PostgreSQL 9.0 | Drupal 7 | Drush | Boxee Box | Calibre | Qt

LINUX JOURNAL™

Since 1994: The Original Magazine of

APRIL 2011 | ISSUE 204 | www.linuxjournal.com

ANGELA BYRON on DRUPAL 7

Create Complex Web Sites Easily with ZOTONIC

Manage Your E-Books with CALIBRE

Writing RICH INTERNET APPS for Users

+ Qt Quick for User Interfaces

How-to: Google Maps API

DRUSH for Drupal on the Command Line

REVIEWED: D-LINK'S BOXEE BOX

$5.99US $5.99CAN

04>

More TFLOPS, Fewer WATTS

Microway delivers the fastest and greenest floating point throughput in history

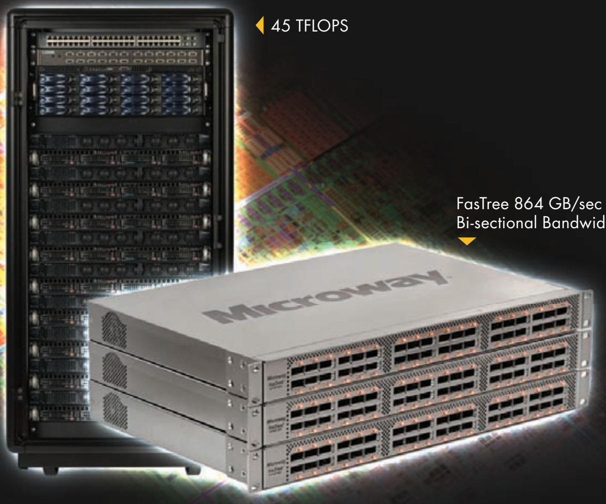
Enhanced GPU Computing with Tesla Fermi

- 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

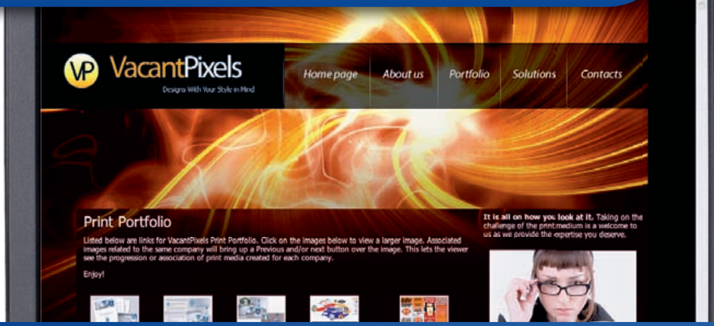
intel Channel Partner Premier 2010

AMD Fusion Partner Premier

GSA

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

Microway®
Technology you can count on℠



CONTENTS

APRIL 2011
Issue 204

FEATURES

44 Drupal 7: the Webchick behind the Wheel
An interview with Angela Byron, co-maintainer of Drupal 7.
Katherine Druckman

50 Drush: Drupal for People Who Hate Mice
It makes life easier for those of us who spend some of our working hours hacking away at the command prompt.
James Walker

54 Simplicity and Performance: JavaScript on the Server
Surprisingly, the language of the browser is powerful, easy to use and well-suited to high-performance server-side programming—when done right.
Avi Deitcher

58 Zotonic: the Erlang Content Management System
It's easy to use and open source.
Michael Connors

COVER PHOTO:
Kris Krug, staticphotography.com

ON THE COVER

- Create Complex Web Sites Easily with Zotonic, p. 58
- Manage Your E-Books with Calibre, p. 75
- Angela Byron on Drupal 7, p. 44
- Writing Rich Internet Apps for Users, p. 68
- Drush for Drupal on the Command Line, p. 50
- Qt Quick for User Interfaces, p. 71
- How-To: Google Maps API, p. 64
- Reviewed: D-Link's Boxee Box, p. 40

KRIS KRUG

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	✗	✓	✓	✓
Native Unified Storage: NFS, CiFS	✗	✗	✓	✓
Virtualized SAN	✗	✗	✗	✓
Deduplication	✗	✗	✗	✓
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**

Powerful.
Intelligent.

Above specific configurations obtained from the respective websites on Feb. 1, 2011. Intel, Intel Logo, Intel Inside, Intel Inside Logo, Pentium, Xeon, and Xeon Inside are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries. All trademarks are the property of their respective owners. All rights reserved. For terms and conditions, please see www.aberdeeninc.com/abpoly/abterms.htm. lj038

888-297-7409
www.aberdeeninc.com/lj038

CONTENTS APRIL 2011 Issue 204

COLUMNS

20 Reuven M. Lerner's At the Forge
PostgreSQL 9.0

24 Dave Taylor's Work the Shell
Mad Libs Generator, Part II

26 Mick Bauer's Paranoid Penguin
Interview with a Ninja, Part II

30 Kyle Rankin's Hack and /
Your Own Personal Server: DNS

80 Doc Searls' EOF
Hacking with Humor

REVIEW

40 D-Link's Boxee Box
Shawn Powers

INDEPTH

64 Find Yourself with the Google Maps API
DIY Google Maps.
Mike Diehl

68 Rich Internet Apps That Just Work—Writing for the User
With the right tools, you can build rich apps that work with, not against, the user.
Avi Deitcher

71 Quick User Interfaces with Qt
Qt Quick is transforming user interfaces.
Johan Thelin

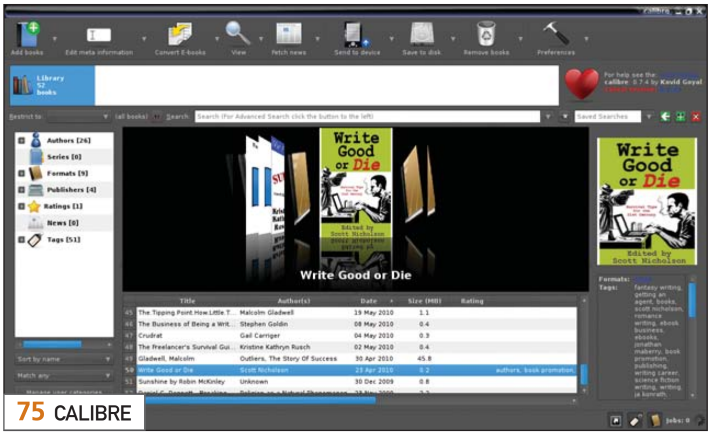
75 Calibre
Organize your e-book collection before it gets (even more) out of control.
Dan Sawyer

IN EVERY ISSUE

8 Current_Issue.tar.gz
10 Letters
14 UPFRONT
34 New Products
36 New Projects
65 Advertisers Index
79 Marketplace

40 D-LINK'S BOXEE BOX

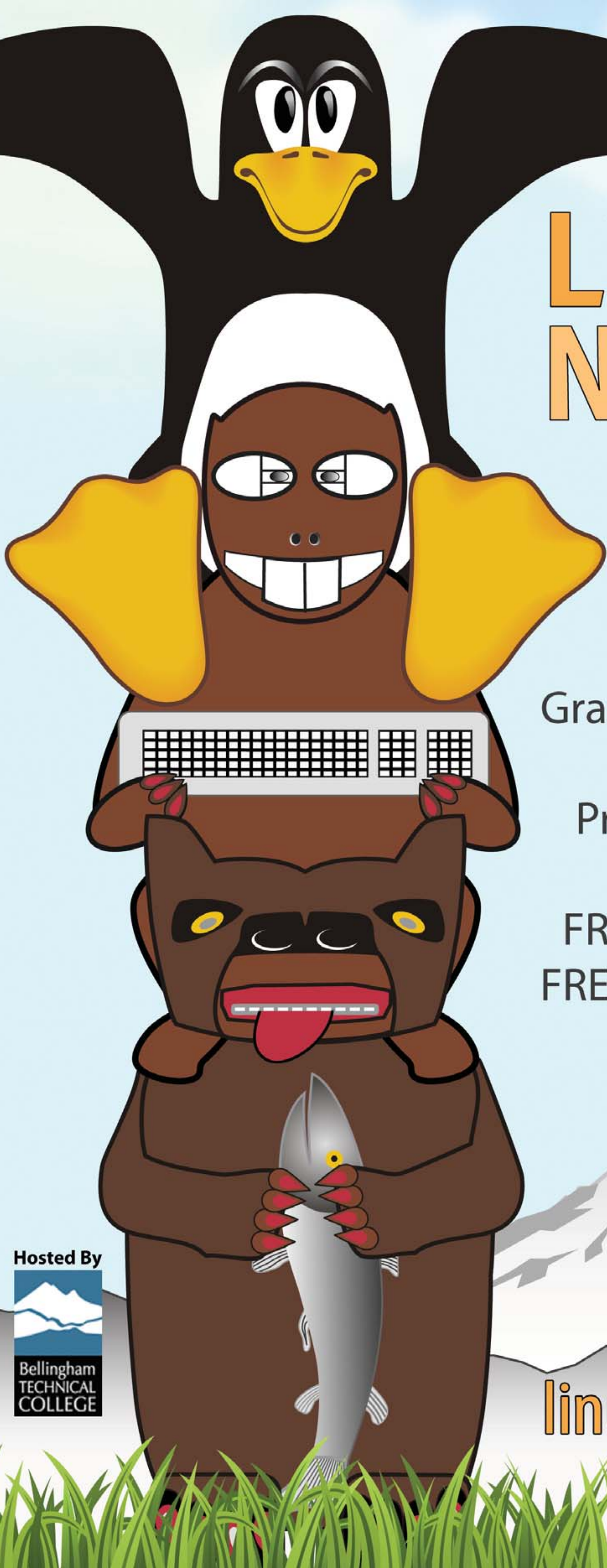
64 GOOGLE MAPS API



	Title	Author(s)	Date	Size (MB)	Rating
45	The Tipping Point How Little T...	Malcolm Gladwell	19 May 2010	1.1	
46	The Business of Being a Writ...	Stephen Goldin	08 May 2010	0.4	
47	Crudrat	Gail Carriger	04 May 2010	0.3	
48	The Freelancer's Survival Gui...	Kristine Kathryn Rusch	02 May 2010	0.4	
49	Gladwell, Malcolm	Outliers, The Story Of Success	30 Apr 2010	45.8	
50	Write Good or Die	Scott Nicholson	25 Apr 2010	0.2	authors, book promotion,
51	Sunshine by Robin McKinley	Unknown	30 Dec 2009	0.8	

75 CALIBRE

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



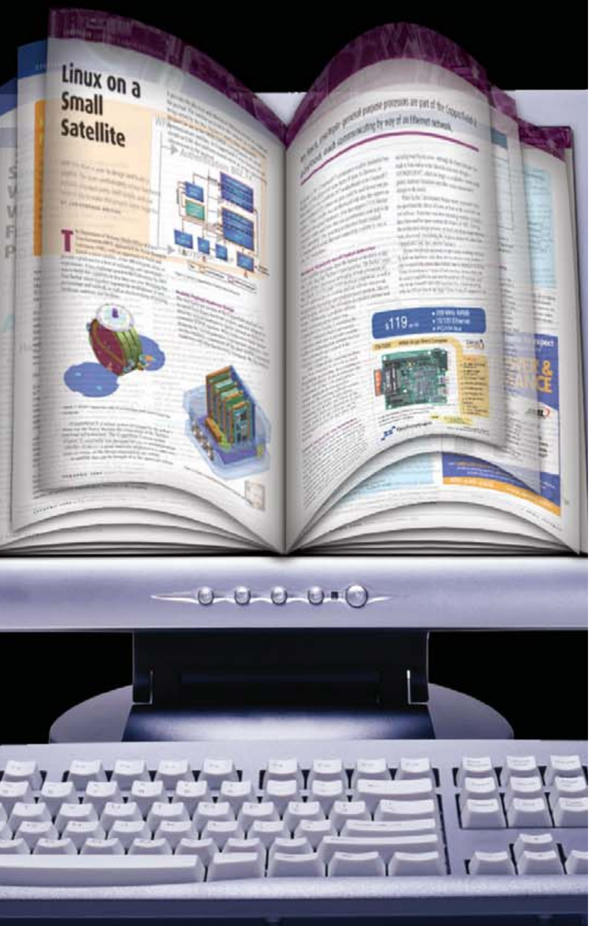
LinuxFest Northwest

April 30 - May 1, 2011
Bellingham, WA

Grassroots Linux gathering
Exhibits of all flavors
Presentations of all levels
Prizes and after party
FREE admission & parking
FREE open source software
Bring the whole family!

Hosted By
Bellingham Technical College

linuxfestnorthwest.org



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

SHAWN POWERS

Warm Up Those Spinnerets!

Instant gratification is quite gratifying. Because I haven't yet perfected the Arduino Wetwired Electronic Semi-Omniscient Memory Enhancer (A.W.E.S.O.M.E. for short), we figured the best way to serve our fast-paced world was to have a Web Development issue. Let's face it, if you want to see the weather forecast, you visit a weather Web site. Waiting for the six o'clock news takes too long, and I'm not willing to admit I'm old enough to watch The Weather Channel. The Web, along with its Twitter-esque companions, is currently the fastest way to get information to the masses. And we want to help.

Reuven M. Lerner starts things off with the back end of the Internet. No, I don't mean my personal Web site; I mean databases. This month, Reuven looks at PostgreSQL 9.0. Even if you're a MySQL fan, PostgreSQL is hard to hate, and Reuven explains why. Dave Taylor is hard to hate as well, and this month, he takes us back to our youth with the second part of his series on creating *Mad Libs*. I did my first *Mad Lib* when I was [ANY_NUMBER] years old, while I was [ACTIVE_VERB] in [PLACE], and it was really [DESCRIPTIVE_ADJECTIVE].

Kyle Rankin and Mick Bauer teach us about servers this time around. Kyle describes creating a DNS server as part of his series "Building Your Own Personal Server". DNS can be complicated, but Kyle walks through setting up the industry standard, BIND. Mick is on the opposite end of the spectrum with his "Interview with a Ninja". What Kyle shows how to build, Mick and his ninja talk about hacking into—good information from both guys.

We've all been 37 steps into an on-line form, only to have a click of the Back button completely ruin the form on which we worked so hard. Creating desktop-like Web apps is popular, but sometimes the sites we visit don't work the way we expect. We want the Back button to take us back a page, but with Web programming, that's not always as easy as it sounds. Avi Deitcher addresses the problem and explains how to make AJAX applications honor the venerable Back button. Avi also shows why JavaScript itself is so awesome in another article, "Simplicity and Performance: JavaScript on the Server". Go get a cup of java, and check out his JavaScript articles. You'll be a better programmer for it.

Some of us aren't hard-core programmers, but we still need to get information onto the Web and have it look good and perform well. That's when content management systems are awesome. Michael Connors introduces Zotonic, a content management system based on Erlang. If you're new to the idea of a CMS, you'll want to check out Zotonic, which is a CMS, but also a Web framework. Drupal, on the other hand, is a traditional CMS that is used in thousands of Web sites across the planet. Our own LinuxJournal.com Web site runs Drupal, and this month, Webmistress Katherine Druckman interviews Angela Byron, the co-maintainer of Drupal 7. Reading the interview will teach you a bit about Drupal, but even more than that, it will give you a behind-the-scenes glimpse at managing a large open-source project. Drupal 7 sounds like a huge step forward in usability, and it's Angela's job to make sure that step doesn't trip and fall along the way.

I know some of you are annoyed by the Web and its dependence on mouse clicking. I suspect Kyle Rankin and his terminal window are as well. James Walker feels the same way and introduces Drush, a command-line interface for Drupal. Sure, a nice GUI is great, but sometimes it's hard to beat a simple command line. James demonstrates a click-free way to interact with Drupal. It's pretty cool for those of us addicted to the command line.

If you're not interested in the Web at all, we haven't forgotten you this month. Whether you want to make a quick user interface with Qt (Johan Thelin shows how) or manage and convert your e-book collection with Calibre (Dan Sawyer explains the process), this issue is bound to please. We also have a review of D-Link's Boxee Box by yours truly, and we have instructions on how to find yourself—with Google Maps (Mike Diehl shows how to manipulate the API).

This is a very fun issue, and until my A.W.E.S.O.M.E. is fully developed, turning the page and reading is the best way to assimilate the information. I'm not sure whether my Arduino brain implant will be ready for the Cool Projects issue, but those interested in beta testing should feel free to contact me. Just think real hard and the message should get to me, assuming my A.W.E.S.O.M.E. is working correctly. Until then, have a [DESCRIPTIVE_ADJECTIVE] [TIME_PERIOD], and we'll [ACTION_VERB] you next month! ■

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.

Understanding Drupal
Administering Drupal
Site Building with Drupal
Learning CCK for Drupal
Learning Views for Drupal
Theming Basics for Drupal
Advanced Theming for Drupal
Introduction to jQuery
jQuery & JavaScript in Drupal
Drupal Module Development

HTML5 | Giada Slim-N20 | Jolicloud | GNOME 3 | Unity | Nepomuk
LINUX JOURNAL
Since 1994: The Original Magazine of the Linux Community
The Ayatana Project
Improving Social Media on the Ubuntu Desktop
Introducing Nepomuk
KDE's Semantic Desktop Technology
The Linux Desktop
What's New with the GNOME Project and Ubuntu?
REVIEWED: Giada Slim-N20 Jolicloud and Ubuntu 10.10 Maverick Meerkat Amazon EC2
BUILD a Portable Custom Desktop
PLUS: Check Out Alternate Desktop Managers
FEBRUARY 2011 | ISSUE 202
www.linuxjournal.com
HOW-TO: Kexi, the KOffice Database Management Program

Letters Complaining about Linux

I get annoyed every time a Linux magazine publishes a letter complaining about some supposed shortcoming of Linux compared to Windows. And, since *Linux Journal* seems to do it at least every other issue, I get annoyed a lot.

Sure, Linux isn't perfect. No OS is. But the suggestion that things just work in Windows or that it is easier to correct problems is wrong. In my experience, things are more likely to "just work" in Linux.

In Windows, you often have to hunt down an install disk. And, forget about getting your problem solved when Windows messes up. Instead of being able to fix a config file, you usually end up going through multiple un-installs, reboots and re-installs hoping that eventually it will work.

I suspect that the people saying they're going back to Windows are actually marketing trolls employed by a certain closed-source company to promote its OS. Otherwise, there is no way any rational person could really imagine that Windows is in any way superior to Linux in the ease-of-use department.

Windows wins in only one area so far as I can see. It comes pre-installed, while you usually have to install Linux. Otherwise, it's just painful compared to any recent Linux distribution.

--
Gary Dale

I do think a fair number of people legitimately switch back to Windows. And, that's fine, what with free will and all. For a very large amount of computer needs, Linux fits the bill perfectly. There are a few instances when it does not. Commercial gaming is one of them for sure. (That's not to say there aren't many games for Linux, commercial and otherwise, just that the majority still are Windows-only.)

I think an even bigger pull, however, is peer support. Everyone has a cousin or uncle that can help with computer problems. Not too many people have someone who can help with Linux problems. Again, that's not to say help is unavailable, it's just nontraditional. While calling weird Uncle Marvin for Windows support isn't always the smartest thing to do, it's what many people are comfortable doing. Plus, with Windows 7, the "searching for drivers" game is largely over. Perhaps Microsoft took its cues from the Linux world, but for the most part, Windows now has drivers for most products by default.

Either way, it's hard to beat free, and Linux has several varieties of free from which to choose. I don't expect Microsoft to follow suit any time soon.—Ed.

Net Neutrality Legislation

We in the FOSS world often talk about freedom and stress its importance. That is why I am so disappointed in the recent legislation on Net Neutrality. We honor people's freedom to use their belongings as they best see fit, yet we fall for the canard that if people associate, and freely pool their resources, maybe even incorporate, then their property and freedoms should be curtailed.

And, typically, such actions have unintended consequences. Rather than see a small number of large corporations negotiate terms of the values of different kinds of Net traffic with each other or their customers, we will have those same corporations working with Washington representatives to mold and modify the regulations of Net Neutrality on an ongoing basis. Google and Verizon are not leaders in the Net Neutrality effort from the desire to have their traffic treated like any other—they are there to influence the rules and rule-makers right from the start. So there will be Net non-Neutrality, but instead of being based on who has the best deal, with the consumer-driven marketplace to correct the eventual mistakes, such rules will be based on those who have the best lobbyists and the most influence to pedal. These deals will be forged in back rooms and after-hours deals. The small guy or startup will have a harder time getting established, and the consumer will have less visibility and opportunity to correct problems.

This Net Neutrality regulation replaces freedom with centralized control, and that's a danger any FOSS proponent should recognize.

--
Keith Reed

Scary 1984 references aside, Net Neutrality is a worrisome topic. I fear that since the world now depends on data more than ever before, the waters are going to get murkier rather than clearer. Sadly, I don't really have anything to add to your comment, other than an urge to don my tin-foil hat.—Ed.

Unity Desktop

Thank you for the February 2011 issue about the Linux desktop. I am running Ubuntu 10.10 on an Eee PC and find it far superior to the Microsoft XP system that was delivered with it, with one exception (and Mr Shuttleworth wanted some feedback about the Unity desktop). I found it to be difficult to use. Most of the applications I need are buried way down in the menus, often requiring longer to find them than I was willing to spend. I could have simply customized the icon mix on the icon list; however, there is a shocking lack of any resource to instruct me on changing the behavior. Until setting it up with the user icon selection is intuitive, the

Unity interface will not be widely adopted, and worse, it will not win any Linux converts from the Windows crowd. Fortunately, I was able to find the key to starting the more traditional desktop, and I am off and running with it. Mr Shuttleworth should consider studying not only confirmed Linux users, but more casual users as well. I like the concepts behind Unity, but the implementation needs a bit of intuition built in, but then again, that's what innovation is all about. Please keep up the good work.

--
Jim O.

I'm completely with you on this one. I recently put the Ubuntu 10.10 Netbook edition on my Netbook as well, and I absolutely hate Unity. Like you, I find it hard to get to applications, and although a simple menu is quick and responsive on an underpowered Netbook, I found the Unity interface slow and kludgy.

Thankfully, Unity is just the default, and it can be changed easily. As to why Canonical decided Unity was the ideal interface, I have no idea. Perhaps it will be better when it hits the 11.04 desktop edition, but as it is now, I'm not a fan.—Ed.

No Chance!

I read Stuart Jarvis' article "Organize Your Life with Nepomuk" with great interest [*LJ*, February 2011]. With open source, I appreciate the accessibility of information. If something interesting exists, it will be known by the community.

The problem is the name of the project! Nepomuk is also the name of Hitler's grandmother. The family history is complicated, not glorious, and a great part of it has been removed by Hitler himself, so I cannot be more precise.

The project has nothing to do with that, but this coincidence can be harmful for us. I don't know what to do. Changing the name is not a good idea; doing nothing is certainly the best, hoping that nobody knows the grandmother's name! Happy New Year and thank you for the quality of your work.

--
Denis

Stuart Jarvis replies: *I'm glad you liked the article. You are right, sharing information is one of the great things about free software.*

About the name Nepomuk, as I'm sure you know, it is an acronym and one that was chosen by the original research project rather than by KDE. We had not made any association with Hitler's grandmother, but there are plenty of other more wholesome connections that can be made. There is, for example, John of Nepomuk, a national saint in the Czech Republic, and Nepomuk is also a friendly dragon in a German children's book.

Within KDE software itself, we prefer to talk about the "semantic desktop" (or the results of the technology, such as desktop search), as that is a more descriptive and understandable name. So, we do not see a need to change the name of the technology, but we already were planning not to use "Nepomuk" very widely.

I hope you will enjoy using KDE software that benefits from Nepomuk as the technology continues to mature.

Feedback Error

In the February 2011 issue's Letters section, Jeffrey Brendecke said that in Dave Taylor's article in the November 2010 issue "Scripting Common File Rename Operations" the code snippet `f=foo.bar.txt echo "$f" | cut -d. -f2` results in `bar.txt`. In fact, it returns `bar`. Using `-f3` in the cut would have gotten us `txt`.

--
Scott Field

xwinwrap

I love the ability of many "smart" phones to have an animated background. I searched the Internet for a way to replicate the effect on my computer and stumbled on the program xwinwrap from different forums. After a little bit of digging, I found a .deb and now enjoy various video loops as my desktop. It would be awesome if an article was written highlighting this program to other *Linux Journal* readers.

--
Milton Pleasant

Thanks Milton! You just gave me an idea for a Linux Journal *Tech Tip video. I'll be sure to give you credit.—Ed.*

Why Not to Use Gmail

I've been using Linux for 11 years and Gmail for almost six years. Recently, I've slowly started to realize I don't own or control my e-mail. Astonishing, I know. Anyway, I did a quick Google search (silly, right?) and was unable to find anything on "reasons to avoid using Gmail". I was wondering if *LJ* would be interested in doing an article on this. I know we all love Google for its genuine greatness when it comes to selling us free items, but I really think it would be awesome to see the flip side of this. Thanks a bunch!

--
Greg

Oddly enough, many of our readers already don't use Gmail. I must admit, I do myself, but I get chided for it often. I do think Gmail gets targeted rather specifically in these things though, when in reality Hotmail and Yahoo are just as creepy. Sure, you can delete e-mail easier with them, but does it really get deleted? How would we know? Yes, it's scary how much information is stored on servers we don't control. And, don't even get me started on Facebook!—Ed.

WebOS and Technology

WebOS lies dormant on the playing field and slowly begins to rebuild itself. The challenge as a user is finding consumer confidence. With HP bringing various assets to the front, it can grow confidence. I had a choice between a Pixi Plus and a Droid and chose the open productivity of the Palm.

Of all things, the Open Source community grows stronger daily, in large part because consumerism drives trends. It is not always a matter that is driven by passing fads. Open source is one of the best ways to go in the large and expanding community. I look forward to Palm and HP expanding the market and forming confidence in the end product.

--
Joseph Ziehm

We have at least one WebOS fan here on staff at Linux Journal. *Most of us just haven't had an opportunity to try it. In the past, you had to be a Sprint user to get a Palm, so that really hurt adoption. Perhaps now that HP is running things, we'll see a bit wider availability, and perhaps WebOS will catch on. Android has a huge head start though. Hopefully, WebOS will do well, and we'll continue to have choices that are open. Those are my favorite types of choices.—Ed.*

Tablets?

I have to agree with Bill Childers in the recent Point/Counterpoint on the Tablet PC [*LJ*, February 2011]. I also would add that I am waiting for a major improvement before I buy one, and that is multitasking. I feel like it is 1994, and we are waiting for Windows 95 to come out. Pardon me, but that is the best analogy I could come up with. Also, at that time I had not found Linux yet.

The iPad has one very specific and professional use—in sales presentations and on-site meetings. The salesman I work with has told me that it is a much easier and intuitive medium for making a sales presentation to a client, because it is hands-on, a novelty item and reduces the need for a projector and laptop. Otherwise, great job with the debate.

--
hal98x

XFCE Is Much Lighter on Resources

In the February 2011 article "The Second-String Desktop", Shawn Powers compared Ubuntu to Xubuntu to see which used less RAM. He was shocked when they both used about the same amount of RAM, but he should not have been. The reason is that Xubuntu is not trying to be lighter on resources but instead simply an alternative to the traditional GNOME desktop environment. If you want to see how light XFCE really can be, you should look at CrunchBang XFCE. It uses less than 100MB of RAM when initially installed. It's better than Ubuntu, Xubuntu and Lubuntu and perfect for low-resource systems. My point is that Ubuntu pulls in all sorts of unneeded and bloated dependencies that defeat the purpose of a low-resource system.

--
Jacob

At the time of this response, the Xubuntu home page says, "An official version of Ubuntu Linux that uses the XFCE desktop environment. Designed for low-specification computers". So I would argue that Xubuntu is meant to be lighter than Ubuntu. You are correct that CrunchBang is a great example of an easy-on-the-resources implementation of XFCE. As far as "better", well, that's unfortunately a matter of preference. I don't think any of them are "better" than GNOME, but I freely admit that's my own opinion! Thanks for pointing out CrunchBang though. It's a great distro.—Ed.

Re: "Netflix and Linux, It's a DRM Shame"

In the February 2011 issue's Upfront section, Shawn Powers mentions that Netflix should release its proprietary software to users running GNU/Linux on their desktops, rather than just for devices, such as Roku. He mentions that "the motivation to reverse-engineer [the software] would be close to zero". I think Shawn is missing the point.

The free (as in freedom) software community is ethically opposed to proprietary software and, as a consequence, DRM (DRM cannot exist within free software). Should Netflix release its software, it never would be an option for the Free Software community. In fact, the only ethical use of proprietary software is to develop a free alternative. However, Netflix will never be a thought in my mind, even with a free decoder, until it removes its restrictions and provides a DRM-free, preferably open, format.

More information, including a petition to Netflix to remove its DRM, can be found at DefectiveByDesign.org: **www.defectivebydesign.org/blog/1093**.

--
Mike Gerwitz

I didn't miss the point; I was simply making a different point. Although I would love to see Netflix exist without DRM, I also would like to see Netflix exist in any form for Linux users. Whether individuals agree or disagree with Netflix's business decisions, as it stands, the entire Linux Desktop community is unable to use Netflix. My point was not whether or not they should, *but that they* can't.

I am a proponent of free as in freedom. I'm also a proponent of choice. I continue to believe that people should be allowed to make choices with which we disagree. I could get my soapbox out, but I think I'll leave it there. Thanks for the feedback.—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**.

Mid-America GNU/Linux Networkers conference St. Louis

No restrictions. No limits. Just freedom. Be free.

Don't miss our inaugural event. Technical classes, social networking and fun
for free software experts, enthusiasts and community newcomers alike.

May 2011

MAGNETCon.info

diff -u

WHAT'S NEW IN KERNEL DEVELOPMENT

The **stable kernel efforts** once again have gotten out of control. In the old days, **Linus Torvalds** had a cute x.even and x.odd version-numbering scheme, where even releases were part of a stable series that would last six months or a year or thereabouts, and odd releases were part of a development series that would last roughly the same amount of time. During the stable series, only bug fixes would be accepted; during the development series, new features also could be added. Over time, that made various folks wish they were dead, because the long duration of the stable series meant that tons of people had to sit around and wait for the development series to open up again.

With the 2.6 kernel, Linus abandoned the concept of a stable series, and this too led to some serious uncertainty among people who no longer had any kernel release they really could rely on. After some vicissitudes, **Greg Kroah-Hartman** and others started forking off various 2.6 releases that they would maintain as their own "stable series", simultaneously with Linus' ongoing 2.6 release schedule. The world stopped its desperate careening, and everyone took an easy breath.

Greg and his cohorts chose which 2.6 kernels to fork into a new stable series, largely based on what would work best for them. From the outside, it seemed to be a fairly arbitrary and unpredictable process, which gradually led to pain and suffering among a very specific subset of users—the distribution maintainers. Anyone maintaining a Linux distribution, or developing a Linux-based phone or other embedded device, really wanted to know which kernels were going to have their own stable series, so they could base all their hard work on that kernel. But because there was no way to predict it, those folks were running into problems organizing their work, making market predictions and so forth. They began begging and pleading with Greg to start a new stable series for the particular kernel versions on which they each depended—or at least to make it more predictable which kernels would get a stable series.

Greg and the other stable series folks couldn't accommodate this without putting in a metric ton of additional work, so recently, Greg essentially decided to ditch the whole project.

From now on, he announced, he will no longer maintain long-term stable kernel trees. Instead, what he'll do is put out a few stable releases for each kernel, just until Linus releases the next development kernel version, at which point Greg will start putting out stable releases for that version and so on. This way, Greg says, people will be less likely to fixate on kernels that are super old and out of date and concentrate on the more-recent, better, faster, stronger and more-beautiful kernel of the moment.

The existing stable trees, particularly 2.6.27, 2.6.34 and 2.6.35, will not be going away. Instead, he's handed them off to other folks, who will maintain them as they see fit. Presumably other maintainers may be found to continue the stable patch series of future kernels as well.

Willy Tarreau will be taking over the **2.6.27 kernel**, and he says he plans to make it the new "ultra-stable" replacement for 2.4 kernels. His intention is to give the remaining 2.4 users an incentive finally to upgrade to 2.6 kernels.

Paul Gortmaker will be taking over the **2.6.34 tree**, essentially for his employer **Wind River**, who has products that rely on that kernel.

And, **Andi Kleen** will be taking over the **2.6.35 tree**, also essentially for his employer **Intel**, who plans to release its own distribution based on that series.

So, that's how things stand. It's not clear how future stable series will come into existence, or who will be responsible for them, or how they'll be maintained. But, if the current situation is any indication, distribution maintainers may start assigning their engineers to become stable tree maintainers. Maybe that's the direction things have been going for a long time.

—ZACK BROWN

Get Green, with Brown!

