LINUX JOURNAL™

Since 1994: The Original Magazine of the Linux Community

SPONSORED BY

CHEF™
GETCHEF.COM

AUGUST 2014 | ISSUE 244 | www.linuxjournal.com

PROGRAMMING

HOW-TO: OpenGL Programming

USE VAGRANT for an Easier Development Workflow

PROMISE THEORY An In-Depth Look

+

Build, Develop and Validate Creation of RPMs

Sysadmin Cloud Troubleshooting with dhclient

Tips for Becoming a Web Developer

A Rundown of Linux for Recreation

WATCH:
ISSUE OVERVIEW

Get the automation platform that makes it easy to:

Build

Deploy

Manage

›

Infrastructure

Applications

In your data center or in the cloud.

getchef.com

CHEF™
CODE CAN

SILICON MECHANICS
SERVERS • STORAGE • HPC

Are you tired of dealing with proprietary storage?

zStax®
ZFS Unified Storage
Powered by NexentaStor

zStax StorCore ZFS Unified Storage from Silicon Mechanics is truly software-defined storage.

From modest data storage needs to a multi-tiered production storage environment, **zStax StorCore** ZFS unified storage appliances have the right mix of performance, capacity, and reliability to fit your needs.

zStax StorCore 64

The **zStax StorCore 64** utilizes the latest in dual-processor Intel® Xeon® platforms and fast SAS SSDs for caching. The zStax StorCore 64 platform is perfect for:

- small-medium office file servers
- streaming video hosts
- small data archives

zStax StorCore 104

The **zStax StorCore 104** is the flagship of the zStax product line. With its highly available configurations and scalable architecture, the zStax StorCore 104 platform is ideal for:

- backend storage for virtualized environments
- mission critical database applications
- always available active archives

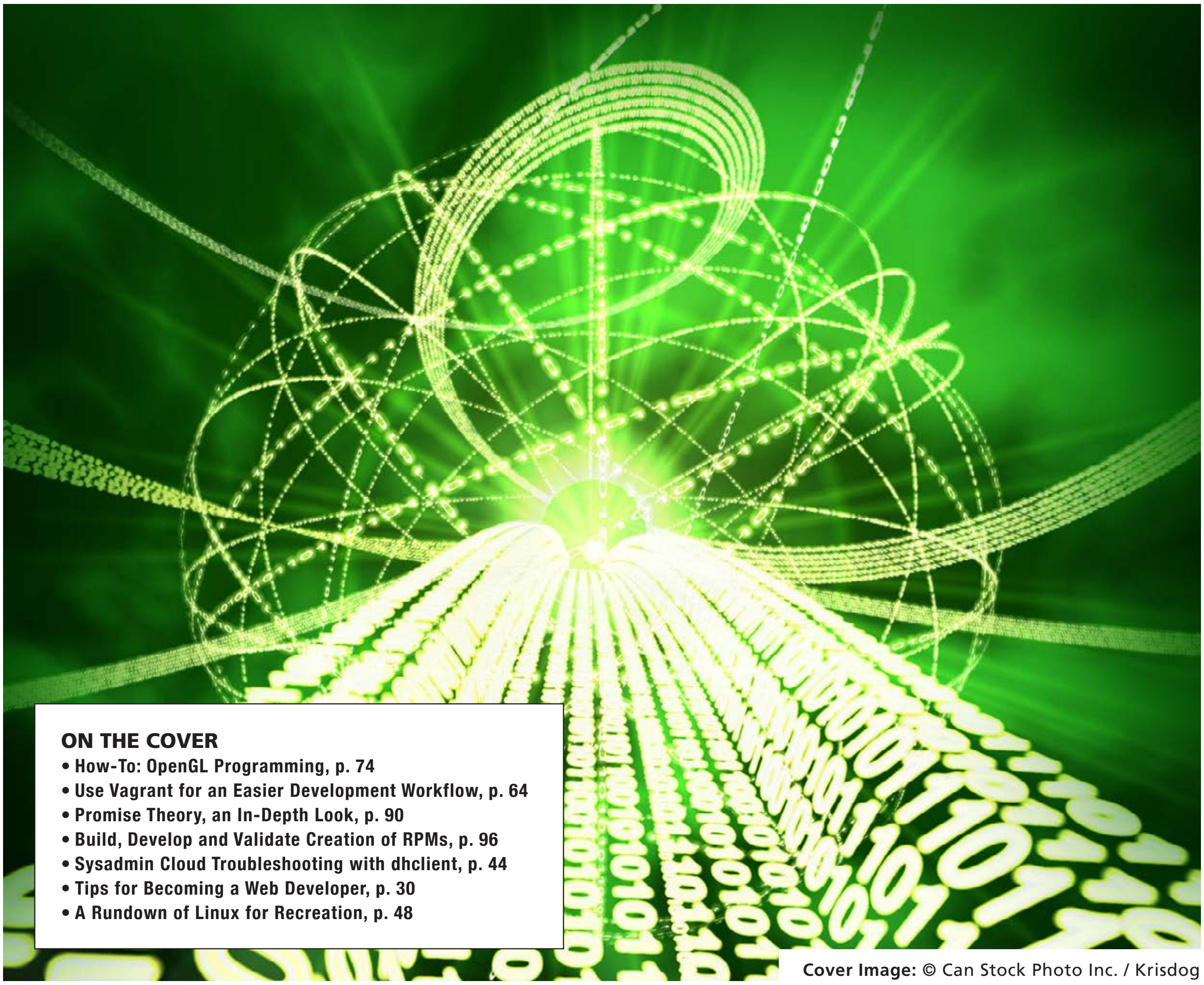
Talk with an expert today: 866-352-1173 - http://www.siliconmechanics.com/zstax

CONTENTS

AUGUST 2014
ISSUE 244

PROGRAMMING

FEATURES



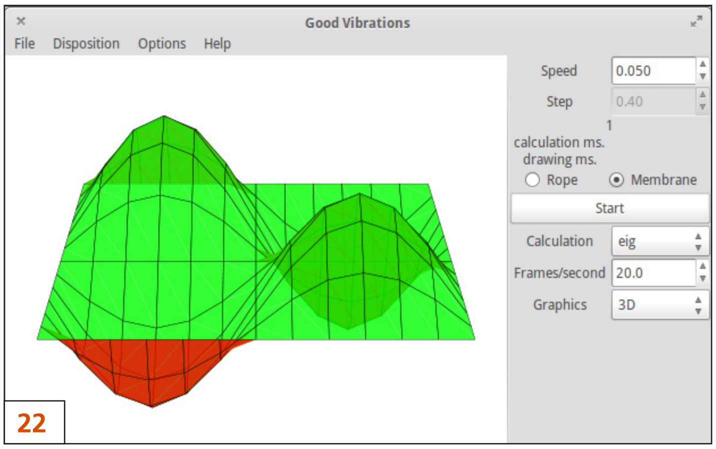
ON THE COVER

- How-To: OpenGL Programming, p. 74
- Use Vagrant for an Easier Development Workflow, p. 64
- Promise Theory, an In-Depth Look, p. 90
- Build, Develop and Validate Creation of RPMs, p. 96
- Sysadmin Cloud Troubleshooting with dhclient, p. 44
- Tips for Becoming a Web Developer, p. 30
- A Rundown of Linux for Recreation, p. 48

Cover Image: © Can Stock Photo Inc. / Krisdog

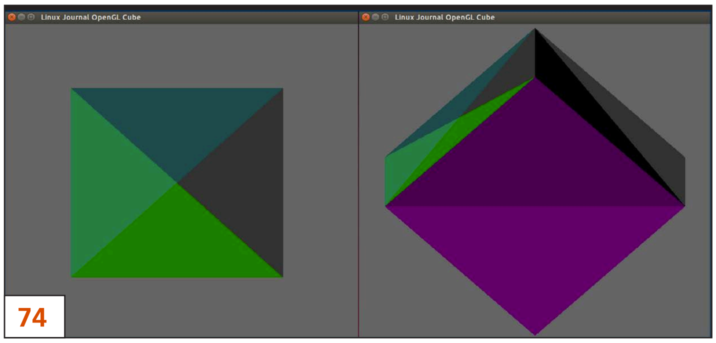
INDEPTH

COLUMNS



22

48



74

IN EVERY ISSUE

LINUX JOURNAL (ISSN 1075-3583) is published monthly by Belltown Media, Inc., 2121 Sage Road, Ste. 395, Houston, TX 77056 USA. Subscription rate is $29.50/year. Subscriptions start with the next issue.

