced with this:

```
<script type="text/javascript">
  $(document).ready(function () {
    %# Grab our current location
    var ws_host = window.location.href;

    %# We are requesting websocket data...
    %# So change the http: part to ws:
```

```
    ws_host = ws_host.replace(/http:/,"ws:") + "-ws";
    %# I also tacked on the "-ws" at the end

    %# Connect the remote socket
    var socket = new WebSocket(ws_host);

    %# When we receive data from the websocket do the following
    %# with "msg" as the content.
    socket.onmessage = function (msg) {
      %# Append the new content to the end of our page
      $('#command-content').append(msg.data);

      %# Scroll down to the bottom
      $('html, body').animate({scrollTop:
      ➥$(document).height()}, 'slow');
    }

    %# Scroll down to the bottom
    $('html, body').animate({scrollTop:
    ➥$(document).height()}, 'slow');
</script>
```

The WebSocket URL is formed when the document initially is loaded. A connection is made using the variable `socket`. New content is appended to the document when it is received, followed by an animated scroll.



```
Removing old data...
Merging elevation tiles...
Generating slope dataset...
Generating aspect dataset...
Calculating solar period...
... Gah! We need to calculate solar for 90 days
Estimated processing time is... 23 days, 7 hours.
Generating solar model for day 320...
Generating solar model for day 321...
Generating solar model for day 322...
Generating solar model for day 323...
Generating solar model for day 324...
```

Figure 4. Watching a fairly lengthy process on the iPhone, and the page seamlessly communicates with the server through a WebSocket.

Into the Wild

Only one thing is holding us back from sitting at a coffee shop with our phones and calling it work—the firewall. Most processes I run are fairly safe when it comes to output. I chose to follow a security-by-obscurity philosophy. When I have a process running, other people can view the output, assuming they know the port number and URL. When the process ends, the port is closed and there is nothing to see. The nice thing about this approach is I can share the URL with clients if the need arises.

Have Fun

There is no reason to have a specialty app on your phone to view output from your computer. With the help of a modern Web framework like Mojolicious, any smartphone can consume command-line output right in the Web browser. I hope this article inspires people to come up with their own unique approaches to this concept.■

Jamie Popkin lives in Lantzville, British Columbia, with his wife and four kids. He is a consultant specializing in geographic data portrayal on the Web. Recently, he has started developing for smartphones, utilizing modern Web/HTML5 technology. He can be reached via Twitter (@jamiepopkin) or e-mail (popkinj@littleearth.ca).

NOTE:
All code for this article is at **https://github.com/popkinj/gobble**.

NVIDIA® ION2 System
Features GeForce® Graphics, flexible storage options and Wi-Fi.

Assembled & tested Ubuntu Linux compatible systems...

Genuine expertise. LOGIC SUPPLY

www.logicsupply.com/linux

Kinect with Linux

Control Linux with a wave of your hand! Hook up a Kinect to your PC and help invent the user interface of the future. RICK ROGERS

The Minority Report has been in rotation on cable lately, and you've probably seen the futuristic vision of Tom Cruise standing in front of a large screen, manipulating information with waves of his hands. That vision is a bit closer to reality, thanks in part to the economies of scale of the game industry. I don't often have reason to sing the praises of Microsoft, particularly not in a magazine devoted to Linux and all things open. But one thing our friends in Redmond do very well is to commoditize hardware. They've done just that with the Kinect by creating it as a natural interface for the Xbox 360 game console. What's more, they've allowed open-source developers to create drivers for the device, and they've even allowed the third party who developed the technology, PrimeSense, Inc., to release its own device drivers for Linux, Windows and OS X.

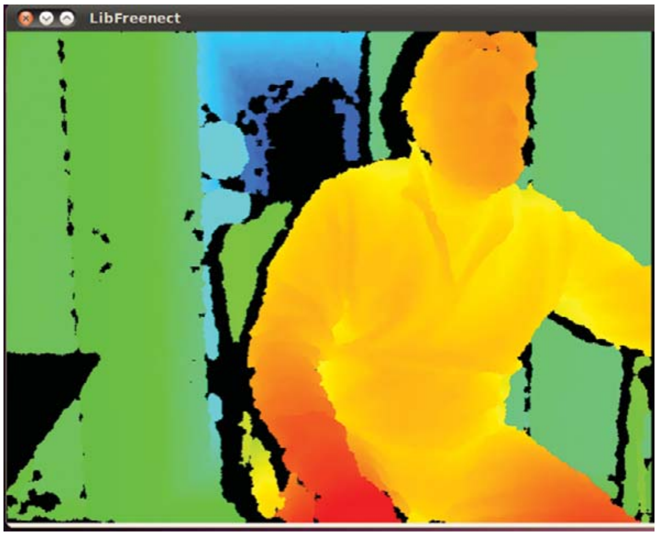


Figure 1. The Author, as Seen in Kinect's Depth View

Natural Interfaces

Computer interfaces that use "natural" interaction have been termed, appropriately, natural user interfaces, or NUI. The implication is not so much that the interface itself is natural (let's face it, there is no natural, unlearned way to interact with a machine), but that the interface is very easy to learn, and in that sense natural. Natural user interfaces are a very active research topic, and a wide variety of prototypes have been demonstrated. The list of projects exploring natural interfaces and Kinect is ever-expanding. Here are some that have been ventured on the Net:

- Robotic vision systems.
- Interior room mapping.
- Light control with gestures.
- Sign-language computer interfaces.
- Standardized gesture interfaces.
- Music with gestures (including the floor piano from *Toys*).
- Virtual keyboards.
- Instruction (such as dance, karate and tai chi).
- Gesture-based presentations.

The Kinect—Sensor Heaven

The Kinect is tightly packed with an array of sensors and specialized devices to preprocess the information received. Communication between the Kinect and the game console or Linux is through a single USB cable.

Several different estimates have been made of the cost to manufacture Kinect, ranging from about $56 to $150. Whatever the real cost, we have to hand it to Microsoft for reducing the cost of its original prototype (reportedly $30,000) by at least two orders of magnitude to create a viable commercial product.

You can view a complete teardown of the Kinect on YouTube (see Resources), and inside you will find the following:

- A projector that projects a field of infrared beams, used to detect depth.
- Two cameras (each 640x480): a monochrome infrared camera that is used to detect the array of reflected infrared beams (this is the information used to construct the image of depth) and a color camera that can be used to capture snapshots or as a Webcam.
- Four microphones.
- A motor to tilt the sensor array up and down for best view of the scene.
- An accelerometer to sense position.
- Camera interfaces and preprocessors for the two cameras.
- 512MB of DDR memory and 8MB of Flash.
- USB interfaces.

In just a few days, the Open Source community had seized the opportunity provided by Microsoft and created an open-source alternative that created an explosion of possibilities for research into natural interfaces.

The infrared beams are encoded, and as they are reflected off surfaces in front of the sensor array and detected by the infrared camera, a preprocessor in the Kinect calculates the distance from the array to the reflecting surfaces. Proprietary software can use that information to identify likely humans in the scene and the likely positions of their arms and legs.

Kinect Open Source

The short history of the Kinect and open source is interesting, instructive, and it demonstrates the awesome power of open source:

- On November 4, 2010, Microsoft released the Kinect in the United States.
- That same day, Adafruit Industries announced it would pay anyone $1,000 for an open-source driver for the device on Windows or any other operating system.
- Hours later, Microsoft announced that it would not condone hacking of any of its devices. Adafruit responded by raising the bounty to $2,000.
- By Saturday, November 6, a hacker going by the name AlexP claimed to have hacked the interface to the Kinect's motor. Microsoft said it wasn't true. Adafruit raised the reward to $3,000.
- By that Monday, AlexP had posted video proving his ability not only to control the motor, but also to interface to the depth perception and color camera on the Kinect. He could have chosen to release the code and collect the prize, but instead, that Tuesday, he posted a message saying he would release the code if $10,000 were contributed to fund his further work.
- Tuesday evening, Adafruit released a large dataset recorded by a USB analyzer watching the data stream between the Kinect and Xbox. Hackers worldwide started using the dataset to work out the details of the interface.
- On Wednesday, the Kinect was released in Europe. Hector Martin, near Bilbao, Spain, purchased one that morning, and using the Adafruit published data, was able to get it connected to his PC by noon. The results were published on the

Advertiser Index

CHECK OUT OUR BUYER'S GUIDE ON-LINE.
Go to www.linuxjournal.com/buyersguide where you can learn more about our advertisers or link directly to their Web sites.