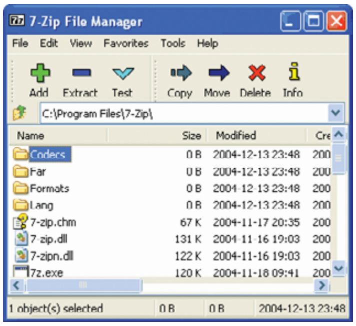
The folks at Recompute have taken the notion of "Going Green" to a whole new level. They've made computer cases out of recyclable cardboard. We had the pleasure of speaking with Recompute's Brenden Macaluso and took one of their computers for a test-drive. Here's what we found:

- The computers living inside the cardboard boxes are actually quite functional. Although they're not super-fast gaming machines, the computer options aren't just a bunch of low-end Atom machines.
- The cases feel sturdy. We were leery about using a computer case made of cardboard, but it didn't feel flimsy at all.
- Although a cardboard case doesn't make the computer internals any more recyclable, it does actually make it easier to recycle those innards. They literally rip right out.

There are many skeptics when it comes to the Recompute idea. Some see the cardboard case as a gimmick, and some think a computer wrapped in brown craft paper is a fire hazard. If you have questions about the Recompute computer, check out the FAQ on the Web site: **www.recomputepc.com**. For my full video review, check out our Web site: **www.linuxjournal.com/video/review-recompute-pc**.

—SHAWN POWERS

NON-LINUX FOSS



Name	Size	Modified	Cre
Codecs	0 B	2004-12-13 23:48	200
Far	0 B	2004-12-13 23:48	200
Formats	0 B	2004-12-13 23:48	200
Lang	0 B	2004-12-13 23:48	200
7-zip.chm	67 K	2004-11-17 20:35	200
7-zip.dll	131 K	2004-11-16 19:03	200
7-zipn.dll	122 K	2004-11-16 19:03	200
7z.exe	120 K	2004-11-18 09:41	200

For Linux users, file decompression tools are as close as an `apt-get` or `yum` away. For Windows users who want to extract more than ZIP files, it means installing a third-party application. That can mean paying for a program like WinRAR, or it can mean installing a program like 7-Zip.

7-Zip will decompress (and compress!) just about any compressed file you run across on the Internet. Sure, it supports its native .7z file format, but it doesn't force you to use that rather uncommon (but awesome) file format. It integrates nicely into the right-click context menu, and it's basically the only compression program a Windows user will ever need. Download it for free at **www.7-zip.org**.

—**SHAWN POWERS**

PINT-SIZE PPA PRIMER

Package management in Linux is great, but unfortunately, it comes with a few cons. Granted, most distributions keep all your software, not just system software like Apple and Microsoft, updated. The downside is that software packages aren't always the latest versions. Whatever is in the repository is what you get. Another frustration is when the software you want to install isn't in the distribution repositories at all.

Usually, it's possible to add software packages, even if they're not in the repos. For Red Hat-based systems, those are RPM files. For Debian-based, they're DEBs. Unfortunately, installing applications that way doesn't give you upgrades when they're available; you need to keep them updated yourself. Most package management systems also have the ability to add third-party repos, but those don't always have the packages you want.

Canonical has a feature in newer versions of Ubuntu that allow the best of both worlds. They're called PPAs (Personal Package Archives). Instead of distributing .deb files, developers simply can distribute their PPAs. With a PPA, the software is updated automatically along with being installed in the first place. While installing PPAs hopefully will become simpler, in the short term, they're still pretty easy to install. You just need to find the right PPA structure, usually given by the developers that support the idea. For example, to install the Mozilla Daily Build PPA, simply type:

```
sudo apt-add-repository ppa:ubuntu-mozilla-daily/ppa
```

Someday, installing a third-party application will be as easy in Linux as it is in Windows and Macintosh. With ideas like PPA repositories, however, your software will stay updated. And, that sounds P-P-Perfect to me.

—**SHAWN POWERS**

MAXIMUM CALCULUS WITH MAXIMA

We looked at Maxima in the February 2011 issue to do algebra and rearrange some equations. But those aren't the only tricks up Maxima's sleeve. This month, I describe how Maxima can help with differential equations, but I'm going to leave out some of the intermediate results to save some space.

A lot of science involves figuring out how systems change over time and what causes those changes. When you start looking at changes, and especially rates of change, that is essentially calculus. Calculus and rates of change also are linked to slopes of lines on graphs. When you plot data and find an equation that describes it, you can find the slope of the line by taking the derivative of the equation. Let's look at a falling object and see what theory has to say about it.

You should start by looking at how you get a derivative. Let's say you have the equation:

```
(%i1) f(x):= 2 + x^2;
                                            2
(%o1)                          f(x) := 2 + x
```

You would find the first derivative by calling the function diff, giving it the equation to differentiate along with the variable to differentiate by. So, you would write:

```
(%i2) answer:diff(f(x),x);
(%o2)                            2x
```

Maxima can do differentiation of expressions too. If you have a couple equations, you can derive their ratio with:

```
(%i3) g(x):= x^(1/2);
(%i4) ratio_diff:diff(g(x)/f(x),x);
                                        3/2
                     1           2 x
(%o4)         ----------------- - -----------
                          2          2    2
              2 sqrt(x) (x + 1)    (x + 1)
```

This might be a bit messy to work with, so you might want to refactor it to a more concise form:

```
(%i5) factor(ratio_diff);
                               2
                            3 x  - 1
(%o5)                - ------------------
                                    2    2
                        2 sqrt(x) (x + 1)
```

Maxima also can handle trigonometric functions, but there are lots of identities you can use to help simplify equations with trig functions in them. By default, Maxima does not try to apply these unless you specifically say so, using special functions. As an example, let's say you have the following equation:

```
(%i6) diff(sin(x)/(1 + cos(x)),x);
                               2
                            sin (x)         cos(x)
(%o6)                    ------------- + ----------
                                     2   cos(x) + 1
                         (cos(x) + 1)
(%i7) factor(%);
                              2          2
                           sin (x) + cos (x) + cos(x)
(%o7)                      --------------------------
                                                    2
                                 (cos(x) + 1)
```

That's still not very simple. If you then apply the function trigsimp, you can force Maxima to apply trigonometric simplification rules to the equation and see what you get:

```
(%i8) trigsimp(%);
                                1
(%o8)                    ----------
                         cos(x) + 1
```

You should be aware of some important caveats regarding how Maxima treats trig functions. The first is that `sin(x)^(-1)` is the reciprocal of sine, not arcsine. To get the arcsine, you would use `asin(x)`. The other is another trig simplification function, trigreduce. This function is used to reduce the powers of trig functions by using the multiple angle formulas. For example:

```
(%i9) trigsimp(cos(x)^2 + 2*sin(x)^2);
                          2
(%o9)                  sin (x) + 1
(%i10) trigreduce(cos(x)^2 + 2*sin(x)^2);
                  cos(2 x) + 1       1   cos(2 x)
(%o10)             ------------ + 2 (- - --------)
                        2            2       2
```

That may not look simpler than what you would get from trigsimp, but it is an easier form of the equation to use with other functions, like integration.

Maxima can apply the chain rule when doing a derivative. Say you have the equation:

```
(%i11) f(x):= x^3);
                                 3
(%o11)                  f(x) := x
(%i12) depends(x,u)$
(%i13) diff(f(x),u);
                          2 dx
(%o13)                 3 x  --
                            du
```

The line at `%i12` uses a new function, `depends`. This is a way of telling Maxima that x is a function of u, without explicitly defining a function describing this relationship. If you decide later that you want to define an actual equation for this relation, you can use:

```
(%i14) remove([x,u],dependency);
(%o14)                     done
(%i15) x:sin(u);
(%o15)                   sin(u)
(%i16) diff(f(x),u);
                               2
(%o16)             3 cos(u) sin (u)
```

Along the same lines, Maxima can handle implicit differentiation. Say you have the equation x^2 + y^2 = 25, and you want to find dy/dx. You need to use the depends function I just mentioned to handle this:

```
(%i17) eqn := x^2 + y^2 = 25;
                    2    2
(%o17)             y  + x  = 25
(%i18) depends(y,x);
(%o18)               [y(x)]
(%i19) deriv_of_eqn:diff(eqn,x);
                        dy
(%o19)              2 y -- + 2 x = 0
                        dx
(%i20) solve(deriv_of_eqn,'diff(y,x));
                       dy      x
(%o20)                [-- = - -]
                       dx      y
```

The other side of calculus is integration. The basic function to do that in Maxima is called `integrate`. This function can do both definite and indefinite integrals. Indefinite integrals are the symbolic form of integration you likely learned in school. For example:

```
(%i21) integrate(x^2,x);
                             3
                            x
(%o21)                      -
                            3
```

A definite integral actually is evaluated over an interval. This form of an integral can be visualized as the area under the curve defined by the equation you are integrating. To do definite integrals, simply add two arguments giving the start and end points of the interval:

```
(%i22) integrate(x^2,x,0,1);
                          1
(%o22)                    -
                          3
```

Putting all these techniques together, you can solve a differential equation for a given variable—for example, solve dy/dx = f(x) for y. You can do this by doing all the required algebra and calculus, but you don't really need to. Maxima has the very powerful function, `ode2`, which can do it in one step. Start with your equation:

```
(%i23) eq: 'diff(y,x) = sqrt(1/x^2 - 1/x^3);
                      dy        1    1
(%o23)                -- = sqrt(-- - --)
                      dx         2    3
                                x    x
(%i24) ode2(eq,y,x);
                                                       2
                          2                   2 sqrt(x  - x)
(%o24)    y = log(2 sqrt(x  - x) + 2 x - 1) - ------------- + %c
                                                     x
```

This one function call does the integration and the solve steps and gives you a final answer to the differential equation.

Let's say you're doing an experiment dropping a coin and timing how long it takes to fall. How do you know whether the times you are measuring actually make sense? Let's start with the most basic law: force = mass * acceleration.

The mass of the coin is a constant, so ignore that for now. The force is the force due to gravity, pulling the coin down to the ground, and the acceleration describes the coin's motion due to this force. The force due to gravity is a constant, at least here on Earth, and it depends linearly on the mass, so you can define the force as:

```
(%i1) force: mass * g;
(%o1)                g mass
```

The acceleration also is a constant, because both the mass and the force are constants. Acceleration is simply the rate of change of the velocity, and the velocity is the rate of change of the position, so you can set that up as:

```
(%i2) depends(y,t);
(%o2)                [y(t)]
(%i3) acceleration: 'diff('diff(y,t),t);
                       2
                      d y
(%o3)                 ---
                        2
                      dt
```

Putting it all together, you get:

```
(%i4) eq_of_motion: force = mass * acceleration;
                                          2
                                         d y
(%o4)                      g mass = mass ---
                                           2
                                         dt
```

```
(%i5) solve(eq_of_motion, y);
                              2
                             d y
(%o5)                       [--- = g]
                               2
                             dt
```

You can see right away that how fast an object falls doesn't depend on the mass at all. Galileo was right! The next step is to do some integrating and see what you end up with:

```
(%i6) integrate(%,t);
                 dy
(%o6)           [-- = g t + %c1]
                 dt
```

At this step, you would be able to find out the velocity (dy/dt) at time t. The additional term `%c1` is a constant of integration. In this case, you can see that it represents the initial velocity of your penny. One more round of integration gives this:

```
(%i7) integrate(%,t);
                   /             2
                   [ dy       g t
(%o7)            [I   -- dt = ---- + %c1 t + %c2]
                   ] dt        2
                   /
```

Now you can find the position, y, of your coin at any time, t. Again, a new constant of integration is introduced, `%c2`. In this case, you can see that this represents the starting height of your coin. But that's not what you were measuring. You were measuring how long it took the coin to drop a given distance. So you need to do a bit of rearranging. Because you are dropping your coin, you know that the start velocity is 0 (that is, `%c1=0`). You can rewrite things a little to make it a bit clearer:

```
(%i8) eqn: y = (g * t^2)/2 + y0;
                                   2
                                g t
(%o8)                 y = y0 + ----
                                 2
(%i9) solve(eqn,t);
                              y   y0                    y   y0
(%o9)     [t = - sqrt(2) sqrt(- - --), t = sqrt(2) sqrt(- - --)]
                              g   g                     g   g
```

There you go. You now have an equation for the time, given a height that your coin is dropping. With this theoretical relation under your belt, you can check to see whether gravity is working correctly in your local lab. If not, you should contact the Nobel committee straightaway.

This only scratches the surface of Maxima's capabilities in dealing with calculus and differential equations, but hopefully, this article gives you a starting point. Happy integrating.

—JOEY BERNARD

They Said It

I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones.
—Linus Torvalds

I have an ego the size of a small planet.
—Linus Torvalds

Do you pine for the days when men were men and wrote their own device drivers?
—Linus Torvalds and David Diamond

If you need more than 3 levels of indentation, you're screwed anyway, and should fix your program.
—Linus Torvalds

Some people have told me they don't think a fat penguin really embodies the grace of Linux, which just tells me they have never seen an angry penguin charging at them in excess of 100 mph. They'd be a lot more careful about what they say if they had.
—Linus Torvalds

LinuxJournal.com

This month, *Linux Journal* focuses on my favorite topic: Web development. We've compiled a fantastic collection of information here in the pages of this issue, and I look forward to learning a thing or two along with you.

With the recent release of Drupal 7, I am excited to focus my attention once again on making significant improvements to LinuxJournal.com, and I hope you'll check in from time to time to see what's new on-line. A new major Drupal release is a great excuse to take a look at things that could use improvement on our site, as well as add new features to improve the overall experience for our readers. With the numerous improvements in Drupal 7 beckoning, I can't wait to get started on what should be the best version of LinuxJournal.com yet.

Web development best practices are constantly evolving, and we all struggle to stay current or even ahead of the curve. *Linux Journal* and LinuxJournal.com are two important sources of the knowledge necessary to keep up with Web technology trends. Join us on-line at **www.linuxjournal.com/tag/web-development** to find a wealth of information in one place. See you in the comment queue?

—KATHERINE DRUCKMAN

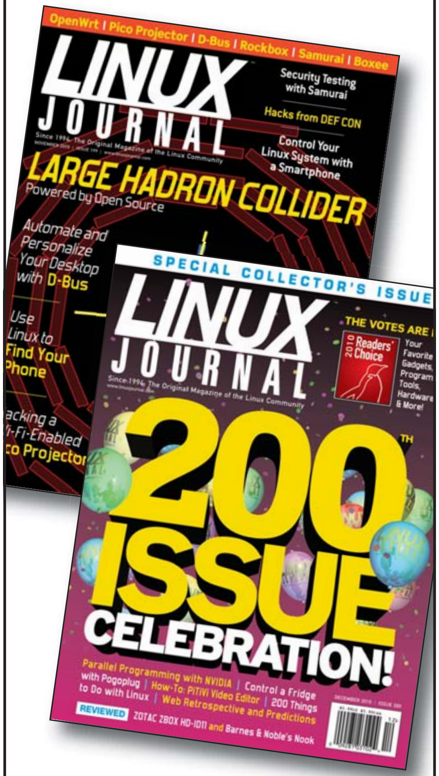
DROP YOUR DROPBOX AND SPARKLESHARE INSTEAD!

We love Dropbox here at *Linux Journal*. It's cross-platform, offers a decent free offering and generally "just works". It has some problems though. Dropbox is proprietary. Dropbox stores a copy of your data in its own data repositories. Dropbox is limited in size, especially with its free accounts.

Enter SparkleShare. SparkleShare is an open-source project that allows you to start a Dropbox-like service on your own. It's a very new project and needs time to mature, but the beta is promising. Also, because you run the server yourself, there are no limits to the amount of data you can store. It's also cross-platform and has some of the same sharing features offered by its proprietary brother.

Check out the early stages of SparkleShare at **www.sparkleshare.org**, and if you're a programmer, consider contributing. I'm excited for a stable alternative to Dropbox that I can host myself.

—SHAWN POWERS



PostgreSQL 9.0

Looking for a relational database? The latest version of PostgreSQL makes a great database even greater.

REUVEN M. LERNER

If you want to build a Web application, you're going to need a few pieces of software. You'll need a server operating system, such as Linux. You'll also need a server-side language and framework. Although I personally use Ruby on Rails nowadays, plenty of other good choices exist. For client-side interactions, you'll need to use a JavaScript framework, such as jQuery. And, to store information about (and for) your users, you'll need to select a persistent storage mechanism, otherwise known as a database.

Until just a few years ago, anyone using the term database almost always was referring to a relational database—that is, one based on linked two-dimensional tables, which could be queried using the relatively standard SQL language. Sure, there were a variety of other types of databases, but they weren't in widespread use or taken very seriously.

In the past few years, however, there has been tremendous growth in the use and mindshare of so-called NoSQL databases. There's not much in common between these various databases—from Redis to MongoDB to CouchDB to Hadoop—that offer the promise of greater scalability and flexibility.

Now, I'm not one to say these non-relational databases are bad. On the contrary, I've used several of them in my own projects and found them to be a perfect fit for certain needs. But for day-to-day use, I continue to find relational databases an excellent fit. And when I need a relational database, I always prefer to use PostgreSQL. Indeed, when I'm working on a Web application, PostgreSQL is the default option. Only if I see that it won't work well (for whatever reason) do I switch some or all of the project to use a different database.

Why do I love PostgreSQL so much? First, because it's rock solid. The developers have an almost obsessive dedication to the safety of the data you store in the database, and they do their best to ensure that data won't ever be deleted or corrupted. This doesn't mean such situations never happen, but they're quite rare. PostgreSQL not only tries to ensure the safety of your data, but it also provides you with a variety of tools to help you constrain the values that may be stored in the database, ensuring that anything stored is guaranteed to be valid.

Second, PostgreSQL offers a wealth of features that never cease to impress and amaze me. Whether it's the number of languages in which you can write server-side functions or the different ways you can define indexes, the clever use of MVCC (multi-version concurrency control) to avoid writers blocking readers or the ongoing statistical analysis that feeds into the query optimizer, PostgreSQL hasn't failed me in more than 15 years of day-to-day use.

Every time I use another relational database, I find myself wishing for one or more functions that PostgreSQL provides or thinking that PostgreSQL wouldn't even enter into certain problematic situations, because of its high-quality architecture. This doesn't mean PostgreSQL is perfect, but its mix of features has served me quite well, and I'm not the only one to feel this way.

This month, I want to look at PostgreSQL 9.0, which was released in late 2010. In particular, I want to consider what advantages it offers over other open-source databases (mainly MySQL). For people already using PostgreSQL, what new features does the 9.0 release bring to the table?

I won't describe installing PostgreSQL, simply because versions are available for all major Linux distributions. Downloading and compiling the source code is relatively straightforward—although doing so as the dedicated "postgres" user, rather than as root, will save you many headaches during the installation process—but I have found binaries to be just fine for my day-to-day needs.

Background

PostgreSQL is an open-source (BSD-licensed) relational database developed by a core group of developers working for a number of different companies. (Indeed, the rules of the core development group forbid more than a handful of core developers from working for the same company to ensure that no one organization controls development directly.) It grew out of the Postgres Project developed by Michael Stonebreaker at the University of California, Berkeley. Postgres itself was an attempt to improve upon the original Ingres database Stonebreaker had developed and commercialized.

Postgres used its own query language, known as QUEL, and had a number of advanced ideas, including many taken from the world of object-oriented programming. Several developers joined forces to take the Postgres database, replace QUEL with SQL, stabilize the code and release it under an open-source license. The first release of PostgreSQL, under its new name, came in 1996. Informally, a large number of PostgreSQL users and developers continue to call it Postgres, although that technically refers to Stonebreaker's project at Berkeley and not the current implementation.

One of the main goals of the PostgreSQL team has been to adhere to the SQL standard wherever possible. In addition, as I mentioned previously, PostgreSQL developers pride themselves on a rock-solid implementation, using the standard ACID (atomicity-consistency-isolation-durability) paradigm for database storage under all circumstances. They further try to balance between a powerful set of features and configurable options and overwhelming people who don't want to be full-time database administrators.

All PostgreSQL storage is transactional, using a system known as MVCC (multi-version concurrency control). MVCC, which also is used in Oracle and CouchDB, largely avoids conflicts between readers and writers, ensuring that neither rows nor tables need to be locked under most circumstances. MVCC does have the side effect of keeping "dead rows" around in the database, which occasionally need to be "vacuumed", similar to garbage collection in many programming languages. For many years, vacuuming needed to be done manually, typically by running the `vacuum` command from a cron job. Nowadays, the autovacuum dæmon runs in the background, marking space as reusable when added or updated rows reach a configurable threshold.

Vacuum also can be run in "analyze" mode, in which case it examines tables and indexes, updating the statistics that are used when planning queries. This is one of the reasons PostgreSQL is able to operate so efficiently, even when handed complex queries. By keeping an up-to-date description of data stored in each table, it can make an intelligent decision regarding how to optimize the query. If the guesses turn out to be wrong for your particular needs, you can configure some configuration parameters.

PostgreSQL offers strong support for foreign keys, making it possible to ensure that connections between tables exist and work in the right way. That is, you can define a People table and then an Addresses table that refers to it:

```
CREATE TABLE People (
id SERIAL,
first_name TEXT,
last_name TEXT,
email_address TEXT,
PRIMARY KEY(id)
);

CREATE TABLE Addresses (
id SERIAL,
person_id INTEGER REFERENCES People,
address TEXT,
PRIMARY KEY(id)
);
```

Now, let's try to insert a new record into the Addresses table, without inserting anything into People first:

```
INSERT INTO Addresses (person_id) VALUES (5);
```

This results in the following:

```
ERROR:  insert or update on table "addresses" violates foreign key
 ➥constraint "addresses_person_id_fkey"
DETAIL:  Key (person_id)=(5) is not present in table "people".
```

Because I have defined person_id as a foreign key, PostgreSQL won't let me have an invalid value there, period. PostgreSQL also will refuse to let you remove a record from the People table if there are rows in the Addresses table referring to it. You can override such restrictions with the CASCADE keyword, but the database never will remove things on its own without your explicit approval.

You can, of course, also ensure that e-mail addresses are unique in your People table:

```
CREATE UNIQUE INDEX people_email_idx ON People(email_address);
```

But wait, what if someone enters an e-mail address in capital letters and then enters the same e-mail address in lowercase letters? You can ensure uniqueness by taking advantage of one of my favorite PostgreSQL features, a functional index:

```
CREATE UNIQUE INDEX people_email_idx ON
 ➥People(lower(email_address));
```

Now PostgreSQL will ensure its uniqueness, regardless of the case. This feature also comes in handy if you're trying to index a column containing long text strings. You can index only the first 1,000 characters:

```
CREATE UNIQUE INDEX people_email_idx ON
 ➥People(substring(email_address, 1, 1000));
```

Finally, another powerful feature is CTEs (common table expressions). If you often find yourself performing a query, putting the results in a temporary table and then querying that temp table, CTEs almost certainly are something you'll want to use. You basically describe the temporary table query, give it a name using the "with" statement, and then query that pseudo table. For example:

```
WITH Mini_users
AS (SELECT id * 2 AS id2, email FROM Users)
SELECT id2 FROM Mini_users;
```

In the example above, I create a new temporary table named mini_users and define it, and then query it as if it already existed. I already have found CTEs to be quite useful in simplifying complex queries. PostgreSQL also makes it easy to define recursive CTEs, allowing you to work with hierarchical data with a single, simple query.

Advanced Features

One of the features I have grown to love over the years is the ability to write my own server-side functions—analogous to "stored procedures" in other databases—in a variety of languages. Most of the time, I use the built-in Pl/PgSQL language, which marries SQL queries with a procedural language. But in cases when I need a bit more power, or want to use existing libraries, I can write functions in PL/Perl, PL/Python or a number of other languages. Regardless of what language I use, these functions integrate seamlessly into the database, working no differently from built-in functions and allowing me to process data inside the database, rather than in my application.

Over time, these server-side languages have become increasingly sophisticated, and although the Pl/PgSQL syntax is not the most modern or expressive, it is easy to learn and handles a large number of cases easily and quickly. I've seen a wide variety of uses for server-side functions, from encapsulating most application logic in such functions to handling specific data-manipulation routines that don't logically belong in the application layer.

One of my favorite uses for functions is in triggers—actions the database executes automatically when a certain action occurs. For example, I once worked on a project into which we wanted to insert a URL, but in which we also wanted to have a (separate) column containing the hostname for that URL. I wrote a function that used regular expressions to retrieve the hostname and then inserted the hostname automatically into the appropriate column. From the application's perspective, it was able to insert a URL but then retrieve either a URL or a hostname. Triggers can be used to perform all sorts of actions before or after an insert, delete or update query is performed on a table.

One of the many advances in the 9.0 release was the improvement of "window functions", functions introduced in 8.4 that make it possible to create aggregations (such as sum and average) over a portion of other rows in the table, rather than over the entire table. Thus, you could calculate the difference between someone's salary and other salaries in that person's department, or rank the number of citations a department has in a bibliographic index, or find which was the longest-running action in each controller, within an MVC Web application. I've only started to work with windowing functions, but the combination of built-in functionality from 8.4, plus some additions to 9.0 that make it easy to peek at preceding and following rows, lead me to believe this is a particularly powerful feature.

PostgreSQL 9.0

The above features are nice improvements, but they're icing on the cake when it comes to reasons for a full-version upgrade to 9.0.

First, this version makes it fairly simple to upgrade. Earlier versions required dumping the database to disk, upgrading the server, and then restoring the database from its dumped form. This might not be a big problem for small databases that can afford to be off-line for a while, but it is a major issue for large sites that cannot afford such a long shutdown period.

The core PostgreSQL developers heard the criticism and have responded with pg_upgrade. Now, pg_upgrade still is considered to be somewhat experimental, and it hasn't been given the official seal of approval by the development team, so it is placed in the contrib directory, rather than in any official location. However, pg_upgrade, which allows you to upgrade your PostgreSQL server without a dump or restore, is certainly one of the major innovations and accomplishments of 9.0, and it has been greeted with great excitement by people who were too nervous or busy to upgrade previously.

Another major feature—and perhaps the most impressive accomplishment of this version—is what's known as "hot streaming replication". This feature is actually the combination of two different features, which work together to make for an amazing new backup and high-availability system.

The basic idea is as follows. When you commit a transaction to PostgreSQL, it doesn't immediately update the tables on disk. Rather, it writes a record of the transaction to a separate "write-ahead log", or WAL, describing the change that should be made to the database. Only after enough WALs have accumulated does PostgreSQL actually update the data.

Starting in version 8.4, you could copy the WALs from one PostgreSQL server to another, typically using a program like rsync. When the WALs appeared on the second server, they were read into that system. If the first server were ever to go down, the second server could be put into active mode, picking up from where the first server had left off.

Although this was better than nothing, it had at least two problems. Not only was transferring the WALs via "log shipping" far from an ideal mechanism, but the secondary server could not be used while it was receiving data from the primary server.

Both of these problems have gone away in PostgreSQL 9.0, providing a robust and easy-to-configure master-slave mechanism for high availability. WALs now can be transferred from the primary server to the secondary using the network by configuring the secondary as a special kind of client program. As each transaction occurs, the WAL can be sent over the network to one or more remote servers, allowing them to be synchronized almost immediately with the primary. The other improvement is that the secondary server can answer read-only queries even while it's receiving data from the primary.

Hot streaming replication, as the combination of these two features is known, doesn't cover all situations. For example, I recently was working with a company that has multiple PostgreSQL servers whose data it would like to put onto a secondary server. For such purposes, hot streaming replication won't work. Fortunately, there are other solutions, such as Slony, that might be a better fit for the company's needs. But, for many sites that want to ensure their servers won't go down, this solution is more than adequate.

Conclusion

Am I gushing? Yes, absolutely. There are very few pieces of software that give me such pleasure to use and on which I feel I can utterly depend, and PostgreSQL is a select member of that club. It keeps my data safe, offers me a huge number of features, allows me to extend the database in whatever ways I want and need, and it gives me excellent performance with a modest time investment. If you're thinking of switching to an open-source database, or if you have already been using one and want to try something powerful but still rather easy to use, you would do well to try PostgreSQL. Especially with the latest features added in version 9.0, I'm sure you'll find something to like.■

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 PostgreSQL is **www.postgresql.org**. From that site, you can download the software, subscribe to a number of (heavily trafficked and extremely friendly) e-mail lists and read through the documentation. There is also a #postgresql IRC channel on Freenode that is generally populated and available for answering questions.

One of the best-known commercial PostgreSQL support and product companies, EnterpriseDB, provides binary packages for a number of operating systems at its Web site, **enterprisedb.com**. It also offers a number of video and audio podcasts and screencasts, including those by PostgreSQL core contributor Bruce Momjian, which are of high quality.

Finally, two excellent books about PostgreSQL recently were published by Packt Press, filling in a long gap since the excellent (but outdated) *PostgreSQL, 2nd edition* was written by Korry Douglas a number of years ago. The two new books are *PostgreSQL 9 Administration Cookbook* by Simon Riggs and Hannu Krosing, and *PostgreSQL 9.0 High Performance* by Gregory Smith. I have learned a great deal from both books, and although the latter book spends a bit more time than I might have liked on hardware, it also taught me that even a software guy like myself occasionally needs to consider the physical side of software and database design. If you use PostgreSQL on a regular basis, I strongly recommend these books.

DAVE TAYLOR

Mad Libs Generator, Part II

Choosing only the interesting words out of a text passage for a *Mad Libs*-style game proves to be a darn difficult task within a shell script, but Dave's up to the challenge.

Last month, we dug in to creating a *Mad Libs* generator—a program that you could give a snippet of English prose, and then it would select words randomly and replace them with their parts of speech, so you could have friends or family create their own amusing alternatives.

So, instead of "the quick brown fox jumping over that lazy dog", it could be "the quick ((adjective)) fox jumps over the ((adjective)) dog", for example.

The problem is that selecting random words from a sentence also can produce something far more boring, like "((definite article)) quick brown fox jumps over ((definite article)) lazy dog".

This month, I take that random word-selection tool and add some smarts so that it is biased toward longer words and words that are nouns or adjectives.

This month, I take that random word-selection tool and add some smarts so that it is biased toward longer words and words that are nouns or adjectives.

Selecting Words by Length

Last month, you'll recall that our script had a word-selection snippet that looked like this:

```
while read sentence ; do
  for word in $sentence ; do
    if [ $(( $RANDOM % $density )) -eq 1 ] ; then
      echo "(($word))"
    else
      echo $word
    fi
  done
```

Where we'll need to expand the code is within the conditional that currently just puts the word in parentheses. The first step is to analyze length: if the word is three or less letters long, we'll be much less likely to select it:

```
if [ $(( $RANDOM % $density )) -eq 1 ] ; then
  length=$(/bin/echo -n $word | wc -c | sed 's/ //g')
  if [ $length -lt 4 -a $(( $RANDOM % 2 )) -eq 1 ] ; then
    echo \{$word\}     # too short
  else
    echo "(($word))"
  fi
else
```

This works pretty well—actually, every time a word is selected, its length is checked. Words less than four letters long have a 50% chance of being ignored. With a simple input sample, here's what we get:

```
{the} ((quick)) brown fox jumped ((over)) the lazy black dog
```

It's still not great, but at least it recognized that "the" wasn't interesting due to length. I'm still not entirely satisfied with which words it chooses to substitute, but let's move on to the second part of this project, testing part of speech, and come back to the selection criteria later.

Figuring Out the Part of Speech

The core code for this already was presented last month, utilizing Princeton's handy WordNet, so here it is:

```
pos="$(curl --silent "$dictionary$word" | grep '<h3>' | head -1 \
  | tr '[:upper:]' '[:lower:]' | sed 's/<h3>//;s/<\/h3>//')"
if [ ! -z "$(echo $pos | grep "not return any results")" ] ; then
  echo \[$word\]    # failed to figure out part of speech
else
  echo "((${word}:$pos))"
fi
```

Notice that we have to worry about failed lookups. Some words just aren't found in the

WordNet dictionary, and we need to be prepared. I'll tie these together, as written, and here's what we get as an output:

```
Note: {} = too short, [] = POS undefined
((I:noun)) {am} {by} ((birth:noun)) {a} Genovese, and
{my} family {is} one of the most ((distinguished:verb))
of that ((republic:noun))
```

As the header reminds us, at this point, we're denoting words selected but skipped because they're too short with {} and those that have an undefined part of speech with [].

I've also changed the word replacement density factor to have more words tested. As you can see, most of the words in our sample input are now evaluated one way or the other.

Now, let's add a test so that only nouns or adjectives are eligible for substitution too:

```
if [ ! -z "$(echo $pos | grep "not return")" ] ; then
  echo \[$word\]        # failed to figure POS
else
  if [ -n "$(echo $pos | grep -E '(noun|adjective)')" ] ; then
    echo "((${word}:$pos))"
  else
    echo "<${word}:$pos>"
  fi
fi
```

I'll give it that same first sentence to Mary Shelley's *Frankenstein*, and let's see what transpires:

```
Note: {} = too short, [] = POS undefined, <> = uninteresting POS
I {am} <by:adverb> birth {a} Genovese, [and] my
family ((is:noun)) {one} {of} {the} ((most:adjective))
<distinguished:verb> {of} [that] ((republic:noun))
```

We're definitely getting there, but I think we still need to add something to the selection criteria—something that will help us produce more interesting *Mad Libs*.