Subscribe to *Linux Journal* Digital Edition for only **$2.45 an issue.**

ENJOY:

Timely delivery

Off-line reading

Easy navigation

Phrase search and highlighting

Ability to save, clip and share articles

Embedded videos

Android & iOS apps, desktop and e-Reader versions

SUBSCRIBE TODAY!

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 • Adam Monsen

Publisher	Carlie Fairchild publisher@linuxjournal.com
Director of Sales	John Grogan john@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
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
MAIL: PO Box 980985, Houston, TX 77098 USA

LINUX is a registered trademark of Linus Torvalds.

SAVE THE DATE

11th Annual

2014 High Performance Computing for Wall Street Show and Conference

September 22, 2014 (Monday) **Roosevelt Hotel, NYC**
Madison Ave and 45th St, next to Grand Central Station

Big Data, Cloud, Linux, Low Latency, Networks, Data Centers, Cost Savings.

Wall Street IT professionals and programmers will assemble at this 2014 HPC Show and Conference, Sept. 22.

New for 2014 – Database Month programmers to speak on the program.

This 11th Annual HPC networking opportunity will assemble 600 Wall Street IT professionals at one time and one place in New York on September 22.

This HPC for Wall Street conference is focused on High Put-through, Low Latency, Networks, Data Centers, lowering costs of operation.

Our Show is an efficient one-day showcase and networking opportunity.

Leading companies will be showing their newest systems live on-the-show floor.

Register in advance for the full conference program which includes general sessions, drill-down sessions, an industry luncheon, coffee breaks, exclusive viewing times in the exhibits, and more. Save $100. $295 in advance. $395 on site.

Don't have time for the full Conference? Attend the free Show. Register in advance at: www.flaggmgmt.com/hpc

Show Hours: Mon, Sept 22	8:00 - 4:00
Conference Hours:	8:30 - 4:50

Wall Street programmers will lead drill-down sessions in the Grand Ballroom program.

September 2013 Sponsors

Cisco · IBM · redhat · hp · Intersect360 Research · Solarflare · Azul Systems · LSI

Media Sponsors

Linux Journal · Linux Pro Magazine · Securities Technology Monitor · datanami · Information Management · Traders Magazine · HPC Wire · Integration developer news · Money Management Executive · Finance on Windows · WSTA Wall Street Technology Association

Show & Conference: Flagg Management Inc
353 Lexington Avenue, New York 10016
(212) 286 0333 fax: (212) 286 0086
flaggmgmt@msn.com

Visit: www.flaggmgmt.com/hpc

SHAWN POWERS

Chocolate in My Peanut Butter

Programming always has been that "thing" people did that I never understood. You've heard me lament about my lack of programming skills through the years, and honestly, I never thought I'd need to learn. Then along came the DevOps mentality, and I started introducing programmatic ways of managing my system administration world. And you know what? Programming is pretty awesome.

Reuven M. Lerner starts off the programming issue with a great article on how to begin in the Web development field. If you're like me, and have been ushered into the programmer's world blindfolded and scared, Reuven has a great starting point for you. Dave Taylor also teaches us programming as he continues his lessons on scripting dates and counting days. For some reason, shell scripting doesn't seem like programming, and I'm far more comfortable doing it.

VIDEO: Shawn Powers runs through the latest issue.

If that sounds silly, that's because it is! Shell scripting is programming, and if you've ever created sysadmin scripts or even written batch files in Windows, your programming skills are better than you think.

Kyle Rankin addresses an issue this month that can be really frustrating if you're automating a cloud environment. If you spin up a server that gets assigned a random IP address, how do you address it from the rest of your infrastructure? He shows us his method of dealing with that scenario and teaches us about DHCP along the way.

I decided to continue our summer vacation a bit with my Open-Source Classroom column and talk about recreation. Last month, I covered health-related technology, and so this month, I'm discussing mental-health-related technology—namely, games, TV, books and music. Next month, will be all command line and learning, but here I focus on enjoying the technology that is supposed to make our lives so much easier!

Next up is Richard Delaney. If you've ever tried to automate your infrastructure with something like Chef or Puppet, you know that spinning up VMs on demand can be challenging. With Vagrant, creating a brand-new virtual machine is a simple one-liner. It supports multiple virtualization platforms, multiple host OSes and abstracts the underlying process. Richard walks through using Vagrant in your VM infrastructure.

Mihalis Tsoukalos dives deep into programming as he shows how to create three-dimensional graphics with OpenGL. While creating a cube on the screen isn't exactly a contender for Game of the Year, the process is an invaluable building "block" for developing with OpenGL. If you want to manipulate graphics in your applications, Mihalis' article is a must-read.

As data centers grow, and IT budgets shrink, managing massive numbers of systems gets overwhelming quickly. In fact, gone are the days when a crusty old system administrator (ahem, me) can configure all the servers one by one and maintain them all him or herself. Mark Burgess describes Promise Theory, which is a framework on which to design and configure massive distributed systems. If you need to re-invent the way your data center works, Promise Theory is a great way to think.

Linux traditionally has been so flexible that it often is considered a one-size-fits-all solution to technology needs. For many use cases, that's still true. If your organization needs a custom Linux distribution or needs to develop something slightly different from what a mainstream distro offers, that flexibility proves to be malleable as well. David Brown walks through customizing a Linux operating system to fit specific needs, including custom development and system modifications. Rather than create a new 'buntu variant, David shows how to tweak existing distros to meet your needs.

I'll admit, programming still is scary territory for some of us, myself included. In this issue, we ease into some of the "development mindset" where we are no longer just consumers, but creators. The beauty of open source is that we get to build on the brilliance of others, and knowledge is freely shared, not guarded like a secret.

We've also included a plethora of non-development content this month. Whether you're a sysadmin, a Linux enthusiast or just someone looking to play some cool games, the August issue aims to please. We hope you enjoy this programming issue of *Linux Journal*. ■

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.

LINUX JOURNAL
Since 1994: The Original Magazine of the Linux Community
TINY CORE LINUX FOR FIREFIGHTERS
HOW TO USE MACROS—A TIME-SAVING FEATURE OF VIM
MOBILE
TIPS FOR USING OLD DEVICES
ANDROID APPS FOR REMOTE SYSTEM ADMINISTRATION
A LOOK AT PRIVATE CLOUDS
OPEN-SOURCE NASA SOFTWARE
GADGETS FOR A HEALTHY LIFESTYLE
TRY FLASK, A POWERFUL MICRO-FRAMEWORK
WATCH: ISSUE OVERVIEW

I Must Be an Extremist

Wow—you should have had the NSA declare *Linux Journal* an extremist organization months ago.

I had, sadly, let my subscription expire a few months ago. However, after reading that *Linux Journal* and its readers are considered "extremists" by the NSA, I have resubscribed for two years with automatic renewal after that.

Take *that* NSA!