Thank you as always for supporting our advertisers by buying their products!

Advertiser	URL	Page #
1&1 Internet Inc.	www.oneandone.com	20, 21, 23
Aberdeen, LLC	www.aberdeeninc.com	C3
Emac, Inc.	www.emacinc.com	53
EmperorLinux	www.emperorlinux.com	37
Genstor Systems, Inc.	www.genstor.com	31
Giada Technology, Inc.	www.giadatech.com	79
iXsystems, Inc.	www.ixsystems.com	C2, 3
Kiwi PyCon	nz.pycon.org	9
Linode, LLC	www.linode.com	47
Linux Foundation	www.linuxfoundation.org	7
Linux Journal Store	www.linuxjournalstore.com	71
Logic Supply, Inc.	www.logicsupply.com	39, 63
Microway, Inc.	www.microway.com	C4, 5
Mikro Tik	www.routerboard.com	1
MonoSpace	monospace.us	49
OEM Production	www.oemproduction.com	29
O'Reilly OSCON	oscon.com	77
Polywell Computers, Inc.	www.polywell.com	79
RackMountPro	www.rackmountpro.com	27
Silicon Mechanics	www.siliconmechanics.com	16, 17, 43
Technologic Systems	www.embeddedx86.com	57
USENIX Security Symposium	www.usenix.org/sec11/lj	19

ATTENTION ADVERTISERS

October 2011 Issue #210 Deadlines
Space Close: July 25; **Material Close:** August 2

Theme: Networking

BONUS DISTRIBUTIONS:
Atlanta LinuxFest, Windycity Rails, LinuxCon, Ohio LinuxFest, High Performance on Wall St, VMWorld

Contact **Joseph Krack, +1-713-344-1956 ext. 118, joseph@linuxjournal.com** libfreenect site, and the prize was won.

- But the story doesn't stop there. By Friday of the same week, Microsoft reconsidered its position on the open-source drivers. In a remarkable bit of semantic derring-do, Microsoft said the Kinect had not been hacked by its definition, and actually praised the developers expanding the use of the device.

In just a few days, the Open Source community had seized the opportunity provided by Microsoft and created an open-source alternative that created an explosion of possibilities for research into natural interfaces.

OpenKinect and OpenNI

The Adafruit contest had been hosted on a GitHub site called libfreenect. Once the contest had been won, Josh Blake and some others founded a community called OpenKinect. From their Web site (see Resources):

> OpenKinect is an open community of people interested in making use of the amazing Xbox Kinect hardware with our PCs and other devices. We are working on free, open-source libraries that will enable the Kinect to be used with Windows, Linux and Mac.
>
> The OpenKinect community consists of over 2,000 members contributing their time and code to the Project. Our members have joined this Project with the mission of creating the best possible suite of applications for the Kinect. OpenKinect is a true "open source" community!
>
> Our primary focus is currently the libfreenect software. Code contributed to OpenKinect where possible is made available under an Apache20[sic] or optional GPL2 license.

Around the same time, a number of companies with interests in commercializing natural interfaces formed a group called OpenNI. According to their Web site (see Resources):

> The OpenNI organization is an industry-led, not-for-profit organization formed to certify and promote the compatibility and interoperability of Natural Interaction devices, applications and middleware. One of the OpenNI organization's goals is to accelerate the introduction of Natural Interaction applications into the marketplace.

OpenNI offers its software under several different licenses—LGPL for the open-source bits and just binaries for some proprietary parts (like skeletal identification). The founders of OpenNI include:

- PrimeSense, Inc., the company that supplied Microsoft with the technology for "Project Natal", which became the Kinect.
- Willow Garage, a company focused on hardware and software for personal robotics.
- SideKick, a game software company that develops motion-based games.
- ASUS, the computer OEM, who is selling a different 3-D depth sensor based on PrimeSense technology called X-tion Pro.

In general, OpenKinect appears to be approaching NUI from the bottom up, starting with the libfreenect driver and building on top of that, all strictly open source. OpenNI has more of an architectural vision for how NUI devices from different vendors can interoperate. There's plenty of room for the two organizations to work together.

Try It Yourself

The drivers and demo software for Kinect are readily available, both from OpenNI and from openkinect.org (see Resources). Installation on Ubuntu 10.10 is particularly easy, as both organizations provide prebuilt packages. If you're running a different Linux distro, RPMs and debs also are available, along with ample build instructions, so you shouldn't have a problem.

With the OpenKinect packages, you get some demo programs that show the depth perception and color image capability of the Kinect. Utilities are included to record the Kinect data stream and to emulate a Kinect so the software can be used without having the hardware connected. There also are language interfaces in various stages of development for the following:

- Python.
- Synchronous C interface (functions instead of callbacks).
- ActionScript.
- C++.
- C#.
- Java JNI.
- Java JNA.
- JavaScript.
- Common Lisp.

The OpenNI package has similar capabilities, and it includes

Utilities are included to record the Kinect data stream and to emulate a Kinect so the software can be used without having the hardware connected.

detailed documentation of the C/C++ interface to the underlying layers. The OpenNI documentation and sample code is oriented toward Visual Studio, but most of it also is applicable to gcc.

Reality Bites

Okay, wrong movie, but there are some not-so-obvious realities to using Kinect with Linux. They might or might not affect your explorations:

1. The USB connector on the Kinect device is nonstandard. That's okay if you buy the Kinect as a standalone device. It comes with a power supply and an adapter to use standard USB. If you buy the Kinect as part of an Xbox bundle, you will need to buy the power supply/adapter separately.

2. The depth perception technology in Kinect works best at distances of 6–8 feet. It's not a mouse-and-keyboard distance interface, it's a stand-across-the-room-and-wave interface.

3. The Kinect software is able to determine body position and infer the positions of arms and legs, but it isn't able to do things like individual finger resolution. That makes it difficult to do things like sign-language interpretation.

Let the Games Begin

So, now for $150 you can have open-source access to hardware that would have cost you $30,000 a few years ago. What clever ideas can you come up with for NUI?■

Rick Rogers has been a professional embedded developer for more than 30 years. Now specializing in mobile application software, when Rick isn't writing software for a living, he's writing books and magazine articles like this one. He welcomes feedback on the article at portmobileapps@gmail.com.

Resources

FemtoLinux: **femtolinux.com**

uClinux: **www.uclinux.org**

uClibc: **www.uclibc.org**

Buildroot: **buildroot.uclibc.org**

OpenEmbedded: **www.openembedded.org**

Data Deduplication with Linux

Lessfs offers a flexible solution to utilize data deduplication on affordable commodity hardware. PETROS KOUTOUPIS

In recent years, the storage industry has been busy providing some of the most advanced features to its customers, including data deduplication. Data deduplication is a unique data compression technique used to eliminate redundant data and decrease the total capacities consumed on an enabled storage volume. A volume can refer to a disk device, a partition or a grouped set of disk devices all represented as single device. During the process of deduplication, redundant data is deleted, leaving a single copy of the data to be stored on the storage volume.

One ideal use-case scenario is when multiple copies of a large e-mail message are distributed and stored on a mail server. An e-mail message the size of just a couple megabytes does not seem too bad, but if it were sent and forwarded to more than 100 recipients—that's more than 200MB of copies of the same file(s).