But, let's leave that for next month as we've already dug through a lot of code in this column.■

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.

MICK BAUER

Interview with a Ninja, Part II

Mick continues chatting with a figurative and literal hacker ninja.

Last month, I introduced my good friend Ninja G, an accomplished professional penetration tester for a large financial institution (who also happens to be a practitioner of ancient martial arts developed by ninjas). We talked about his daily activities, his unusual career path, his perspectives on WLAN and Linux security, and the uncanny preoccupation among Ninja G and his coworkers with ninjutsu.

This month, I wrap up the interview. I think this installment is every bit as wide-ranging, thought-provoking and entertaining as part I. I hope you agree!

MB: I've got to ask a somewhat selfish question, because so much of my own career has involved firewalls. In the age of Web applications, where the browser is the platform and so much of the world's business is conducted over TCP ports 80 and 443, do traditional layer 3 firewalls still have a useful role?

NG: I believe so; however, I think most people would agree that a traditional (state-aware) firewall by itself isn't a complete solution. It is only the first of many layers of defense. Intrusion detection (and prevention) systems, system hardening, secure centralized logging and application-aware firewalls should supplement a state-aware firewall to help round out areas where it would otherwise be lacking.

MB: What's your opinion of the new generation of (Web) application firewalls that can be trained to block exceptions to "expected" Web application behavior? Are we finally reaching a point in this class of technology where it's possible to create a complete enough notion of "expected" application or network behavior to reduce the very high rate of false positives we've traditionally associated with this approach?

NG: I don't think I ever would recommend a Web application firewall in lieu of Web application hardening. A lot of the problems that are being caught by Web application firewalls usually can be solved with proper tainted variable checks on *all* user-supplied input. This includes the Web browser's user-agent string (which may find its way into log files), all cookies and, of course, GET/POST data, AJAX requests and so on.

In an ideal world, the Web application should provide real-time logging of things like cross-site script, SQL injection and brute-force attempts, but that is more often the exception than the rule. Often organizations rely on (SSL breaking) Web application firewalls to obtain the same types of data; however, you have to be *very* confident in low false positive rates when deploying in any inline device that actually prevents malicious traffic from reaching the target host, or else you could be creating your own denial-of-service condition for what could be business-critical applications. One should keep in mind that even the top-of-the-line Web application firewalls occasionally will generate false positives and plan one's use of them accordingly.

MB: Some people are very skeptical not only about the worth of pen testing, but also especially of the trustworthiness of anybody who has amassed experience in system cracking, which they (the skeptics) seem to think is inherently corrupting. But my own experience has been that overall, security researchers and penetration testers, precisely because they understand so well how easy it is to get caught, tend to be very responsible.

And, I'd unhesitatingly put you in that category. I think of you as a highly ethical and trustworthy person. So, what's your take on this? Are security ninjas generally like you, or is criminal recidivism (à la Alberto Gonzales) rampant?

NG: Of course, I can't speak for anyone else but myself. People have their own unique motivations, passions or "demons" that drive them along this path. I can say it has been my experience that most people who perform penetration testing for a living generally wouldn't risk their livelihood by hacking illegally. For a person like myself, I view my work almost like a flip-side version of the Buddhist concept of Right Livelihood, which states that people should try to find an occupation where they won't sacrifice their moral code. Except for me, I found a job where I get to express myself creatively in a criminal-like way, yet with the most ethical of outcomes.

At the end of testing, I create detailed write-ups of all of my security findings, write proof-of-concept code, help vendors understand and re-create these issues, and assign a risk rating to help others understand their severity—like a lot of the other "unsung heroes" of this field, who never release their exploit code and provide full details directly to vendors and code writers. I would like to think I am helping improve the overall security of not only the company I work for, but also indirectly helping all the other companies who use the same systems, services and products.

MB: It's easy for someone in my position (and that of my readers, I hope!) to see the value of your methods and your reports. I don't think "unsung hero" is any exaggeration, having seen firsthand how tangibly things tend to be improved after an unfavorable penetration test. But how do vendors and developers react to your work? How do you get them past what I'm sure tends to be an initially defensive reaction?

NG: Yes, you're correct. The typical *initial* response is to view the request for a penetration test as simply a check-box item on a list somewhere, which may or may not prove any sort of real due diligence. My assumption is that this is their usual experience with dealing with other companies with limited information security personnel and budgets.

Once they actually start hearing about high- or critical-severity security vulnerabilities, they almost always shift to a defensive position. At this point, I give them my (somewhat canned) speech about how they are in essence getting weeks of free security consulting. I assure them that I will share the details of all security findings completely and recommend steps for remediation where appropriate.

It is usually around this point when they realize that although their experience may be painful or humbling, they really are getting something for nothing, and far better to receive the information in *this* way, rather than having the same issues discovered by others who may not give the vendor *any* sort of warning before releasing the information to the general public.

MB: In my own consulting work years ago, now and then I'd be called on to do port scanning or security scanning (though not actual penetration testing) of live, production systems, and I always found that nerve-wracking. Have you ever inadvertently (or, come to think of it, intentionally) disrupted an important production system?

NG: I once took down a large group of firewalls simply by port scanning them. This was completely inadvertent, as I didn't know that they had been configured to effectively "turn themselves off" when they detected heavy port scanning. (?!) This caused a large production outage (and several people were howling for my blood), yet the configuration was beyond foolish, so by helping to "correct" the issue, I managed to escape unscathed.

I have never intentionally disrupted an important production system. Even in my lawless years, the goal was never to "take down" a system. I viewed that action as providing a *very* clear sign that an intrusion had occurred. The end result would be one less system to explore, which I viewed as sub-optimal.

MB: There still seems to be tension between two camps in information security: those of us who self-identify as hackers and those who don't. Yet attendance at Black Hat Briefings, DEFCON and other hacker conventions continues to balloon. Do you think the "hackers vs. suits" situation is getting better, or are we simply gaining numeric superiority?

NG: I would say that there was *much* more tension a decade or two ago, back when the corporate world wouldn't dare ever send someone off to a "hacker conference", let alone pay for it. Now conferences such as Black Hat and DEFCON are considered valued sources of information for those defending rather than attacking corporate resources.

Personally, I think there is a rather new element at hacker cons that may soon upset the delicate "hacker/suit" ecosystem: the US military—and I'm not talking about their old guys, I'm talking a new wave of young military. So don't be surprised in the coming years if that guy sitting next to you at DEFCON happens to be a SIGINT analyst for the US Navy.

MB: I've already had that type of experience, and agree with your observation about the military (which, I supposed, tends to be a youthful demographic, so maybe this isn't too surprising). DEFCON's "Spot the Fed" contest stopped being easy ages ago!

I think I'd like to wrap up with some more or less random questions—feel free to answer as tersely or completely as you like and, of course, omit names to protect whomever you like!

MB: What's the stupidest design decision you've ever seen?

NG: Okay, this one goes back way over a decade, so I feel comfortable discussing it. I once saw a large financial organization solve their backup issues by installing FIDDI interfaces in all of their UNIX servers. They dual-homed all of the machines into one large FIDDI ring and then kicked off all of the backups via rhost. I rooted *one* backup server in the FIDDI ring and ended up owning the entire back office settlement environment across the globe in minutes, due to rhost trusting the backup server. Keep in mind these hosts' primary interfaces were segregated into protected zones by firewalls, and access was limited into each zone based on your job function.

Epic fail.

MB: What's the coolest security control you've encountered lately?

NG: People are probably sick of hearing about it, but DNSSEC is actually pretty cool, especially when you factor in the potential impact associated with traditional DNS. (Yes, I have drunk Dan Kaminsky's Kool-Aid!)

MB: What's your favorite (non-secret) ninja weapon or fighting technique?

NG: Hmmm...for this question, I believe a little bit of ninja lore is necessary. First, forget everything you know about ninjas being the bad guys dressed head to toe in black uniforms; rather, they were often highly skilled martial artists acting covertly in enemy territory much like a modern-day spy. The worst possible fate would be to be caught, so contrary to all of those bad ninja movies of the 1980s, escape was always *way* more desirable than fighting.

So in the study of ninjutsu, you will encounter things like the Santo Tonko kata (forms of the escaping rat), which includes techniques for escaping grabs, the use of eye-blinding powders and using small thrown objects (such as shuriken) to dissuade continued attack or pursuit. Nowadays, a small tactical flashlight can be used instead of the irritating blinding power, and as shuriken have been outlawed in most states, a pocket full of loose change does the same thing when unexpectedly tossed into the face of a would-be attacker. "Oh! These are a few of my favorite things...!"

In one particular situation, I was greatly outnumbered, so I decided I needed to even the odds a bit. Using a SureFire E2E flashlight, I blinded the advancing "front row" only to have my "fighting with light" completely misidentified as a police tactic. Next thing I heard was, "This guy is a cop!" and everyone scattered. *That* alone was worth the price of the flashlight.

So rather than doing something silly or dangerous (like arming your loved ones), instead get them an inexpensive high lumen output flashlight. If their attacker disarms them, it isn't anything that could really be used against them. Plus, you would be surprised how often a flashlight comes in handy, even when you aren't being jumped by a large group of people.

MB: Which is harder, being stealthful in meat-space or being stealthful in cyberspace?

NG: If you understand both environments, then one really isn't more difficult to manage than the other. These are not set in stone, but I generally think of four basic principles that apply fairly equally to both physical and cyberspace stealth considerations.

The first principle would be disguise. In physical space, this includes concepts, such as wearing appropriate clothing to blend in with the rest of the population in the area, and the use of multiuse, hidden or improvised tools/weapons. In cyberspace, this would include avoiding the generation of network traffic that would be typically associated with either reconnaissance or attack. Instead, the stealthy attacker would choose to generate completely "normal" network traffic that could *also* be used to glean useful information.

A good modern example of this would be the tool named FOCA, which uses search engines to identify the location of documents that commonly contain metadata. The tool downloads these files a few at a time and then rips out the metadata and presents the extracted data. All of the generated Web traffic is 100% normal. All of the documents downloaded were *intended* to be downloaded. Unless a company is militant about scrubbing metadata, it isn't unusual to find about one file out of 100 that actually contains something useful enough to be leveraged in further attacks, such as an employee name, e-mail address or user ID. With larger companies, it is not uncommon to discover thousands of metadata-laden files available for download.

The second principle would be distraction (or attention control). In physical space, there are lots of variations here, as the human mind is *very* limited. We may claim to multitask, but the reality is that we can think of only one thing at a time. All you have to do is fill the mind with something interesting, and it will miss everything else. A good example of this can be found on-line by Googling Daniel J. Simons' video titled "selective attention test". In cyberspace, the same effect can be achieved by intentionally generating known malicious traffic to create overt "noise" in which to hide. Tools like snot and stick were designed to do exactly this sort of thing.

The third principle would be exhaustion (or frustration). In physical space, a modern analog would be to repeatedly set off a building's burglar alarm. The first time it goes off it usually will get all sorts of attention. The police will show up first, then eventually, the building manager will arrive on-site. After about the fifth or sixth time (in the same night) of the alarm going off with no visible signs of attempted forced entry, that alarm is going to be turned off until morning and a repair call will be made to the burglar alarm vendor.

The exact same thing happens in cyberspace with intrusion detection systems that constantly generate alarms that seem to be false positives. A security-conscious person will put up with an interruption or two in the middle of the night, but nobody can tolerate unwarranted nonstop interruptions. Discover this "breaking point" limit, and you can help "tune" remote intrusion detection systems to be more "friendly" toward your future stealthy endeavors.

The fourth principle I would call fu-sui (wind and water), but most people know it as feng shui. This is intentionally selecting the most advantageous positioning, terrain, weather or timing. In physical space, this could be something as simple as choosing to attack at dawn or dusk with the sun low in the sky behind you. Your enemy will be staring directly into the sun allowing you to hide in bright glare. Another example would be to choose a foggy night to hide in the darkness and mist. In cyberspace, this could include performing your activities during holidays celebrated by your target, choosing late hours when monitoring personnel may be more scarce or choosing peak traffic hours if you are trying to "blend in" with normal activities.

These principles are not mutually exclusive; in fact, it is often better to blend them together when creating your plan of attack.

MB: When does the fun of Brute Force trump the righteousness of the Elegant Solution?

NG: Wow, that definitely applies to both penetration testing and ninjutsu: "When it is the quickest and most direct solution to the problem."

MB: Who's more elite, pirates or ninjas, and why?

NG: Wow, that *is* the eternal debate. Here is my take.

If you research the origin of each of the nine martial arts that I study, you will find that one originally was used as a naval military art, and as a result, the body positions and movements were designed to be used on ships that were constantly rocking and slippery. Even to this day, techniques from this school are sometimes practiced using a boat oar as a weapon. This means that historically, some ninjas were pirates; however, I seriously doubt that the contrary was ever true. I'm sure this won't surprise you, but I vote ninjas!

MB: And on that lighthearted note, I'll sadly but gratefully wrap up what I hope, dear readers, you agree was a fun and informative conversation. Thanks so much for playing, Ninja G!■

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

KYLE RANKIN

Your Own Personal Server: DNS

Why let registrars have all the fun? Learn how to set up your own DNS server completely under your own control.

In this day and age, it's simple and popular to have someone else change your oil, grow your vegetables, remodel your house and host your services. However, I'd argue that it's far more rewarding, educational and not very difficult to manage these things yourself. This column is the second in a series about how to manage your own services on your own server. In the first column, I discussed how to make sure your home network is ready to host your own services. In this column, I start to get into the meat of the topic and discuss the first service you can (and should) set up at home: DNS.

A Short Primer on DNS

DNS (Domain Name System) is a system you use every day and one on which the Internet heavily depends. Every server (including your own) that has a presence on the Internet should have a public IP address. Since last month's column, you should have your home network set up for your server and have at least one public IP (hopefully static) you can use. It's true that all you really need to host many services on the Internet is an IP address; however, in practice, there are only so many IP addresses (like phone numbers) that the average person is going to commit to memory. As IPv6 becomes commonplace, this will be even more true. DNS allows you to register a domain name and associate individual host names (like www.example.com and mail.example.com) to IP addresses.

For instance, how many of you (besides you, Katherine) have www.linuxjournal.com's IP address memorized? If you did want to know the IP address, all you would need to do is perform a simple nslookup command:

```
$ nslookup www.linuxjournal.com
Server:         192.168.0.1
Address:        192.168.0.1#53

Non-authoritative answer:
Name:   www.linuxjournal.com
Address: 76.74.252.198
```

In this example, the first bit of output tells me that I'm getting this answer from a DNS server at 192.168.0.1 (my own personal DNS server) and that the IP address for www.linuxjournal.com is currently 76.75.252.198. There isn't enough space in this column to describe everything that happened to allow me to get that IP address, but essentially, my DNS server asked other DNS servers on the Internet for this IP address and was subsequently redirected to more and more DNS servers until it finally found the one that knew the answer. If you are interested in more detail on how this works, books like *DNS and BIND* do a good job of explaining it, or from the command line, you could run `dig www.linuxjournal.com +trace`.

Your Own DNS Server

DNS seems like a complicated service, yet it's relatively simple to set up a DNS server of your own. Now, there are a number of different DNS server software from which to choose that are easier to configure or that have fancy database back ends, but for this article, I'm going to choose the old standby, BIND. Although it's not as simple as other DNS servers, it isn't so bad, once you get the hang of it.

BIND should be packaged for most major distributions; however, there are slight differences in how each distribution packages BIND. For instance, under Red Hat, you install the bind package, but under Debian-based systems (like Ubuntu), you install bind9. Red Hat stores its core BIND configuration file at /etc/named.conf and all its zone files (files that contain name→IP address mappings for a domain, such as example.org, a subdomain, such as ny.example.org, or possibly both) under /var/named, while Debian-based systems put named.conf and any zone files under the /etc/bind/ directory. Even the init script is different on both systems: Red Hat uses /etc/init.d/named, and Debian-based systems use /etc/init.d/bind9. Once you get past the differences, however, the syntax inside the files should be similar. Just to simplify things, I'm going to base the rest of this article off a standard Ubuntu 10.04 LTS server, so we have some sort of baseline. If you use a different distribution, however, it shouldn't be too difficult to adapt these instructions to the different file paths.

Once BIND is installed on the system, the package should create a basic named.conf file and all of the base directories. In the case of this sample Ubuntu system, the default named.conf actually will be set up to act as a caching name server. So, out of the box you should be able to point to this server with other hosts on your network, and it will be able to resolve other domains on the Internet just like with your ISP's DNS server. In this case, though, we want to create a DNS master.

Master DNS Configuration

A DNS master contains its own zone files that have name→IP address mappings, and it doesn't have to consult any other source to answer queries for those names. By contrast, a DNS slave is configured to load all of its zone configurations from a DNS master. Any future changes are made on the master and propagate to each of the slaves. Any individual BIND instance acts as a DNS master, a DNS slave or a caching name server, or all three at the same time (although it can be a master or a slave only to any individual zone, not both).

For this example, let's set up a DNS master for example.org, and this master will have the following records:

- ns1.example.org, which points to 123.12.34.56 (the public IP of the name server itself).
- example.org, which points to 123.12.34.57.
- www.example.org also points to 123.12.34.57.

To start, I create the zone file at /etc/bind/db.example.org (remember Red Hat stores these zones in a different places) and put the following information in it:

```
;
; BIND data file for example.org
;
$TTL 4h
@  IN  SOA ns1.example.org. root.example.org. (
             2           ; Serial
             604800      ; Refresh
             86400       ; Retry
             2419200     ; Expire
             604800 )    ; Negative Cache TTL
;
@       IN NS   ns1.example.org.
@       IN A    123.12.34.57
www     IN A    123.12.34.57
ns1     IN A    123.12.34.56
```

Make sure this file has similar permissions to the other zone files you find in the /etc/bind directory. The first non-comment line in the file sets the TTL or Time To Live, the default time in which a remote DNS server will cache any answers it gets from your DNS server before it will ask it again. The value you put here will help determine how fast changes you make will propagate. BIND accepts seconds in this field, or you can use shorthand values like 1d for one day, 4h for four hours or 20m for 20 minutes. I set the TTL to four hours here; however, if you make frequent changes to your records (or know you are going to soon), you may want to make the TTL shorter. On the other hand, if you find you hardly ever change these values, you might want to bump up the TTL to a day to reduce load on your DNS server.

Something to note is that zone files use semicolons not hashes at the beginning of a line for comments. A common mistake is to put hashes in a zone file to make a comment, reload BIND and then wonder why your changes didn't take. When BIND sees a mistake like that, it just skips that particular zone.

To keep things simple, I'm going to skip the Retry, Refresh and other values here—just keep them with these defaults unless you know what you are doing. The Serial line is for DNS slaves, which I discuss later. Below those values, however, you'll see the syntax I used to define the different records:

```
@       IN NS   ns1.example.org.
@       IN A    123.12.34.57
www     IN A    123.12.34.57
ns1     IN A    123.12.34.56
```

The first record starts with @, which means it is a record for example.org itself. In this case, it is an NS record that defines the hostname I'm going to use for my name server. You can use any hostname you control here (including hostnames on a different domain, actually), but one popular convention is to use hostnames like ns1 and ns2 for the first and second name servers. The second record begins with an @ as well, only in this case, it's an A record. An A record is a fundamental DNS record that maps a hostname (like www) to an IP address (like 123.12.34.57). In this case, because the record starts with @, I am setting the IP address for example.org itself. The next two lines define two more A records, one for www.example.org and one for ns1.example.org. It's important if you used a name within this same domain for your name server (like ns1.example.org) that you be sure to add an A record so that it has an IP address.

Now that I have created my zone, next I need to modify the /etc/bind/named.conf file and add a new section at the end of the file to point to the /etc/bind/db.example.org file I just created:

```
zone "example.org" {
        type master;
        file "/etc/bind/db.example.org";
};
```

After the file is changed, I reload BIND, and I should be able to send DNS requests to my new DNS server:

```
$ sudo /etc/init.d/bind9 reload
* Reloading domain name server... bind [OK]

$ nslookup www.example.org localhost
Server:          localhost
Address:         127.0.0.1#53

Name:   www.example.org
Address: 123.12.34.56
```

If there is a problem with the BIND reload, it should tell you on the command line. Otherwise, if it still doesn't work, you may have to look in your syslog file (/var/log/syslog on Debian-based systems and /var/log/messages on Red Hat) for clues.

Slave DNS Configuration

Many registrars on the Internet require that any domain you register have at least two DNS servers configured with it. It's a good practice to have, because if you have a single DNS server and it goes down, it effectively will make all your servers under that domain inaccessible. This means you need to set up a second DNS server on a different IP, ideally on a different network, or have a friend with a DNS server act as a slave to your master DNS server. In either case, it's a relatively simple process. Let's say that my second DNS server is going to be at the IP address 98.76.54.32. First, I would log in to my Master DNS server and add the new NS and A records to my zone file:

```
;
; BIND data file for example.org
;
$TTL 4h
@  IN  SOA ns1.example.org. root.example.org. (
             2           ; Serial
             604800      ; Refresh
             86400       ; Retry
             2419200     ; Expire
             604800 )    ; Negative Cache TTL
;
@       IN NS    ns1.example.org.
@       IN NS    ns2.example.org.
@       IN A     123.12.34.57
www     IN A     123.12.34.57
ns1     IN A     123.12.34.56
ns2     IN A     98.76.54.32
```

Next, I edit named.conf and add a line to the configuration of example.org so that it will allow zone transfers from my DNS slave:

```
zone "example.org" {
        type master;
        file "/etc/bind/db.example.org";
        allow-transfer { 98.76.54.32; };
};
```

Finally, I would install BIND on the second server, or if it already exists, all I would have to do is add a new entry at the end of the named.conf file to define the example.org zone and tell this server the IP address of the master:

```
zone "example.org" {
        type slave;
        file "/var/cache/bind/db.example.org";
        masters { 123.12.34.56; };
};
```

Note that in this case the slave zone is being stored under /var/cache/bind. That's the default location for slave zone files under Debian-based systems. Under Red Hat, you would store them under /var/named/. Once I reload BIND on the slave server, it will pull the new zone information from the master, and I should be able to perform DNS queries against it.

Once you have set up a slave, keep in mind that anytime you make a change to the master, you will need to increment the Serial field in the Master's zone file (in my example, it is set to 2, but a lot of administrators like to set it to the current date plus two extra number fields, such as 2010120500). When the slave needs to know whether its zone information is up to date, it compares its serial number with the serial number on the master. If the master's serial number for a zone is higher, it copies down the new zone information; otherwise, it sticks with what it has cached.

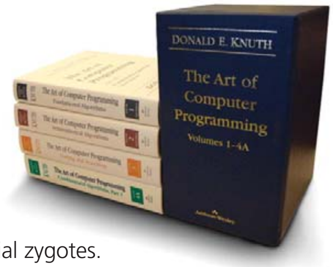
Domain Registration

Once you have a functioning DNS server, all that's left is to tell the world to use it. If you haven't already registered your domain with a registrar, find a domain registration service on the Internet (there are too many for me to list here, but a search for domain name registration should turn up plenty). When you register the domain, most registrars will let you use their own DNS servers for your domain, but you don't need them! When you get to the point in the registration process where it asks you about your DNS servers, just give them the public IP address for your own DNS server (in my case, it would be ns1.example.org or 123.12.34.56). Note that many registrars require you to have two DNS servers defined for a domain, so in that case, set up a slave DNS server and add its IP address as well. Once you complete the registration process and allow the new domain information time to propagate around the Internet, you will have the ability make IP changes for your Web, mail and other servers all from your own machines.■

Kyle Rankin is a Systems Architect 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.

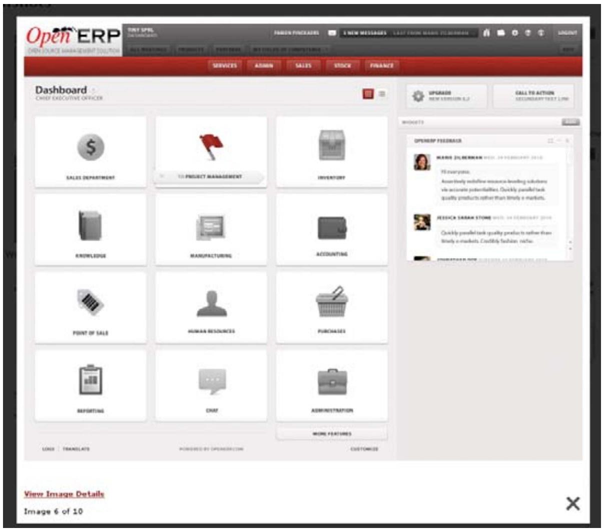
the ecosystem and beyond

Donald E. Knuth's *The Art of Computer Programming*, Volumes 1–4A Box Set, 3rd ed. (Addison-Wesley)



Donald E. Knuth's monumental book series *The Art of Computer Programming* has deep roots. When the distinguished Stanford Professor Emeritus Knuth began putting his ideas to paper, John F. Kennedy was president, Don Draper was Madison Avenue's hottest ad man, and most of us were merely potential zygotes. Publisher Addison-Wesley says that the first three volumes of *Art of Computer Programming* are "widely recognized as the definitive description of classical computer science". Practicing programmers have long applied his "cookbook" solutions to their day-to-day problems. Now comes the long-awaited fourth volume to compose a new four-volume set. The new volume 4 brings together definitive new coverage of broadword computation, combinatorial generation, fundamental combinatorial objects and other topics. Bill Gates has said that people who read the entire set should send him their résumé. If you get that far, we imagine Linus would love to see it too!

www.informit.com



OpenERP

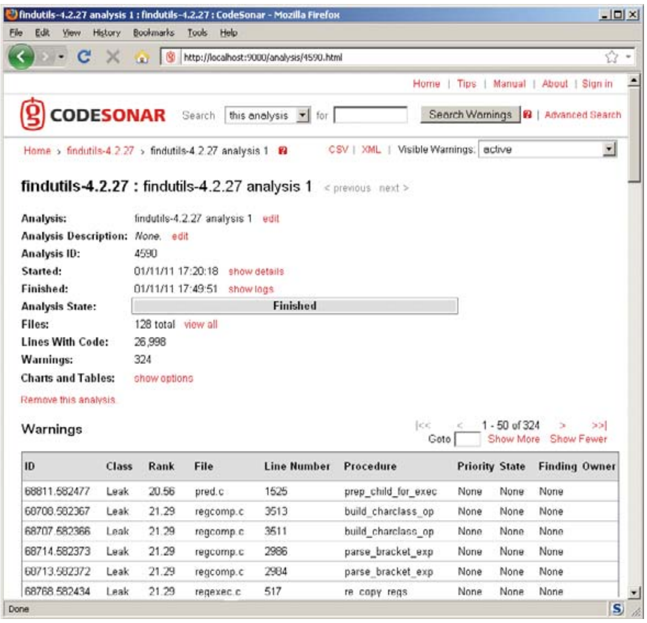
The OpenERP open-source suite of comprehensive business applications recently bounded up to version 6.0. OpenERP's eponymous developer noted that a number of factors warranted the release's designation, including advancement in simplicity and ease of deployment, the ability to build an ERP system at one's own pace, greatly improved affordability and accessibility for companies of all sizes, and more than 800 contributions from its global community of open-source developers. Further, OpenERP v6 now is available not only as an on-site version, but also as an SaaS platform, which the firm says "radically reduces the cost and complexity of an ERP deployment". Some of the hundreds of additional new features include extended multicompany functionality marketing campaign management, simplified accounting interface, tracking of tickets for support and after-sales services, push and pulled logistics flows, talent acquisition and manufacturing scrap management.

www.openerp.com

GrammaTech's CodeSonar

The developers at GrammaTech have released a fresh new version 3.6 of CodeSonar, a source code analysis tool that performs a whole-program, interprocedural analysis on code and identifies complex programming bugs. Version 3.6 adds two significant improvements, namely a significantly improved GUI, which streamlines developer interaction and boosts productivity, as well as a more efficient analysis engine, which can reduce analysis time on large code bases, says GrammaTech, by up to a third. GrammaTech also says that CodeSonar's unique strength is "its ability to identify far more program-crashing defects and security vulnerabilities than competing static-analysis tools". Another advantage is CodeSonar's new GUI that "enables developers to quickly digest key information, understand and identify the most important issues and prioritize fixes". CodeSonar runs on Linux, Solaris, Windows and Mac OS X operating systems and supports most compilers.

www.grammatech.com


findutils-4.2.27 analysis 1 : findutils-4.2.27 : CodeSonar - Mozilla Firefox
File
Edit
View
History
Bookmarks
Tools
Help
http://localhost:9000/analysis/4590.html
Home | Tips | Manual | About | Sign in
CODESONAR
Search
this analysis
for
Search Warnings
Advanced Search
Home › findutils-4.2.27 › findutils-4.2.27 analysis 1
CSV | XML | Visible Warnings: active
< previous next >
Done

findutils-4.2.27 : findutils-4.2.27 analysis 1

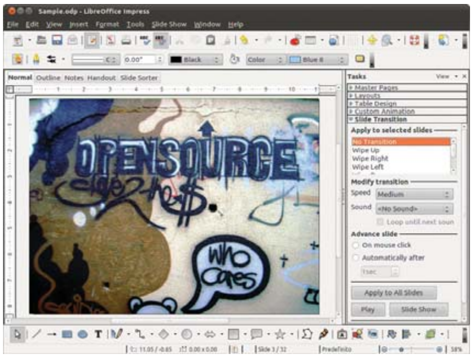
Analysis: findutils-4.2.27 analysis 1 edit
Analysis Description: *None.* edit
Analysis ID: 4590
Started: 01/11/11 17:20:18 show details
Finished: 01/11/11 17:49:51 show logs
Analysis State: Finished
Files: 128 total view all
Lines With Code: 26,998
Warnings: 324
Charts and Tables: show options

Remove this analysis

Warnings

|<< < 1 - 50 of 324 > >>|
Goto Show More Show Fewer

ID	Class	Rank	File	Line Number	Procedure	Priority	State	Finding	Owner
68811.582477	Leak	20.56	pred.c	1525	prep_child_for_exec	None	None	None	
68708.582367	Leak	21.29	regcomp.c	3513	build_charclass_op	None	None	None	
68707.582366	Leak	21.29	regcomp.c	3511	build_charclass_op	None	None	None	
68714.582373	Leak	21.29	regcomp.c	2986	parse_bracket_exp	None	None	None	
68713.582372	Leak	21.29	regcomp.c	2984	parse_bracket_exp	None	None	None	
68768.582434	Leak	21.29	regexec.c	517	re_copy_regs	None	None	None	

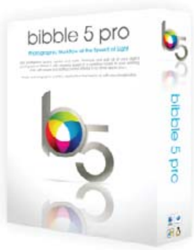


The Document Foundation's LibreOffice

The developers over at The Document Foundation are giddy about their new LibreOffice 3.3, the first stable release of this free, power-packed and open-source personal productivity suite for Linux, Windows and Macintosh. Based on and containing all features of OpenOffice.org, LibreOffice contains the Writer, Calc, Impress, Draw, Math and Base applications. Some of the many new features include compatibility with SVG files, improved ergonomics in Calc, and Microsoft Works and Lotus Word Pro document import filters. The Document Foundation says it now has more than 100 developers working on LibreOffice.

www.libreoffice.org

Bibble Labs' Bibble Pro and Bibble Lite



Bibble Labs' bills its new Bibble Pro and Bibble Lite, both nudging up to version 5.2, as "an ambitious project to revolutionize digital photographic workflow", streamlining it to run "at the speed of light". The applications, according to Bibble Labs, offer tools for photographic editing and organizing capabilities all at "blazing speed in a sleek, modern interface". Version 5.2 adds, among other things, support for 14 new RAW formats, including Nikon D3100, D7000, P7000 and Panasonic LX5, GF2 and GH2, and includes significant improvements to the application's selective editing capability. Both Bibble 5.2 Pro and Bibble 5.2 Lite are available for Linux, Windows and Mac OS.

bibblelabs.com

TYAN's S8225 Workstation Motherboard

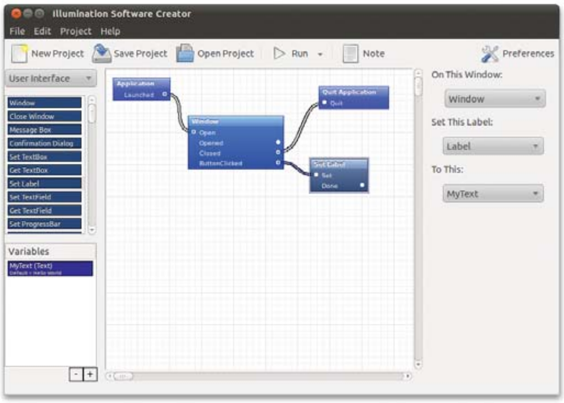
TYAN is targeting its newly released S8225 Workstation Motherboard at graphics workstation and personal supercomputing applications. The power-efficient S8225 supports two six-core AMD Opteron 4100 series processors and up to four double-wide PCI-E 2.0 GPGPU compute accelerators. Additional features include four 1-Gbit Ethernet ports, integrated support for IPMI 2.0, an optional LSI 2008 SAS controller, integrated audio, IEEE 1394a headers and TPM 1.2 support in an EATX 12" x 13" form factor.

www.tyan.com

Radical Breeze's Illumination Software Creator

Illumination Software Creator from Radical Breeze makes application development not only accessible to everyone but also "just plain fun". Now in version 3.0, this cross-platform, 100% visual application creation suite allows even those with absolutely no programming experience to visually design and create their own software applications that run on a wide variety of platforms. This latest version adds support for native iPhone and iPad applications using the exact same projects that already build for Android, Flash and so on. Apps produced for iOS platforms by Illumination Software Creator are able to be submitted to the iTunes App Store. Support for Windows Phone 7 applications is coming soon. Illumination Software Creator is available on on Linux, Windows and Mac OS X platforms.

radicalbreeze.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

Q4Wine—GUI Front End for Wine

q4wine.brezblock.org.ua

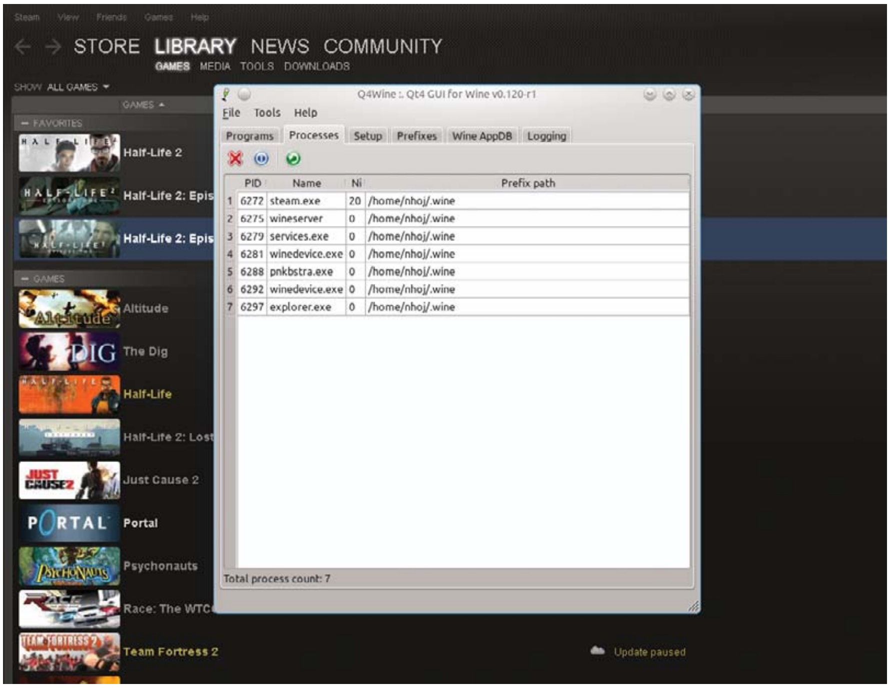
There have been plenty of GUI front ends for Wine over the years, and to be honest, I've hated pretty much all of them. However, this one has turned me around, and I recommend it to you as well.