And, thanks also for a great publication. Keep those extremist articles about privacy and freedom coming.
—David Both

Thanks David. I have to admit, when I first heard of the XKEYSCORE/NSA stuff, I went to our Web site to see what on earth might have attracted attention. The first thing I saw was a video still shot of me doing the issue intro with a really goofy look on my face. "This is the face of terror", I said to my wife, who replied with something like, "we're all doomed."

We're happy to have you as a subscriber, and we'll try to continue with our extremist ways—whatever that might mean!—Shawn Powers

Extremists?

A glance at the XKEYSCORE snippet shows "These variables define terms and websites relating to the TAILs (The AmnesicIncognito Live System) software program, a comsec mechanism advocated by extremists on extremist forums.", followed by:

```
"$TAILS_terms=word('tails' or 'Amnesiac Incognito
 ➥Live System') and word('linux' or ' USB '
 ➥or ' CD ' or 'secure desktop' or ' IRC ' or
 ➥'truecrypt' or ' tor ');
$TAILS_websites=('tails.boum.org/') or
('linuxjournal.com/content/linux*');"
```

Note that it is quite likely that extremist forums do advocate use of Tor and TAILs,

as do most comsec concerned folks, such as forensic specialists and malware investigators, and perhaps, Three Letter Agencies. Since *Linux Journal* is a publicly available and informative forum on Linux items of interest, such as Tor and TAILs, I would suggest that the proximity of "extremists on extremist forums" to "linuxjournal.com/content/linux" does not associate "linuxjournal" with "extremist forums". However, your point is that access to your Web site in conjunction with the Tor or TAILs query may put one on a "watch list" for further scrutiny—well, given the forensic and malware workload these days, that will push the noise-to-signal ratio up quite a bit (no pun intended). Do you think DistroWatch is somewhere on the list?
—Arnold Hayes

It's so crazy, it's really hard to tell what the exact reason or triggers we might be associated with are. The concept at all is just...crazy. I can't think of a better word. The Linux community and, in kind, the Linux Journal *community are obviously advocates of freedom, but when standing for the very things that make America what it is marks us for scrutiny? Again, crazy.—Shawn Powers*

No More Letters in the Magazine?

I am disappointed that the Letters portion has moved to the Web site instead of being printed in *Linux Journal*. I read *LJ* off-line with my Kindle and look forward to the Letters every issue. Now that it's no longer there, I have to view it on my desktop, which defeats the purpose. Trying out the Letters link on my Kindle when I'm on-line does not provide me with content that is easy to read on the device.

I remember this with paper magazines of old, where they provided "extra content" on the Web. I rarely visited them. And if I wanted to, the magazine often wasn't with me anymore, and I'd forget the link.

Granted, I can view *LJ* on my Nexus 10, or my MacBook, but nothing provides the pleasure of reading that is E Ink.

Also, looking at the link **http://www.linuxjournal.com/letters**, how would I know what was the response to what particular issue?

I propose an alternative, since you wanted to allow for comments: publish the Letters to the Editor in the magazine, with the replies if any, then also provide a permalink to the letter on the Web. This will:

- Allow me to read while in a relaxed mood, either the PDF, .epub or .mobi versions, without the need to switch context and jump to the Web (and from there, go down the rabbit hole of surfing).

- If I'm passionate about a letter, I can click on the link to see if there are more responses.

- I can comment on my thoughts, if I have any on the particular topic.

With the letters still in my head, I can refer to the actual letter in question in *LJ* and click on the link for immediate gratification.

Since *LJ* is an electronically delivered magazine, you do have possibilities that are within your power to manipulate to make the user's experience better than what paper publications can do.
—Wari Wahnab

Thank you for the input, Wari. We did shift the Letters out of last issue, but we've included them back in this one because of letters like yours. As you mention, we're trying to make the Letters something that is interactive in a little more timely manner than once a month. We'll keep tweaking how we do things, and hopefully we'll find a good solution. Your input is invaluable, because otherwise we're just guessing!—Shawn Powers

I See the NSA Is Labeling *Linux Journal* as an Extremist Web Site

If *Linux Journal* is an American-based company with servers located inside America's borders, I'm sure the Electronic Frontier Foundation (EFF, **https://www.eff.org**) and/or American Civil Liberties Union (ACLU, **https://www.aclu.org**) would be happy to litigate a case on your behalf against the NSA, for free.

Warrantless spying on an American journal and its users is clearly a violation of the first and fourth constitutional Amendments—freedom of speech, association and warrantless surveillance.

The lawsuit would be great publicity for your magazine. *Linux Journal* vs. NSA would be plastered on the front pages of every major news organization across the world.

I believe having a copy of the XKEYSCORE source code specifically listing *Linux Journal* as a "target" selector would give your lawsuit plenty of standing for a case against the NSA.
—Ralf

Thank you, Ralf. As of this response, we're still just reeling a bit from the notion that our readers, and most likely our staff, are being monitored and/or spied on. While we do have an international readership, we are indeed an entirely US-based magazine. The worrisome part is, if we are targeted, I can only imagine who else must be! Scary stuff.—Shawn Powers

Days Between Dates Script

Regarding Dave Taylor's "Days Between Dates?" article in the July 2014 issue:

```
grep '29' => Yes, 1729 was a leapyear, so February 29,
 1729 is a valid date.
grep ' 29' => Oops, 1729 wasn't a leapyear, so February
 only had 28 days.
```

—Jyrki Kajala

Dave Taylor replies: *You're right, the addition of a leading space does make this invocation more accurate! This problem will, of course, show up any time there's a "29" in the year, including in 2029, by which time we'll likely have a different way of figuring out whether it's a leap year. "Siri? Is this a leap year?" "Jyrki, you asked me that 4,503,343 milliseconds ago. Why are you asking me again? Have you become obsolete?" "AUuuuggghhhh!"*

More on the Days Between Dates Script

In the sidebar in Dave Taylor's "Days Between Dates?" article in the July 2014 issue, he states that `date` doesn't work for dates before 1970.

If you are using the GNU corelib date utility, this is incorrect:

```
$ date +%s '1776-1-1'
-6122019600

$ date -d '@-6122019600'
Mon Jan  1 00:00:00 MST 1776
```

Also, `date` handles some pretty free-form date entries, although the date has to be a complete date—no attempts with only a month and year, for example.

On a separate note, `date +%j -d 'Dec 31, $year'` will give you either 365 or 366, depending on whether or not it's a leap year. I'm not sure how it handles years in which there were less than 365 days. I suspect it just treats all dates as Gregorian.
—Alan Young

Dave Taylor replies: *Thanks for the helpful message. As an old-school Linux and UNIX guy, negative seconds-since-epoch bugs me, but that's okay, because your observation that the* `%j day-of-the-year` *value easily can be used to test for leap year is terrific and one I'm going to borrow!*

Removing Letters to the Editor from the Digital Magazine

I was surprised to find that you are no longer including the Letters to the Editor section in the *Linux Journal* digital magazine (as of the July 2014 issue). Since your magazine has gone digital-only, I find that I now read it cover to cover (mostly), and I particularly enjoyed the give and take that occurs between the readers and the article writers in the Letters section.

I realize that I can still read this interchange on-line, but doing so adds a little more "friction" to the process and fails to inform me when there is feedback on the various topics.

Since *Linux Journal* is now available only in digital formats (PDF, .epub and so on), it seems that removing the Letters section does not result in much of a cost-savings (possibly a small savings in page layout/markup/design). Do you have a more compelling reason for this change?

If you feel that this must be a permanent change, perhaps you could include a sidebar listing the topics of letters received (with links to enable accessing them on-line).