Another great example is in the arena of host virtualization. In recent years, virtualization has been the hottest trend in server administration. If you are deploying multiple virtual guests across a network that may share the same common operating system image, data deduplication significantly can reduce the total size of capacity consumed to a single copy and, in turn, reference the differences when and where needed.

Again, the primary focus of this technology is to identify large sections of data that can include entire files or large sections of files, which are identical, and store only one copy of it. Other benefits include reduced costs for additional capacities of storage equipment, which, in turn, can be used to increase volume sizes or protect large numbers of existing volumes (such as RAID, archivals and so on). Using less storage equipment also leads to a reduced cost in energy, space and cooling.

Two types of data deduplication exist: post-process and inline deduplication. Each has its advantages and disadvantages.

FUSE

FUSE or Filesystem in USEr Space is a kernel module commonly seen on UNIX-like operating systems, which provides the ability for users to create their own filesystems without touching kernel code. It is designed to run filesystem code in user space while the FUSE module acts as a bridge for communication to the kernel interfaces.

To summarize, post-process deduplication occurs after the data has been written to the storage volume in a separate process. While you are not losing performance in computing the necessary deduplication, multiple copies of a single file will be written multiple times, until post-process deduplication has completed, and this may become problematic if the available capacity becomes low. During inline deduplication, less storage is required, because all deduplication is handled in real time as the data is written to the storage volume, although you will notice a degradation in performance as the process attempts to identify redundant copies of the data coming in.

Storage technology manufacturers have been providing the technology as part of their proprietary and external storage solutions, but with Linux, it also is possible to use the same technology on commodity and very affordable hardware. The solutions provided by these storage technology manufacturers are in some cases available only on the physical device level (that is, the block level) and are able to work only with redundant streams of data blocks as opposed to individual files, because the logic is unable to recognize separate files over the most commonly used protocols, such as SCSI, Serial Attached SCSI (SAS), Fibre Channel, InfiniBand and even Serial ATA (SATA). This is referred to as a chunking method. The filesystem I cover here is Lessfs, a block-level-based deduplication and FUSE-enabled Linux filesystem.

NOTE:
In order to use these filesystems, it is required to install FUSE on the system. Most mainstream Linux distributions, such as Ubuntu and Fedora, most likely will have the module and userland tools already preinstalled, most likely to support the ntfs-3g filesystem.

Lessfs

Lessfs is a high-performance inline data deduplication filesystem written for Linux and is currently licensed under the GNU General Public License version 3. It also supports LZO, QuickLZ and BZip compression (among a couple others), and data encryption. At the time of this writing, the latest stable version is 1.3.3.1, which can be downloaded from the SourceForge project page: **sourceforge.net/projects/lessfs/files/lessfs**.

Before installing the lessfs package, make sure you install all known dependencies for it. Some, if not most, of these dependencies may be available in your distribution's package repositories. You will need to install a few manually though, including mhash, tokyocabinet and fuse (if not already installed).

Your distribution may have the libraries for mhash2 either available or installed, but lessfs still requires mhash. This also can be downloaded from SourceForge: **sourceforge.net/projects/mhash/files/mhash**. At the time of this writing, the latest stable build is 0.9.9.9. Download, build and install the package:

```
$ tar xvzf mhash-0.9.9.9.tar.gz
$ cd mhash-0.9.9.9/
$ ./configure
$ make
$ sudo make install
```

Lessfs also requires tokyocabinet (**1978th.net/tokyocabinet**), as it is the main database on which it relies. The latest stable build is 1-4.47. To build tokyocabinet, you need to have zlib1g-dev and libbz2-dev already installed, which usually are provided by most, if not all, mainstream Linux distributions.

Download, build and install the package using the same `configure`, `make` and `sudo make install` commands from earlier. On 32-bit systems, you need to append `--enable-off64` to the configure command. Failure to use `--enable-off64` limits the databases to a 2GB file size.

If it is not already installed or if you want to use the latest and greatest stable build of FUSE, download it from SourceForge: **sourceforge.net/projects/fuse**. At the time of this writing, the latest stable build is 2.8.5. Download, build and install the package using the same `configure`, `make` and `sudo make install` commands from earlier.

After resolving all the more obscure dependencies, you're ready to build and install the lessfs package. Download, build and install the package using the same `configure`, `make` and `sudo make install` commands from earlier.

Now you're ready to go, but before you can do anything, some preparation is needed. In the lessfs source directory, there is a subdirectory called etc/, and in it is a configuration file. Copy the configuration file to the system's /etc directory path:

```
$ sudo cp etc/lessfs.cfg /etc/
```

This file defines the location of the databases among a few other details (which I discuss later in this article, but for now let's concentrate on getting the filesystem up and running). You will need to create the directory path for the file data (default is /data/dta) and also for the metadata (default is /data/mta) for all file I/O operations sent to/from the lessfs filesystem. Create the directory paths:

```
$ sudo mkdir -p /data/{dta,mta}
```

Initialize the databases in the directory paths with the mklessfs command:

```
$ sudo mklessfs -c /etc/lessfs.cfg
```

The -c option is used to specify the path and name of the configuration file. A man page does not exist for the command, but you still can invoke the on-line menu with the -h command option.

Now that the databases have been initialized, you're ready to mount a lessfs-enabled filesystem. In the following example, let's mount it to the /mnt path:

```
$ sudo lessfs /etc/lessfs.cfg /mnt
```

When mounted, the filesystem assumes the total capacity of the filesystem to which it is being mounted. In my case, it is the filesystem on /dev/sda1:

```
$ df -t fuse.lessfs
Filesystem          1K-blocks      Used Available Use% Mounted on
lessfs                5871080   3031812   2541028  55% /mnt

$ df -t ext4
Filesystem          1K-blocks      Used Available Use% Mounted on
/dev/sda1             5871080   3031812   2541028  55% /
```

Using less storage equipment also leads to a reduced cost in energy, space and cooling.

Currently, you should see nothing but a hidden .lessfs subdirectory when listing the contents of the newly mounted lessfs volume:

```
$ ls -a /mnt/
.   ..   .lessfs
```

Once mounted, the lessfs volume can be unmounted like any other volume:

```
$ sudo umount /mnt
```

Let's put the volume to the test. Writing file data to a lessfs volume is no different from what it would be to any other filesystem. In the example below, I'm using the dd command to write approximately 100MB of all zeros to /mnt/test.dat:

```
$ sudo dd if=/dev/zero of=/mnt/test.dat bs=1M count=100
100+0 records in
100+0 records out
104857600 bytes (105 MB) copied, 5.05418 s, 20.7 MB/s
```

Seeing how the filesystem is designed to eliminate all redundant copies of data and being that a file filled with nothing but zeros qualifies as a prime example of this, you can observe that only 48KB of capacity was consumed, and that may just be nothing more than the necessary data synchronized to the databases:

```
$ df -t fuse.lessfs
Filesystem         1K-blocks      Used Available Use% Mounted on
lessfs               5871080    3031860   2540980  55% /mnt
```

If you list a detailed listing of that same file in the lessfs-enabled directory, it appears that all 100MB have been written. Utilizing its embedded logic, lessfs reconstructs all data on the fly when additional read and write operations are initiated to the file(s):

```
$ ls -l
total 102400
-rw-r--r-- 1 root root 104857600 2011-02-26 13:57 test.dat
```

Now, let's work with something a bit more complex—something containing a lot of random data. For this example, I decided to download the latest stable release candidate of the Linux kernel source from **www.kernel.org**, but before I did, I listed the total capacity consumed available on the lessfs volume as a reference point:

```
$ df -t fuse.lessfs
Filesystem         1K-blocks      Used Available Use% Mounted on
lessfs               5871080    3031896   2540944  55% /mnt

$ sudo wget http://www.kernel.org/pub/linux/kernel/v2.6/
➥testing/linux-2.6.38-rc6.tar.bz2
```