To quote the Web site: "Q4Wine is a Qt4 GUI for W.I.N.E. It will help you manage Wine prefixes and installed applications." Also according to the Web site, Q4Wine includes the following general features (and more):

- It can export the Qt color theme into Wine color settings.
- It can work with different Wine versions easily at the same time.
- It's easy to create, delete and manage prefixes (WINEPREFIX).
- It's easy to control Wine processes.
- It supports autostart icons.
- It provides easy CD image use.
- You can extract icons from PE files (.exe, .dll).
- It provides easy backup and restore for managed prefixes.
- It includes Winetriks support.

Installation Binaries are provided for Debian, Ubuntu, Fedora, FreeBSD, Gentoo, openSUSE, openmamba and Slackware, as well as the usual source tarball. For those running with the source, the documentation says you need the following libraries:

- sudo
- wine
- >=sqlite-3.5.6
- >=which-2.19
- >=icoutils-0.26.0



	PID	Name	Ni	Prefix path
1	6272	steam.exe	20	/home/nhoj/.wine
2	6275	wineserver	0	/home/nhoj/.wine
3	6279	services.exe	0	/home/nhoj/.wine
4	6281	winedevice.exe	0	/home/nhoj/.wine
5	6288	pnkbstra.exe	0	/home/nhoj/.wine
6	6292	winedevice.exe	0	/home/nhoj/.wine
7	6297	explorer.exe	0	/home/nhoj/.wine

Total process count: 7

A key feature of Q4Wine is its Processes tab, which acts as a built-in system guard for misbehaving programs, but only for Wine-specific programs—simple, but very clever.

- >=qt-4.4.0
- >=qt-sql-4.4.0 (note: qt-sql might have sqlite driver support)
- >=qt-gui-4.4.0
- >=qt-core-4.4.0
- >=qt-network-4.4.0
- >=qt-dbus-4.4.0
- wget

FuseISO is listed as an optional dependency, but I thoroughly recommend it. I also had to install qt4-qmake and libqt4-dev.

Grab the latest tarball, extract it, open a terminal in the new folder, and enter the following commands:

```
$ mkdir build
$ cd build
$ cmake -DCMAKE_INSTALL_PREFIX=/usr
..
$ make
```

If your distro uses sudo:

```
$ sudo make install
```

If your distro uses root:

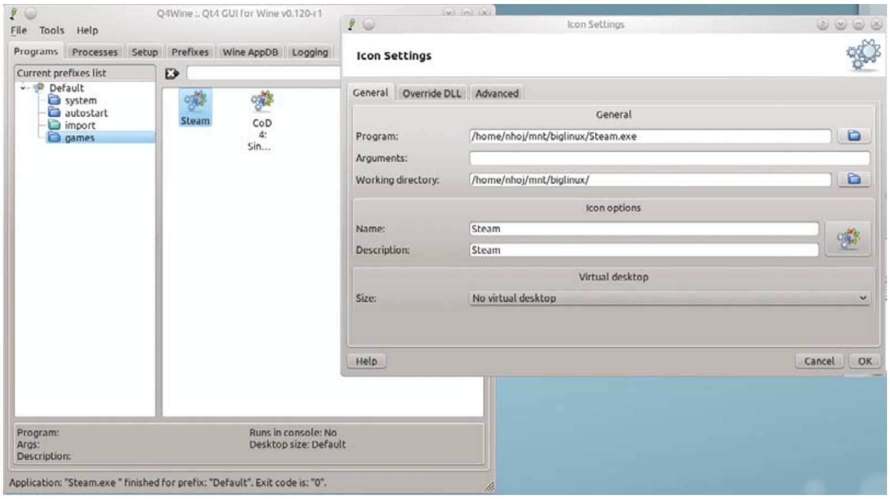
```
$ su
# make install
```

To run the program, enter:

```
$ q4wine
```

Usage Before you can begin using Q4Wine, you'll be taken through a first startup wizard. You'll be asked where the paths are for four Wine components, with the first three most likely filled in already, and the last, wine libs, empty, needing to be defined. This was /usr/lib32/wine on my system, which is perhaps left empty for 32-/64-bit reasons?

Next is a similar screen for System utils paths (which thankfully come pre-filled with defaults), followed by another page of settings also pre-filled, but feel free to tweak them if you know what



Icon Settings

General

Program: /home/nhoj/mnt/biglinux/Steam.exe
Arguments:
Working directory: /home/nhoj/mnt/biglinux/

Icon options

Name: Steam
Description: Steam

Virtual desktop

Size: No virtual desktop

Program:
Args:
Description:
Runs in console: No
Desktop size: Default

Application: "Steam.exe " finished for prefix: "Default". Exit code is: "0".

Q4Wine's real talent lies in letting you define Wine settings for one program without affecting the whole system.

you're doing. Network settings are next, followed by a very important step: a quick mount profile.

This allows you to mount or unmount without root privileges, and hopefully, it will let you eject a disc during multi-CD installations. FuseISO was the default choice with my installation (although I deliberately chose to install FuseISO for this purpose), but those without it may choose from sudo or gui sudo.

A fear of mine when it comes to GUI front ends is that you can't always terminate an errant program the way you can with a basic terminal.

Now that you're in the main GUI, look under the Setup tab. The Current prefix: drop-down box gives you the brilliant option of choosing between Wine versions if you have multiple versions installed.

If you click on the Programs tab, you're presented with a series of default system programs, including wordpad, winecfg, explorer and so on. The important thing for me, however, was the eject program, which let me get through Valve's two-disc installation of *The Orange Box*, on which I previously got stuck on disc one (either I couldn't eject it before, or couldn't get disc two to read). So any *Half-Life* fans out there may want to give this a go if you've also been having trouble.

Now, let's move on to actually adding programs.

Still under the Programs tab is a current prefixes list, which under my new installation was pretty spartan. I added a games folder by right-clicking in the left pane and then clicking New. This brings up a dialog box prompting you to enter a new directory name. Type in a name and click OK.

Now that you have a new folder, click on this folder, and in the right pane, right-click and choose New. This adds a new icon, but first brings up a window with a slew of options; this is where some of the clever stuff lies.

The first two sections are pretty basic: the General section is where you tell Q4Wine where the program is, and the Icon options is where you give the icon a name and description. Thankfully, the General section has a Browse button for both the program and working directory fields, which saves you from having to enter these paths manually. For those advanced users who want to add some arguments to the command, a specific field is provided here.

One of my favorite parts is the Virtual desktop section at the bottom of this tab. Here you can choose whether to have no virtual desktop at all, or you can select from specific resolutions in the drop-down box. This option is a piece of genius, as you don't have to go into winecfg and pick different settings every time you run another program.

Go into the Override DLL tab, and you can choose specific DLLs to override the default options, whether native or built-in. This option will be familiar to anyone who's done this in winecfg, but with the added benefit of applying a DLL override to just one program, without affecting the rest of your system.

Back in the main screen, there's a Wine AppDB tab. Those who are new to Wine may not realize there's a vast database on the Wine Web site of Windows applications and how well they run under Wine. The aging Web site and its database always has been somewhat cumbersome, but this tab lets you skip the middleman and search the database from within the Q4Wine itself.

A fear of mine when it comes to GUI front ends is that you can't always terminate an errant program the way you can with a basic terminal. The Q4Wine team has been extremely canny in that they've included a Processes tab, where you have a system-monitor-style interface, but only with Wine-specific processes. This is a game-changer for me.

Wine has many extensions based around it, but until now, they've mostly been obscure projects that one had to seek out. Q4Wine integrates the important extensions, and by the look of things, it will continue to integrate more of these projects and extensions as they emerge. I know that other GUIs have done this sort of thing in the past, but Q4Wine really has the best interface.

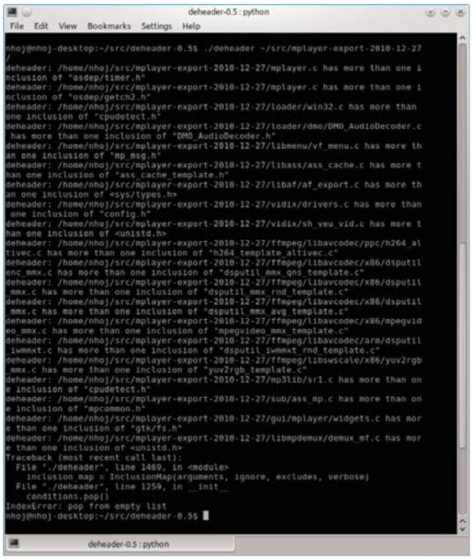
I've been using Wine for about a decade now, and I've seen many GUIs come and go. Either they've been too rudimentary in their features and interface, or they've been slow and clunky, making one give up on the whole process. Q4Wine is none of these. It's sleek, quick, intuitive and packed with features. I'll no longer stick to a terminal and do these things manually. In fact, I just made a Q4Wine desktop shortcut. I'm converted.

deheader—C Header Analysis

www.catb.org/~esr/deheader

In a world of ever-expanding code, it's easy to become sloppy, with lines of redundant code or inelegant design coming into play. Thankfully, deheader steps up to the plate—a simple tool that can save coders a great deal of time.

According to the Web site: "deheader analyzes C and C++ files to determine which header inclusions can be removed while still allowing them to compile. This may result in substantial improvements in compilation time, especially on large C++ projects; it also sometimes exposes dependencies and cohesions of which developers were unaware."



```
nhoj@nhoj-desktop:~/src/deheader-0.5$ ./deheader ~/src/mplayer-export-2010-12-27
/
deheader: /home/nhoj/src/mplayer-export-2010-12-27/mplayer.c has more than one i
nclusion of "osdep/timer.h"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/mplayer.c has more than one i
nclusion of "osdep/getch2.h"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/loader/win32.c has more than
one inclusion of "cpudetect.h"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/loader/dmo/DMO_AudioDecoder.c
 has more than one inclusion of "DMO_AudioDecoder.h"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/libmenu/vf_menu.c has more th
an one inclusion of "mp_msg.h"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/libass/ass_cache.c has more t
han one inclusion of "ass_cache_template.h"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/libaf/af_export.c has more th
an one inclusion of <sys/types.h>
deheader: /home/nhoj/src/mplayer-export-2010-12-27/vidix/drivers.c has more than
 one inclusion of "config.h"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/vidix/sh_veu_vid.c has more t
han one inclusion of <unistd.h>
deheader: /home/nhoj/src/mplayer-export-2010-12-27/ffmpeg/libavcodec/ppc/h264_al
tivec.c has more than one inclusion of "h264_template_altivec.c"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/ffmpeg/libavcodec/x86/dsputil
enc_mmx.c has more than one inclusion of "dsputil_mmx_qns_template.c"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/ffmpeg/libavcodec/x86/dsputil
_mmx.c has more than one inclusion of "dsputil_mmx_rnd_template.c"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/ffmpeg/libavcodec/x86/dsputil
_mmx.c has more than one inclusion of "dsputil_mmx_avg_template.c"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/ffmpeg/libavcodec/x86/mpegvid
eo_mmx.c has more than one inclusion of "mpegvideo_mmx_template.c"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/ffmpeg/libavcodec/arm/dsputil
_iwmmxt.c has more than one inclusion of "dsputil_iwmmxt_rnd_template.c"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/ffmpeg/libswscale/x86/yuv2rgb
_mmx.c has more than one inclusion of "yuv2rgb_template.c"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/mp3lib/sr1.c has more than on
e inclusion of "cpudetect.h"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/sub/ass_mp.c has more than on
e inclusion of "mpcommon.h"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/gui/mplayer/widgets.c has mor
e than one inclusion of "gtk/fs.h"
deheader: /home/nhoj/src/mplayer-export-2010-12-27/libmpdemux/demux_mf.c has mor
e than one inclusion of <unistd.h>
Traceback (most recent call last):
  File "./deheader", line 1469, in <module>
    inclusion_map = InclusionMap(arguments, ignore, excludes, verbose)
  File "./deheader", line 1259, in __init__
    conditions.pop()
IndexError: pop from empty list
nhoj@nhoj-desktop:~/src/deheader-0.5$ █
```

deheader scans code for redundant header inclusions. Here's some output from the MPlayer Project, for instance.

Installation and Usage As far as packages go, at the time of this writing, the only thing available was a source tarball. But, fear not. No compiling is necessary, and because there's no real mention of library requirements, I'm guessing most distros will run it off the bat, assuming they have Python.

Download the latest tarball from the Web site, extract it, and open a terminal in the new folder. Then, it's simply a case of running:

```
$ ./deheader path-of-files
```

If the given path is a directory, deheader scans all the files within. Give it some time to process, and eventually a list of all the unnecessary headers appears on-screen. For instance, I chose to analyze the now ten-year-old MPlayer code, a project that would unavoidably have a lot of legacy code and loose ends simply from being around for such a long time.

As you can see, deheader is a very simple-to-use program with an elegant design.

If you're ready to take things further, add a switch of `-r`, and the unnecessary headers are removed from the files. If you want to do some test compiling, use the `-m` switch. As an example, here's a command I ran against the MPlayer code:

```
$ ./deheader -r  ~/src/mplayer-export-2010-12-27/
```

Those are the basics; refer to the documentation for more information.

As you can see, deheader is a very simple-to-use program with an elegant design. This ideal of coding elegance is manifest in deheader's results. It should save a great deal of compilation time and highlight coding foibles that likely would have remained unnoticed. Although it's still sitting around in tarball source form, hopefully it will make its way into distro repositories soon.■

John Knight is a 26-year-old, drumming- and climbing-obsessed maniac from the world's most isolated city—Perth, Western Australia. He can usually be found either buried in an Audacity screen or thrashing a kick-drum beyond recognition.

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

Projects at a Glance

Razercfg

bu3sch.de/joomla/index.php/razer-nextgen-config-tool

Fans of the hard-core gaming company, Razer, no doubt will want to look up this project. According to the Web site: "This is the next-generation Razer device configuration tool bringing the Razer gaming experience to the free, Open Source world. This utility is supposed to replace the old deathaddercfg utility. The tool architecture is based on razerd, which is a background dæmon doing all of the low-level privileged hardware accesses. The user interface tools are razercfg, a command-line tool; and qrazercfg, a Qt4-based graphical device configuration tool."

And, according to its Freshmeat entry: "Supported devices are the Razer DeathAdder mouse, the Razer Krait mouse, the Razer Lachesis mouse, the Razer Copperhead mouse and the Razer Naga mouse."

GAdmin-Antivirus

freshmeat.net/projects/gadmin-antivirus

I've always been put off by Clam antivirus's command-line interface (I just don't get it), so a GUI front end would be brilliant. Thankfully, it seems numerous front ends exist, and hopefully, I can cover one or two in the coming months. Details on *this* project, however, are scarce, although I'd like to try it anyway. According to its Freshmeat entry: "GAdmin-Antivirus is a fast-and-easy to use GTK+ front end for Clam antivirus. Multiple scan sets can be scheduled to run at specific times via cron. Each of these scan sets can contain multiple directories to be scanned."

SHARE THE PASSION
ELEVATE YOUR GAME
SELF2011
SPARTANBURG SC
JUNE 10-12, 2011
SOUTHEASTLINUXFEST.ORG

HARDWARE

D-Link's Boxee Box

How does the Boxee Box stack up as a living-room entertainment unit? SHAWN POWERS

When it comes to software battles, the Open Source community has its fair share. KDE vs. GNOME, vi vs. Emacs, Firefox vs. Chromium—there seems to be countless rivalries. In my house, one of the biggest rivalries is between XBMC and Boxee. This month, I take a look at Boxee, specifically the Boxee Box from D-Link.

Although Boxee itself is a software package, competing with software like XBMC, Front Row, Plex, MythTV, GeeXboX and so on, the Boxee Box is a hardware device competing directly against things like the Roku, AppleTV and GoogleTV. When compared to the other hardware options out there, it really does shine.

What's in the Box's Box?

The most noticeable physical feature of the Boxee Box is its odd melted-ice-cube shape (Figure 1). It looks like someone took a cube and put a corner of it on a hot surface to make it melt. I'm sure it's very artsy, and I've read that it efficiently manages the cables in the rear. But for me, it annoyingly doesn't stack anywhere in my entertainment center. I know creative-yet-impractical designs work well for some companies, but I'm of the opinion that square is good. I'm often called a square, and it works for me—but I digress. It's a design choice and really doesn't affect my opinion that much.

In contrast to the Box's design, the remote control is a thing of beauty. It's like the mullet of remote controls—business on the front and party on the back. The top side has a very simplistic button design (much like an Apple remote to be honest). The flip side of the remote has a complete qwerty keyboard. It's not something you'd want to write the Great American Novel with, but it fares about as well as a cell-phone keyboard does (more on the keyboard later).

As you can see on the back of the Boxee Box in Figure 3, this unit is designed for high-definition (HD) systems.

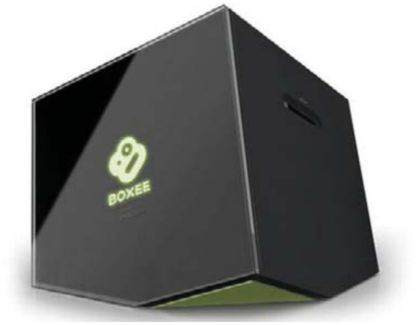


Figure 1. The Boxee Box is remarkably not boxy. It is shaped like a cube that someone modified with a belt sander.

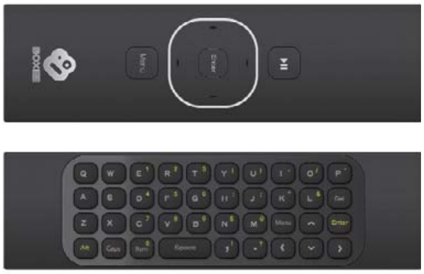


Figure 2. I love the idea of the double-sided remote. Apart from a few nagging issues, it's perfect.

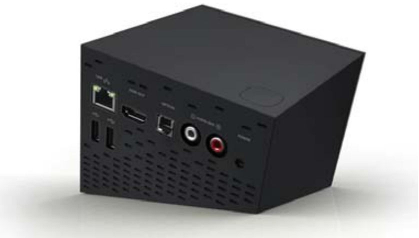


Figure 3. The ports are easily accessible in spite of the weird angle.

The only video output option is HDMI. Thankfully, there are both optical and analog audio outputs to go along with the HDMI audio, but if you have a television without HDMI support, you'll need to be creative with adapters and such.

The unit also has some USB ports in the back and an SD card slot on the side—or perhaps the SD card slot is on the top. With the melted-ice-cube design, it's really hard to determine what's considered up. Add to that a wall-wart power supply, and you have the contents of the Boxee Box's Box.

Box Guts

Because the Boxee Box is an appliance, the internal hardware isn't really as important as what it does, but we're all geeks here, so this part is interesting if nothing else. Originally, the Boxee Box was going to use the NVIDIA Tegra2 for video playback. In a last-minute switch, however, the Boxee team went with the Intel CE 4100 and a PowerVR GPU. The Boxee team is confident with its decision, and as long as it performs well, the end user really doesn't interact with the guts anyway. The Box boasts:

- Atom processor, at 1.2GHz.
- PowerVR SGX535 graphics processor.
- 1GB of RAM.
- 1GB NAND Flash memory.
- HDMI 1.3 (audio and video).
- S/PDIF optical audio.
- Composite (RCA) audio.
- Two USB ports.
- SDHC card slot.
- 100Mbps Ethernet port.
- 802.11n Wi-Fi.
- Two-sided RF remote control.

The Two-Faced Remote

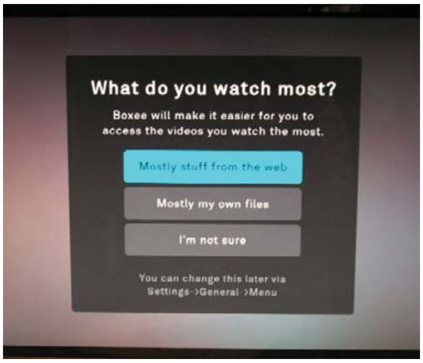
The remote that comes with the Boxee Box is really a perfect addition to an entertainment device into which you occasionally need to type. Those of us who have typed on the Nintendo Wii's on-screen keyboards fully understand

Those familiar with the Boxee software will recognize the Boxee Box's interface right away.

how frustrating it can be to "type" with nothing more than directional keys or a pointing device. Although the idea behind the two-sided remote is brilliant, it does have its flaws.

The simplicity of the top side makes for a simple browsing experience, but unfortunately, it's so simple (and symmetric), it's easy to grab upside down. When you grab the remote upside down, up becomes down, right becomes left, and because the remote uses RF instead of infrared, it doesn't matter if the remote is facing the wrong way. When you press a button, it registers. There is a raised logo on one side of the remote, but it's not really clear which end is up on the logo either, so you have to look closely to tell which is up and which is down. It's not a terrible problem, but it's annoying at times.

The keyboard is rather nice for something so small. It's not too bad to hold, and the keys give enough feedback that you can tell you're typing. The enter and arrow keys aren't in obvious positions, so you have to look when you're typing. That's not a big problem either, unless you happen to be in the dark. The keys don't have any backlight, so typing in the dark is just about impossible. Because typing isn't required for normal operation, that's not a showstopper, but I'd like to see future versions have some sort of illumination for the keyboard buttons.



What do you watch most?

Boxee will make it easier for you to access the videos you watch the most.

Mostly stuff from the web

Mostly my own files

I'm not sure

You can change this later via Settings->General->Menu

Figure 4. This dialog box just sets up the default media the Boxee Box plays back. It doesn't hinder your ability to watch on-line content.

The Software

Those familiar with the Boxee software will recognize the Boxee Box's interface right away. There are a few big changes, but for the most part, it is similar to the traditional Boxee experience. I'm currently using the firmware released in mid-December 2010, which makes some significant (and welcome) changes. The big difference in functionality (apart from Wi-Fi working, which it did not for me out of the box) was a change in how the Boxee Box displays media. Figure 4 shows the dialog box that appears on boot, asking how you prefer to view media. This question is a little confusing, as it doesn't explain what difference your choice will make, but simply put, if you choose Local Media, the Boxee Box defaults to showing you media stored locally as opposed to Web content. You're still able to watch Web content, and it's not difficult to do so, but by default, Boxee points to local media.

The interface is functional, but it's not always intuitive or easy to navigate. For example, it takes a series of clicks to get back to the home screen, with no apparent shortcut to get there quickly. The menu button doubles as a back button, but it isn't labeled as such. Even with those frustrations, however, the menu system isn't difficult to figure out. I have read other reviews claiming the Boxee Box menus are sluggish, but I've noticed that only when identifying media or making significant system preference changes. Usually the remote and interface are quite responsive.

The video playback for local media is stellar.

The video playback for local media is stellar. In my 5TB file server, I can't find a single file that won't play. Playback of 1080p MKV files is flawless, and even Web streams are clear and watchable. (That is likely due to my beefy Internet connection, but the Boxee renders the Web streams well, so kudos.)

Third-Party Software

Boxee, and in turn the Boxee Box, has the ability to use "applications" in order to access other on-line services. These are almost identical to the Roku's application system, if you are familiar with that. At the time of this writing, many applications exist that add value to the Boxee experience. Pandora Radio has a great interface, and many on-line sites, such as College Humor and Revision 3, have applications that will guide you through their catalogs. Netflix and Hulu are glaringly absent. By the time this goes to press, hopefully both on-line services will be available, but their absence makes the Boxee seem quite lacking, especially if you don't have an extensive library of local media.

Firmware updates certainly will improve the Boxee Box as time goes on, but the application framework is where the magic could really happen. Third-party applications turned the Roku from a Netflix-streaming device into a full-blown living-room entertainment unit. Because the Boxee does (or will do shortly) all the things the Roku can do, *plus* plays any local media you throw at it, it's really positioned to be the perfect set-top box.

Final Thoughts

The Boxee Box, like its software-only counterpart, Boxee, takes a different approach to media from XBMC. Where XBMC is fast, clean and elegant in its simplicity, Boxee does everything it can to immerse users into a multimedia universe. XBMC plays media, whereas Boxee is an entertainment system. Even after taking the Boxee Box through its paces, I'm not convinced one is better than the other. They're just different. At $199, the Boxee Box is an affordable way to get an extremely robust media center. Although I have XBMC on a $199 nettop device, and it runs fine, the integration of specialized hardware is hard to beat. I heartily recommend the Boxee Box for anyone considering an HD media center. Although the unit I reviewed didn't have Netflix, Hulu or even VUDU, by the time you read this, it should have access to all three, which really will make Boxee a tough box to beat when it comes to home entertainment.

My only recommendation before buying the Boxee Box is to download the Boxee software and see if you like its interface. Granted, the downloadable version is aesthetically different from the Box's interface, but it's a good way to see whether you are comfortable with the Boxee way of doing things. Some like it; some hate it. (See the Local Media, Boxee Style sidebar for a little more information.) It's free, so you can try it before you buy it. Check it out at **www.boxee.tv**.■

Local Media, Boxee Style

Boxee always has handled local media in two ways. Users can access their filesystems directly and peruse their video collection folder by folder. Where Boxee really shines, however, is with its media library function. If files are named correctly (basically with the name of the movie and year), Boxee downloads genre information, cover art, ratings and other metadata. It's possible to browse your media collection like you'd look through movies at the rental store. Because Boxee downloads metadata, you even can sort your collection to find that perfect popcorn companion. Granted, the media scraper isn't perfect, but the Boxee Box gives you the ability to identify media it can't figure out automatically.

Boxee makes your media collection look beautiful. The initial scanning takes a long time, but the cover art and metadata are really nice.

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.

Go to **www.linuxjournal.com/video/review-boxee** to see Shawn's video review of D-Link's Boxee Box.

BAD CAMP
BAY AREA DRUPAL CAMP

Drupal 7: the Webchick behind the Wheel

Katherine Druckman talks to Angela Byron about Drupal 7 and managing a distributed open-source development team.

KATHERINE DRUCKMAN

Photos by Kris Krug

Angela Byron is one of the most respected contributors in the Open Source community. She has been recognized for her efforts by Google, receiving the 2008 Google-O'Reilly Open Source Award for Best Contributor. She's passionate about the Drupal Project and has worked tirelessly to ensure that the latest release of the popular Web platform and CMS is the best version yet. We talked to Angie about her role as co-maintainer of Drupal 7 and about what it takes to manage such an important, massive open-source project.

KD: You've had a very busy three years taking the lead as core maintainer on the newly released 7th version of Drupal. Tell us a little more about what being the Drupal 7 core maintainer means. What does it entail?

AB: Dries Buytaert is the Drupal Project's founder and project lead, also known as our Benevolent Dictator for Life. He holds the keys (as in commit access) to Drupal core. Each new release of Drupal, Dries also appoints one other person, called a co-maintainer, who is also given commit access and the responsibility to help set the vision and direction of the release alongside him.

Neil Drumm was the core co-maintainer in Drupal 5, and he prioritized usability improvements, including a graphical installer. Gábor Hojtsy was the core co-maintainer of Drupal 6, and he prioritized numerous internationalization improvements. In Drupal 7, I was selected. Things I prioritized included usability, quality assurance, "developer experience" (API consistency and other things that make Drupal more enjoyable to work with), and making Drupal more accessible to designers, themers and people with disabilities.

Image handling now is included out of the box without the need to download six or seven additional modules.

"On paper", a core co-maintainer's responsibility is to review and commit patches submitted by the core developers. In practice, I found that 80% or so of my time was spent *not* doing that, but instead on more "human" endeavors: identifying people working in similar areas and encouraging them to work together, helping brainstorm architectural direction on certain patches, mediating heated arguments, helping new contributors get involved, promoting major initiatives and so on. So in some ways, it was a community management role, but with commit access thrown in.

KD: Which part of the three years has been the most intense? I would guess these past few months, but then I wonder what the pattern has been over the course of working on this version. How was it at the beginning versus the middle and end?

AB: Drupal's release cycle goes in three phases:

- "code thaw" where we make our wildest dreams come true and add new features, break APIs and generally pursue world domination.
- "code slush" where we focus on polishing the rough edges from code thaw, API consistency, UI cleanups and performance.
- "code freeze" where the APIs get severely locked down, and bug fixes/stabilization is the name of the game.

The intensity varies, and generally whenever we have a major deadline (for example, a Drupalcon is coming up or a code freeze date is about to be declared), things heat up a lot. Most of our major features were introduced a week or so before one of those major deadlines.

One of the most challenging things was maintaining momentum after code thaw (aka the "fun" part) completed. People could no longer add their pet features; instead, the attention turned to "the slog" of bug fixes and incremental improvements. The number of core contributors died off significantly as they focused back on their contributed modules or other projects, and those who were left had to carry the pretty tremendous burden of taking us from some 150 critical issues down to zero so we could release.

KD: Drupal has a *lot* of core contributors. It's up to around 1,000 now, isn't it? Obviously a small number of those contribute the bulk of the code, but it's still an enormous group of people to coordinate. Can you tell us about that experience?

AB: Yeah, the count from the commit logs was a little less than 1,000, but that doesn't include people who reported bugs, tested patches and so on. That number includes a set of maybe 50–75 hard-core people who spend their lives in the Drupal core queue daily and then an enormously long tail of others who contribute only a handful of patches. And, bear in mind that every single one of those patches has to be reviewed and then committed by Dries or me.

It's both incredibly exciting and challenging to work in this environment. My favourite parts were forming battle plans and kicking around ideas on how to make Drupal 7 the most amazing release possible, helping new contributors submit their first patch, seeing the community consciousness about things like usability, design and accessibility gradually evolve and accept their importance. The core contributors are amazing, smart, dedicated and passionate folks who pour their hearts and souls into Drupal, and I'm honored to work with them every day on IRC and the issue queues.