I also miss the "Photo of the Month", although I realize you may not always have a photo to share with us every month.
—Gerald

Thanks Gerald! Yes, we shook things up a bit taking out the Letters to the Editor. They're (obviously) back this month. We'll eventually find the perfect mix of interactivity and convenience. Last month, was our first step toward that goal. Thank you for letting us know your thoughts; it really helps us shape the future! We'll continue having Photo of the Month as well, as long as we receive good submissions, so please send your Linux-related photos to ljeditor@linuxjournal.com.—Shawn Powers

Keep Up the Great Work!

I just subscribed to *LJ* two months ago, and I love it! I have been involved with computers and electronics since I was ten years old when I started with the Apple IIE. I also love high-power rocketry! I plan on building a Linux computer to control a high-power rocket I plan on doing for Level 2 NAR certification soon. Also, I love health and fitness, as I was a personal trainer for nine years. Now I train and workout hard-core for myself and train clients part time. I am a technology and fitness freak! I love it!

I love the PDF format of *LJ*, as this is way of the future, since we can eliminate paper clutter and have links to click on when we want more information on details. I love it! I have tons of paper books and old magazines that I want to convert to PDF. I started doing it myself with a good Fujitsu scanner, but I have so many, I want to hire someone or some company to scan most of these books for me.

Keep up all your amazing content in the PDF format!
—Christian Lombardo

Welcome aboard, Christian! I'm still struggling with my fitness routine. The diet is going well (40 pounds lost as of today, July 5th), but getting motivated to exercise is harder. I still try to walk as much as possible, but I'd like to be a runner someday.

Thanks for sharing your appreciation for the digital publication. The feelings on that subject vary widely from user to user. Thankfully, I think the tablet/reader market has matured to the point where reading digitally is really a painless experience.—Shawn Powers

WRITE *LJ* A LETTER

We love hearing from our readers. Please send us your comments and feedback via http://www.linuxjournal.com/contact.

PHOTO OF THE MONTH
Remember, send your Linux-related photos to ljeditor@linuxjournal.com!

LINUX JOURNAL

At Your Service

SUBSCRIPTIONS: *Linux Journal* is available in a variety of digital formats, including PDF, .epub, .mobi and an on-line digital edition, as well as apps for iOS and Android devices. Renewing your subscription, changing your e-mail address for issue delivery, paying your invoice, viewing your account details or other subscription inquiries can be done instantly on-line: **http://www.linuxjournal.com/subs**. E-mail us at subs@linuxjournal.com or reach us via postal mail at *Linux Journal*, PO Box 980985, Houston, TX 77098 USA. Please remember to include your complete name and address when contacting us.

ACCESSING THE DIGITAL ARCHIVE: Your monthly download notifications will have links to the various formats and to the digital archive. To access the digital archive at any time, log in at **http://www.linuxjournal.com/digital**.

LETTERS TO THE EDITOR: We welcome your letters and encourage you to submit them at **http://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: **http://www.linuxjournal.com/author**.

FREE e-NEWSLETTERS: *Linux Journal* editors publish newsletters on both a weekly and monthly basis. Receive late-breaking news, technical tips and tricks, an inside look at upcoming issues and links to in-depth stories featured on **http://www.linuxjournal.com**. Subscribe for free today: **http://www.linuxjournal.com/enewsletters**.

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: **http://ww.linuxjournal.com/advertising**. Contact us directly for further information: ads@linuxjournal.com or +1 713-344-1956 ext. 2.

diff -u
WHAT'S NEW IN KERNEL DEVELOPMENT

One problem with Linux has been its implementation of **system calls**. As **Andy Lutomirski** pointed out recently, it's very messy. Even identifying which system calls were implemented for which architectures, he said, was very difficult, as was identifying the mapping between a call's name and its number, and mapping between call argument registers and system call arguments.

Some user programs like **strace** and **glibc** needed to know this sort of information, but their way of gathering it together—although well accomplished—was very messy too.

Andy proposed slogging through the kernel code and writing up a text file that would serve as a "master list" of system calls, giving the call name, the corresponding call number, the supported architectures and other information. Among other things, this would allow tools like glibc to eliminate their ugly implementations and use a simple library to get this information out of the kernel.

H. Peter Anvin liked the idea, but said it would take a lot of work to get it right. He mentioned that he'd been advocating something along the same lines for a long time, dating back to his work on **klibc**.

Various other folks liked Andy's idea as well—particularly anyone involved with user code that currently had to deduce system call organization piecemeal. **David Howells** remarked that it would be wonderful if strace could rely on Andy's master list as well. And, **Michael Kerrisk** said the **manpages project** also would be interested in tracking the progress of the master list.

There's always a special case that would benefit from tweaking the **process scheduler** just a little bit beyond The Good. Recently, **Khalid Aziz** from **Oracle** submitted some code to allow user processes to claim additional timeslices. Typically, the kernel itself controls that sort of resource allocation, because otherwise the system is dependent on the friendliness or well-codedness of user applications.

But, Khalid's database folks had noticed a problem with large numbers of threads vying for the same mutex. If one of those threads had the mutex and was almost ready to give it up, the scheduler might run through the whole queue of other processes, none of which could actually run because they were all waiting for that one mutex. And like a thumb in the eye, the process holding the mutex was all set to give it up, but couldn't, since it had been preempted. Much better, Khalid said, would be to allow the process holding the mutex to delay preemption, long enough to give up that mutex. Then all the other processes could take their turn and do actual work, rather than spend their precious timeslices spinning on an unavailable lock.

Khalid said his code showed a 3–5% speedup relative to the previous case. But, there was still a fair bit of reluctance to accept his code into the kernel.

In particular, H. Peter Anvin pointed out that Khalid's code allowed userspace to transform the kernel's natural preemptive multitasking into a cooperative multitasking model, in which processes all had to agree on who would get timeslices, and when—and some processes could aggressively claim timeslices at the expense of the others.

Davidlohr Bueso pointed out that a voluntary preemption model might work better with the kernel's existing implementation, allowing processes to give up their timeslice to another process voluntarily. There was no danger from hostile processes there.

There were various suggestions for alternatives to Khalid's design, but Khalid always pointed out that his way was fastest. But, **Thomas Gleixner** said that "It's a horrible idea. What you are creating is a crystal ball-based form of time-bound priority ceiling with the worst userspace interface I've ever seen."

That was the real problem, apparently. Giving user code the ability to preempt the normal scheduling process meant that neither the kernel nor other userspace processes could predict the behavior of the system, or even properly debug problems.

At one point Thomas said, "What you're trying to do is essentially creating an ABI which we have to support and maintain forever. And that definitely is worth a few serious questions." He added, "If we allow you to special-case your database workload, then we have no argument why we should not do the same thing for real-time workloads where the **SCHED_FAIR** housekeeping thread can hold a lock shortly to access some important data in the **SCHED_FIFO** real-time computation thread. Of course the RT people want to avoid the lock contention as much as you do, just for different reasons."

Eric W. Biederman also objected to Khalid's code, saying, "You allow any task to extend its timeslice. Which means I will get the question why does why does really_important_job only miss its latency guarantees when running on the same box as sched_preempt_using_job?" And he said, "Your change appears to have extremely difficult to debug non-local effects."

There seems to be a lot of interest in implementing a feature like what Khalid has proposed, but there also seems to be security concerns, debugability concerns and maintainability concerns that make the whole thing very iffy. But, it's still possible that Khalid could address those concerns and come up with a patch that does what the database people want, without the mess.