Listing the contents, you can see that the package is approximately 75MB:

```
$ ls -l linux-2.6.38-rc6.tar.bz2
-rw-r--r-- 1 root root 74783787 2011-02-21 19:50
 ➥linux-2.6.38-rc6.tar.bz2
```

Listing the capacity used to store the Linux kernel source archive yields a difference of roughly 75MB:

```
$ df -t fuse.lessfs
Filesystem         1K-blocks      Used Available Use% Mounted on
lessfs               5871080    3106440   2466400  56% /mnt
```

Now, let's create a copy of the archived kernel source:

```
$ sudo cp linux-2.6.38-rc6.tar.bz2 linux-2.6.38-rc6.tar.bz2-bak

$ ls -l linux-2.6.38-rc6.tar.bz2*
-rw-r--r-- 1 root root 74783787 2011-02-21 19:50
 ➥linux-2.6.38-rc6.tar.bz2
-rw-r--r-- 1 root root 74783787 2011-02-26 14:43
 ➥linux-2.6.38-rc6.tar.bz2-bak
```

By having a redundant copy of the same file, an additional 44KB is consumed—not nearly as much as an additional 75MB:

```
$ df -t fuse.lessfs
Filesystem         1K-blocks      Used Available Use% Mounted on
lessfs               5871080    3106484   2466356  56% /mnt
```

And, because the databases contain the actual file and metadata, if an accidental or intentional system reboot occurred, or if for whatever reason you need to unmount the filesystem, the physical data will not be lost. All you need to do is invoke the same mount command and everything is restored:

```
$ sudo umount /mnt/
$ sudo lessfs /etc/lessfs.cfg /mnt
$ ls
linux-2.6.38-rc6.tar.bz2  linux-2.6.38-rc6.tar.bz2-bak
```

In the situation when a system suffers from an accidental reboot, possibly due to power loss, as of version 1.0.4, lessfs supports transactions, which eliminates the need for an `fsck` after a crash.

Shifting focus back to lessfs preparation, note that the lessfs volume's options can be defined by the user when mounting. For instance, you can define the desired options for big_write, max_read and max_write. The big_write improves throughput when used for backup purposes, and both max_read and max_write must be defined to use it. The max_read and max_write options always must be equal to one another and define the block size for lessfs to use: 4, 8, 16, 32, 64 and 128KB.

The definition of a block size can be used to tune the filesystem. For example, a larger block size, such as 128KB (131072), offers faster performance but, unfortunately, at the cost of less deduplication (remember from earlier that lessfs uses block-level deduplication). All other options are FUSE-generic options defined in the FUSE documentation. An example of the use of supported mount options can be found in the lessfs man page:

```
$ man 1 lessfs
```

The following example is given to mount lessfs with a 128KB block size:

```
$ sudo lessfs /etc/lessfs.cfg /fuse -o negative_timeout=0,\
        entry_timeout=0,attr_timeout=0,use_ino,\
        readdir_ino, default_permissions,allow_other,big_writes,\
        max_read=131072,max_write=131072
```

Additional configurable options for the database exist in your lessfs.cfg file (the same file you copied over to the /etc directory path earlier). The block size can be defined here as well as even the method of additional data compression to use on the deduplicated data and more. Below is an excerpt of what the configuration file contains. In order to define a new value for various options clearly, just uncomment the option desired and, in turn, comment everything else:

```
BLKSIZE=131072
#BLKSIZE=65536
#BLKSIZE=32768
```

```
#BLKSIZE=16384
#BLKSIZE=4096
#COMPRESSION=none
COMPRESSION=qlz
#COMPRESSION=lzo
#COMPRESSION=bzip
#COMPRESSION=deflate
#COMPRESSION=disabled
```

This excerpt defines the default block size to 128KB and the default compression method to QuickLZ. If the defaults are not to your liking, in this file you also can define the commit to disk intervals (default is 30 seconds) or a new path for your databases, but make sure to initialize the databases before use; otherwise, you'll get an error when you try to mount the lessfs filesystem.

Summary

Now, Linux is not limited to a single data deduplication solution. There also is SDFS, a file-level deduplication filesystem that also runs on the FUSE module. SDFS is a freely available cross-platform solution (Linux and Windows) made available by the Opendedup Project. On its official Web site, the project highlights the filesystem's scalability (it can dedup a petabyte or more of data); speed, performing deduplication/reduplication at a line speed of 290MB/s and higher; support for VMware while also mentioning its usage in Xen and KVM; flexibility in storage, as deduplicated data can be stored locally, on the network across multiple nodes (NFS/CIFS and iSCSI), or in the cloud; inline and batch mode deduplication (a method of post-process deduplication); and file and folder snapshot support. The project seems to be pushing itself as an enterprise-class solution, and with features like these, Opendedup means business.

It is also not surprising that since 2008, data deduplication has been a requested feature for Btrfs, the next-generation Linux filesystem. Although that also may be in response to Sun Microsystem's (now Oracle's) development of data deduplication into its advanced ZFS filesystem. Unfortunately, at this point in time, it is unknown if and when Btrfs will introduce data deduplication support, although it already contains support for various types of data compression (such as zlib and LZO).

Currently, the lessfs2 release is under development, and it is supposed to introduce snapshot support, fast inode cloning, new databases (including hamsterdb and possibly BerkeleyDB) apart from tokyocabinet, self-healing RAID (to repair corrupted chunks) and more.

As you can see, with a little time and effort, it is relatively simple to utilize the recent trend of data deduplication to reduce the total capacity consumed on a storage volume by removing all redundant copies of data. I recommend its usage in not only server administration but even for personal use, primarily because with implementations such as lessfs, even if there isn't too much redundant data, the additional data compression will help reduce the total size of the file when it is eventually written to disk. It is also worth mentioning that the lessfs-enabled volume does not need to remain local to the host system, but it also can be exported across a network via NFS to even iSCSI and utilized by other devices within that same network, providing a more flexible solution.■

Petros Koutoupis is a full-time Linux kernel, device-driver and application developer for embedded and server platforms. He has been working in the data storage industry for more than six years and enjoys discussing the same technologies.

Resources

Official Lessfs Project Web Site: **www.lessfs.com**

Lessfs SourceForge Project: **sourceforge.net/projects/lessfs**

Opendedup (SDFS) Project: **www.opendedup.org**

Wikipedia: Data Deduplication: **en.wikipedia.org/wiki/Data_deduplication**

Notes on the Integration of Lessfs into Fedora 15: **fedoraproject.org/wiki/Features/LessFS**

Lessfs with SCST How-To: **www.lessfs.com/wordpress/?page_id=577**

release date: december 2010
LINUX JOURNAL
ARCHIVE 1994-2010
Includes issues 1–200 of Linux Journal
ALL 200 ISSUES!

Puppet

How-to: build a managed environment using Puppet. JES FRASER

Puppet is an open-source platform for systems management. It allows for rapid deployment of system configuration for hundreds or even thousands of servers. Puppet enables you to build an environment that is robust with systems that bounce back from unauthorized changes, reliable with systems that are in a known-good configuration, and above all, repeatable. Not only does Puppet allow servers to be duplicated easily, but it also makes rebuilding failed systems a snap.

Puppet manages system configuration as a collection of resources. A resource could include a user account, a configuration file or a running service, among others. Each resource to be managed is defined using Puppet's simple declarative syntax and then the definition applied to the appropriate systems. Today, I describe how to build a managed environment using Puppet with some sample modules to get you started.

Installing Puppet is very simple with packages available for most distributions. For this project, I demonstrate on Ubuntu Meerkat 10.10 Server:

```
apt-get install puppet puppetmaster
```

NOTE:

Most other distributions include packages for Puppet. For Puppet packages for enterprise Linux, such as RHEL or OEL, please see EPEL (**fedoraproject.org/wiki/EPEL**).