However, this level of closeness also meant things also could get a bit personal sometimes. And when they did, it really, really hurt, because these folks are all my friends, and we have worked so closely together for years. There definitely were days when I needed to step away from the computer for a few hours and get some perspective, and even considered throwing in the towel altogether more than once.

On the whole though, the experience was absolutely amazing. I'm infinitely proud of the team and what we've managed to accomplish with Drupal 7.

KD: What can you tell us about the decision-making processes involved in developing and maintaining this sort of a project? How do you collectively decide on a course of action or the best approach to a problem? Programmers can be married to their ideologies. How do you deal with that?

AB: Drupal is very much a consensus-based community. Problems are identified, solutions proposed, code written and then discussed among at least two but possibly dozens or hundreds of people. No change, big or small, makes it into the upstream code unless it has been marked "reviewed & tested by the community", which means at least one other person has looked at it and given it a nod of approval. This is a great "community engineering" strategy, because it ensures better quality code and encourages developers to be civil to each other so they can find reviewers for their patches.

There's typically very little contention around straightforward bug fixes. On some of the more esoteric or architecturally facing issues, however, lots of people start throwing around opinions on approaches, and sometimes heated ones.

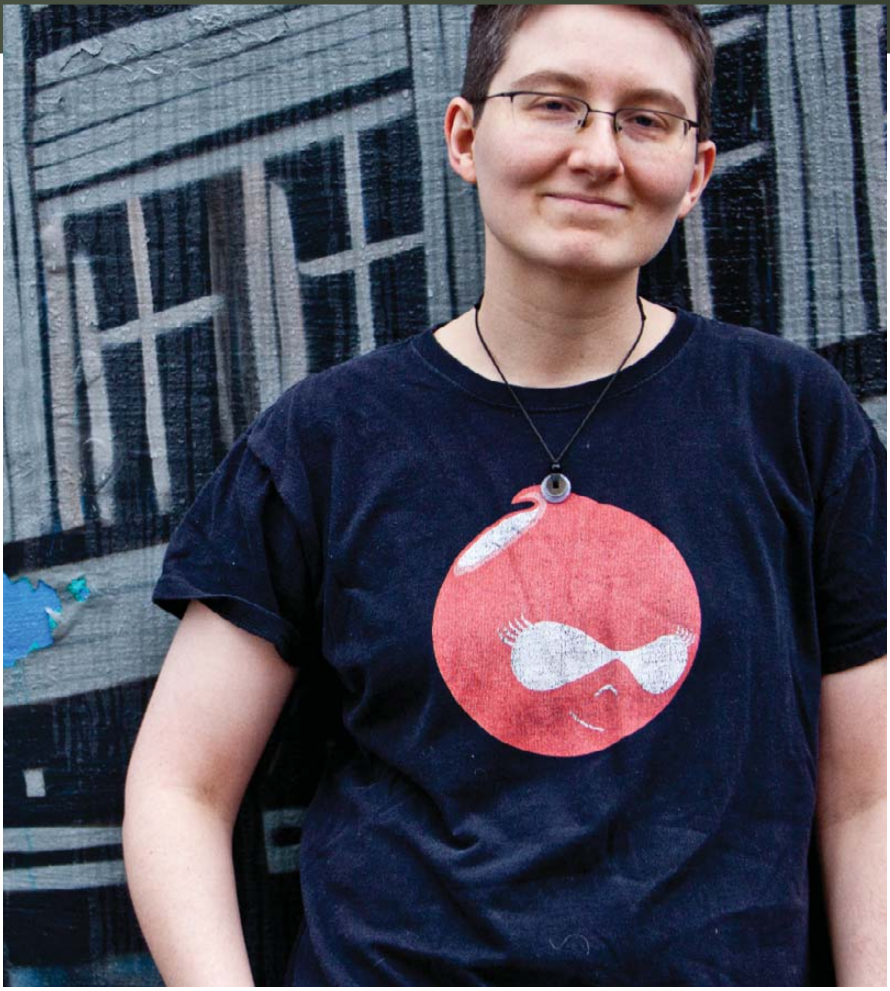
In my role as core maintainer, the best thing I could do in those situations usually was simply monitor the discussion carefully, but keep mostly quiet except when things escalated to personal attacks. In almost all cases, if developers are given free reign to hash things out among themselves, they're able to come to a mutual resolution without any intervention. It's important that this happen *most of the time*, in order to build camaraderie and respect among the development team.

Occasionally, however, a stalemate was reached and intervention by a core maintainer was needed. In this situation, I normally try to take the folks aside in IRC and see if we can hash out their differences together. Oftentimes a heated discussion on the issue queue dragging on for days can be settled in minutes when both parties are brought together with a mediator. I try to summarize the opposing views and explain what is good about each perspective in an attempt to disarm some of the defensiveness that might be brought into such a meeting. Neither person's idea is "wrong"; they both have pros, but we need to come to the right decision that might be some mixture of both.

In cases where we weren't able to come to resolution, I'd post my best attempt at a mutually agreeable solution. This actually very rarely was the *actual* solution we went with, but it had a way of "resetting" the conversation to be around a new suggestion rather than the old ones, which usually helped everyone play nicely again—usually. On the whole, I just tried to remind people that we're all here to make Drupal better, even that stubborn git who can't see your point of view yet.

KD: What excites you most about Drupal 7? Do you think Drupal 7 improvements will increase Drupal adoption?

AB: Feature-wise, it's hard to pick—there were *so* many things we added to Drupal 7—but I'd say overall it comes down to the following things:



- Image handling now is included out of the box without the need to download six or seven additional modules. This is *huge* for adoption, at least for people who want something other than text on their Web sites, who apparently exist. It also brought with it a host of improvements with our native file API, which has interesting implications for extending Drupal for use as a document management system.

- The new database abstraction layer has a lot of people really excited. Both from its support of new features, such as transactions, and support for more database back ends, but also its new object-oriented syntax. Basically, everything that ever sucked about our database abstraction layer has been fixed in Drupal 7.

- The new entity and field paradigm is an important shift in the Drupal Project. In the past, Drupal was very "content-oriented", and lots of features were developed that would extend "nodes" or pieces of content in the system. This resulted in a lot of people trying to shoehorn things that weren't actually nodes (such as users, comments and so on) into nodes so they could benefit from these features. In the future, these same features will be developed as fields, which then can be used across any entity in the system—users, comments, content, taxonomy terms and more.

- The automated testing framework and more than 30,000 tests we added to Drupal 7 has had profound effects on our community development process. We know instantly if any patch in the issue queue works or breaks existing tests. We are free to refactor subsystems knowing we didn't break any of the existing functionality, and the "test-driven development" mindset is slowly working its way into our development community's consciousness as a best practice.

- Overall, I'm simply thrilled with the amount of diversity in the core development team now as compared to the last release. We have an accessibility team, a usability team, a markup and design team, and a documentation team dedicated to improving the core in ways that go beyond the code. My hope is that the leadership shown by early pioneers in these fields will open the floodgates for new contributors in the Drupal 8 process, and that Drupal 8 improves upon Drupal 7 in all of these areas and more.

KD: I have played with Drupal 7 quite a bit, but I can't say I've really put it to full use yet, and I keep hearing about how much more usable it is. How much do you think this will affect the Drupal learning curve?

AB: Usability was something that saw tremendous attention during the Drupal 7 release cycle, and we saw a radical transformation in the culture of the Drupal development community and how seriously this barrier of entry was treated.

For folks coming from other CMSes, such as WordPress or Joomla, the biggest hurdle to getting started with Drupal is often the "LEGO block" approach Drupal takes to building sites with modules.

Just after Drupal 6 was released in February 2008, Dries, myself and several other major contributors went to the University of Minnesota (yes, Minneapolis in February—that's how much we love Drupal) to perform our project's first formalized usability study. We were given a room with one-way glass, tools like eye-tracking software to tell where people were looking on the screen, and the University found several participants who had previous Web development experience with tools like WordPress, Movable Type and Dreamweaver but not with Drupal. In other words, people in our project's *direct* target audience.

The results were absolutely shocking and completely transformed the way I look at Drupal. We found that participants were completely lost as to whether they were on the front end or back end of their Web sites. They were unable to find major administrative sections in order to perform basic tasks. They were mystified by Drupal jargon, and on and on.

A usability team was formed who set about fixing many of the problems identified in testing, and they fixed a number of important and obvious problems in the existing UI. Additionally, Acquia funded Mark Boulton and Leisa Reichelt, a design and usability expert, to take a holistic view of the Drupal administrative experience and make more wide-sweeping changes, in a community-driven, collaborative usability experiment called D7UX. Both initiatives worked in tandem to provide Drupal 7 with a new administration theme; a set of common administration patterns, such as a toolbar and contextual links; and a task-based administrative information architecture. We haven't yet been able to test formally how the usability work we did in Drupal 7 improved the situation over the results we saw with Drupal 6, but preliminary feedback from the broader community has been pretty awesome.

This usability work doesn't fully address the dreaded "Drupal learning cliff", per se. There still are an awful lot of things you need to know in order to be a successful Drupal site builder, like what modules you should use for what and what the heck weird words like "taxonomy" and "node" mean. However, Drupal 7 hopefully should require a lot less customization from site builders to put it in front of their clients and give them a better leg up on answering the question, "Great. I have a Drupal site installed...now what?"

KD: It's supposed to be more scalable too. Can you tell us how?

AB: A number of new features in Drupal 7 help with the situation where your site needs to accommodate huge blitzes of additional traffic, assuring all visitors of a speedy experience:

- Master/slave replication support in the database abstraction layer—database writes are slower than database reads, and reads happen way more often. A master/slave setup allows you to separate reads from writes, so that the main database storing all the information isn't additionally overburdened with read requests, which can be routed instead to read-only slave databases.
- Reverse-proxy support—a reverse proxy, such as Varnish or Pound, can greatly speed up Web site access by caching copies of pages and then intercepting requests to serve them, saving the Web server from having to handle page requests directly. Drupal's settings.php configuration file now contains directives to enable reverse-proxy support.
- Support for content delivery networks (CDNs)—services exist for caching static files, such as images, CSS and JS, across multiple geographically distributed computers, which then can be served to visitors more quickly than a round-trip to the Web server where Drupal is stored. By invoking hook_file_url_alter(), modules can re-route requests from Drupal's files directory to services, such as Akamai or Amazon CloudFront.

There's a high-performance distribution of Drupal 6 core called Pressflow (**pressflow.org**) from which a lot of these scalability improvements originated.

KD: For our readers who are more comfortable with WordPress, Joomla or even platforms like Ruby on Rails and Django, what do they need to know about Drupal, and in particular Drupal 7, in order to have the best experience getting started?

AB: For folks coming from other CMSes, such as WordPress or Joomla, the biggest hurdle to getting started with Drupal is often the "LEGO block" approach Drupal takes to building sites with modules. It's common in other CMSes that if you want to add a photo gallery to your site, you simply search for a photo gallery extension and choose from a list of prebuilt all-in-one options.

In Drupal, however, the trend in

modules is more toward small, generic, re-usable components that can be combined and mixed and matched in lots of various ways. There's not much in the way of off-the-shelf photo gallery modules for Drupal. Building a photo gallery in Drupal typically involves creating a content type to provide a data entry form for images, adding an image field in order to upload photos to the content type, creating a view of photo images, and so on. Although more elbow grease is required at the outset, the advantage is that the photo gallery you end up with can behave exactly as you want. And, the same module that provides an image field for photo galleries also can be re-used to provide album covers and user avatars, while the same module that provides a photo gallery view can be used to create event calendars, RSS feeds and other types of listing pages. This level of customization and re-usability is what attracts people to Drupal, but it definitely requires a tinkerer's mindset.

For folks coming from frameworks, such as Django or Ruby on Rails, the main thing to realize about Drupal is that it's more of a "framlication" than a pure framework. Drupal provides ample APIs for dealing with file handling, session management, internationalization and so on, and it also provides a "hook" system from which Drupal's base behavior can be extended. However, it also makes some base assumptions that what you're building is a Web-based application tracking things like users and content. The advantage of this is you don't need to recode a new login system every time you build a site on Drupal; this type of low-level functionality is provided for you in an extensible way. But, it does mean if you don't agree with some of the base assumptions Drupal makes, you'll need to spend a bit of effort developing a module to alter behavior you want to change. The best advice I probably could give to folks coming from more traditional Web frameworks is to take the time to explore what Drupal can do without writing a line of code, which is fairly substantial. Then, learn the extension mechanisms Drupal provides—hooks, the theme system, the localization layer and so on—to make customizations in a forward-portable way.

KD: You've worked on some major, large-scale Drupal projects, and Drupal was also selected as the platform for Whitehouse.gov. What about Drupal lends itself to those types of sites?

AB: I think Drupal hits a sweet spot in that it's free, open source and an extremely capable framework that's constantly evolving. It can be highly customized to particular use cases, and it has an ever-growing community with a *lot* of expertise. Many of the enterprise-level clients we work with move to Drupal from less-capable, proprietary CMSes that have thousands or even hundreds of thousands of dollars per year in licensing fees, with bugs they can't fix themselves because they're beholden to a vendor's schedule. So the idea of moving to something they can be trained on internally or hire outside expertise to implement quickly is very appealing.■

Katherine Druckman is an HTML-flinging, PHP-hacking *Linux Journal* Webmistress by day and a refined connoisseur of historic architecture and fine Chinese ceramics by night. She usually can be found surrounded by the charm of aging Texas buildings from the pioneer days or appreciating ceramics of the Song and Qing dynasties. Well, either that or sitting in a comfy chair with a laptop. Yeah, probably the laptop thing. Now in her fourth year heading all things LinuxJournal.com, Katherine's experiences in Web publishing for the open-source audience have strengthened her stance as an impassioned Drupal fangirl.

We couldn't fit the entire interview in the magazine, so to read more, go to **www.linuxjournal.com/content/interview-angie-byron-drupal**.

Drush: Drupal for People Who Hate Mice

Read on for more about this **command-line shell and scripting interface** for Drupal.

James Walker

During its ten-year rise to prominence, Drupal dramatically has changed the way people use it to build Web sites. Without needing to write a single line of code, Drupal users are able to build complex, custom Web sites through the point-and-click Web interface. The problem is I hate Web interfaces.

To be fair, the rise of keyboard shortcuts in modern Web applications has made them far more comfortable for me, but I don't like mice (and don't even get me started on trackpads). I am a keyboard kind of guy; hands on home row make me happy.

Even for those of you proficient with your pointing devices, there still are times when you just can't beat the elegant power of the command line. As Linux users, you have all experienced the sheer joy of a simple shell script to automate repetitive tasks (and if you haven't, why not?). There is a long list of scenarios where using Drupal's Web interface is either inefficient or inconvenient, but the truth is, for most Web developers and sysadmins, the command line is a comfortable place.

Enter Drush

Drush (DRUpal SHell) is a command-line shell and scripting interface for Drupal, a veritable Swiss Army knife designed to make life easier for those of us who spend some of our working hours hacking away at the command prompt.

Drush began several years ago as a project to provide a command-line interface to Drupal. Although somewhat limited in its early utility, Drush has matured to become an essential tool for many Drupal developers and administrators (myself included).

The current version, 4.1, brings a ton of great new built-in commands, plenty of internal cleanup and better support for Drupal 7. Arguably more exciting, however, is the growth of add-ons and extensions for Drush. Although not a traditional Drupal module, Drush is highly extensible via its own API. A large number of existing Drupal modules now include Drush commands in their official releases.

Getting Started

Initial versions of Drush required a working Drupal installation prior to installing Drush. Since version 2, however, Drush has become Drupal-version-independent. You don't need a working Drupal site to install Drush. In fact, as you will see shortly, you can use Drush to install your Drupal site fully.

Installing Drush on Debian/Ubuntu systems is as simple as running `apt-get install drush`. However, the current stable Debian/Ubuntu versions include packages for Drush 3. To get the latest and greatest, or if you're on a system without a Drush package, you need to go through a few extra steps:

1. Make sure you have a working version of PHP CLI. This generally can be accomplished by installing the php5-cli package on your system.

2. Download Drush from **drupal.org/project/drush** (I am currently using the "All-versions-4.1" release). Extract the downloaded package (.zip or .tar.gz) and place it where you won't lose it. I typically place Drush in ~/local/drush on my development machines and /usr/local/drush on servers, but the location is flexible.

3. Get the drush command into your $PATH. You can make this happen in several ways. Drush includes a shell script (called drush) that tries to determine the "best" PHP interpreter to use and runs the drush.php PHP script. I typically symlink this script into a directory already in my $PATH (`ln -s /usr/local/drush/drush /usr/local/bin/drush`). Alternatively, you can add the path of your Drush install to your $PATH or provide your own shell alias to the PHP script itself. See the included README.txt file for full details.

To test your new Drush install, you should be able to run `drush status` in your terminal and see output something like this:

```
PHP configuration     :  /etc/php5/cli/php.ini
Drush version         :  4.1
Drush configuration   :
Drush alias files     :
```

If that worked, it's time to let the real fun begin!

Look, Ma! No Mouse!

Although Drush works without an existing Drupal installation, it is certainly much more fun with one around. In fact, one of the most fun uses of Drush is to install new Drupal sites. Normally, this process would include something like: browse to **drupal.org**, click Get Started, then click Download Drupal, then click to download the tar.gz file. Extract that package into your Web server's document root. Create a database for your new Drupal site. Browse to your new Drupal install (that is, http://localhost/drupal) where you will be tossed into the installer. Begin clicking your way through the installer (where you'll likely be asked to `chmod` your settings file), and after seven or more screens, you'll have a working Drupal install. Of course, chances are good that from here you'll go on to download a handful of modules (again, browsing drupal.org and extracting zip files) and clicking through the interface to turn them on and configure them.

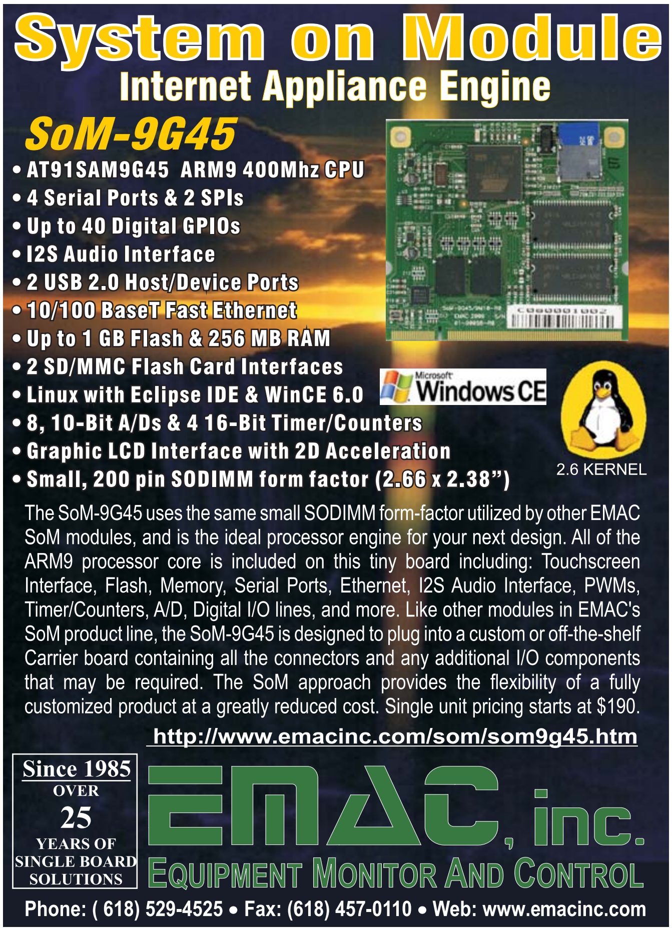
Phew, that's a lot of clicking!

Okay, it's not so bad when you are installing your first Drupal site, but it takes a lot of time with your mouse, and if you install (or re-install) Drupal sites a lot, it can become tedious. Let's look at that process with Drush:

1. Change into your Web server's root directory (`cd /var/www`).

2. Run `drush dl drupal-7.0`. This downloads Drupal 7.0 from drupal.org and extracts the files into /var/www/drupal-7.0.

3. Change into the newly created directory (`cd /var/www/drupal-7.0`) and run the `site-install` command:



```
drush site-install --db-url=mysql://root:secret@localhost/drupal
```

Answer "y" to the following prompt, and in a few short moments, you'll have a working Drupal install (using the "drupal" database on localhost). To verify, you can browse to http://localhost/drupal-7.0/ and log in using admin/admin as the user name/password, respectively. The site-install command has several additional options; run `drush help site-install` for full details.

What if you need a handful of Drupal-contributed modules for your site (as most people do)? Don't reach for your mouse yet! Let's grab "views" and "devel" while we're here (still in our Drupal directory):

```
drush dl ctools views devel
drush en views views_ui devel
```

(Note that ctools is a required dependency for Views in Drupal 7.)

The Drush dl command (a shortcut for pm-download) is very clever about how it downloads the modules and where it places the extracted folders. By default, it grabs the .tar.gz file from drupal.org and places folders inside the sites/all/modules directory, but you have full control (including using CVS checkouts or where the folders belong). See `drush help dl` for full details.

As with most tools, people develop their own habits with Drush, but here are a few built-in commands you are likely to appreciate (at least in time):

- `drush cc` (or cache-clear): Drupal developers learn early on to "clear the cache" while developing, because Drupal aggressively caches its more complex internal data structures. Chances are good that if you're writing code, you have a terminal window nearby, and running `drush cc all` will ensure that when you check your latest work in the browser, you'll accurately see the most recent changes.

- `drush cron`: the Drupal installation guide (INSTALL.txt) recommends running Drupal's periodic maintenance tasks by adding a crontab entry that uses wget to fetch http://example.com/cron.php periodically. In Drupal 7, this now requires a generated "cron_key" parameter as part of the URL to inhibit nefarious requests. Using Drush eliminates the need for the key and avoids tying up a Web server process to do periodic maintenance.

- `drush up` (or pm-update): this command checks drupal.org for new releases for all installed modules and offers to download all updates and run the database upgrades for you. It even handles that pesky, easily forgotten "backup first" step.

Running `drush help` in your Drupal directory always will give you the list of currently available commands (based on modules installed and so forth). For more information on each command, run `drush help <command>`.

The Power of Aliases

You may have noticed that we ran our drush commands above from inside the Drupal install directory. This allows Drush to determine which Drupal site (and configuration options) to use. However, for those looking for a bit more freedom and particularly those working within Drupal "multi-site" environments, you need to make use of the --root and --uri switches for Drush. This can lead to a lot of repetitive typing. If you have only a single Drupal site, you can specify these options (and lots more) by creating a .drushrc.php file in your home directory. See the example.drushrc.php file in the examples folder of your Drush installation for more details.

For those of you who work on several Drupal sites across various versions and installations, Drush provides an "alias" mechanism to define the common parameters. To create an alias for this example site here, you can create a file ~/.drush/aliases.drushrc.php containing the following PHP code:

```
$aliases['example'] = array(
  'root' => '/var/www/example/drupal',
  'uri' => 'example.com',
);
```

Now you can run drush commands for your example site from any directory using `drush @example <command>` (for example, `drush @example status`). To make things even better, aliases can reference remote sites (accessible via SSH) by including the remote-host and remote-user options. See the example.aliases.drushrc.php file from the examples folder in your Drush install directory for full details.

The Sky Is the Limit

With each new release, Drush includes more commands, and Drupal modules are increasingly offering Drush commands with their releases. If that's not enough (and you aren't afraid of writing some PHP), you can write your own Drush commands. The examples folder in your Drush install includes a sample command, `make-me-a-sandwich` (XKCD fans rejoice!), for those interested in learning more.

One of the greatest testaments to the power of Drush is the Aegir Hosting Platform (see Resources), which provides a mass-hosting environment (with support for multiserver installations). Although it includes a Web interface, Aegir does all of its heavy lifting via custom Drush commands.

If you work with Drupal and hate using your mouse (or prefer your terminal), Drush can save you countless hours of clicking and open a whole new world of scripted Drupal tasks.■

James Walker is a longtime Drupal developer, consultant, trainer and advocate. He is a co-author of the O'Reilly book *Using Drupal* and often can be found speaking at Drupal conferences and events. Find him on-line as "walkah" or via walkah.net.

Resources

Download Drush 4: **drupal.org/project/drush**

Handy On-Line Reference (for those who still prefer Web pages): **drush.ws**

Modules That Already Include Drush Support: **drupal.org/project/modules?filters=tid:4654**

Drush Aliases: **developmentseed.org/blog/2010/mar/10/drush-30-more-powerful-flexible-and-magical**

For Shell Completion Junkies: **drupal.org/node/437568**

Aegir for Mass-Hosting Drupal: **www.aegirproject.org**

2011 High Performance Computing Financial Markets – Plan to attend.
Two successful Wall Street events – April 4 and Sept 19, Roosevelt Hotel, NY

2011 8th Annual High Performance Linux Financial Markets Show and Conference

April 4, 2011 (Monday) **Roosevelt Hotel, NYC**
Madison Ave and 45th St, next to Grand Central Station

8th Annual High Performance Computing Financial markets Show and Conference

September 19, 2011 (Monday) **Roosevelt Hotel, NYC**
Madison Ave and 45th St, next to Grand Central Station

The 2011 High Performance Computing Wall Street Shows will assemble 800 -1,000 Wall Street professionals at the convenient and affordable Roosevelt Hotel, April 4 and September 19.

Plan to be there. See live demonstrations of speed, low latency, network solutions, data storage solutions and where to reduce costs for your tight budgets.

The 2011 8th Annual High Performance Computing Linux, April 4, Monday, Roosevelt Hotel will focus on High Performance Computing, Low Latency, Linux and Open Source, Virtualization, Cloud, Grid, Blade, Cluster, Ultra High Speed Networking and HPC Data Center Solutions for Wall Street.

The 2011 8th Annual High Performance Computing, Sept 19, Monday, Roosevelt Hotel will assemble IT professionals and Wall street business managers to see the latest trends in High Performance Computing, Low Latency, Virtualization, Cloud, Linux, Grid, Blade, ClusterUltra High Speed Networking and HPC Data Center Solutions for Wall Street.

The Roosevelt Hotel is convenient. A short walk from most midtown offices.

Plan to attend. Show is Free. 8-4 pm, but you must register in advance to receive your free show badge.

Conference program 8:30-4:50 pm with Plenary Sessions, Drilldown Sessions, a Wall Street luncheon, handouts and more at $295. Save $100. $395 on site.

"The show floor was packed with Wall Street solutions including HPC, low latency, storage and data center hot new products.""

This is the largest New York Show for HPC and low latency.

2010-11 Sponsors

Microsoft IBM intel hp cisco Novell
ORACLE DELL FUJITSU redhat FUSION-io
Solace Systems SUPERMICRO Myricom EXAR sgi
ScaleMP VERSANT TIBCO ARISTA FORSYTHE Realize the business value of IT.
BLADE NETWORK TECHNOLOGIES Corvil APPRO vmware FULCRUM

Media Sponsors

LINUX JOURNAL Global Investment Technology SECURITIES INDUSTRY NEWS LINUX MAGAZINE ADMIN Network & Security
UBUNTU user TRADERS MAGAZINE HPCwire SECURITIES TECHNOLOGY MONITOR waters
Inside Market Data HPC In the Cloud DWT WINDOWS IN FINANCIAL SERVICES WSTA

Show & Conference:
Flagg Management Inc
353 Lexington Ave, NY10016
(212) 286 0333 flaggmgmt@msn.com

Register today. Register online at:

www.flaggmgmt.com/linux
www. flaggmgmt.com/hpc

SIMPLICITY AND PERFORMANCE: JavaScript on the Server

The award for the hottest new server Web development language goes to...JavaScript! JavaScript, language of the browser since the early days, is the hottest language for server-side development. See why JavaScript is the language for developing quick, easy and incredibly powerful server-side applications.

Avi Deitcher

For years, Douglas Crockford, the high priest of JavaScript (JS), has claimed that it is a powerful, flexible language suited to a multitude of tasks, especially if you can separate it from the ugly browser-side piece that is the Document Object Model, or DOM. Because of the browser, JavaScript is the most popular programming language around by number of users. Job sites dice.com and monster.com post more jobs for JavaScript than any other language, except Java. Of course, if JavaScript runs in a browser, or anywhere, it must have an engine. Those engines have been around since the earliest JS-capable browsers, and they have been available as separate standalone entities for several years. Thus, the *potential* for running JS on its own always has been available. However, JavaScript always has missed two critical elements to make it *worthwhile* to run on the server side.

FREE TRADE AGREEMENT

The first missing piece was a common set of libraries. Quite simply, because JS was so focused on the browser, it was missing basic I/O libraries, such as file reading and writing, network port creation and listening, and other elements that can be found in any decent standalone language. Ruby includes them natively; Java includes them in its java.io and java.net packages. For JavaScript, running alone when all you can do is process text and data structures, but not communicate with the outside world, was rather useless. Over the years, several attempts have been made to make some form of JS I/O and Net packages, mostly wrapped around native C calls, if the JS engine was written in C, such as SpiderMonkey, or java.io, and java.net calls, if the JS engine was written in Java, for example, Rhino.

This began to change in early 2009 with the creation of the CommonJS Project (which, for some mystical reason, stands for Common JavaScript), which unified these efforts under a common namespace, with JS-specific semantics and included a package-inclusion system to boot.

Using Rhino as an example, you could read from a file using:

```
defineClass("File");
var f = new File("myfile.txt"), line;
while ((line = f.readLine()) !== null) {
   // do some processing
}

// this example slightly modified and simplified
// from the Mozilla Rhino site
```

As you can see, this is not file processing in JavaScript; it is file processing in Java! All I have done is opened the Java API to JavaScript. It is great if you really intend to program in Java, but it's of limited help if you are trying to do pure JS, and especially if your engine is not Java-based.

With CommonJS, there emerged a standard JavaScript-native interface to include a package, for example an I/O package or http package, and define many of the standard functionalities. Under the covers, the implementation may be C, Java, Erlang or Gobbledygook. All that matters is that the interface to the developer is platform-agnostic and portable from one interpreter to another.

THE MISSING NODE

The second missing piece was a server, similar either to Tomcat/Jetty for Java or Mongrel/Thin for Ruby, that provides a real environment, includes the necessary modules and is easy to use. Most important, it needed to take advantage of JavaScript's strengths, rather than attempt to copy a system that works for Java or Ruby. The real breakthrough was Ryan Dahl's Node.JS. Ryan combined Google's highly performant V8 engine, JavaScript's natural asynchronous semantics, a module system and the basic modules to create a server that suits JavaScript to a tee.

Most Web servers have a primary process that receives each new request. It then either forks a new process to handle the specific request, while the parent listens for more requests, or creates a new thread to do the same, essentially the same method if somewhat more efficient. The problem with processes or threads is threefold. First, they require significant resource usage (memory and CPU) for a small amount of differing code. Second, these threads often will block on various activities, such as filesystem or network access, tying up precious resources. Finally, threads and processes require context switches in the CPU. As good as modern operating systems are, context switches still are expensive.

The alternative, gaining in popularity, is event-driven, asynchronous callbacks. In an event model, everything runs in one thread. However, each request does not have its own thread. Rather, each request has a callback that is invoked when an event—like a new connection request—occurs. Several products already take advantage of the event-driven model. Nginx is a Web server with similar CPU utilization characteristics to dominant Apache, but with constant memory usage, no matter how many simultaneous requests it serves. The same model has been taken to Ruby using EventMachine.

As anyone who has programmed in JavaScript, and especially in asynchronous AJAX, knows, JS is extremely well suited to event-driven programming. Node.JS brilliantly combines packaging and an asynchronous event-driven model with a first-rate JS engine to create an incredibly lightweight, easy-to-use yet powerful server-side engine. Node has been in existence for less than two years and was first released to the world at large only at the end of May 2009, yet it has seen widespread adoption and has served as a catalyst for many other frameworks and projects. Quite simply, Node changes the way we write high-performance server-side nodes (pun intended) and opens up a whole new vista.

The rest of this article explores installing Node and creating two sample applications. One is the classic “hello world”, a starting point for every programming example, and the other is a simple static file Web server. More complex applications, Node-based development frameworks, package managers for Node, available hosting environments and how to host your own Node environment, will be subjects for future articles.

INSTALLING NODE

Node will install on almost any platform, but it is ideally suited to UNIX-like environments, such as Linux, UNIX and Mac OS X. It can be installed on Windows, using Cygwin, but it is not as easy as the other platforms and there are plenty of gotchas. Like most server-side packages, if you want to do anything serious, do it on UNIX/Linux/BSD.

On Linux or UNIX, installation follows typical UNIX program installation: download, configure, make, make install.

First, download the latest package. At the time of this writing, the latest unstable version is 0.3.2, and the latest stable is 0.2.5. I recommend moving toward 0.3+ as soon as possible. Don’t be fooled by the low version numbers; plenty of production sites are using Node right now for at least part of their environment, including github.com.