—ZACK BROWN

They Said It

It is not always the same thing to be a good man and a good citizen.
—Aristotle

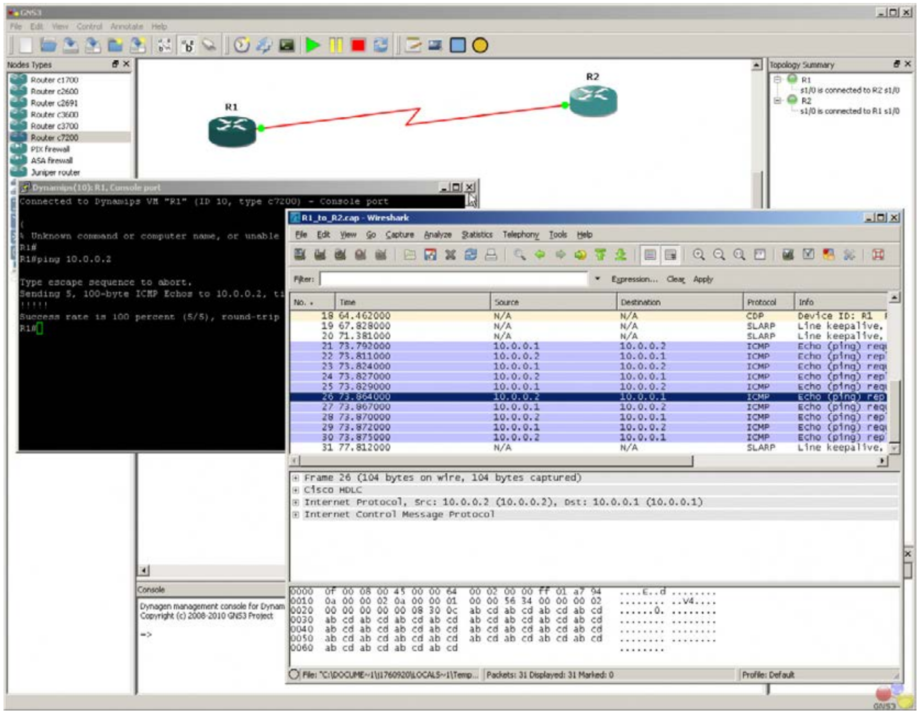
Life is a reciprocal exchange. To move forward, you have to give back.
—Oprah Winfrey

It is only possible to live happily ever after on a day-to-day basis.
—Margaret Bonnano

The greater man the greater courtesy.
—Alfred Lord Tennyson

That is not what Geek means to me. We are more than the hobbies that we do or the things that we like. To me, Geek means an outsider, a rebel, a dreamer, a creator, a fighter. It's a person who dares to love something that isn't conventional.
—Felicia Day

Non-Linux FOSS: a Virtualized Cisco Infrastructure?



```
Connected to Dynamips VM "R1" (ID 10, type c7200) - Console port

(
% Unknown command or computer name, or unable
R1#
R1#ping 10.0.0.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.2, ti
!!!!!
Success rate is 100 percent (5/5), round-trip
R1#
```

```
No.  Time        Source    Destination  Protocol  Info
 18  64.462000   N/A       N/A          CDP       Device ID: R1
 19  67.828000   N/A       N/A          SLARP     Line keepalive,
 20  71.381000   N/A       N/A          SLARP     Line keepalive,
 21  73.792000   10.0.0.1  10.0.0.2     ICMP      Echo (ping) requ
 22  73.811000   10.0.0.2  10.0.0.1     ICMP      Echo (ping) rep
 23  73.824000   10.0.0.1  10.0.0.2     ICMP      Echo (ping) requ
 24  73.827000   10.0.0.2  10.0.0.1     ICMP      Echo (ping) rep
 25  73.829000   10.0.0.1  10.0.0.2     ICMP      Echo (ping) requ
 26  73.864000   10.0.0.2  10.0.0.1     ICMP      Echo (ping) rep
 27  73.867000   10.0.0.1  10.0.0.2     ICMP      Echo (ping) requ
 28  73.870000   10.0.0.2  10.0.0.1     ICMP      Echo (ping) rep
 29  73.872000   10.0.0.1  10.0.0.2     ICMP      Echo (ping) requ
 30  73.875000   10.0.0.2  10.0.0.1     ICMP      Echo (ping) rep
 31  77.812000   N/A       N/A          SLARP     Line keepalive,

⊞ Frame 26 (104 bytes on wire, 104 bytes captured)
⊞ Cisco HDLC
⊞ Internet Protocol, Src: 10.0.0.2 (10.0.0.2), Dst: 10.0.0.1 (10.0.0.1)
⊞ Internet Control Message Protocol

0000  0f 00 08 00 45 00 00 64  00 02 00 00 ff 01 a7 94   ....E..d ........
0010  0a 00 00 02 0a 00 00 01  00 00 56 34 00 00 00 02   ........ ..V4....
0020  00 00 00 00 00 08 30 0c  ab cd ab cd ab cd ab cd   ......0. ........
0030  ab cd ab cd ab cd ab cd  ab cd ab cd ab cd ab cd   ........ ........
0040  ab cd ab cd ab cd ab cd  ab cd ab cd ab cd ab cd   ........ ........
0050  ab cd ab cd ab cd ab cd  ab cd ab cd ab cd ab cd   ........ ........
0060  ab cd ab cd ab cd ab cd                            ........

File: "C:\DOCUME~1\j1760920\LOCALS~1\Temp...  Packets: 31 Displayed: 31 Marked: 0  Profile: Default
```

```
Dynagen management console for Dynam
Copyright (c) 2008-2010 GNS3 Project

=>
```

Screenshot provided by http://www.gns3.net.

We're all familiar with the idea of virtualized computers. Not only are they a great way to better utilize resources in a server room, but they also allow you to create and destroy servers in the blink of an eye. That's perfect for a lab or training environment. Unfortunately, it's always taken a rack of actual hardware to create a training lab for Cisco hardware. Thanks to GNS3 (Graphical Network Simulator 3), that's no longer the case.

GNS3 is an open-source application that creates a virtual infrastructure of Cisco (or other) hardware. Not only can you watch the traffic flow, but you also can connect directly to the virtual devices and configure them like the actual hardware devices they represent. On the surface, it looks like a Visio diagram, but it's a diagram that actually *does* something!

Whether you're actively trying to learn to use Cisco devices or just want to try out some networking scenarios, GNS3 is an incredible tool for sandboxing network hardware. It does require you to provide the IOS firmware (they can't provide Cisco's operating systems themselves), but that's it. It works under Windows, along with OS X and Linux. If you're interested, download a copy today at http://www.gns3.net.—**SHAWN POWERS**

IndieBox: for Gamers Who Miss Boxes!

There are lots of cool ideas on the Internet that never really make it out of the "startup" phase. IndieBox has been around only for a few months, but I really, really hope it catches on.

Here's the idea:

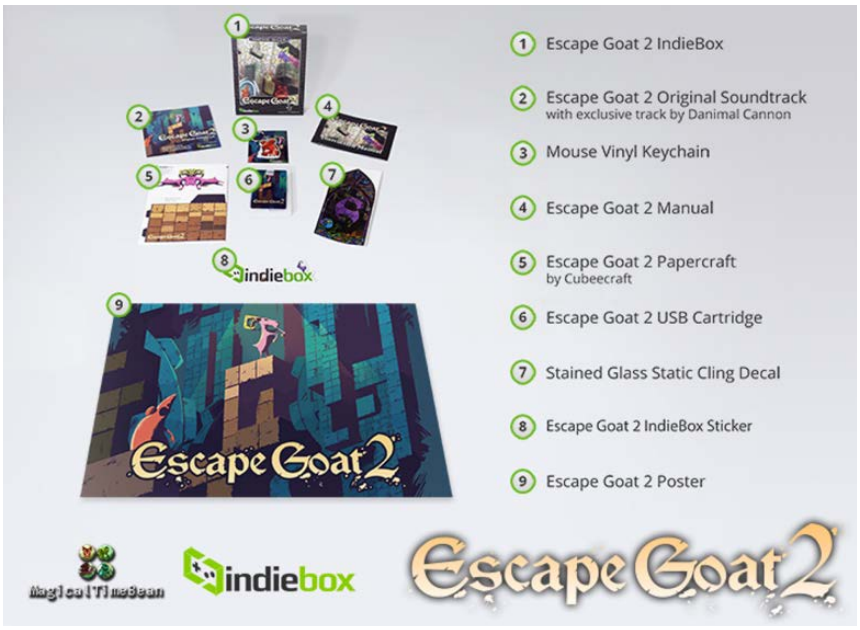

1 Escape Goat 2 IndieBox
2 Escape Goat 2 Original Soundtrack with exclusive track by Danimal Cannon
3 Mouse Vinyl Keychain
4 Escape Goat 2 Manual
5 Escape Goat 2 Papercraft by Cubeecraft
6 Escape Goat 2 USB Cartridge
7 Stained Glass Static Cling Decal
8 Escape Goat 2 IndieBox Sticker
9 Escape Goat 2 Poster
indiebox
Escape Goat 2
MagicalTimeBean

Photo courtesy of http://www.theindiebox.com.