If all else fails, Puppet can be installed on any distribution with a recent Ruby version using RubyGems:

```
gem install puppet
```

Although Puppet is most powerfully used in a client-server configuration, it's possible to apply and test Puppet manifests without running a dæmon using the `puppet apply` command. Now that Puppet is installed, let's write a short snippet of Puppet code and apply it to your system to test the installation:

```
root@localhost # vim /tmp/test.pp

user { "test":
        ensure => present,
}

root@localhost # puppet apply /tmp/test.pp
notice: /Stage[main]//User[test]/ensure: created

root@localhost #  id test
uid=1001(test) gid=1001(test) groups=1001(test)
```

Puppet configuration is a list of resources with `keyword => value` pairs to define characteristics about these resources. Here, I've declared the resource user and named it test. In this scenario, the name of the resource becomes the name of the user. The ensure keyword is used with a value of present, which means that Puppet creates the user if it is not already present. If the user is present, Puppet does nothing.

If you examine /etc/passwd, you can see the test user has been applied to the system. You can remove this user through Puppet by changing the keyword `ensure => present` to read `ensure => absent`:

```
user { 'test':
        ensure => absent,
}

root@localhost # puppet apply /tmp/test.pp
notice: /Stage[main]//User[test]/ensure: removed
```

However, /tmp is a poor place to keep your configuration examples. By default in Ubuntu, both settings pertaining to the Puppet processes and configuration to be applied to systems reside in /etc/puppet.

Puppet files to note:

- /etc/puppet/puppet.conf: the puppet.conf file contains both server and client options. For the purposes of this project, I leave the default configuration intact for the Puppet master.

- /etc/puppet/manifests/site.pp: the site.pp file defines options general to the site. This example site.pp imports a list of nodes, defines a "filebucket" to back up the original copies of files it modifies and sets the default $PATH for executing commands on remote systems. Create the /etc/puppet/manifests/site.pp file on your Puppet master using this example:

  ```
  root@localhost # vim /etc/puppet/manifests/site.pp

  import "nodes"

  #filebucket

  filebucket { main: server => 'puppet.example.com' }

  File { backup => main }
  Exec { path => "/usr/bin:/usr/sbin:/bin:/sbin:"}
  ```

- /etc/puppet/manifests/nodes.pp: the nodes.pp file defines the nodes that will be controlled by Puppet and what configuration is applied to them.

Puppet uses modules to collect code, known as manifests, and related files into a central location. By default on Ubuntu, Puppet looks for modules in /etc/puppet/modules. Modules have a very specific structure:

```
root@localhost # cd /etc/puppet/modules
root@localhost # mkdir ntp/{manifests,files,templates}
root@localhost # touch ntp/manifests/init.pp
```

Every module has an init.pp in the manifests directory that declares the class and optionally includes other files. If the module includes files or templates, they are stored in the files and templates subdirectories. Here, let's create a simple module to manage NTP configuration on our servers:

```
root@localhost # vim /etc/puppet/modules/ntp/manifests/init.pp

class ntp {

        package { "ntp":
                ensure => installed,
        }

}
```

Here, I've defined a package resource called ntp, and the name of the resource is also the name of the package. I've used the keyword `ensure => installed`, which causes Puppet to install the package if it is not already present on the system.

That's great, but ntp won't do us much good unless it's running. Next, let's define a service resource:

```
root@localhost # vim /etc/puppet/modules/ntp/manifests/init.pp

class ntp {

        package { "ntp":
                ensure => installed,
        }
        service { "ntp":
                ensure => running,
                hasstatus => true,
                hasrestart => true,
        }

}
```

This resource defines the service ntp, ensures that it's running, and also specifies that its init script supports "status" and "restart". Puppet supports various types of UNIX, and if these options are not specified, it uses methods that are compatible with older init systems instead.

There's one last step before considering this module complete, and that's to establish a relationship between the package ntp and the service ntp. Obviously, the service can't be started until the package is installed, and Puppet does not always execute manifests in order. To ensure Puppet is aware of the order in which events need to occur, use the relationship-chaining syntax:

```
class ntp {

        package { "ntp":
                ensure => installed,
        } -> service { "ntp":
                ensure => running,
                hasstatus => true,
                hasrestart => true,
        }

}
```

Here, I've inserted `->` between the end of the package block and the start of the service block. This sets up a relationship that says the package needs to come before the service. This also could be done by using the keyword `require` to state that the service requires the package:

```
service { "ntp":
        ensure => running,
        hasstatus => true,
        hasrestart => true,
        require => Package["ntp"],
}
```

Other relationships can be defined using both the chaining syntax and keywords, such as require, subscribe and notify. For more information, see the Puppet Language Guide at **docs.puppetlabs.com/guides/language_guide.html**.

Connecting Servers to Puppet

Now that we have some configuration, let's set up some client systems to be managed by Puppet. In order to have a few client servers to manage, you may want to create a virtual machine and then clone it several times. These client servers need to be able to access the Puppet master on the network on the default port, 8140.

Install Puppet on the client machine:

```
root@localhost # apt-get install puppet
```

On Ubuntu, you also need to edit /etc/default/puppet to enable the service to start on boot:

```
root@localhost # vim /etc/default/puppet
# Defaults for puppet - sourced by /etc/init.d/puppet

# Start puppet on boot?
START=yes
```

Edit /etc/puppet/puppet.conf and add the FQDN of the Puppet master. If you don't have DNS, you also need to add the Puppet master's name and IP address to /etc/hosts:

```
root@localhost # vim /etc/puppet/puppet.conf

[main]
logdir=/var/log/puppet
vardir=/var/lib/puppet
ssldir=/var/lib/puppet/ssl
rundir=/var/run/puppet
factpath=$vardir/lib/facter
templatedir=$confdir/templates
prerun_command=/etc/puppet/etckeeper-commit-pre
postrun_command=/etc/puppet/etckeeper-commit-post
server = puppet.example.com

root@localhost # vim /etc/hosts

127.0.0.1    localhost.localdomain    localhost
192.168.0.1       puppet.example.com puppet
```

It's very important that all clients have an FQDN defined in /etc/hostname and can resolve their own names and the names of the Puppet master as they use SSL certificates to authenticate. On the Puppet master, configure your node in nodes.pp and apply some configuration to it:

```
root@localhost # vim /etc/puppet/manifests/nodes.pp

node test {

        include ntp
}
```

Certificate Exchange

Puppet authenticates itself using SSL certificates to ensure a secure, trusted connection. Before you can apply any configuration to nodes, you need to exchange and sign certificates. On the client node, type the following to connect to the Puppet master:

```
root@localhost # puppetd --test
```

If all goes well, you should see the following output:

```
info: Creating a new certificate request for test.example.com
info: Creating a new SSL key at /etc/puppet/ssl/private_keys/
➥test.example.com.pem
warning: peer certificate won't be verified in this SSL session
notice: Did not receive certificate
notice: Set to run 'one time'; exiting with no certificate
```

Now, on the Puppet master, check for the client certificate, and sign it:

```
root@localhost # puppetca -l
test.example.com

root@localhost # puppetca -s test.example.com
notice: Signed certificate request for test.example.com
notice: Removing file Puppet::SSL::CertificateRequest test.example.com
 ➥at '/var/lib/puppet/ssl/ca/requests/test.example.com.pem'
```

Now the client is authenticated, so re-running `puppetd --test` on the client machine should connect to the server and pull down your NTP config:

```
root@localhost # puppetd --test
iinfo: Caching catalog for test.example.com
info: Applying configuration version '1301969812'
notice: /Stage[main]/Ntp/Package[ntp]/ensure: created
notice: Finished catalog run in 2.42 seconds
```

Repeat optionally for as many nodes as you would like to manage, creating a definition for each in nodes.pp.

Now that you're managing some nodes, let's create some more configuration, so that every server in your environment will have a certain set of base configuration applied. This will include a sane hosts file, certain required packages, some standard users and the NTP configuration created earlier.