You can download the tarball directly from **nodejs.org**, or clone the github repository, my preferred method. If you don’t have git installed already, do so via your preferred package manager or directly. Before you get started, make sure you have the prerequisites. Although I could include them here, the details of building git are beyond the scope of this article.

On Mac OS X:

```
# install XCode from the Apple developer Web site
$ brew install git
```

On Linux or similar with the apt packaging system:

```
$ sudo apt-get install g++ curl libssl-dev apache2-utils
$ sudo apt-get install git-core
```

Now, you are ready to download, compile and install Node. First, you need to `cd` to the appropriate directory. At that point, clone the git repository:

```
$ git clone git://github.com/ry/node.git
# if you have problems with git protocol, http works fine
$ git clone http://github.com/ry/node.git
```

Next, make sure you are in the right version. Because git clones the entire repository, make sure you switch to the correct version:

```
$ cd node
$ git checkout <version>
# version can be whichever you want,
# but I recommend v0.3.2 as of this writing
```

Then, run `configure`. As usual, configure will check whether you have all of the prerequisites installed. Configure also will determine where to install Node when it is ready. Unless you are working on a production machine, I highly recommend installing Node in a local writable repository under your home directory, such as ~/local/.

Installing git in the default /usr/local/bin/ leads to all sorts of interesting permission issues when installing packages and running everything as sudo during installs. Unless it is going to be shared among everyone and used in production, installation makes a lot more sense in your own directory. It is also quite small. The entire installation on my laptop, including binaries, man pages and several add-on packages, is less than 50MB. The Node binary itself is less than 5MB:

```
# installing in the default
$ ./configure

# installing in your own local directory,
# my preferred method
$ ./configure --prefix=~/local
```

Then, compile and install:

```
$ make
$ make install
```

At this point, Node is installed and ready to run. If you installed Node in ~/local/, you need to add ~/local/bin to your path, which depends on your shell.

A NEW MINDSET

The critical thing to remember about Node development is that everything important is asynchronous. Sure, you could do many things synchronously, but why would you?

For example, a traditional Web programming model might look something like this:

```
// pseudo-code
conn = connection.waitForRequest();
if (conn != null) {
    request = conn.getRequest();
    response = conn.getResponse();
    data = database.getData(query);
    response.write(someData);
}
```

In asynchronous Node, you would do something more like this:

```
server.handleRequest(function(request,response) {
    // we need some data from the database
    database.submitQuery(query,function(data) {
        response.write(data);
    });
});
```

Notice how everything is in callbacks, an event-driven asynchronous model.

SAMPLE PROGRAM ONE: HELLO WORLD

Everything starts with hello world. This example demonstrates the basics of modules and asynchronous handling of requests.

First, include the necessary http module:

```
var http = require('http');
```

http is a standard module included with Node. If you wanted a module that was not in the standard path, you would preface it with `./`, which is executed relative to the path in which the app is running. For example, `require("./mymodule");`.

Next, create the server, which is as simple as `createServer()`, as well as the callback function to handle each request:

```
http.createServer( function(request, response) {
    // handling code here
});
```

Next, put in the handling code. You know you want the response to be hello world and the http status code to be 200, which is basic success:

```
http.createServer( function(request, response) {
    // set your status code to 200 and content to plain text,
    // since "hello, world!" is as plain as it gets
    response.writeHead(200,{"Content-Type": "text/plain"});
    // write out our content
    response.write("Hello, world!\n");
    // indicate that we are done
    response.end();
});
```

The above is a callback function. It will be called each and every time a new connection request comes in.

Finally, you need to tell the server to listen and on which port. For now, let's put it on 8080 (just to annoy Tomcat):

```
http.createServer( callbackFunction ).listen(8080);
```

Pulling it all together, you get a very simple program:

```
var http = require('http');
http.createServer( function(request, response) {
    // set your status code to 200 and content to plain text,
    // since "hello, world!" is as plain as it gets
    response.writeHead(200,{"Content-Type": "text/plain"});
    // write out our content
    response.write("Hello, world!\n");
    // indicate that we are done
    response.end();
}).listen(8080);
```

Six lines of code, and a functioning Web server that says "Hello, world!" Save the file as app.js, and then run it:

```
# cd to your development directory
$ cd workingdir
$ node ./app.js
```

Connect your browser to http://localhost:8080, or use curl or wget, and you will see "Hello, world!"

SAMPLE PROGRAM TWO: SIMPLE FILE SERVER

For the next example, let's serve up files from the local filesystem. If the file is available in the document root, let's return it with a 200 response; if it is not, let's return a 404 status code and an error message.

Like last time, you need the http module. Unlike last time, you also need the modules to read from the filesystem, and an ability to process URLs:

```
var http = require('http'), fs = require('fs'),
 ➥path = require('path'), url = require('url');
```

Create the server and its handler, and listen on port 8080 (just to annoy Tomcat) in the same way as last time:

```
http.createServer( function(request, response) {
   // handling code
}).listen(8080);
```

The difference is in the handling code. Now, when you get a request, you want to see whether it exists in the filesystem, and if so, return it:

```
http.createServer( function(request, response) {
   // __dirname is a special variable set by node
   var file = __dirname+path;
   // check if the requested path exists
   path.exists(file, function(exists) {
      if (exists) {
      } else {
      });
   });
}).listen(8080);
```

You use the path module to check whether the file is available, but you do it asynchronously. Normally, file access is very slow, and everything in the thread or process will block. With Node's event-driven model, nothing blocks; rather, the system continues to move and calls the function(exists) callback when it has an answer if the file exists.

If the file does exist, you need to read it using the "file" module and send it back. If it doesn't, you send back a 404 error. First, let's look at the simple file-not-found case:

```
if (exists) {
   // do some handling
} else {
   response.writeHead(404, {"Content-Type": "text/plain"});
   response.write("404 Not Found\n");
   response.end();
}
```

Now, let's look at reading the file and sending it back when it does exist. Once again, read the file asynchronously:

```
if (exists) {
   // read the file asynchronously
   fs.readFile(file,"binary",function(err,file) {
      if (err) {
         // we got some kind of error, report it
         response.writeHead(500,{"Content-Type":"text/plain"});
         response.write(err+"\n");
         response.end();
      } else {
         response.writeHead(200,{"Content-Type":"text/html"});
         response.write(file,"binary");
         response.end();
      }
   });
}
```

Tying it all together and cleaning it up a bit, you get a nice tidy, asynchronous, event-driven Web file server:

```
var http = require('http'), fs = require('fs'),
 ➥path = require('path'), url = require('url');
http.createServer( function(request, response) {
   var file = __dirname+url.parse('url').pathname;
   // check if the requested path exists
   path.exists(file, function(exists) {
      if (exists) {
         fs.readFile(file,"binary",function(err,file) {
            if (err) {
               response.writeHead(500,{"Content-Type":"text/plain"});
               response.write(err+"\n");
               response.end();
            } else {
               response.writeHead(200,{"Content-Type":"text/html"});
               response.write(file,"binary");
               response.end();
            }
         });
      } else {
         response.writeHead(404, {"Content-Type": "text/plain"});
         response.write("404 Not Found\n");
         response.end();
      }
   });
}).listen(8080);
```

A static Web file server, which will outperform most such servers on the market, in just 23 lines of code—it's a work of Art.

SUMMARY

Node.JS is an incredibly powerful, simple and elegant engine to run event-driven server-side JavaScript, and it has been a catalyst for an enormous amount of fermentation in the server-side world during the past year and a half.■

Avi Deitcher is an operations and technology consultant based in New York and Israel who has been involved in technology since the days of the Z80 and Apple II. He has a BS in Electrical Engineering from Columbia University and an MBA from Duke University. He can be reached at avi@atomicinc.com.

Resources

Node.JS: **nodejs.org**

Node.JS Git Repo: **github.com/ry/node**

CommonJS: **www.commonjs.org**

Cygwin: **www.cygwin.com**

Nginx: **nginx.org**

Douglas Crockford: **www.crockford.com**

Language Popularity:
www.webdirections.org/the-state-of-the-web-2008

Zotonic: THE ERLANG CONTENT MANAGEMENT SYSTEM

It's more than just a CMS. Create complicated Web sites quickly with Zotonic.

Michael Connors

Described by its authors as a pragmatic and modern CMS, Zotonic is that and much more. When I started using Zotonic, it was because of its efficiency and the fact that I could pack several client CMS sites onto a machine with only humble resources. I soon discovered, however, that Zotonic is not only a CMS, but also a Web framework, which allows me to create very complicated Web sites in a fraction of the time it would have taken me using more traditional languages and frameworks. Zotonic won't fall over if it encounters an error, and it does not need to be poked with a stick and awakened every time a request comes in.

Zotonic is written in Erlang, a functional language that was designed for programming telephone switches. The logic behind using Erlang for Web development is that modern Web sites, with their plethora of connections from users and robots, are starting to look more and more like telephone exchanges. "I have never programmed in a functional language, and Erlang looks like Dutch to me!", I hear you say. Well, the authors of Zotonic are fluent in Erlang (and Dutch, incidentally), and they have done a good job of creating a piece of software that is useful out of the box, regardless of whether you know Erlang, and Zotonic could be just the killer app you need to dive in and learn Erlang.

Another attractive feature of Zotonic is its PostgreSQL database (see sidebar). As someone who has toyed with learning Erlang for a while, probably one of the big barriers was that on top of learning a completely new programming paradigm, I also would have to learn a new database in the form of mnesia. Zotonic's use of PostgreSQL means one less new thing to learn and at least allows me to feel in familiar territory when I am designing my data.

NOTE:

The only database offered by Zotonic at the moment is PostgreSQL; however, there are plans to create "Elastic Zotonic", which will use a distributed store. Although Zotonic does use a relational database, in most cases, what you are inserting into the database will be a Zotonic resource with a key (ID) and a document (Props). Relationships between resources are defined using predicates, such as has_relation and has_part. For a lot of Web development, this is all you need; however, if you do need it, the power of the relational database is available to you.

Dependencies

I am running the latest version of Ubuntu, which has Erlang preinstalled. You can test whether you have Erlang by typing `erl` at the command line. If you get the Erlang shell, you are good to go. Press Ctrl-c, followed by the letter a and carriage return to exit Erlang. If you don't have Erlang on your system, you can download it from the Erlang Web site or install it with your distribution's package manager.

Another dependency is ImageMagick; to check whether it's installed, run:

```
convert -version
```

You, of course, need to have PostgreSQL installed, and you need Mercurial installed to fetch the latest version of Zotonic from the Google code site.

Installing and Configuring Zotonic

Fetch the Zotonic source and build it:

```
hg clone https://zotonic.googlecode.com/hg/ zotonic
cd zotonic
make
```

Now, create a database for Zotonic:

```
CREATE USER zotonic WITH PASSWORD 'yourdbpassword';
CREATE DATABASE zotonic
    WITH OWNER = zotonic ENCODING = 'UTF8';
GRANT ALL ON DATABASE zotonic TO zotonic;
\c zotonic
CREATE LANGUAGE "plpgsql";
```

The Default Site

Zotonic comes complete with an example site, which implements a simple blog. You can find the code for this default site in priv/sites/default/, and you can get this default site running by creating a config file and starting Zotonic.

Find the sample config file in priv/sites/default/config.in, and rename it or create a copy with no extension:

```
cp priv/sites/default/config.in priv/sites/default/config
```

Open config in your favourite text editor, and modify it to use the database you just created:

```
% Hostname for virtual host support
{hostname, "127.0.0.1:8000"},
{hostalias, "localhost:8000"},
% PostgreSQL database connection
{dbhost, "127.0.0.1"},
{dbport, 5432},
{dbuser, "zotonic"},
{dbpassword, "yourdbpassword"},
{dbdatabase, "zotonic"},
```

Now, start Zotonic in debug mode using start.sh:

```
./start.sh
```

ZOTONIC'S USE OF POSTGRESQL MEANS ONE LESS NEW THING TO LEARN AND AT LEAST ALLOWS ME TO FEEL IN FAMILIAR TERRITORY WHEN I AM DESIGNING MY DATA.

You should see text fly by on the console that suggests some tables are being created. Point your browser at 127.0.0.1:8000, and you should see your new blog.

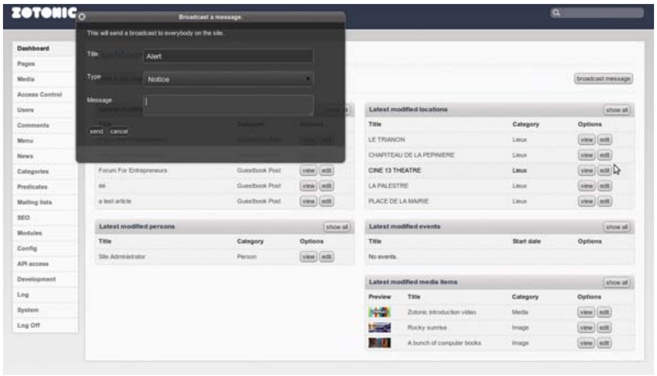
Using Zotonic for Content Management

Zotonic is first and foremost a content management system—that is what it says on the tin. Now that you have a running version of Zotonic, you can try out the content management features.

Point your browser at http://127.0.0.1:8000/admin. You will see a login screen, and as this is your first login, you need to set the password.

Use the login name "admin", leave the password field empty, and click Log On. You will see the create password form.

Once you have set your password, you are presented with the Zotonic admin dashboard. Down on the left-hand side is the admin menu. Much of it is straightforward, but one of the more interesting items here is modules.



Broadcast a message.

This will send a broadcast to everybody on the site.

Title: Alert

Type: Notice

Message:

Forum For Entrepreneurs	Guestbook Post	view edit
dd	Guestbook Post	view edit
a test article	Guestbook Post	view edit

Latest modified locations

Title	Category	Options
LE TRIANON	Lieux	view edit
CHAPITEAU DE LA PEPINIERE	Lieux	view edit
CINE 13 THEATRE	Lieux	view edit
LA PALESTRE	Lieux	view edit
PLACE DE LA MAIRIE	Lieux	view edit

Latest modified persons

Title	Category	Options
Site Administrator	Person	view edit

Latest modified events

Title	Start date	Options
No events.		

Latest modified media items

Preview	Title	Category	Options
	Zotonic introduction video	Media	view edit
	Rocky sunrise	Image	view edit
	A bunch of computer books	Image	view edit

Figure 1. Zotonic Dashboard and Broadcast Dialog

In the Modules menu item, there's a list of available modules, some of them activated, others not. Activate the comments module to see the comments form appear at the bottom of your blog posts.

You can create a new blog post by creating a new page of category "article".

In the list of pages, find the home page. You also can find it by searching for "home" using the search box on the top right of the admin page.

Open this page for editing, and scroll down it until you see advanced options. Expand the advanced options section, and notice that the home page has a unique name of page_home set. This is useful for referring to this page later in your code.

Customizing the Front End

Zotonic uses a modified version of Erlydtl for templating. Erlydtl is an Erlang implementation of the Django templating language.

Look in the templates folder of the default Zotonic site (priv/sites/default/templates). Here you will find a collection of .tpl files, which are templates that define the site. Templates that start with an underscore are not intended to be rendered alone, but rather can be included in another template.

Most of the templates in this directory inherit from base.tpl, which includes the site's header, menu and footer. This site uses article.tpl for displaying pages of the category "article" and uses home.tpl to display the home page.

Dispatch rules map URIs to resources. Look in the file priv/sites/default/dispatch/dispatch. The following two dispatch rules are defined:

```
{
     home,  [],
     resource_page,
     [ {template, "home.tpl"}, {id, page_home} ]
 },
{
     article, ["article", id, slug],
     resource_page,
     [ {template, "article.tpl"}, {cat, article} ]
},
```

The first rule states that you render the page with the unique name of page_home using the template home.tpl.

The second rule says that you want to render all pages of category article to article.tpl. You also define the structure of the URL in this rule. Each article will have an address of this form: /article/id/page-name.

In both of these examples, you use resource_page to do the actual rendering. This renders a resource as an HTML page and gives you access to the ID and category of the page from the template.

Other text pages you create will be rendered to page.tpl by default.

An about page, with the unique name page_about already exists in the default Zotonic site; it is currently rendered to page.tpl. Let's try making our own template to display the about page.

Create a template in the templates directory called about.tpl, and put the following code in it:

```
{% extends "base.tpl" %}

{% block title %}
    {{ m.rsc[id].title }}
{% endblock %}

{% block content %}

    <h1>{{ m.rsc[id].title }}
             -- {{ m.rsc[id].summary }}</h1>
   <h2>Hello, this is my about page!</h2>

    {{ m.rsc[id].body|show_media }}

{% endblock %}
```

Add the following to your dispatch rules:

```
{about,      ["about"],
                  resource_page,
                  [
                      {template, "about.tpl"},
                      {id, page_about}
                  ]
}
```

Stop Zotonic, and run `make`.

Start Zotonic again. Now, if you go to http://127.0.0.1:8000/about in your browser, you will see the text from the default about page rendered using the new template you created.

Having access to the ID of the about resource from the template, you can make calls to the database to retrieve other information to display. As you can see from the template above, I used the title (m.rsc[id].title), the summary (m.rsc[id].summary) and the body (m.rsc[id].body). I also used a "filter" called show_media to convert image markers in the body text into actual image tags for display.

Summary of Some Other Front-End Tools

You already have seen the show_media filter above, and many other filters exist to transform data for output. Other than filters, front-end development in Zotonic is aided by tags and scomps.

In the example above, I used the block tag to replace the content area in the template that I'm extending. Other tags that I use often are if, for and lib:

```
{% if id == 1 %}
    <p>The ID is 1</p>
{% endif %}

{% for color in ["bleu", "blanc", "rouge"] %}
    <p>{{ color }}</p>
{% endfor %}
```

The lib tag can be used to import an aggregate of stylesheets or scripts to reduce the number of requests to the server:

```
{% lib
        "css/zp-menu.css"
        "css/zp-project.css"

%}
```

Scomps, or screen components, are used when tags are not powerful enough and more logic is needed. The scomps that I use most frequently are menu and validate.

Menu is used to insert the standard Zotonic menu into your site:

```
{% menu id=id %}
```

Validate is used to validate a form field at both the front end and the back end:

```
<input
    type="password"
    id="password"
    name="password" value="" />
<input
    type="password"
    id="password2"
    name="password2" value="" />
{% validate id="password"
    type={confirmation match="password2"} %}
```

Extending the Back End

If you are willing to write some Erlang code, Zotonic can become much more than just a content management system. You can extend Zotonic with modules. The modules can be stored in the modules subdirectory of your site.

To make a module, create a modules directory within your site if it does not already exist:

```
mdkir priv/sites/default/modules
```

Let's create a simple module that implements a Web site guestbook. Users will be able to see the existing guestbook posts and add a new post.

Create a directory called mod_guestbook within the modules directory:

```
mkdir priv/sites/default/modules/mod_guestbook
```

Using your favorite text editor, create a file in this directory called mod_guestbook.erl, and in this file, put the following code:

```
%% @author Michael Connors
%% @doc A guestbook module.
-module(mod_guestbook).
-author("Michael Connors <michael@bring42.net>").
-mod_title("Guestbook").
-mod_description("A simple guestbook module.").
-mod_prio(500).

%% interface functions
-export([
    init/1,
    datamodel/0
]).

-include_lib("zotonic.hrl").

%% @doc Initiates the server.
init(Context) ->
    %% Manage our data model
    z_datamodel:manage(?MODULE,
                       datamodel(),
                       Context).
datamodel() ->
    [{categories,
      [
        {gp,
         text,
         [{title, <<"Guestbook Post">>}]}
      ]
     }
    ].
```

YOU CAN EXTEND ZOTONIC WITH MODULES.

Stop Zotonic and run `make` again. This will build your new module. Now, restart Zotonic, and log in to the admin. Go to the modules page, and observe that there now is a new module called Guestbook.

You can see here the values defined in the code for author, mod_title, mod_description and mod_prio. The Prio value indicates the importance of the module—the highest being 1 and the default being 500. Modules with a higher priority are checked first for templates and scomps.

A percentage symbol in Erlang indicates a comment, so any of the lines in this code preceded by percentage symbols are ignored by the compiler. The first two lines, although comments, also contain special notation, which is used to document the program.

Here, I exported init/1. This is because it must be called by external modules; init has an arity of 1, meaning it takes 1 argument:

```
-export([
    init/1
]).
```

If I had a function that took two arguments, I would export it like this:

```
-export([
    itsname/2
]).
```

You do not need to export datamodel, because it is used only by the init function in this module.

Init will be called whenever your module is loaded, and the first time it is called, it will create a new category in Zotonic called guestbook_post. This will be a subcategory of "text" and will have the display name Guestbook Post.

For each guestbook post, you should have a title and summary—luckily, all pages in Zotonic already have a title and summary, so without doing anything else, you can add posts to your guestbook by creating pages of the category Guestbook Post. Create a few Guestbook Posts now, ensuring that you fill in the titles and summaries. Also, don't forget to tick Published; otherwise, they won't be visible to users who are not logged in. You can use these to test your guestbook's display, which I discuss next.

Templates

Create a new subdirectory in mod_guestbook called templates:

```
mkdir priv/sites/default/mod_guestbook/templates
```

Using your favorite text editor, create the following file named guestbook.tpl:

```
{% extends "base.tpl" %}

{% block content %}
<h1>Guestbook</h1>
<ul id="guestbook-posts" class="guestbook-posts">
{% with
    m.search[
        {query cat='gp' sort='-publication_start'}
    ] as posts %}
    {% for post in posts %}
        {% include "_guestbook_post.tpl" %}
    {% endfor %}
{% endwith %}
</ul>
{% include "_guestbook_form.tpl" %}
{% endblock %}
```

This template fetches the pages of category guestbook_post in order of publication date; it extends the base template of the site in which it is used and overrides the “content block” of that base template.

I also am including two other templates, _guestbook_post.tpl and _guestbook_form.tpl. I’ll create these templates later.

Next, you need a dispatch rule. Create a new subdirectory of mod_guestbook named dispatch:

```
mkdir priv/sites/default/mod_guestbook/dispatch
```

Using a text editor, create a file called dispatch (with no extension) in the dispatch folder. It should contain the following dispatch rule:

```
[
    {guestbook, ["guestbook"],
                resource_template,
                [ {template, "guestbook.tpl"}]}
].
```

The first parameter above is the name of the rule. This is followed by a list containing the URI scheme; in this case, it’s simply /guestbook. Let’s use a premade Zotonic resource called resource_template to do the rendering, and the template that you actually will be rendering is called guestbook.tpl.

Save everything, run `make` and then restart Zotonic. When you navigate to 127.0.0.1:8000/guestbook, you will see a page that simply contains the heading Guestbook.

In the template above, I included another template called _guestbook_post.tpl, which I did not create yet. This template will contain the details of each guestbook post and be rendered once for every guestbook post. Let’s create it now in the templates subdirectory of mod_guestbook:

```
<li class="guestbook-post">
        <p>{{ post.title }}-{{post.summary}}</p>
    </li>
```

Run `make` and reload Zotonic. You now should see the guestbook posts you created earlier in the admin.

The next step is to allow users to sign the guestbook by creating the _guestbook_form.tpl template:

```
{% wire id="guestbook-form"
           type="submit"
           postback={np}
           delegate="mod_guestbook" %}
<form id="guestbook-form"
            method="post" action="postback">
  <div>
    <div class="form-item">
      <label for="title">Title</label>
        <input type="text" name="title" id="title" />
        {% validate id="title" type={presence} %}
    </div>
    <div class="form-item">
      <label for="summary">Summary</label>
      <input type="text" name="summary" id="summary" />
      {% validate id="summary" type={presence} %}
    </div>
    <div class="form-item button-wrapper">
      <button type="submit">{_ Post _}</button>
    </div>
  </div>
</form>
```

You use the “wire” scomp to specify that the form with the `id="guestbook-form"` will be handled by the event function of mod_guestbook. You also use the “validate” scomp to check for the presence of the required fields. If you wanted the summary field to be optional, you could leave out the validate scomp for the summary field. Here, you just use the presence validator, but there are others, such as numericality, length, confirmation (for making sure two fields match) and the very useful format validator, which takes a regular expression.

Now, you need to implement the event function of mod_guestbook to handle this post:

```
%% @doc Handle the submit event of guestbook
event({submit, {np, _}, _TriggerId, _TargetId}, C) ->
    T = z_context:get_q_validated("title", C),
    S = z_context:get_q_validated("summary", C),
    CatId =
    m_category:name_to_id_check(gp, C),
    AC = z_acl:sudo(C),
    Props = [
          {category_id, CatId},
          {title, T},
          {summary, S},
          {is_published, true}],
     {ok, RscID} = m_rsc:insert(Props, AC),
     Post = m_rsc:get(RscId, C),
     TemplateProps = [
          {post, Post}
     ],
Html = z_template:render("_guestbook_post.tpl",
                                    TemplateProps, AC),
z_render:insert_top("guestbook-posts",
                                    Html, AC).
```

Don't forget to export event/2. Now you are writing Erlang code and making use of some support functions that come with Zotonic.

If you are new to Erlang, the first thing to note is that once a variable has been bound to a value, it cannot be changed. This may seem strange, but the idea is to avoid side effects, so that you can write distributed applications. A nice side effect (I know) is that it makes Erlang easier to debug.

In Erlang, you use lists and tuples for storing aggregates of data. A list is enclosed in square brackets and a tuple in curly brackets. Variables in Erlang start with a capital letter (Props), and you also use atoms, which are lowercase. Atoms do not have any value associated with themselves; in essence, they are the value.

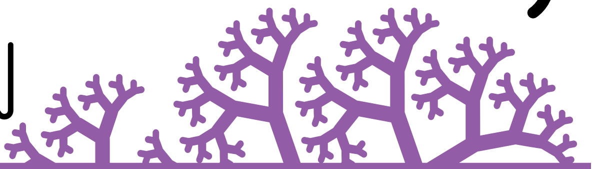
Functions that relate to access control are found in z_acl, and in this case, you use `z_acl:sudo` to gain superuser rights. `z_context:get_q_validated` allows you to get the contents of a validated field from the post; `z_template:render` returns a rendered template, and `z_render:insert_top` inserts some text at the top of an HTML element with a given ID. More support functions can be found in src/support.

IF YOU ARE NEW TO ERLANG, THE FIRST THING TO NOTE IS THAT ONCE A VARIABLE HAS BEEN BOUND TO A VALUE, IT CANNOT BE CHANGED.

The code for accessing the database is found in src/models. Here, I accessed the database to check the ID of a category (m_category:name_to_id_check) and also to insert a new resource (m_rsc:insert).

The guestbook is not completely finished. You still need to add the name of the person that signed it. This is easy, however, and you don't need to go near the database to do it. Simply add a new field to your form template, modify your event function to handle that field, and your guestbook will be complete.■

Michael Connors is a freelance software developer from Ireland, but he currently lives in Normandy, France. These days, he mostly develops software in Erlang.



Find Yourself with the Google Maps API

Making maps the Internet way.

MIKE DIEHL

I don't think a day goes by that I don't use Google to find something, and lately, I've seen an increasing number of businesses that post Google maps to their locations on their Web sites. Sometimes, these companies even go so far as to put little push-pins on the map indicating each of their locations, which makes things very convenient for navigationally challenged people like myself.

Listing 1. HTML for Google Map

```
1 <html>
2 <head>
3 <title>My Google Map</title>
4
5 <style>
6 #map {
7 position: relative;
8 left: 5px;
9 top: 5px;
10 width: 764px;
11 height: 520px;
12 }
13 </style>
14
15 <script src="http://maps.google.com/
➥maps?file=api&v=3&key=thisisasecret&sensor=false"
16 type="text/javascript">
17 </script>
18
19 <script type="text/javascript" src="/main.js"></script>
20
21 </head>
22
23 <body onload="initialize('map');" onunload="GUnload();">
24
25 <div id=map>
26 Map goes here.
27 </div>
28
29 <p>
30
31 SW Corner: <span id=debug_sw> </span><br>
32 NE Corner: <span id=debug_ne> </span><br>
33 Zoom: <span id=debug_zoom> </span><br>
34
35 </body>
36 </html>
```

I guess it would be possible to take a screenshot of a Google map and post it on a company Web site. I also guess you could open that image up in the GIMP and manually add a bunch of pushpins. Additionally, I guess you could use an image map to make the pushpins clickable and interactive. Yes, you could do things this way, and in a pinch, it might make sense, but it sure wouldn't be as much fun as using the Google Maps API and doing it right.

With the Google Maps API, you can create a map centered at a particular location. You can place colored push pins anywhere on the map, and you can draw geometric shapes on the map. Perhaps you even want to draw borders around delivery regions or school districts. The Google Maps API is incredibly powerful, and I've scratched the only surface myself.

However, before you can get started, you need to get an API key by registering with Google. This registration is free, and you receive your key instantly. You do, however, have to agree to some usage restrictions. Most of the restrictions seem reasonable. The only surprise is the Web site that uses the Google Maps API must be publicly accessible; it can't be on an intranet nor can it be password-protected. If you need to create an application that will not be publicly accessible, you can make other arrangements with Google. To sign up for a key, point a browser at **code.google.com/apis/maps/signup.html**. You will be asked for your Web site's domain name, and you'll need a separate key for each domain.

Once you've got an API key, you're ready to start coding. First, you need to create a simple Web page to display your map (Listing 1).

Lines 1–14 are simple boilerplate HTML. Note that I include some in-line styling for a div container called "map". Here I'm mostly just interested in setting the size of the resulting rectangle.

Lines 15–17 are where you load the Google Maps API. The section of the URI that looks like "v=3" indicates that I'm using version 3 of the API. This is also where you include the API key you obtained earlier. Finally, you see the "sensor=false" section of the URI. This indicates that I'm not using any type of location sensor, such as a GPS, to select the appropriate map. Accurately configuring this field is required by the Google Maps API EULA.

The JavaScript program that I wrote to load and manipulate the map is loaded on line 19. On line 23, I arrange for an initialization function that I wrote to be called when the page finishes loading and another function, that Google provides, when I close the page. I discuss the initialize() function shortly.

The rest of the HTML simply creates a container (mentioned earlier) to hold the map and a few other containers to hold debugging information. You might not want to display this information in a production application, but it's instructional to see what type of information is available from the API and what methods are available to the programmer for keeping the display up to date as the user interacts with the map.

The rest of the map is created in JavaScript, so let's take a look at Listing 2.

In lines 1 and 2, I create a global variable to hold the "map" object that the API will create. I also configure the latitude and longitude to point the map.

The initialize() function is found in lines 4–27 and does all the work of creating the map. In lines 5–10, I test to make sure that the user's Web client is able to display the map, and if so, I create the map object. Lines 12–18 configure the map. First, I select the location for the map to display. Then, I add the map type and map navigation controls. The map type control allows the user to select between a simple map, satellite map or hybrid map. The map navigation control allows the user to pan the map around and to zoom in and out. Finally, I configure the map to display as the hybrid map by default.

The update_gui() function referred to on lines 14, 20 and 26 simply updates the debugging information below the map and probably wouldn't be used in a production application. Line 20 is interesting because it demonstrates how to have your application react when the user scrolls or zooms the map to other locations. In this case, the application simply updates the lat/long coordinates below the map. I discuss the update_gui() function a bit more later.

At this point, if you did nothing else, you'd have a map that users could interact with. They'd be able to select the type of map, move it around and zoom in and out. But, let's go a bit further.

The ajax_get() function called on lines 22 and 24 isn't included in Listing 2, but it's relatively easy to write. This function simply accepts a URL and the name of a JavaScript function as parameters. Then, the function makes an AJAX call and fetches the data at the given URL. This data is assumed to be XML, which is passed to the indicated function.

The parse_markers() function referenced on line 22 accepts an XML string that describes where to put markers on the map. This XML resembles Listing 3. As you can see, it's simply a list of assets; each asset has an ID, a name, a description and a lat/long location.

The parse_zones() function on line 24 of Listing 2 works similarly and describes shapes to be drawn on the map. The corresponding XML looks like that shown in Listing 4.

Here, you see two containers and a list of lat/long points that define their boundaries. So, with what you have so far, and some slightly different data, you get a map that resembles Figure 1. By default, the map centers on a location just outside of Albuquerque, New Mexico. You see a red triangle that's filled in with green. The triangle is semi-transparent, so you can see the map through it. You also see one of the markers in the lower-right corner of the triangle. In Figure 1, I've clicked on the marker to demonstrate the information window, which I discuss a bit more later.