- Every month, you get a Linux/Mac/Windows-compatible Indie game in the mail.

- The games come on a really cool USB drive shaped like a credit card covered in awesome game art. It's like a modern-day Nintendo cartridge.

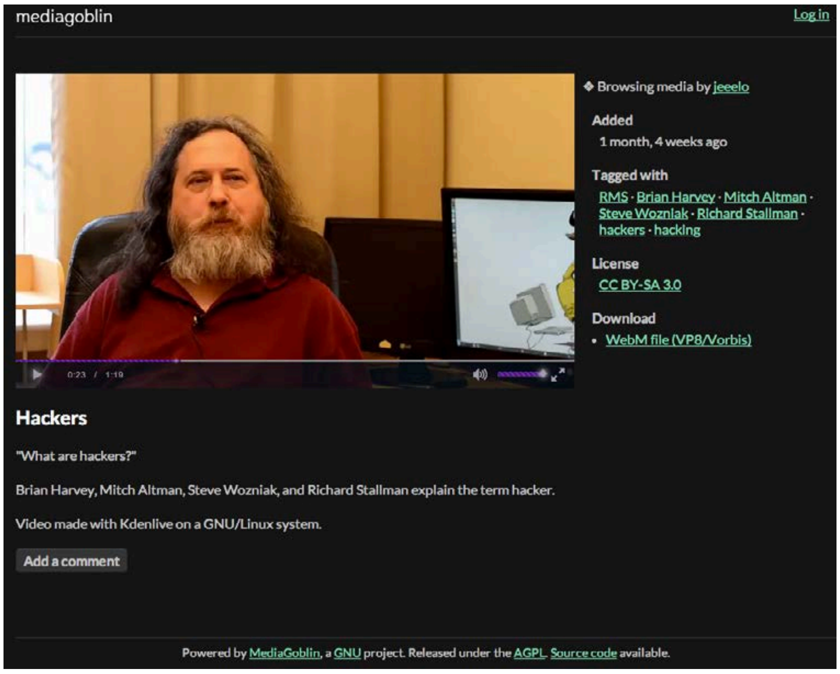
- Inside the game box (which is also covered in game art), you'll find posters, keychains, cardboard cutouts, CDs of game music or anything else the folks at IndieBox can scrape together to make the monthly game awesome and nostalgic.

- Most months you'll also get a Steam code, because although old-school boxed games are awesome, Steam still is really convenient.

As someone who belongs to a "coffee of the month" club, I can assure you that getting a fresh new thing every month in the mail is fun and exciting. When that new thing is a nostalgic trip to my youth, plus a really fun cross-platform game? For me, it's worth the $16.99 a month. If you miss the days of buying games in a box from the department store, you'll love the IndieBox experience. Check it out at **http://www.theindiebox.com**.

—SHAWN POWERS

Roll Your Own YouTube/ Flickr with MediaGoblin


mediagoblin

Log in

◆ Browsing media by jeeelo

Added
1 month, 4 weeks ago

Tagged with
RMS · Brian Harvey · Mitch Altman · Steve Wozniak · Richard Stallman · hackers · hacking

License
CC BY-SA 3.0

Download
- WebM file (VP8/Vorbis)

Hackers

"What are hackers?"

Brian Harvey, Mitch Altman, Steve Wozniak, and Richard Stallman explain the term hacker.

Video made with Kdenlive on a GNU/Linux system.

Add a comment

Powered by MediaGoblin, a GNU project. Released under the AGPL. Source code available.

Screenshot from http://roaming-initiative.com/mediagoblin/u/jeeelo/m/hackers.

Everyone has wasted an afternoon on YouTube clicking through videos of talking cats, screaming goats and bad-lip-reading renditions of popular movies. Heck, there are plenty of YouTube videos of me doing odd and silly things as well. (Does anyone remember 'Buntu Family Theater?) For important family videos, however, I much prefer to control my own data. I've tried over the years to keep an archive of home movies and such in a folder on a server somewhere, but they never get seen because getting to them in inconvenient. That's where MediaGoblin comes in.

MediaGoblin is an open-source Web-based application that allows you to host audio, video and photographic media on your own server. It still allows that media to be easily viewed via the Web, however, so you get the convenience of YouTube with the security of hosting files yourself.

Hosting your own local version of YouTube might not be your cup of tea, but for me, it's a great compromise between convenience and data privacy. If you want to give MediaGoblin a try, head over to **http://www.mediagoblin.org** and download a copy today. Or, check out one of the many publicly hosted installations to see if you like the interface. The screenshot here is from **http://roaming-initiative.com/mediagoblin/u/jeeelo/m/hackers** and includes an interview that hits home for nerds like us!—**SHAWN POWERS**

Getting Good Vibrations with Linux

Vibrations and wave motions describe many different physical systems. In fact, most systems that dissipate energy do so through waves of one form or another. In this article, I take a look at gvb (Good ViBrations, http://www.pietrobattiston.it/gvb), a Linux application you can use to visualize and model wave motion and vibrations.

Installation should be relatively easy. It already should exist in the packaging system of most distributions. For example, on Debian-based systems, you can install it with this:

```
sudo apt-get install gvb
```

You will notice that a large number of Python requirements also are installed. This gives you an idea of what you need as requirements if you want to build gvb from source.