Create the directories for a module called packages:

```
root@localhost # cd /etc/puppet/modules
root@localhost # mkdir -p packages/{manifests,files,templates}
```

Now, create the class. Create the file manifests/init.pp in your favourite editor:

```
root@localhost # vim /etc/puppet/modules/packages/manifests/init.pp

class packages {

        package { "vim":
                ensure => installed,
        }

        package { "screen":
                ensure => installed,
        }

        package { "sysstat":
                ensure => installed,
        }
}
```

Here, we've defined three packages for Puppet to install. On Ubuntu, however, there's one more step to getting sysstat to collect metrics: enabling it in /etc/default/sysstat. To do this, grab a copy of this file from a system where you've manually installed the package, copy it into the right location in the module, and configure Puppet to push out this file:

```
root@localhost # cp /etc/default/sysstat
 ➥/etc/puppet/modules/packages/files
```

Edit the file, and ensure ENABLED is equal to true:

```
vim /etc/default/sysstat

# Default settings for /etc/init.d/sysstat, /etc/cron.d/sysstat
# and /etc/cron.daily/sysstat files
#

# Should sadc collect system activity information? Valid values
# are "true" and "false". Please do not put other values; they
# will be overwritten by debconf!
ENABLED="true"
```

Now, edit the packages class, and add the file resource:

```
root@localhost # vim /etc/puppet/modules/packages/manifests/init.pp

class packages {

        package { "vim":
                ensure => installed,
        }

        package { "screen":
                ensure => installed,
        }

        package { "sysstat":
                ensure => installed,
        } -> file { "/etc/default/sysstat":
                  ensure => present,
                  owner => root,
                  group => root,
                  mode => 644,
                  source => "puppet:///modules/packages/sysstat"
        }
}
```

Here, we've added a file resource and the name of the resource is the full path of the file to be deployed on the remote system. We've set the owner and group to root and the mode to 644, which is standard for configuration files that don't contain sensitive data. The source line specifies that Puppet will copy out a file called sysstat from the files directory of the module packages. Note that "files" doesn't need to be specified in the path—it's implied. We've used the arrow chaining syntax again to show the relationship between the package sysstat and this file, ensuring that the package is installed before Puppet tries to copy out the file. This isn't strictly necessary as the directory /etc/default already exists and it won't really hurt the system to have this file before the package, but it's good practice to get into specifying relationships.

Now when Puppet runs on a remote system that has this class added to it, it will check to see if the md5sum of /etc/default/sysstat matches the md5sum of the copy it has. If they don't match, Puppet will overwrite the remote file with the modified version.

Now, let's ensure sane hosts files for the environment. As the hosts file will need to contain the IP address and hostname of each node, you can't just use the same static file for each like you have for the sysstat package. Enter templates.

Create the directories for the hosts module:

```
root@localhost # cd /etc/puppet/modules/
root@localhost # mkdir -p hosts/{manifests,templates,files}
```

Now, copy the hosts file from the Puppet master into the templates subdirectory to give you something to work from. Give it the suffix .erb, as this is the extension used by Puppet templates:

```
root@localhost # cp /etc/hosts
 ➥/etc/puppet/modules/hosts/templates/hosts.erb
```

Open this file in an editor and insert a line beneath the localhost line:

```
root@localhost # vim /etc/puppet/modules/hosts/templates/hosts.erb

127.0.0.1 localhost
127.0.1.1   <%= fqdn %> <%= hostname %>
```

Where did these variables come from?

Every host managed by Puppet has a set of variables that can be accessed called facts. To get a list of facts currently known about a system, log on to that system and run:

```
root@localhost #  facter
architecture => i386
domain => compute-1.internal
facterversion => 1.5.8
hardwareisa => unknown
hardwaremodel => i686
hostname => domU
id => root
interfaces => eth0
ipaddress => 10.214.115.2
ipaddress_eth0 => 10.214.115.2
is_virtual => true
<snip>
```

Here you can see some of the available variables, although many more are not shown here.

Now, populate the rest of the hosts file with addresses important to your environment. If you use DNS internally, you may want to leave the rest of the file blank. It may be a good idea to define the Puppet master's IP address and hostname in /etc/hosts, so that Puppet still can manage the environment if DNS goes down.

Save the hosts file template, and create an init.pp for the module:

```
root@localhost # cd /etc/puppet/modules/hosts
root@localhost # vim manifests/init.pp

class hosts {

    file { "/etc/hosts":
        ensure => present,
        owner => "root",
        group => "root",
        mode => 644,
        content => template("hosts/hosts.erb"),
    }

}
```

The syntax to use a template is a little different from sending out a file whole. Rather than use the source keyword, we've used content and defined that this content is laid out in the template hosts.erb in the templates directory of the module hosts. Again, templates isn't required in the path; it's implicit.

For the final example module, let's look at some different resource types to manage users.

Create the directory structure for a users module, and create the init.pp:

```
root@localhost # cd /etc/puppet/modules
root@localhost # mkdir -p users/{manifests,files,templates}
root@localhost # touch users/manifests/init.pp
```

Let's create user accounts for Jane and Chris, two of our sysadmins, and for Robert and Sara, two developers. The two sets of users require different access levels, which you manage through UNIX groups. To keep the init.pp from becoming unwieldy, let's break the users out into two different files within the manifests directory:

```
root@localhost # vim /etc/puppet/modules/users/manifests/init.pp

class users {

  include users::admin, users::devel

}

root@localhost # touch /etc/puppet/modules/users/manifests/admin.pp
root@localhost # touch /etc/puppet/modules/users/manifests/devel.pp
root@localhost # ls /etc/puppet/modules/users/manifests/
admin.pp
devel.pp
init.pp
```

Here, we've created two manifests within the namespace of the module users. This certainly isn't necessary with only four users, but later when your users grow, it will help keep them organized within their categories.

Now that we have our empty manifests, we're going to cheat. Instead of typing the manifests by hand, we are going to extract the data from a system that already has these user accounts by using a tool called ralsh.

Ralsh is a simple tool for converting current system state into Puppet code. It can be used to interrogate a resource—a file, a user and so on—and returns everything it can find out about that resource, formatted in Puppet syntax:

```
root@localhost # ralsh user jane
user { 'jane':
    home => '/home/jane',
    shell => '/bin/bash',
    uid => '1001',
    ensure => 'present',
    password => '$6$N0V80Bci$.wBuBpSNj4fnTpvoSv3hC5UpzP/kTj2/
    ➥Sdkj50P16YUlP7aybpz9VkjHo4r6FHooNoO0t79iEHl748wcC4zL70',
    groups => ['adm','dialout','cdrom','plugdev','lpadmin',
    ➥'admin','sambashare'],
    comment => ',,,'
}
```