Let's take a closer look at the parse_markers() function defined on lines 29–51 of Listing 2. This function is fairly straightforward

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	1
Aberdeen, LLC	www.aberdeeninc.com	3
Emac, Inc.	www.emacinc.com	51
Flagg Management	www.flaggmgmt.com/hpc	53
Flourish! 2011	flourishconf.com	63
Genstor Systems, Inc.	www.genstor.com	49
Giada Technology, Inc.	www.giadatech.com	79
iXsystems, Inc.	www.ixsystems.com	C3
Linode, LLC	www.linode.com	41
LinuxFest Northwest	www.linuxfestnorthwest.org	5
Linux Journal Store	www.linuxjournalstore.com	7
Logic Supply, Inc.	www.logicsupply.com	25, 55
Lullabot	www.lullabot.com	9
MagnetCon	magnetcon.info	13
Microway, Inc.	www.microway.com	C2, C4
O'Reilly MySQL Conference	conferences.oreilly.com/mysql	33
O'Reilly Where 2.0 Conference	conferences.oreilly.com/where	33
Polywell Computers, Inc.	www.polywell.com	79
Silicon Mechanics	www.siliconmechanics.com	21, 29
SouthEast LinuxFest	southeastlinuxfest.org	39
Technologic Systems	www.embeddedx86.com	15
Texas Linux Fest	texaslinuxfest.org	43

ATTENTION ADVERTISERS

July 2011 Issue #207 Deadlines

Space Close: April 25; **Material Close:** May 3

Theme: Cool Projects

BONUS DISTRIBUTIONS:
USENIX Annual Tech, O'Reilly's Velocity

Contact **Joseph Krack, +1-713-344-1956 ext. 118, joseph@linuxjournal.com**

Listing 2. JavaScript for Google Map

```
var map;
var default_map = "35.181804,-105.40625,8";

function initialize (el) {
if (!GBrowserIsCompatible()) {
document.getElementById(el).innerHTML = "Incompatible Browser";
return;
}

map = new GMap2(document.getElementById("map"));

var l = default_map.split(",");
map.setCenter(new GLatLng(parseFloat(l[0]), parseFloat(l[1]))
➥,parseInt(l[2]));
update_gui();

map.addControl(new GMapTypeControl());
map.addControl(new GSmallMapControl());
map.setMapType(G_HYBRID_MAP);

GEvent.addListener(map, "mousemove", function () {update_gui();});

ajax_get("/markers.xml", "parse_markers");

ajax_get("/zones.xml", "parse_zones");

update_gui();
}

function parse_markers (e) {
var i, lon, lat;
var assets = e.getElementsByTagName("asset");

for (i=0; i<assets.length; i++) {
lon = parseFloat(assets[i].getAttribute("long"));
lat = parseFloat(assets[i].getAttribute("lat"));

var marker = new GMarker(new GLatLng(lat,lon));

marker.id = assets[i].getAttribute("id");
marker.name = assets[i].getAttribute("name");
marker.desc = assets[i].getAttribute("desc");

marker.long = lon;
marker.lat = lat;

map.addOverlay(marker);

//GEvent.addListener(marker, "mouseover", function ()
 ➥{marker_mouseover(this);});
GEvent.addListener(marker, "click", function ()
 ➥{marker_click(this);});
}
}

function parse_zones (e) {
var i,j;
var containers = e.getElementsByTagName("container");

for (i=0; i<containers.length; i++) {
var bounds = new Array;

var id = containers[i].getAttribute("id");
var name = containers[i].getAttribute("name");
var desc = containers[i].getAttribute("description");

var points = containers[i].getElementsByTagName("point");
for (j=0; j<points.length; j++) {
var p = new Object;

p.long = points[j].getAttribute("long");
p.lat = points[j].getAttribute("lat");

bounds.push(new GLatLng(p.lat,p.long));
}

var container = new GPolygon(bounds, "#ff0000",
 ➥5, .5, "00ff00", .2);

container.id = id;
container.name= name;
container.desc = desc;

map.addOverlay(container);
GEvent.addListener(container, "click", function ()
 ➥{zone_click(this);});

}
}

function marker_mouseover(who) {
map.openInfoWindow(new GlatLng(who.lat,who.long), who.name);
}

function marker_click(who) {
map.openInfoWindow(new GLatLng(who.lat,who.long), who.desc);
}

function zone_click(who) {
map.openInfoWindow(new GLatLng(who.lat,who.long), who.desc);
}

function update_gui () {
var sw = map.getBounds().getSouthWest();
var ne = map.getBounds().getNorthEast();

document.getElementById("debug_sw").innerHTML= sw.toString();
document.getElementById("debug_ne").innerHTML= ne.toString();
document.getElementById("debug_zoom").innerHTML= map.getZoom();
}
```

Listing 3. XML Listing of Markers

```
<assets>
<asset id="1" name="Home" desc="The home base" lat="35"
 ➥long="-105"></asset>
<asset id="2" name="BC Site" desc="The off-site site"
 ➥lat="34" long="-106"></asset>
</assets>
```

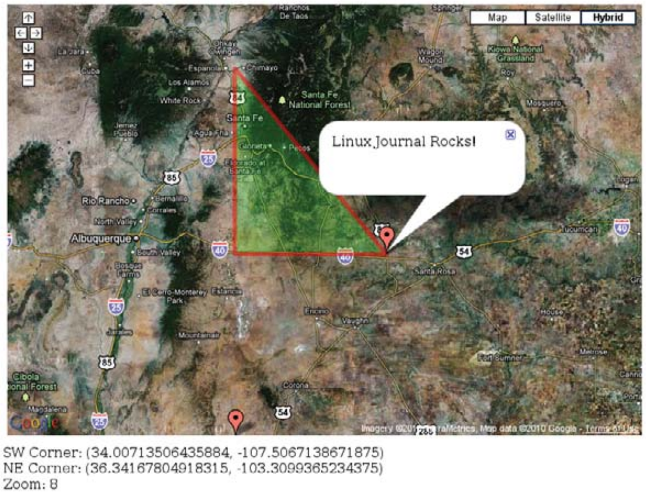
Listing 4. XML Listing of Zones

```
<containers>
<container id="1" name="HQ" description="This is HeadQuarters">
<point lat="35.0" long="-105.0" />
<point lat="35.0" long="-106.0" />
<point lat="36.0" long="-106.0" />
<point lat="35.0" long="-105.0" />
</container>
<container id="2" name="OffSite"
 ➥description="This is the Offsite Site">
<point lat="37.0" long="-104.0" />
<point lat="37.0" long="-105.0" />
<point lat="38.0" long="-105.0" />
<point lat="37.0" long="-104.0" />
</container>
</containers>
```

and simply loops over a list of assets. For each asset, the function parses the lat/long coordinates of the marker and creates a marker object with them (lines 34–37). Notice on line 37 that I had to use the lat and long variables to create an object to pass to the marker constructor. Next, I set some additional attributes for each marker for later use. Then, on line 46, I added the marker to the map. Lines 48 and 49 are interesting—they allow users to click on (or mouse over) a marker and display additional information. I've commented one of the lines out because it doesn't seem to work well to have a mouse-over and mouse-click event at the same time. I probably need some additional logic, but you get the idea.

In a real-world application, you might use the id attribute to make an AJAX callback to a server-side database and do some really cool things.

The parse_zones() function is only slightly more complex, because it has to build a GPolygon object from the points listed for each container. Lines 53–62 are similar to the first part of the parse_marker() function. The main difference is in lines 64–72 where I loop over each point that delineates the zone, create a GLatLng object for each point and push the object onto the bounds array. Then, on line 74, I create a GPolygon object with this array of points. The GPolygon constructor also allows you to specify a border color and size, as well as a fill color and opacity setting. I add the polygon to the map on line 80. I add an on-click event handler on line 81, so that when users click on a given zone, the application can perhaps provide more information about the zone.



SW Corner: (34.00713506435884, -107.5067138671875)
NE Corner: (36.34167804918315, -103.3099365234375)
Zoom: 8

Figure 1. Final Results

The event handlers defined on lines 86–96 are almost identical and quite trivial, so let me make some passing comments about them all at once. They each call the map object's openInfoWindow method to open the little message bubble and display a message. In these cases, I simply display the name or description of the object the user clicks on. In a real-world application, you might use the id attribute to make an AJAX callback to a server-side database and do some really cool things.

Finally, the update_gui() function on lines 98–105 is responsible for updating some of the display information at the bottom of the map. To do this, the function uses some of the map object's methods to get the coordinates of the southwest and northeast corners of the map. Then, those coordinates are converted to strings and placed inside the appropriate container on the Web page. The map's zoom factor is handled very similarly.

As you can see, using the Google Maps API is pretty easy. The API is fairly intuitive and exhaustively documented at Google. As I mentioned in the beginning of this article, it's pretty easy to plot businesses' office locations or the locations of their customers. But, this is the Web, and it should be fun. How about a geography quiz game with real maps? Or a *Risk*-like game, or any number of military simulations set on a virtual Earth? I recall playing a racing game that allowed you to drive a car around a Google Map. The Google Maps API is simple and powerful, and lots of interesting things are waiting to be done with it.■

Mike Diehl is a contract programmer and consultant in Albuquerque, New Mexico. Mike lives with his wife and three small boys and can be reached via e-mail at mdiehl@diehlnet.com.

Rich Internet Apps That Just Work—Writing for the User

AJAX is power. It makes Internet applications look, feel and perform in the eyes of the user like desktop apps, all while run from the server and written in the platform-agnostic languages of HTML and JavaScript. But, it carries a heavy price: breaking the browser. AVI DEITCHER

"The customer is always right." This time-worn adage—attributed to either Harry Selfridge, founder of the famous British Selfridges department store, or Marshall Field, of the Chicago department store that bears his name—has been discussed and dissected to no end. Undoubtedly, every one of us can come up with plenty of cases when customers aren't right, and it does not make sense to treat them that way. What is true, however, is that if you want to sell (or develop) something that's useful to customers, you must build it for the way they *actually* work, not the way you *want* them to work.

In the Web's early days, we were all entranced by the ability to access any application anywhere, without installing anything more than a browser. Developers loved the idea of writing in a single universal language. Even better, HTML is declarative—no interesting components and callbacks, no per-platform or per-OS-version oddities (more or less). Users loved the simple book paradigm. You could go back and forward (which, unsurprisingly, were the names of the buttons), and even click reload. The semantics were simple; writing for the platform was easy, and deployment, compared to managing each desktop, felt like the new Enlightenment.

The downsides, of course, were obvious, but a fair price to pay. If each page was statically generated with just HTML, every change, however small—say a change in text or adding a warning—required a complete page reload. Besides the headache for the user, it was unnatural and slow. Some pundits in the 1990s suggested that the Web would never be a dominant platform for this very reason. Dynamic HTML based on JavaScript, which allowed DOM manipulation, gave us some leeway, but anything that came from the server—real data—required a reload.

Enter AJAX

In the early to mid-2000s, developers began to explore how to communicate with the server without requiring page reload. Microsoft introduced the XMLHTTP ActiveX control in 1999, later adopted by every other browser. In 2005, Jesse Garrett, cofounder of Adaptive Path, coined the term Asynchronous JavaScript with XML, or AJAX. Although Jesse didn't invent it, he certainly popularized it, which once again underscores the importance of marketing that we engineers tend to overlook. As an interesting aside, one of the earliest known usages of AJAX occurred in...1596, by Sir John Harington, to describe his new invention: the flush toilet.

AJAX was wonderful. We could get what we wanted from the server without reloading the entire Web page. We could process it in the background. We could get as little or as much as we wanted. It seemed Web apps, now called Rich Internet Apps, finally were fully competitive with desktop apps in terms of ease of deployment *and* performance. It enabled such ubiquitous apps as Google Maps, which would have been impossible without AJAX.

The User Is the Problem?

The big problem with AJAX apps is that they broke Web semantics. The Refresh, Back and Forward buttons work entirely on the address in the URL bar of the browser. In the days of static pages, that mostly indicated where you were: http://example.com/store?product=12345 was definitely different from http://example.com/store?product=99999.

In the modern RIA AJAX world, however, the URL was http://example.com/store. With the product rendered using AJAX, the URL unchanged, reloading was highly unlikely to bring you back to where you were.

First Attempts

The first responses were to add complex state to the server. JavaEE, PHP frameworks and others all added session variables in which you could store oodles of information about what the user's last request was, and so you could roughly attempt to reconstruct it for the next request. The entire JavaServer Faces (JSF) framework is built around such complex state semantics. These did the job, more or less, but they were very complex and required lots of effort with which to work.

The next attempts essentially said, "we don't support browser buttons!" Put in other terms, "we and the technology are right, and the user is wrong." As anyone who ever has been in business knows, this strategy is doomed to failure. It may work, for a little while, if your customer has no alternative, but customers who are told they are wrong and "just don't get it" quickly will look for alternatives. Silicon Valley is littered with the corpses of startups that whined, "our customer just doesn't get it." Of course, it was the startup (and the engineers) who just didn't get it.

Technology-User Harmony

What we needed, then, was a way to use AJAX apps and modify the URL bar in a way that it would *not* reload the page, yet still give fairly complete indication of where we were. Thus, Back and Forward, not to mention Refresh, would work just fine.

The magic is in one little character, the hash (#). In the HTML specification RFC 1866, you can give a name to an anchor, as follows:

```
<a name="myname"/>
```

If you do so, a browser should be able to go to the named section on the page by appending # and the anchor name to the URL. For example, if you have an HTML page named mypage.html:

```
<html>
<head>
<!-- lots of stuff -->
</head>
```

```
<body>
<div>Lots of content</div>
<a name="part2"/>
<div>Even more content</div>
</body>
</html>
```

To get to the above page, you would go to http://example.com/mypage.html. But, if you wanted to go to that page and directly to part2, you would go to http://example.com/mypage.html#part2.

The most interesting part is that if the browser is *already* on mypage.html and you go to mypage.html#part2, the browser should, and will, go directly to part2 *without reloading the Web page*. Even further, if the browser cannot find an anchor named part2, it will fail silently and graciously. Last, but not least, JavaScript events can capture this change and process it.

With the above, we have the making of a system that uses AJAX for Rich Internet Application dynamism, yet can change the URL to indicate where we are, and thus work with, rather than against, the user. As a matter of fact, if you use Gmail and look closely, you will see that this is exactly how it works.

Of course, remembering to manage the URLs can be difficult and changes the way you work. Wouldn't someone have developed a framework to manage all of this?

Enter Sammy

Sammy is an amazing Web framework developed by Aaron Quint. Not only does it provide the framework for managing the URLs, as well as lots of additional functionality to boot, but it also actually dramatically improves how it writes your client-side apps. You move from programmatically driven to declarative. You return to the ease of use of the early Web 1.0 days, when the URL defined exactly where you were, but without giving up the dynamism of AJAX. Once again, the URL becomes the declarer of location in your app, and you can leverage its full power.

Getting Started

Let's explore a basic Sammy app. For our purposes, let's use a contact application. To keep things simple, let's not do any data updating in this article, although Sammy's semantics fully support it. Let's stick with simple GETs. In the contact application, we have ten contacts, each with the ID of 1 through 10 (complicated!), and each with properties of First Name, Last Name and Email. Our application view has a left pane, wherein contacts are listed, and a right pane, wherein contact details are shown. Remember, we want this to be a Rich Internet Application, all running in a single page.

Word of warning: the code in this article may be incomplete. If you want to download and run it, get the sample app off the Web (see Resources).

First, let's define our single HTML page contacts.html:

```
<html>
<head>
<script type="text/javascript" charset="utf-8"
 ➥src="jquery.min.js"></script>
<script type="text/javascript" charset="utf-8"
 ➥src="sammy.min.js"></script>
<script type="text/javascript" charset="utf-8"
 ➥src="contactapp.js"></script>
<style type="text/css">
  #list {float: left; width: 48%;}
  #details {float: left; width: 48%;}
</style>
</head>
<body>
<h2>Contact Application</h2>
<p>Click on a contact to view the details</p>
<div id="list">
  <table></table>
</div>
<div id="details">
<table>
  <tr><td>First Name:</td><td id="firstName"></td></tr>
  <tr><td>Last Name:</td><td id="lastName"></td></tr>
  <tr><td>Email:</td><td id="email"></td></tr>
</table>
</div>
</body>
</html>
```

Notice several elements:

- Installation: we included jQuery, a prerequisite for Sammy (and a really great library to boot).
- Installation: we included Sammy, after jQuery.
- HTML: the page is really simple. There are two blank divs, one with the ID list, the other with the ID details. They are floated.

Declaring Our Paths

Next, we need to declare what all the states are in which the application can exist. These will determine what paths we want. In our contacts app, we really have only two states: 1) listing the contacts and 2) viewing one particular contact (while the main list remains open).

In keeping with RESTful style, let's declare our URLs as follows:

1) Listing the contacts: `contacts.html#/contacts`.

2) Viewing one particular contact: `contacts.html#/contacts/:id` (where :id is replaced by the ID of the viewed contact).

In addition, we want a default path. What happens if the user just opens contacts.html?

3) Default path: `contacts.html`, re-routed to `contacts.html#/contacts`.

Notice something interesting. We are defining various declarative paths. When each of these paths is encountered, we want to take a certain action. Essentially, these are routes. Most Ruby-based frameworks (Sinatra, Rails, Merb/Rails3 and so on) use this exact language, as does Sammy.

So, we have three routes and their actions:

- `contacts.html`→redirect to `contacts.html#/contacts`.
- `contacts.html#/contacts`→list contacts.
- `contacts.html#/contacts/:id`→show details for `contact :id`.

In our included JavaScript file contactapp.js, we declare each of the routes:

```
var app = $.sammy(function(){
  // for the verb GET with the path #/, go to #/contacts
  this.get("#/",function(context){
    this.redirect("#/contacts");
  });

  // for the verb GET with the path #/contacts, render the contacts
  this.get("#/contacts",function(context){
    // get our contact list from the server
    $.get("/contacts",function(res,status) {
      // render the results - should include
      // status-checking for safety

      // jQuery already parsed the response to JSON for us
      var list = res, tr, td, table = $("#list table"), a;

      // clear the existing list
      table.empty();

      // use jQuery to go through each result
      $.each(list,function(i,elm) {
        tr = $("<tr></tr>").appendTo(table);
        td = $("<td></td>").appendTo(tr);
        // the key part: make it a URL
        a = $("<a></a>").attr("href","#/contacts/"+elm.id).text
        ➥(elm.lastName + " " + elm.firstName).appendTo(td);
      });
    },"json");

    // hide the details
    $("#details table").hide();
  });

  // for the verb GET with a specific path #/contacts/:id,
  // render that one contact
  this.get("#/contacts/:id",function(context){
    // get our contact list from the server - access
    // param :id as this.params.id
    $.get("/contacts/"+this.params.id,function(res,status) {
      // render the results - should include
      // status-checking for safety
      // jQuery already parsed the response to JSON for us
      var contact = res, table = $("#details table");

      // find the elements in the table, and fill them with the data
      $("#firstName",table).text(contact.firstName);
      $("#lastName",table).text(contact.lastName);
      $("#email",table).text(contact.email);

      // make sure the table is shown
      table.show();
    },"json");
  });

});

// set up a default route for contacts.html
$(function(){
  app.run("#/");
});
```

Notice several key elements:

1. There are no event handlers here at all. Although we might need some for things like edit buttons or key presses, navigation in the app really happens using URL <a> links. This makes it really easy to manage the app and understand what every change does. Clicking on a contact in the list is clicking on a URL. We just happen to use that URL to control our app.

2. We could have used a handler just as well. Instead of using an <a> link, we could have put on a handler with `$("list td").click(function(e),{...});`.

3. This application is incredibly short and easy to understand. That is the beauty of Sammy.

4. The browser URL changes, but the page does not reload. We remain in the Rich Internet App world, yet browser semantics simply work: Back, Forward, Reload. Try it!

The full sample, without minified JS, is available on-line (see Resources).

Summary

Sammy gives us the power to provide Rich Internet Applications simultaneously, work with the user's mindset rather than against it, and program our apps using routes declaratively, making it much simpler to build yet richer Internet applications.

The Sammy library is open source under the MIT license and available on-line (see Resources).■

Avi Deitcher is an operations and technology consultant based in New York and Israel who has been involved in technology since the days of the Z80 and Apple II. He has a BS in Electrical Engineering from Columbia University and an MBA from Duke University. He can be reached at avi@atomicinc.com.

Resources

jQuery: **jquery.com**

Ruby on Rails: **rubyonrails.org**

Sinatra for Ruby: **www.sinatrarb.com**

Sammy: **code.quirkey.com/sammy**

Sample from This Article: **jsorm.com/doc/samples/contacts/contacts.html**

RFC 1866: **www.rfc-editor.org/rfc/rfc1866.txt**

Douglas Crockford's JavaScript Site: **www.crockford.com**

Quick User Interfaces with Qt

The user interface is progressing quickly these days. It has been 15 years since cool 3-D buttons and the like, as popularized by Windows 95 and other early windowing environments, emerged. Now, we see halo effects, shades, transparency and more—all hardware-accelerated, and all making our computers look better than ever. JOHAN THELIN

A consequence of Nokia's acquisition of Trolltech, a lot of attention has been paid to Qt's abilities in the mobile device space. This not only means speed optimizations and support for more platforms, such as Symbian (and Android, if you look at community efforts), but it also means that what the Nokians refer to as device user interfaces receive quite a lot of attention.

A device user interface is basically a look and feel that integrates well with the device on which it is used. It also provides what modern consumers expect: fluid transitions, graphical effects and a polished look. The consequence of this is a move from a widget-based user interface to a scene-based one.

Qt still supports widgets, and many, if not most, applications still use them. As a matter of fact, new user interfaces are run in a specific widget—QGraphicsView. QGraphicsView, in turn, shows a QGraphicsScene, which contains QGraphicsItem instances. All of this then is managed by Qt Quick.

The Qt Quick concept consists of two parts. The first is the QML language, used to build Qt Quick user interfaces. The other is the QtDeclarative module that provides the means to execute QML components and integrate them with C++ code.

The reason for developing QML was that creating fluid user interfaces with C++ is becoming increasingly complex. By designing a language specifically for the task of doing that, the work effort needed is greatly reduced. This is done in a fashion so that Qt and C++ still can be used for their strong points, by implementing the user interface using QML and the business logic and parts requiring performance in C++. As a side effect, the always-wanted split of user interface code and the rest of the application is enforced, as the parts are implemented using different languages.

To understand how Qt Quick can be used, let's look at three aspects. First, QML in general, then how QML is used to build dynamic user interfaces and finally, how QML and C++ fit together.

Introduction to QML

QML is a declarative language, based on JavaScript. It is based on the concepts of components that are declared and properties that are bound. A simple example of this is an empty, rectangular scene:

```
import Qt 4.7

Rectangle {
    id: theRect

    width: 400
    height: width*1.5
}
```

In this snippet, the component Rectangle is instantiated. All words starting with an uppercase letter instantiate components. In the rectangle declaration, three properties are bound to values. The id property is special; it names items. In the future, the rectangle can be referenced as theRect. To access a property of the rectangle, such as its width, theRect.width can be used.

Next, the width is bound to the value 400, and the height is bound to the width times 1.5. Notice that the height is bound to width*1.5 and not assigned to the result of the multiplication. This means if the width changes, the height is updated automatically.

It also is worth noting the first line, which imports all components that are part of Qt version 4.7. This imports a set of components, such as the rectangle class, defined and implemented using C++. It is possible to import more C++-based components, components written in QML or entire modules of QML components.

I won't go into details on QML components here. Basically, a component is the contents of a given qml source file. Having imported a file named Foo.qml, its contents can be instantiated as Foo { ... }. A module is a directory containing components. Importing a module simply means importing all components of a directory. A really cool feature is that a module can be loaded from a remote location over the Internet.

States and Transitions

One concept that is heavily integrated into QML is states and transitions between states. The Qt 4.6 release saw the introduction of the C++ classes for supporting this. However, with QML, using states and transitions is a natural thing.

The source code shown in Listing 1 demonstrates a number of QML concepts. First is the example of how to declare a hierarchy of items. The scene rectangle contains the red and blue rectangles. The red rectangle, in turn, contains a text item and a mouse area item.

The text item in the red rectangle demonstrates another feature: anchor layouts. Items can be anchored to each other, either to their sides or their center lines. The anchors can be offset using margins, and different items can be used for anchoring different parts of the same item. Basically, all your layout needs should be covered by anchor layouts. In this specific example, the center of the text is anchored to the center of the parent rectangle.

Further down, another text item is declared. This time, it is centered in the blue rectangle. Notice that the text item does not have to be a child of the item on which it is centered. This will have implications later on.

Moving on in the red rectangle, we reach the mouse area. This is another concept in QML—interactive areas are not mapped tightly to the visuals. A mouse area is used to interact with mouse events. Think of it as an invisible rectangle that can be anchored

Listing 1. States and Transitions

```
import Qt 4.7

Rectangle {
  width: 300; height: 150
  id: scene

  Rectangle {
    id: red
    x: 50; y: 50
    width: 50; height: 50
    color: "red"

    Text {
      anchors.centerIn: parent
      text: "Red"
    }

    MouseArea {
      anchors.fill: parent
      onClicked: {
        if(scene.state == "redFocus")
          scene.state="";
        else
          scene.state = "redFocus";
      }
    }
  }

  Rectangle {
    id: blue
    x: 200; y: 50
    width: 50; height: 50
    color: "blue"

    MouseArea {
      anchors.fill: parent
      onClicked: {
        if(scene.state == "blueFocus")
          scene.state="";
        else
          scene.state = "blueFocus";
      }
    }
  }

  Text {
    anchors.centerIn: blue
    text: "Blue"
  }

  states: [
    State {
      name: "redFocus"
      PropertyChanges { target: red; scale: 2.5 }
      PropertyChanges { target: blue; rotation: 30 }
    },

    State {
      name: "blueFocus"
      PropertyChanges { target: red; rotation: 30 }
      PropertyChanges { target: blue; scale: 2.5 }
    }
  ]

  transitions: [
    Transition {
      NumberAnimation { properties: "scale";
        duration: 2000; easing.type: Easing.OutBounce }
      NumberAnimation { properties: "rotation";
        duration: 750; easing.type: Easing.InOutCubic }
    }
  ]
}
```

to other items, just as a visual item.

In the mouse area, the onClicked signal is bound to a piece of JavaScript. In this case, it alters the state property of the scene item. This brings us to the states and transitions.

Items in QML have a list of states and a list of transitions. In the example, the states' list contains two states: redFocus and blueFocus. Each state contains a number of PropertyChange items. These items modify properties of target items. In the case of redFocus, the scale of the red item and the rotation of the blue item are changed. Other items can be used in states—for instance, ParentChange moves items in the item hierarchy.

Looking back on the JavaScript bound to the onClicked event, the change of the state property moves between the states listed in the states property. When the state is set to an empty string, the default state is used. This means all properties are set to their initial, unaltered values.

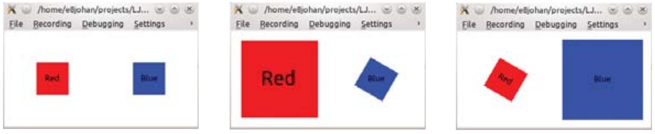

/home/e8johan/projects/LJ...
File Recording Debugging Settings
Red
Blue
/home/e8johan/projects/LJ...
File Recording Debugging Settings
Red
Blue
/home/e8johan/projects/LJ...
File Recording Debugging Settings
Red
Blue

Figure 1. States and Transitions

The final piece of the puzzle is the transitions property, which is a list of behaviors for value changes of different properties. It is possible to control each individual property for each item for each transition direction. In the example, however, we control only each property for all items and all transitions. The NumberAnimation items control how long each change takes and how the change is made. The scale bounces while the rotation accelerates and decelerates according to a cubic curve, forming a smooth motion.

Looking at the screenshots in Figure 1, you can see the difference between the two texts. In the case of the red rectangle, the text is a child of the rectangle. This means the rotation and scaling of the rectangle is applied to the text too. In the case of the blue rectangle, the text simply stays centered. It is not affected by the transformations applied to the rectangle, because it is now a child of it.

Listing 2. A List View

```
import Qt 4.7

Rectangle {
  width: 200; height: 200

  ListModel {
    id: countries

    ListElement {
      name: "Denmark"; capital: "Copenhagen"
    }

    ...

    ListElement {
      name: "Sweden"; capital: "Stockholm"
    }
  }

  Component {
    id: countryDelegate

    Item {
      width: listView.width; height: 50

      MouseArea {
        anchors.fill: parent
        onClicked: { listView.currentIndex = index; }
      }

      Rectangle {
        x: 3; y: 3
        width: parent.width-6
        height: parent.height-6

        color:
listView.currentIndex==index?"white":"lightgray"
        radius: 8

        Column {
          anchors.fill: parent
          anchors.margins: 5
          Text {
            font.bold: true; font.pixelSize: 18
            color: "#444444"; text: name
          }
          Text {
            font.italic: true; font.pixelSize: 10
            color: "#666666"; text: capital
  } } } } }

  Component {
    id: highlightFrame

    Item {
      width: listView.width; height: 50;
      y: listView.currentItem.y
      Rectangle {
        x: 3; y: 3
        width: parent.width-6
        height: parent.height-6
        radius: 8
        border.width: 4; border.color: "darkGray"
      }
    }
  }

  ListView {
    id: listView
    anchors.fill: parent
    model: countries
    delegate: countryDelegate
    highlight: highlightFrame
    focus: true
    highlightFollowsCurrentItem: true
  }
}
```

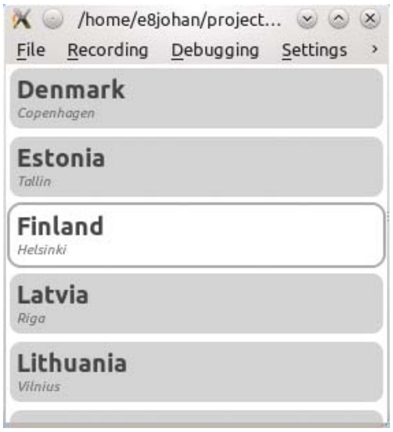
Models and Views

When building user interfaces, a common scenario is to show a list of data. As you already have guessed, QML provides support for this as well. Listing 2 shows how this can be used.

The example in Listing 2 consists of four major parts: the countries model, the countryDelegate component, the highlightFrame component and the ListView item, which puts it all together. Starting from the bottom, the list view item refers to a model, a delegate and a highlight. These are the model and components implemented earlier. In addition to this, some tuning of the view's behavior is needed to allow keyboard navigation in parallel with mouse navigation.

Returning to the top of the example, the model is a ListModel containing a set of ListElement items. The properties of the list elements are made available through the view, as you can see if you continue into the countryDelegate component and its text items.

The countryDelegate component is what the list view uses to visualize each item of the list. It consists of a mouse area and a rectangle with two texts in it. The mouse area sets the current item of the list if an item is clicked, while the texts show the data of the model. Notice that the text property of the items is bound to the property names used in the list elements of the model. This makes it easy to tie items in a delegate to model data.



Denmark
Copenhagen
Estonia
Tallin
Finland
Helsinki
Latvia
Riga
Lithuania
Vilnius

Figure 2. A List View with a Delegate and Highlight Rectangle

Let’s continue to the highlightFrame component. This is a frame that the view places over the current item. In this case, it adds a border to the item. The countryDelegate changes its own background color if it is the current item. This is because the current item is shown without a background color instead of with one. That is not possible to achieve using only a highlight frame.

Finally, the list view puts it all together. The result is shown in Figure 2.

The QML Runtime

It is common to rely on the QML viewer tool when developing QML applications. It also is common to use QML for populating models with data, be it phony or real data. Most state management and work also can be carried out from QML with the aid of JavaScript. However, in most cases, a native application is needed as the runtime environment for any QML application. This is where the QtDeclarative module enters the picture.

For readers who are familiar with Qt, it is good to know that QML consists of a language engine, a context for the scripts to execute in and a QGraphicsScene to operate inside. All these components can be set up manually—it even is possible to add QML components to an existing scene. This way, you can upgrade an existing application gradually.

If you start from scratch with a QML application, the QDeclarativeView encapsulates all these components into a single class, which also happens to be a widget. For an application relying only on QML for its user interface, this is all that it takes.

To integrate C++ objects into QML, the QObject meta-system is used. This means that any QObject-derived class can be exposed to QML. From QML, properties, signals and slots will be available. As QML properties are bound to values, rather than assigned, any changes in the C++ part of the application is reflected automatically in the QML part.

It is beyond the scope of this article to go into details on this, but in the first example, the state could have been driven from C++. This would have let QML handle what it is good at: visuals and dynamic transitions. In the second example, a typical application would provide the model from C++—again, letting QML focus on the visuals.

This approach has a number of benefits. The first one is that the user interface is created quickly using QML as the whole language is focused on that goal. The other is that you are forced to maintain a clear division between the user interface and the rest of the application. This leads to more structure and better code.

The Future

Looking at the future of Qt Quick, many things may happen. In MeeGo, the MeeGo touch initiative is implementing new widgets using Qt Quick. In KDE, Plasma is supporting Qt Quick. One effect of this is that you can write Plasmoids using QML. In the Qt tooling department, the trolls are working on a visual designer for Qt Quick. It already has a few interesting features—for instance, layers can be imported from GIMP and Photoshop directly into the designer.