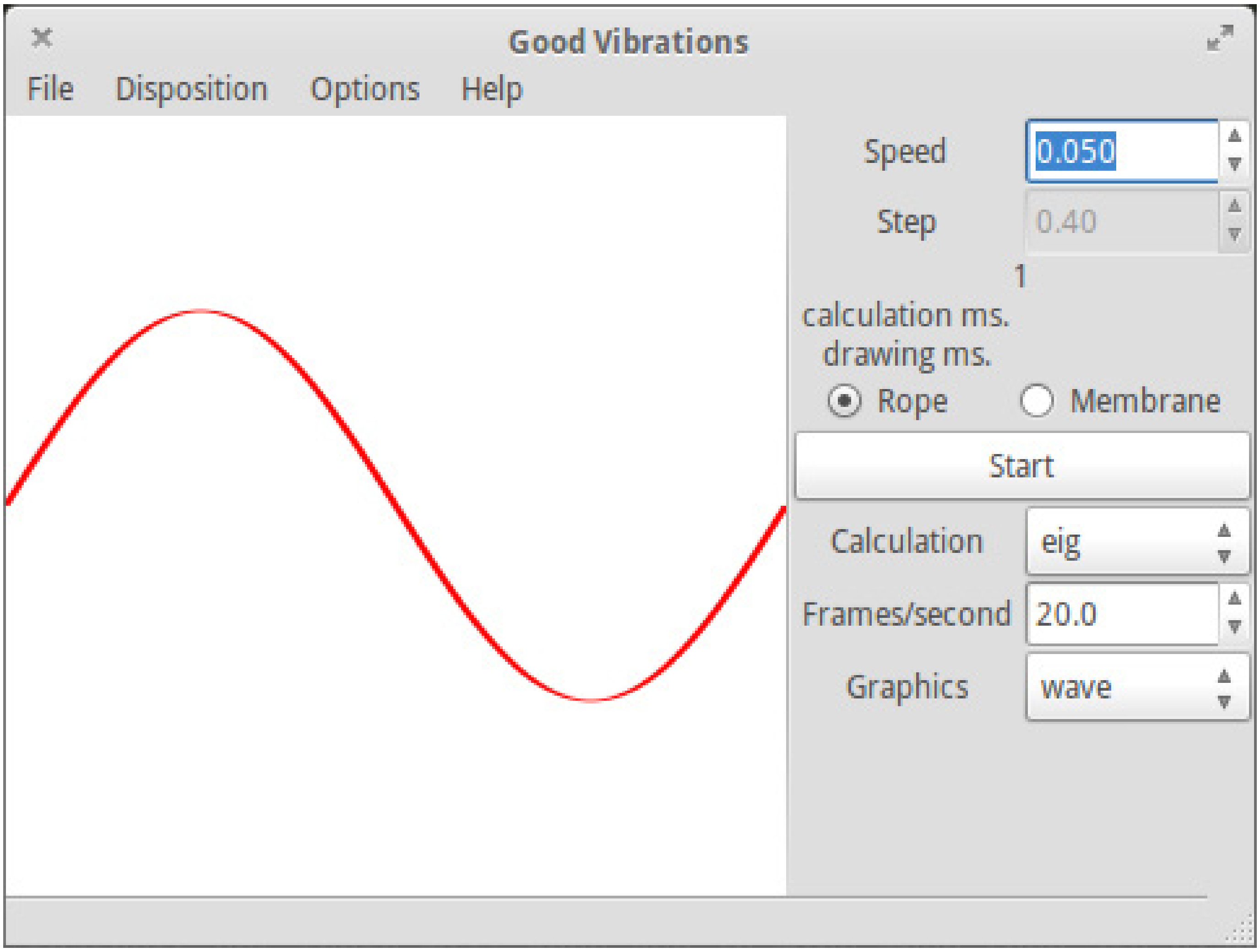
Now that you have gvb and its requirements installed, you can start it by running the command `gvb`. When it starts, you will see a main display window with a pane on the right-hand side with calculation options. In the main window, there will be a starting sine wave form.

Just to make sure that everything is working, you can click on the start button in the right-hand pane. This starts the calculations used to model the wave motion and shows each time step in the main display window. You should see the evolution of a basic sine wave happening.

Let's use this starting waveform to play with some of the available calculation options. The first option is the speed of the wave. You can change this while the wave is being modeled and animated, so you can see in real time how the wave changes with a different speed.

Below the start button are three available options. The calculation drop-down lets you change the algorithm used to do the calculations. The default is eig, and the other three methods are quad, naif and naif matrix. When you select any of the other three methods, the step size option at the top of the pane becomes active.

You then can change the size of the steps used in calculating



Speed 0.050
Step 0.40
1
calculation ms.
drawing ms.
[✓ Rope] Membrane
Calculation eig
Frames/second 20.0
Graphics wave

Figure 1. When you first start gvb, you will see a sine wave ready to start modeling.

the wave motion. You can set the number of frames per second to display. The higher this value is, the smoother your wave animation will be. You probably won't notice any improvement beyond 30 frames per second, simply due to the limits of human vision.

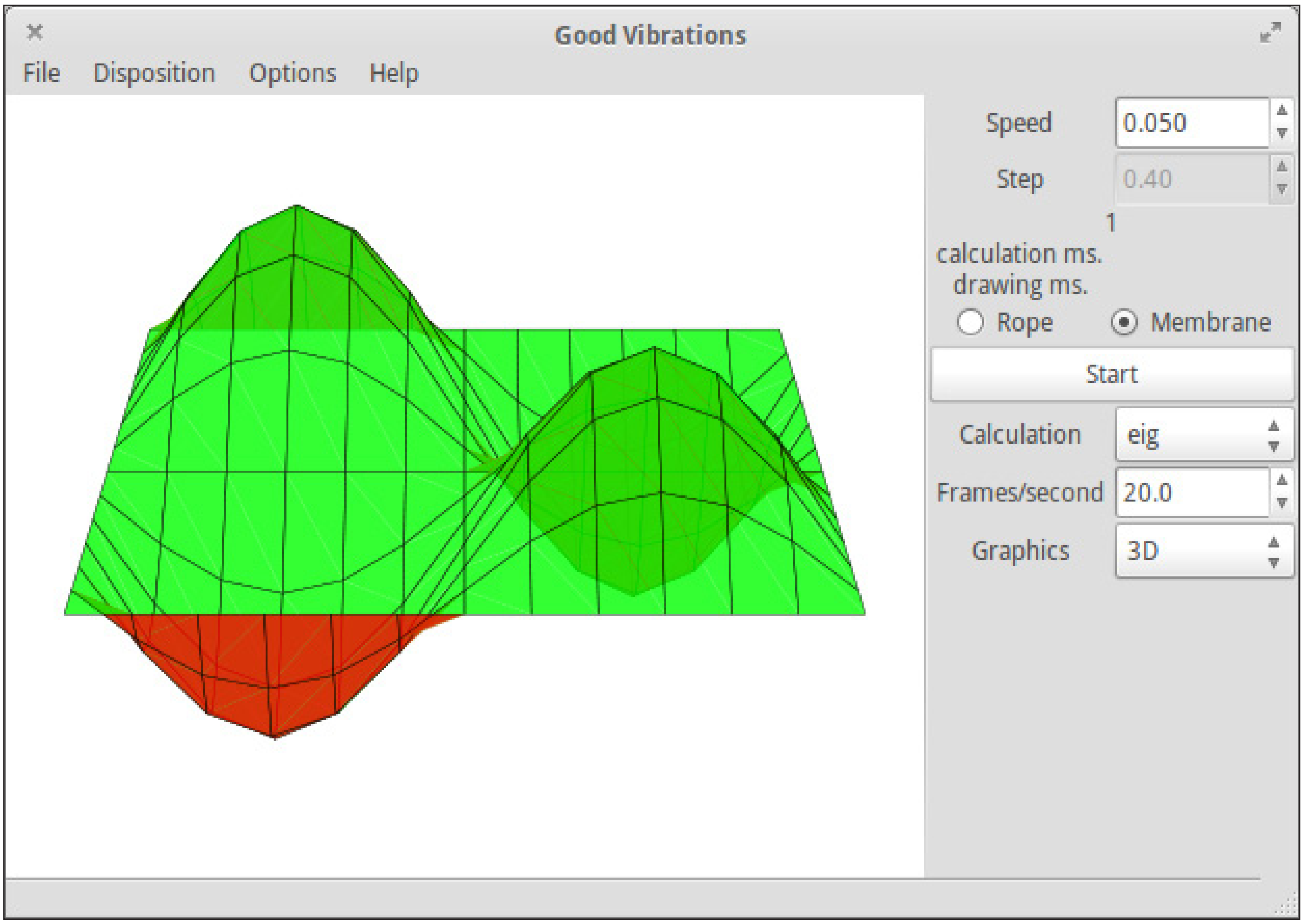
The last option allows you to change what graphics are actually rendered in the display. Wave shows the waveform itself. Temp displays a color map representing the amplitude with different colors. Speed displays the waveform, like Wave, but it also includes a series of bars along the wave indicating the magnitude of the speed at those points along the wave. This display helps students get a feeling for how fast the different parts of the wave are moving.