You can copy and paste this into the manifest, and if ralsh is run as root, it even will return users' password hashes, allowing them to have their usual password on the new system. We'll want to add one keyword: managehome. If this is set to true, when the user is created, Puppet will instruct useradd also to create the home directory. Repeat for the other users:

```
root@localhost # vim /etc/puppet/modules/users/manifests/admin.pp

class users::admin {

        user { 'jane':
              home => '/home/jane',
              managehome => true,
              shell => '/bin/bash',
              uid => '1001',
              ensure => 'present',
              password => '$6$N0V80Bci$.wBuBpSNj4fnTpvoSv3hC5UpzP/
             ➥kTj2/Sdkj50P16YUlP7aybpz9VkjHo4r6FHooNoO0t
             ➥79iEHl748wcC4zL70',
              groups => ['adm','dialout','cdrom','plugdev',
             ➥'lpadmin','admin','sambashare'],
              comment => ',,,'
        }

        user { 'chris':
              home => '/home/chris',
              managehome => true,
              shell => '/bin/bash',
              uid => '1002',
              ensure => 'present',
              password => '$6$N0V80Bci$.wBuBpSNj4fnTpvoSv3hC5UpzP/
             ➥kTj2/Sdkj50P16YUlP7aybpz9VkjHo4r6FHooNoO0t
             ➥79iEHl748wcC4zL70',
              groups => ['adm','dialout','cdrom','plugdev',
             ➥'lpadmin','admin','sambashare'],
              comment => ',,,'
        }
}

root@localhost # vim /etc/puppet/modules/users/manifests/devel.pp

class users::devel {
```

```
user { 'robert':
        home => '/home/robert',
        managehome => true,
        shell => '/bin/bash',
        uid => '1008',
        ensure => 'present',
        password => '$6$N0V80Bci$.wBuBpSNj4fnTpvoSv3hC5UpzP/
      ➥kTj2/Sdkj50P16YUlP7aybpz9VkjHo4r6FHooNo0t
      ➥79iEHl748wcC4zL70',
        groups => ['adm','dialout','cdrom','plugdev',
      ➥'lpadmin','sambashare'],
        comment => ',,,'
}

user { 'sara':
        home => '/home/sara',
        managehome => true,
        shell => '/bin/bash',
        uid => '1009',
        ensure => 'present',
        password => '$6$N0V80Bci$.wBuBpSNj4fnTpvoSv3hC5UpzP/
      ➥kTj2/Sdkj50P16YUlP7aybpz9VkjHo4r6FHooNo0t
      ➥79iEHl748wcC4zL70',
        groups => ['adm','dialout','cdrom','plugdev',
      ➥'lpadmin','sambashare'],
        comment => ',,,'
        }

}
```

It's important to note that the uid supplied must be available on the target systems.

Now that you have your basic modules, you can apply these to your nodes. One way of doing this efficiently is to define a metanode from which other nodes can inherit classes:

```
vim /etc/puppet/manifests/nodes.pp

node common {

        include users
        include packages
        include hosts

}

node test inherits common {

        include ntp

}
```

Save the file, and run `puppetd --test` on the client to check for changes.

Now if all has gone well, you should have a managed environment with a solid core of modules from which to build. The power of Puppet doesn't end here, with built-in types to manage more than 40 system resources, including cron jobs, filesystems and e-mail aliases.

Puppet also excels at configuring systems to role, allowing you to customize your systems further. For example, a LAMP module could install the packages for Apache, PHP and MySQL, configure Apache for virtual hosts, and add developer access to the new LAMP stack. If another Web server was needed to scale, an identical one could be deployed in minutes.

Experiment and build on these examples, and you soon will start to notice the benefits of configuration management with systems that are robust, reliable and above all, repeatable.■

Jes Fraser is an IT Consultant from Open Systems Specialists in New Zealand. She's passionate about promoting open source and Linux in the enterprise.

Resources

For more information about Puppet, see **www.puppetlabs.com**, and for more in-depth documentation, including a handy cheat sheet for the core types, see **docs.puppetlabs.com**.

Giada
Tech · Fashion · Art
www.GiadaPC.com

The Trend - Ultra Mini PC

Giada N20 ION2
Intel® Atom™ D525 **$295**
2G RAM, 320GB HD
Nvidia® ION2™, HDMI/ DVI

Giada i50 Core i5
Intel® Core™ i5 **$449**
2GB RAM, 320G HD,
HDMI/ DVI

Giada A50 E-350
AMD® Fusion E-350 **$322**
2GB RAM, 320G HD
ATI® Radeon™ HD6310

Giada i30/ i32
Intel® Atom™ D525/ D510
2G RAM, 320GB HD

Giada N10U
Intel® Atom™, Nvidia® ION™
2G RAM, 250GB HD

Giada MI-D525 **$99**
Atom™ D525, Dual LAN

Giada MI-E350
AMD® Fusion™+ATI® 6310

Giada MI-H67
Intel® Core™ i7/ i5

Giada MI-ION2
Nvidia® ION2™, HDMI/ DVI

Giada Technology, Inc. 1461-3 San Mateo Ave., South San Francisco, CA 94080 415.202.5441 Fax: 415.727.4947 Info@GiadaPC.com

NVIDIA, ION are trademarks of NVIDIA Corporation. Intel Core, Atom are trademarks of Intel Corporation. Other names are for informational purposes only and may be trademarks of their respective owners.

Polywell Solutions

Quiet Storage NAS/SAN/iSCSI

More Choices, Excellent Service, Great Prices!

NetDisk 8074A
72 Bay 144TB $26,999
74 Bay 222TB $36,999
- 4x Gigabit LAN
- RAID-5, 6, 0, 1, 10
- Hot Swap, Hot Spare
- Linux, Windows, Mac

5048A 5U-48Bay 144TB $26,999
RAID-6, NAS/iSCSI/SAN Storage
SATA, 4x GigaLAN

9020H 20Bay
60TB $9,999
- 4x Gigabit LAN
- RAID-5, 6, 0, 1, 10
- Hot Swap, Hot Spare
- Linux, Windows, Mac
- E-mail Notification
- Tower Case

9015H 15Bay
45TB $7,750
30TB $4,999
- Dual Gigabit LAN
- RAID-5, 6, 0, 1, 10
- Hot Swap, Hot Spare
- Linux, Windows, Mac
- E-mail Notification
- Tower Case

2U12B 2U-12Bay 36TB $6,999
SATA II, RAID-6, 2x GigaLAN
NAS/iSCSI/SAN Storage

4U24A 4U-24Bay 72TB $12,950
RAID-6, NAS/iSCSI/SAN Storage
Mix SAS/SATA, 4x Giga/10Gbit LAN

Netdisk 8000V
Quiet Performance
- Dual Gigabit LAN
- RAID-5, 6, 0, 1, 10
- Hot Swap, Hot Spare
- E-mail Notification
- Tower Case

Silent Eco Green PC
- Intel® / AMD® CPU
- Energy efficient
- Quiet and Low Voltage Platform

starts at **$199**

LD-001

Polywell OEM Services, Your Virtual Manufacturer
Prototype Development with Linux/FreeBSD Support
Small Scale to Mass Production Manufacturing
Fulfillment, Shipping and RMA Repairs

- 20 Years of Customer Satisfaction
- 5-Year Warranty, Industry's Longest
- First Class Customer Service

888.765.9686
linuxsales@polywell.com
www.polywell.com/us

POLY POLYWELL

Polywell Computers, Inc 1461 San Mateo Ave. South San Francisco, CA 94080 650.583.7222 Fax: 650.583.1974

NVIDIA, ION, nForce, GeForce and combinations thereof are trademarks of NVIDIA Corporation. Other names are for informational purposes only and may be trademarks of their respective owners.

A Way off the Ranch

Is Eben Moglen's FreedomBox the ticket out of Facebook's walled garden?

DOC SEARLS

As entities on the Web, we have devolved. Client-server has become calf-cow. The client—that's you—is the calf, and the Web site is the cow. What you get from the cow is milk and cookies. The milk is what you go to the site for. The cookies are what the site gives to you, mostly for its own business purposes, chief among which is tracking you like an animal. There are perhaps a billion or more server-cows now, each with its own "brand" (as marketers and cattle owners like to say).

This is not what the Net's founders had in mind. Nor was it what Tim Berners-Lee meant for his World Wide Web of hypertext documents to become. But it's what we've got, and it's getting worse.

In February 2011, Eben Moglen gave a landmark speech to the Internet Society titled "Freedom in the Cloud" (**www.softwarefreedom.org/events/2010/isoc-ny/FreedomInTheCloud-transcript.html**), in which he unpacked the problem. In the beginning, he said, the Internet was designed as "a network of peers without any intrinsic need for hierarchical or structural control, and assuming that every switch in the Net is an independent, free-standing entity whose volition is equivalent to the volition of the human beings who want to control it". Alas, "it never worked out that way". Specifically:

> If you were an ordinary human, it was hard to perceive that the underlying architecture of the Net was meant to be peerage because the OS software with which you interacted very strongly instantiated the idea of the server and client architecture.
>
> In fact, of course, if you think about it, it was even worse than that. The thing called "Windows" was a degenerate version of a thing called "X Windows". It, too, thought about the world in a server-client architecture, but what we would now think of as backwards. The server was the thing at the human being's end. That was the basic X Windows conception of the world. It served communications with human beings at the end points of the Net to processes located at arbitrary places near the center in the middle, or at the edge of the Net. It was the great idea of (Microsoft) Windows in an odd way to create a political archetype...which reduced the human being to the client and produced a big, centralized computer, which we might have called a server, which now provided things to the human being on take-it-or-leave-it terms.