Looking at Qt, I don’t think we have seen the last widget-based application yet. Actually, when creating serious software for serious tasks, I see no reason not to use widgets. However, with the new focus on mobile, not only within the old Trolltech, but the entire Linux community, I think that Qt Quick will be a very frequently used tool.■

Johan Thelin is a passionate Qt and open-source user. He spends his days at Pelagicore working with Linux and open source in the automotive industry. At night, he works as a consultant and freelance writer.

Limitations of Widgets

There are a number of limitations imposed from building user interfaces with widgets that are addressed when switching to a graphics view-based approach. One obvious limitation is that widgets are rectangular and like to be arranged side by side, which makes it hard to arrange non-rectangular items in a good manner.

Another limitation is that widgets clip their children, which means children cannot extend outside their parent widget. Take a simple effect, such as having parts of a user interface explode. In that case, clipping is a limiting factor.

Another feature that widget-based systems usually do not support is sub-pixel resolution for item dimensions and placement. Also, transformations, such as scaling and rotation, are not supported by widgets. In a scene, all these features can be used to achieve the best possible visual experience.

Taking transformations over time, it becomes obvious that widgets are not meant to slide, bounce or generally move about. They are designed to be arranged in layouts based on grids, columns and rows, and they provide users with a standardized, structured user interface. This is very good when the user is using the computer as a computer. When the user is using a device, this type of computer interface is not the most appropriate solution.

Getting Started with Qt Quick

As Qt 4.7 recently was released, Qt Quick is becoming available through the repositories of most distributions. Some distributions choose to package Qt in several packages, so make sure you get the Qt development package, the Qt Creator package and all Qt modules, especially those referencing to Qt declarative. In the Linux world, I recommend you use the facilities provided by your distribution to install and maintain your software. However, for those of you needing a particular version of the Qt tools, or if you are using a distribution that doesn’t include Qt, you can download the Qt SDK from Nokia’s Web site.

The package that you want is the Qt SDK, available from **qt.nokia.com/downloads**. Simply download the file, `chmod` it to make it executable and run the installer. You can install it in your home directory if you do not have root access. The SDK includes tools, demos, sources and documentation, all in one convenient package.

Calibre

The ultimate in open-source e-book management.

DAN SAWYER

I've got e-books—a lot of e-books. I was one of those geeks who, when he found Projects Gutenberg and Perseus, was convinced the Rapture would happen tomorrow because life had reached its highest pinnacle on the celestial plains.

If the rapture didn't happen, nuclear war might. So I wanted to make sure I'd have all the books I could get my hands on. When Fictionwise and Tor started giving books away, I was there too. When non-DRM-encumbered books became available through Smashwords, I started picking them up as I could afford them. Ditto for Doctorow's stuff and other e-books by friends of mine—and let's not even mention the hundreds of NASA, Navy and Army manuals I use for research.

Fast-forward 13 years since discovering Project Perseus, and I have more e-books than I really know how to deal with. More than an embarrassment of riches, it was a bloody mess and high time to do something about it.

E-book Management: the Old Way

Generation one of my library organization project went the way any competent, non-database-designing sysadmin would do it: with a sensible directory structure. After many hours, I wound up with a system that was excellent for nonfiction, but crap for fiction. After all, the best you can do with a good directory tree is break out by genre, author and series. That's adequate for reference materials, but not great for the inevitable "hmmm...what do I want to read next?"

Enter Calibre, the Python-based, data-fetching, universally device-compatible e-book management and conversion program.

A good library needs good metadata, and directory structures have squat. E-book management software that ships with many readers attempt to do this, but they tend to have proprietary ties to devices and operating systems. They're usually not Linux-friendly, and they're also not very friendly to collection longevity. DRM? Proprietary formats? Thank you, no.

What I needed was something like iTunes or Amarok, but with a decent interface, designed for books. Fortunately, I wasn't the only person with this problem.

E-book Management: the New Way

Enter Calibre, the Python-based, data-fetching, universally device-compatible e-book management and conversion program. The product specs are ambitious, and the implementation is, though occasionally bumpy, pretty darn spectacular.

Unlike most proprietary, device-specific management programs, Calibre converts all major formats into one another (I do not know how it handles DRM, as at the time of this writing the DMCA overthrow has just come down—happy day!—so any such features are as yet undocumented), and allows medium-grained metadata control over all of them. For fine-grained metadata control, you need more specialized tools, or you need to edit the files directly with a compatible editor. For example, Sigil does this quite splendidly for EPUB. For households with multiple species of e-readers, this is a must.

Calibre also can autopopulate your library's metadata, pulling it down from a number of different on-line databases by title, author and ISBN.

It allows rating, so you can keep track of how much you liked your book.

It has a very handsome cover art browser.

It can store Open Document Format files as e-books and create metadata for them. This gives it the nice unintended utility as a version control system for authors.

It comes with a native e-book reader that allows limited annotations and can view a number of the supported formats. For those formats that Calibre does not support reading directly, it will launch your operating system's default viewer with the click of a button.

It also nests multiple formats of the same book under the same entry in the database and directory structure, so when you convert, say, a PDF to Kindle format for your Kindle, you don't have duplicate titles popping up in your book list.

It syncs to more than 20 different makes and models of e-book reader and also allows you to access most readers (even the unsupported ones) in mass storage mode, so you're future-proofed if you change reader platforms later on.

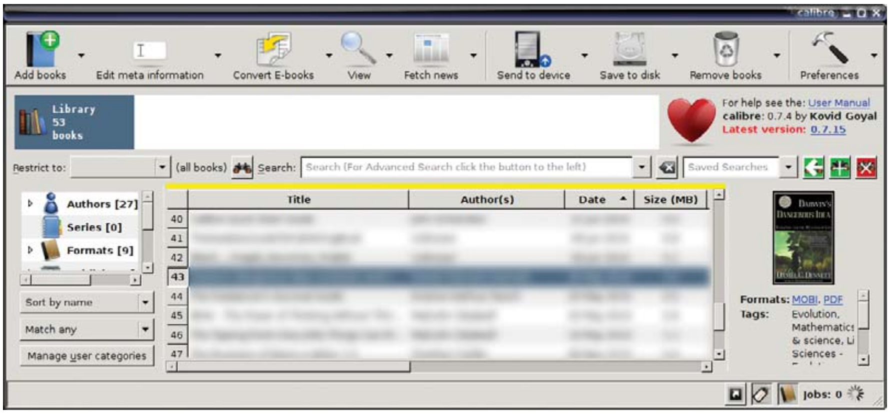
So, what are we waiting for?

Fetching

Calibre can be found at an uncommonly well-designed Web site: **www.calibre-ebook.com**. Click the download button, select Linux from the following screen (you'll notice that it also runs on Mac and Windows—a plus for those of us with multiplatform networks), and read the following screen first.

Calibre is picky about the dependencies; the glibc and Python versions are particularly important. Recent distributions are all in compliance, but older distros might require some updating to work properly (you also may need to compile them yourself—a fairly trivial undertaking—of course, your mileage may vary).

Assuming you're in compliance, copy the code from the



Library
53
books

For help see the: User Manual
calibre: 0.7.4 by Kovid Goyal
Latest version: 0.7.15

Formats: MOBI, PDF
Tags: Evolution, Mathematics & science, Li Sciences -

Figure 1. Calibre Interface

code window into your root terminal (you must run the install as root; otherwise, it tends to fail with nasty comments about your intelligence, heritage and recreational proclivities), and press Enter. If all goes well, a new item should appear in your window manager's start menu. If it didn't, you most likely missed a dependency.

Getting to Know Calibre

Calibre's project manager Kovid Goyal deserves a big pat on the back (and tips in the tip jar). Not only is the program organized well, the Web site easily navigable, and the installation relatively painless, but the documentation is very comprehensive as well.

When you open Calibre, you're presented with the main interface screen (Figure 1).

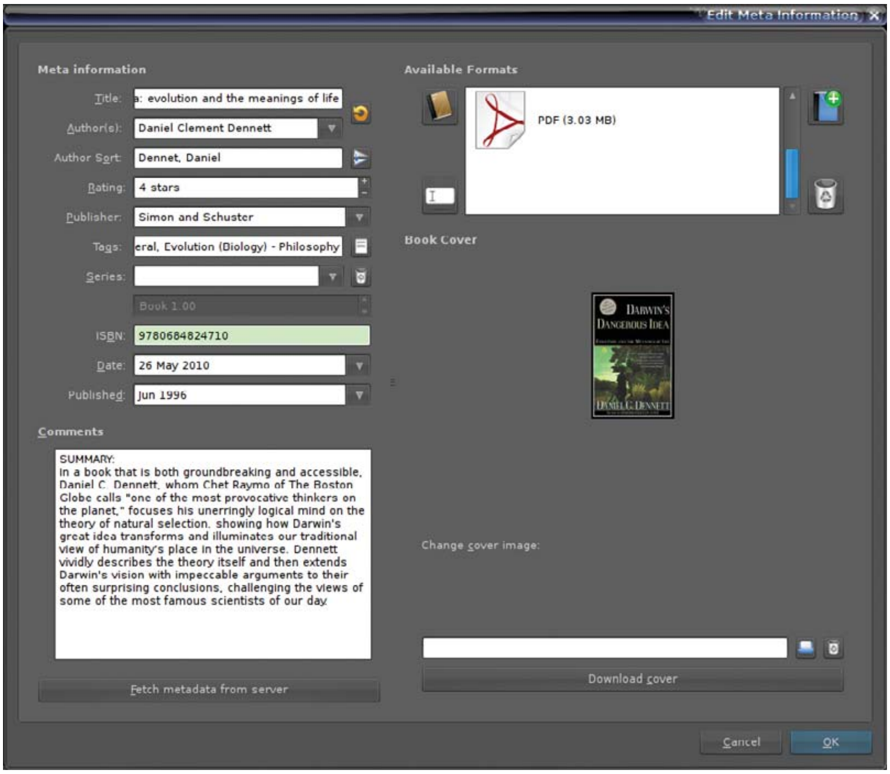
You'll notice the unique interface concept. There is no standard menu bar—just a toolbar with a few basic buttons and drop-downs under the buttons to allow you access to finer-grained controls on each of those tools.

Starting at the top left is the button that starts it all: Add Books. The list box attached to it gives you the option to add a single book, add a directory or a nested directory structure, or to add an empty entry into the database that you can populate later. This last option is useful if you're also adding your physical books to the collection—it can serve as a placeholder with instructions on where the book is shelved.

Moving along to the right is the metadata tool. This is the heart and soul of your database. The metadata are all your obvious tags: ISBN, Author, Series name, Publisher, Copyright date, Publication date for that edition and so on. It also includes the cover art, the listing of the different formats in which you have the book, a comments field and a tags field. Fetching metadata from the remote server populates the fields for you, not including the cover art, and it puts the back-of-book copy in the comments field and the genre in the tags field. Downloading the cover art pulls the cover art linked to that ISBN from the ISBN or Google Book servers. If you get art for an edition you didn't intend, you always can replace it by hand.

Pushing the metadata button brings up the metadata edit screen for the individual book you have selected (Figure 2). Using the drop-down list, you can act upon the metadata of multiple selected books in a variety of ways using batch functions—most handy.

The third button is the conversion tool, which lets you translate one format into another—very handy for devices like the Kindle, which reads only one kind of proprietary file format. You can translate EPUBs, properly formatted PDFs (some formatting conventions, like headers and footers, can cause major headaches), OpenOffice.org documents and so on, into Kindle AZWs for easy viewing on your Kindle. In many cases, the default settings are quite adequate, but for the



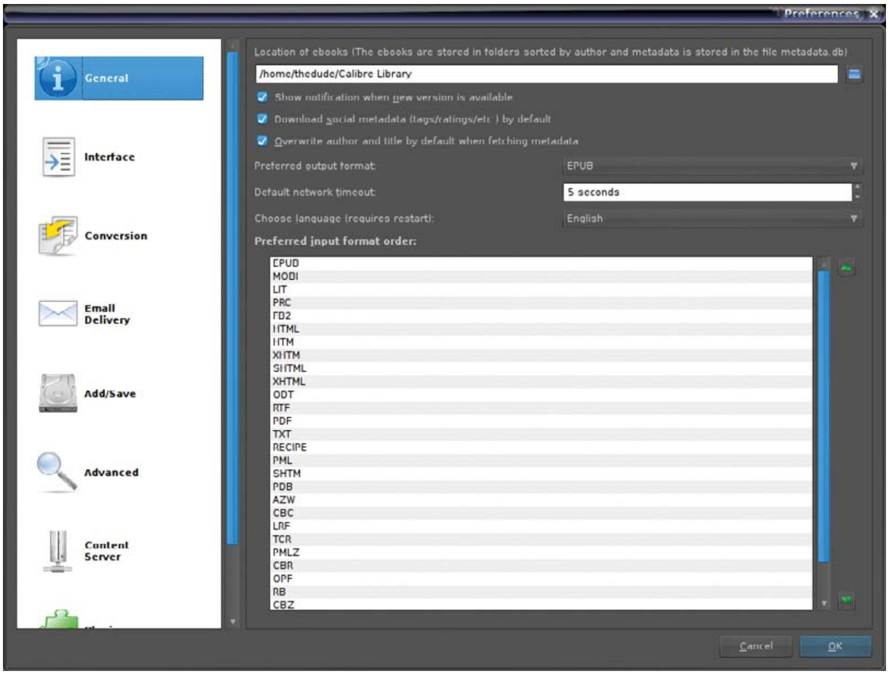
Title: a: evolution and the meanings of life
Author(s): Daniel Clement Dennett
Author Sort: Dennet, Daniel
Rating: 4 stars
Publisher: Simon and Schuster
Tags: eral, Evolution (Diology) - Philosophy
Series:
Book 1.00
ISBN: 9780684824710
Date: 26 May 2010
Published: Jun 1996

PDF (3.03 MB)

Comments

SUMMARY:
In a book that is both groundbreaking and accessible, Daniel C. Dennett, whom Chet Raymo of The Boston Globe calls "one of the most provocative thinkers on the planet," focuses his unerringly logical mind on the theory of natural selection, showing how Darwin's great idea transforms and illuminates our traditional view of humanity's place in the universe. Dennett vividly describes the theory itself and then extends Darwin's vision with impeccable arguments to their often surprising conclusions, challenging the views of some of the most famous scientists of our day.

Figure 2. Editing Metadata



Location of ebooks (The ebooks are stored in folders sorted by author and metadata is stored in the file metadata.db)
/home/thedude/Calibre Library
☑ Show notification when new version is available
☑ Download social metadata (tags/ratings/etc.) by default
☑ Overwrite author and title by default when fetching metadata
Preferred output format: EPUB
Default network timeout: 5 seconds
Choose language (requires restart): English
Preferred input format order:
EPUB
MOBI
LIT
PRC
FB2
HTML
HTM
XHTM
SHTML
XHTML
ODT
RTF
PDF
TXT
RECIPE
PML
SHTM
PDB
AZW
CBC
LRF
TCR
PMLZ
CBR
OPF
RB
CBZ

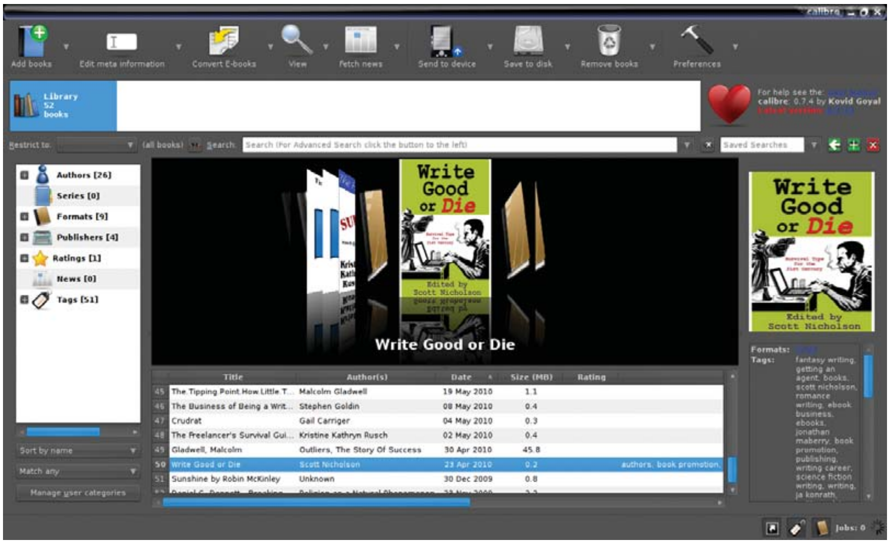
Figure 3. Preferences Screen

occasions when they're not, the tool gives you direct access to the document's structure and several of the XML wild cards. This is the one section of the program that, at the time of this writing, is not well documented, so you'll need to experiment if you're getting wonky results from the default conversion.

The drop-down list also gives you the option of creating a device-exportable library catalog—handy for those who like to compare book lists with their friends or who, like me, are simply nuts for library catalogs. (Don't laugh, there are more of us than you think, and we live on the Internet with vast botnets at our beck and call. Taunt us at your own risk.)

Next is the View button, which is fairly self-explanatory. Clicking it opens the default viewer for the highest-priority format in which the book is available. Clicking on the list box gives you the further option to view a specific format rather than just using the one Calibre picks for you by the numbers.

Next up, there's a button curiously entitled Fetch News. This is actually a very sophisticated RSS reader, and it comes preloaded with more than a thousand news feeds in various languages. If your e-reader doesn't have Wi-Fi or 3G, and you want to batch-spool up your morning news or blogs, this is the tool for you. It can pull down anything with an RSS feed, so you always can have the latest installment of Doctorow's current novel added to your library as soon as it's released.



	Title	Author(s)	Date	Size (MB)	Rating	
45	The Tipping Point How Little T...	Malcolm Gladwell	19 May 2010	1.1		
46	The Business of Being a Writ...	Stephen Goldin	08 May 2010	0.4		
47	Crudrat	Gail Carriger	04 May 2010	0.3		
48	The Freelancer's Survival Gui...	Kristine Kathryn Rusch	02 May 2010	0.4		
49	Gladwell, Malcolm	Outliers, The Story Of Success	30 Apr 2010	45.8		
50	Write Good or Die	Scott Nicholson	23 Apr 2010	0.2		authors, book promotion,
51	Sunshine by Robin McKinley	Unknown	30 Dec 2009	0.8		

Tags: fantasy writing, getting an agent, books, scott nicholson, romance writing, ebook business, ebooks, jonathan maberry, book promotion, publishing, writing career, science fiction writing, writing, ja konrath,

Figure 4. Graphical Bookshelf Browser

Clicking the button brings up the scheduling window if you don't have anything scheduled yet (here you set the download schedule) or it grabs all queued downloads if you have them scheduled. Using the drop-down menu lets you fetch the news and customize the feeds by creating "recipes" for your specific news-reading needs (basically, a list of RSS feeds and the way you want them to appear in your customized electronic newspaper).

The next button along is your device controller. It syncs your selected reading list with your e-reading device. The drop-down menu lets you select the particular driver (if autodetection is not working properly) and tweak other sync settings.

If your device isn't recognized, due to driver problems, kernel issues or the device being so new there isn't a driver for it yet, fear not. The next button is Save To Disk, which will save the selected books to any location in your file tree that you please, including a USB mass storage device, such as your e-reader's internal Flash memory. When using this option, you'll usually need to reboot the device so that it rescans its file tree and updates its database accordingly.

Next is the Remove Books button, which also is self-explanatory. The drop-down gives you the option to remove singles, multiples, specific formats contained within the selected titles only, cover art only, or to remove the selected books from the attached device, but not from the library.

Finally, there's the Preferences button, which brings up the granddaddy of all dialogs (Figure 3). This is the beating heart of the operation, offering lower-level access to the database, debug functions, conversion defaults and a bunch of other stuff. Most interesting, perhaps, is the Calibre server setup, which allows Calibre to operate like a remote library system for other computers on your network (or on the Internet, if you often experience the sudden need for one of your e-books while at the office). Pretty much everything in here, sophisticated though it is, is self-explanatory.

Below the button bar, there's a field listing the available collections. Calibre can support multiple libraries, and this is where you switch between them. Below that, there's a search bar. It doesn't search within the books, alas, but it does search the metadata quite effectively.

Moving down again, is the library itself. In the left pane, you'll find the available hierarchies—Authors, Publishers, Tags, formats and ratings. It offers all of these as pre-sorted searches, and clicking on them will modify the list you see in the center pane.

The center pane is your library itself. All the metadata is listed in easy-to-read rows of alternating colors. You can organize them alphabetically by any of the available fields simply by clicking on the column title, just as you can do with any spreadsheet or other list-driven program.

The rightmost pane shows you the cover art and a quick rundown on the book currently highlighted. File types, comments and so on all appear here.

Finally, along the bottom of the screen are three buttons that control the layout of the program. Toggling these three buttons turns on and off different panes on the screen. The one that's turned off in my default layout is the graphical bookshelf (Figure 4).

This handy interface allows you to flip through the list as it appears (search-restricted and all), like you'd flip through books on a bookshelf. Clicking left shuffles to items farther up the list, clicking right shuffles to items farther down the list.

This handy interface allows you to flip through the list as it appears (search-restricted and all), like you'd flip through books on a bookshelf.

Building the Library

If you're anything like me, you probably have some kind of e-book library already, so you'll want to start off by importing what you've got. Because Calibre's library import feature does some destructive rewriting, it's worth creating a backup copy of your library, just in case.

After importing, you're going to have books in your list—maybe a lot of books. And, because most e-book metadata tends to be poor, the books in your list are probably haphazard and not well organized.

Organizing them can be a laborious process, but there are a few ways to make the job less irritating—bulk-editing the metadata is chief among them, as mentioned earlier. For example, if you had 20 titles by the same author, you could select all of them, right-click on one of them, and select Bulk Edit Metadata from the pop-up menu. In the resulting screen, you can set the author, genre tags, publisher tag and most (but not all) other data fields to help with your sort. You also can check a little box that says Swap Author and Title fields. This one is particularly useful, as many poorly tagged e-books are tagged with these fields the wrong way around.

Once the organization is done to your satisfaction, you're good to go.

Weaknesses and Caveats

As shining as the program is in most respects, it comes up short in a couple areas.

The first, and most annoying, is one that is only partly the responsibility of the development team. The USB stack under the most recent versions of the Linux kernel has been occasionally glitchy. As a result, several recent distributions, including one of the ones I run, have had trouble syncing to external devices. The device would show up as a mass storage device, but attempting to access the device's internal database resulted in crashes, nasty core dumps, segfaults and the occasional exploding computer.

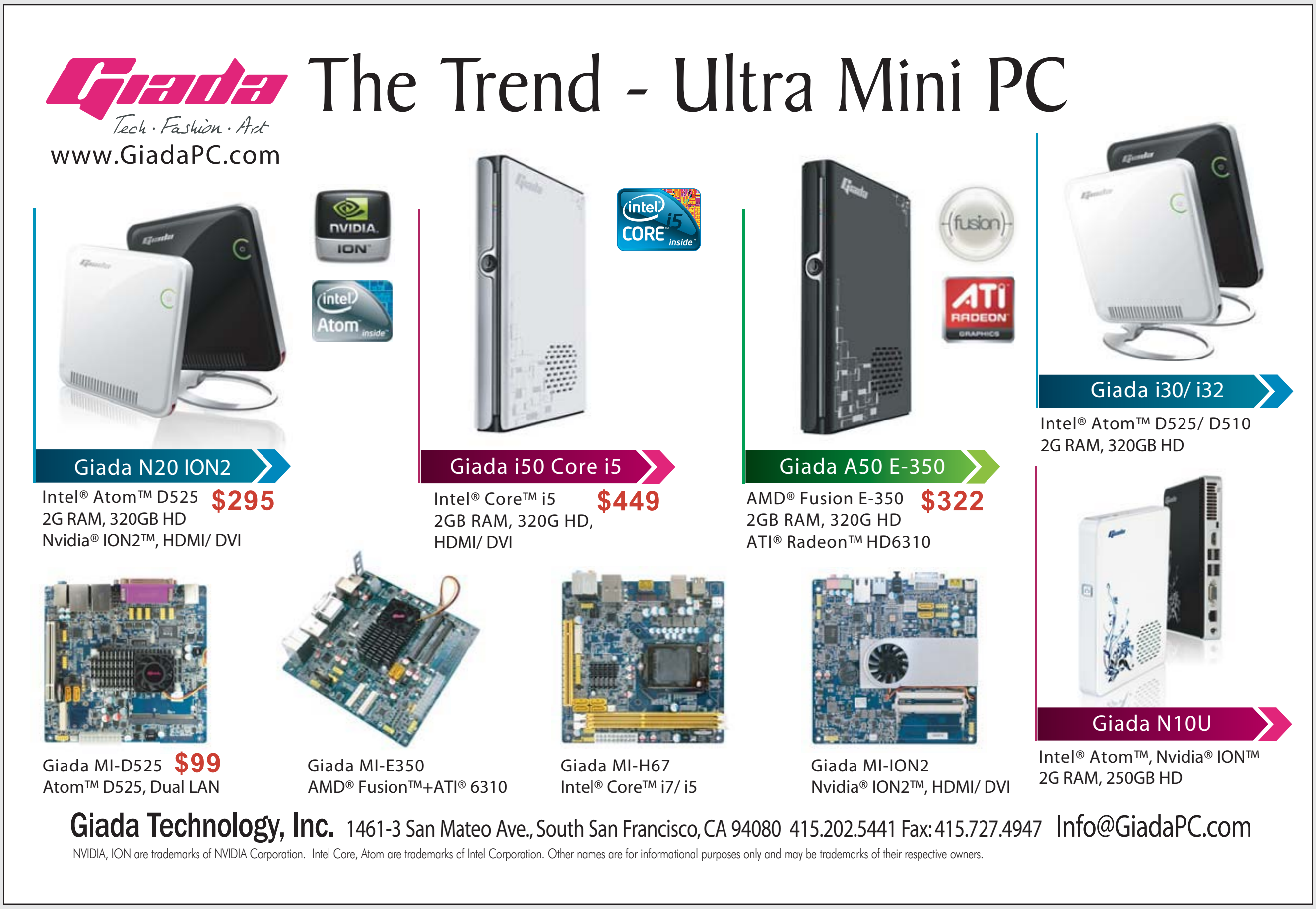
In the most recent kernel versions, this seems no longer to be a problem, but if you encounter it and can't upgrade your kernel or distro, you can usually fall back to mass storage mode and hand-sync your books through the Save To Disk button.

My other gripe is that there's only one comment field. The ability to deeply annotate books or keep reviews on books simply isn't there. This will cramp the style of research junkies and avid bookworms alike. Hopefully, this will be remedied in future versions (it's not like there is any shortage of potential metadata fields).

Wrap-Up

Calibre is the best-of-breed solution currently on offer for any platform, and it is well worth the download if you've got an e-book collection numbering more than a dozen, or if you've been trying to figure out some way to manage things on your Sony, Nook or Kindle without having to boot the Windows image. Enjoy!■

Dan Sawyer is the founder of ArtisticWhispers Productions (www.artisticwhispers.com), a small audio/video studio in the San Francisco Bay Area. He has been an enthusiastic advocate for free and open-source software since the late 1990s. He currently is podcasting his science-fiction thriller *Antithesis* and his short story anthology *Sculpting God*. He also hosts "The Polyschizmatic Reprobates Hour", a cultural commentary podcast. Author contact information is available at www.jdsawyer.net.



Hacking with Humor

When the going gets tough, the creative get funny. DOC SEARLS

Hacking is largely a practical matter—scratching itches and all that. But, some hackers work in service of irony, manipulating social as well as computer code—to effects that are equally interesting, useful and funny. At this line of work, nobody outdoes Tim Hwang (aka @TimHwang), who blogs as Broseph Stalin (after retiring as commissioner of the US Bureau of Fabulous Bitches) and whose CV also includes creating and organizing all of the following:

- ROFLCon ("the most epic Internet culture conference ever assembled", and it's true).
- The Awesome Foundation for the Arts and Sciences ("an ever-growing, worldwide network of people devoted to forwarding the interest of awesomeness in the universe").
- The Web Ecology Project (in which he is also identified as "an analyst with The Barbarian Group"—where he works on issues of group dynamics and Web influence, adding that he "is in the process of watching every homemade flamethrower video on YouTube", the results of which were presented at SXSW last year).
- Robot Robot & Hwang (his future law firm, currently working on "an open-source project to develop the legal infrastructure to allow for large-scale securitizing of lawsuits").
- SOCIALBOTS 2011 ("the first-ever competitive event in the large-scale robotic influence of on-line social groups").

I met Tim when he was a researcher at the Berkman Center and got to know him better through ROFLCon and The Awesome Foundation (in which I now hold a chair in the original Boston chapter). It also seemed like I ran into Tim pretty much everywhere around the MIT-Harvard axis of Cambridge-based social and technical hackery.

I still don't know where Tim went to college, though I'm guessing it was one of the two likely suspects. (Berkman is at Harvard and ROFLCon happens at MIT.) I never asked. From a hacking perspective, where one went to school (or currently goes to school) is not a matter of great importance. This is one of the things that drew me to working with *Linux Journal* and hanging with hackers in the first place. The hacker ethos is a corollary to a line on a T-shirt I used to see a lot in Los Angeles. It read, "It isn't who you are but how you look. After all, who cares who you are?" Around hackerdom, it's like, "It isn't where you went but what you do. After all, who cares where you went?"

These days, Tim goes to UC-Berkeley, where he's a law student. What he's doing (and presumably will do when he gets out) is work with Robot Robot & Hwang. And, since hacking law is an activity on which at least one of my own open-source projects depends, I'm looking forward to Tim's progress on that front.

Tim Hwang

Meanwhile, my immediate interest is in SOCIALBOTS 2011, which will be over by the time you read this. The whole "social network" craze drives me up a wall. We've had "social networks" forever, both off-line and on. Bars, restaurants, churches, town squares, marketplaces, cafés, USENET, FidoNet, SMS, blogging—all those things are no less social than Twitter and Facebook, yet when somebody says "social network", those two things are what they mean. Until I busted them for it, Wikipedia's entry for "social media" read:

> Social media have been modernized to reach consumers through the Internet. Social media have become appealing to big and small businesses. Credible brands are utilizing social media to reach customers and to build or maintain reputation. As social media continue to grow, the ability to reach more consumers globally has also increased. Twitter, for example, has expanded its global reach to Japan, Indonesia and Mexico, among others. This means that brands are now able to advertise in multiple languages and therefore reach a broader range of consumers. Social media have become the new "tool" for effective business marketing and sales.

But, while I mocked the pursuit of influence and rankings on Twitter as "high school with a business model", Tim hacked up a way to prove it, with SOCIALBOTS 2011. The pitch:

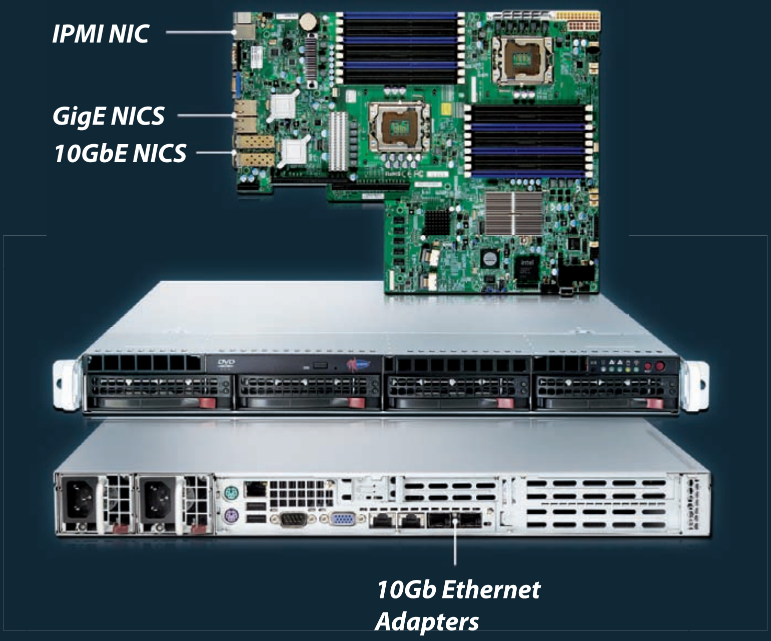
> Teams will program bots to control user accounts on Twitter in a brutal, two-week, all-out, no-holds-barred battle to influence an unsuspecting cluster of 500 on-line users to do their bidding. Points will be given for connections created by the bots and the social behaviors they are able to elicit among the targets. All code to be made open source under the MIT license.
>
> It's blood sport for Internet social science/network analysis nerds. Winner to be rewarded $500, unending fame and glory, and THE SOCIALBOTS CUP.

I look at this as a way of debugging Twitter, plus every other "social network" (or wannabe) whose executives cringe at the results.

Here's betting that Tim's hack will do more good than a thousand complaints like mine.

Oh, and don't forget to check **roflcon.org** for details on ROFLCon 3.■

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.


IPMI NIC
GigE NICS
10GbE NICS
10Gb Ethernet Adapters