True, but let's not forget Netscape's role. The HTTP cookie was created by Lou Montulli in 1994, when he was working at Netscape on the first e-commerce servers. The idea was innocent enough: to recall state and save work by remembering shared data, such as the contents of shopping carts. Today, cookies are used for many other things, including user tracking, mostly so advertising can be personalized. The less-polite word for this is spying. Eben singles out Facebook as an especially egregious offender and describes its offering this way: "I will give you free Web hosting and some PHP doodads and you get spying for free all the time".

This trade-off is the rule, not the exception. In the study that launched its "What They Know" series last year, the *Wall Street Journal* said all but one of the 50 most-popular Web sites installed tracking cookies in users' browsers. (The only exception, no surprise, was Wikipedia.) One, Dictionary.com, installed 234 tracking files, of which all but 11 were from tracking companies.

The end state of this system was forecast with metaphorical precision by *The Matrix*, back in 1999. From the standpoint of its business model, the advertising-supported commercial Web is a machine world in which users are nothing more than batteries to whom "experience" is "delivered" through pipes. In that world, advertising companies are the agents, and tracking cookies are like the electric scorpion that wiggles into Neo through his navel so he can be followed around.

Eben doesn't want us to go there. So he had a plan: FreedomBoxes (**www.nytimes.com/2011/02/16/nyregion/16about.html?_r=1**). These are "cheap, small, low-power plug servers...the size of a cell-phone charger, running on a low-power chip". He sees these boxes starting cheap and dropping in price, eventually costing less than $30 apiece. That's roughly the price of cell-phone earbuds.

Thus, Eben and friends created the FreedomBox Foundation (**https://freedomboxfoundation.org**), set up a KickStarter project (**www.kickstarter.com/projects/721744279/push-the-freedombox-foundation-from-0-to-60-in-30**) and quickly attracted enough donations to get rolling. (They asked for $60,000 and got $86,724.) Others jumped on board. Debian, for example, has its own FreedomBox project (**wiki.debian.org/FreedomBox**) to support the foundation's efforts with working code.

Although Eben didn't aim his appeal to big companies, we can use their help, especially at the retail level. For FreedomBoxes to be popular, they'll need to show up where the populace shops. That means we should see FreedomBoxes on the shelves at Best Buy, Radio Shack and Walmart—and not just at, say, Amazon. Hey, companies don't need to be cows any more than humans need to be calves. When the world fills up with FreedomBoxes, customers can help companies evolve.

The economic case we need to make is that free customers are more valuable than captive ones. But, we can't make it if we're still on the ranch. We need to break out.■

Doc Searls is Senior Editor of *Linux Journal*. He is also a fellow with the Berkman Center for Internet and Society at Harvard University and the Center for Information Technology and Society at UC Santa Barbara.

ABERDEEN
SERVERS AND STORAGE

ANYONE INTERESTED IN SAVING MONEY?

Looks like these guys are comfortable overpaying for enterprise storage. Are You?

"Hewlett-Packard Co. agreed to buy 3Par Inc. for $2.35 billion" — *Bloomberg.com*

"EMC to Buy Isilon Systems Inc. for $2.25 Billion" — *Wall Street Journal*

"Dell to Buy Compellent for $960 Million" — *CNBC*

So what "benefit" will you see by this spending spree, other than higher costs?

The AberSAN Z-Series scalable unified storage platform, featuring the Intel® Xeon® processor 5600 series, brings the simplicity of network attached storage (NAS) to the SAN environment by utilizing the innovative ZFS file system. The AberSAN Z20 is easily found starting under $20,000.

Who gives you the best bang for the buck?

	3Par InServ F200	Compellent Storage Center Series 30	Isilon NL-Series	Aberdeen AberSAN Z20
Storage Scale-Out	✓	✓	✓	✓
Thin Provisioning	✓	✓	✓	✓
HA Clustering	✓	✓	✓	✓
VMware® Ready Certified	✓	✓	✓	✓
Async / Synchronous Replication	✓	✓	✓	✓
iSCSI / Fibre Channel Target	✓	✓	iSCSI Only	✓
Unlimited Snapshots	X	✓	✓	✓
Native Unified Storage: NFS, CiFS	X	X	✓	✓
Virtualized SAN	X	X	X	✓
Deduplication	X	X	X	✓
Native File System	none	none	OneFS	ZFS 128-bit
RAID Level Support	5 and 6	5 and 6	Up to N+4	5, 6 and Z
Raw Array Capacity (max)	128TB	1280TB	2304TB	Unlimited
Warranty	3 Years	5 Years	3 Years	5 Years
Online Configurator with Pricing	Not Available	Not Available	Not Available	**Available**

intel® Xeon® inside™

Powerful. Intelligent.

888-297-7409
www.aberdeeninc.com/lj038

Cut Execution Time by >50% with WhisperStation-GPU

Delivered ready to run new GPU-enabled applications:

Design	Simulation		BioTech
3ds Max Bunkspeed Shot Adobe CS5	ANSYS Mechanical Autodesk Moldflow Mathematica	MATLAB ACUSIM AcuSolve Tech-X GPULib	AMBER GROMACS NAMD, VMD TeraChem

WhisperStation with 4 Tesla Fermi GPUs

Integrating the latest CPUs with NVIDIA Tesla Fermi GPUs, Microway's WhisperStation-GPU delivers 2x-100x the performance of standard workstations. Providing explosive performance, yet quiet, it's custom designed for the power hungry applications you use. Take advantage of existing GPU applications or enable high performance with CUDA C/C++, PGI CUDA FORTRAN, or OpenCL compute kernels.

- Up to Four Tesla Fermi GPUs, each with: 448 cores, 6 GB GDDR5, 1 TFLOP single and 515 GFLOP double precision performance
- Up to 24 cores with the newest Intel and AMD Processors, 128 GB memory, 80 PLUS® certified power supply, and eight hard drives
- Nvidia Quadro for state of the art professional graphics and visualization
- Ultra-quiet fans, strategically placed baffles, and internal sound-proofing
- New: Microway CL-IDE™ for OpenCL programming on CPUs and GPUs

Microway's Latest Servers for Dense Clustering

- 4P, 1U nodes with 48 CPU cores, 512 GB and QDR InfiniBand
- 2P, 1U nodes with 24 CPU cores, 2 Tesla GPUs and QDR InfiniBand
- 2U Twin^2 with 4 Hot-Swap MBs, each with 2 Processors + 256 GB
- 1U S2050 servers with 4 Tesla Fermi GPUs

Microway Puts YOU on the Cutting Edge

Design your next custom configuration with techs who speak HPC. Rely on our integration expertise for complete and thorough testing of your workstations, turnkey clusters and servers. Whether you need Linux or Windows, CUDA or OpenCL, we've been resolving the complicated issues – so you don't have to – since 1982.

OctoPuter™ 4U Server with up to 8 GPUs and 144 GB memory

1U Node with 2 Tesla Fermi GPUs

2U Twin^2 Node with 4 Hot-Swap Motherboards Each with 2 CPUs and 256 GB

Configure your next WhisperStation or Cluster today!

microway.com/quickquote or call 508-746-7341

Sign up for technical newsletters and special GPU promotions at microway.com/newsletter

NVIDIA TESLA · intel Channel Partner Premier 2010 · AMD Fusion Partner Premier · GSA

GSA Schedule Contract Number: GS-35F-0431N

Microway® Technology you can count on℠