Before going on, it would be worth spending a few minutes to take a cursory look at the types of calculations gvb can do. The default calculation method is eig. This method actually uses the `eig()` function found in the linalg part of NumPy. This uses the plane wave expansion method to get an eigenvalue formulation of the problem and solve that.

The naif calculation method is a naive method. It simply takes the current amplitude of the wave at some point and calculates how far it should travel based on a linear application of the current speed at that point.

The naif matrix method just tries to speed up the naif method by treating the wave as a vector rather than point by point. Neither of these two methods should be used for any reason other than to see just how bad they are.

The last method is quad. This



Speed 0.050
Step 0.40
1
calculation ms.
drawing ms.
( ) Rope [✓ Membrane]
Calculation eig
Frames/second 20.0
Graphics 3D

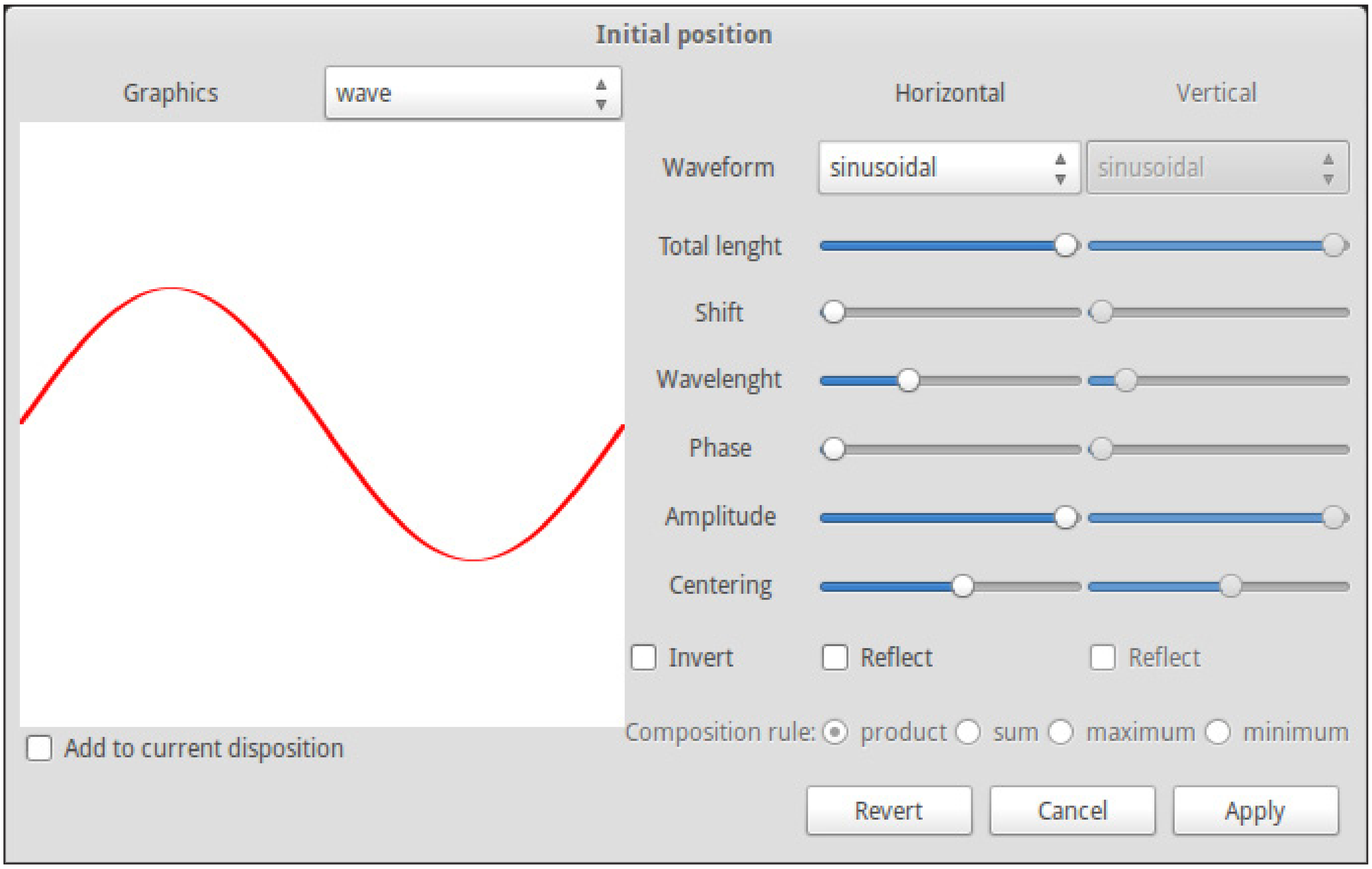
Figure 2. You even can model waves on a membrane with gvb.

method expands on the simplistic speed calculation of the naif method with a more accurate method. In most cases, you simply will want to use the eig method, as it is the most accurate for most situations.

Up until now, I've only been discussing a sinusoidal wave on a string. But really interesting things happen when you can look at other starting waveforms. There are a series of pre-programmed waveforms available for you under the menu entry Disposition→1 dimension: precooked. Here you will find more than a dozen available options.

You also may have noticed that there is also an entry for two-dimensional precooked options. That's right, gvb can model waves on a membrane as well.

If you select one of these precooked options, you should notice that the graphics option is changed to 3-D, and the type of drawing is changed from rope to membrane. Clicking the start button will show you how the starting waveform propagates


Initial position

Graphics: wave

	Horizontal	Vertical
Waveform	sinusoidal	sinusoidal

Total lenght, Shift, Wavelenght, Phase, Amplitude, Centering

[ ] Invert [ ] Reflect [ ] Reflect

Composition rule: [✓ product] sum maximum minimum

[ ] Add to current disposition

Figure 3. The advanced window allows you to create your own customized waveforms.

across the given membrane. Under the Disposition menu entry, there also are entries for advanced one- and two-dimensional systems. Selecting one of these will pop up a new window where you can set the various parameters for your own model.

You still are limited to sinusoidal, triangular, square and peak waves as your base forms. If you have something even more complicated in mind, you actually can define the amplitude and speed at each point in a text file. For example, to model a string, the text file will have three lines. The first line will contain the shape—in this case, the number of points along the string. The second line will contain a series of amplitude values, comma-separated, one for each point. The third line will contain a series of speed values, comma-separated, one for each point. You next need to end the file with a new-line character at the end. Then, you can read this into gvb to provide your starting waveform. This gives you maximum flexibility. You can do the equivalent two-dimensional input file if you want to model a membrane rather than a string.

One final thing you may want to do with gvb is generate a movie of the wave you are modeling. Selecting the menu entry Options→Save frames as png will pull up a file selection window. You need to choose a directory to store all of the image files for each frame of the animation. Once this is done and you click the start button, a PNG file will be written out for each frame. In order to generate a movie out of those, you need to have FFmpeg installed on your system. Then, all you have to do is change directory to the spot where gvb is writing out these image files and run the command:

```
ffmpeg -i *.png video.mp4
```

Now you have a video file that you can share displaying the wave evolution that you are modeling.

As you have seen, gvb can be a very fun application to use. It especially gives students a more intuitive feel for how waves behave, in both one and two dimensions. The ability to visualize the speeds for various sections also helps conceptualize the physics involved. Hopefully, the teachers out there will keep gvb in mind when they are planning their next science class to help students understand wave mechanics.

—JOEY BERNARD

STOP THEM

BEFORE THEY STOP YOU.

LINUX JOURNAL PRESENTS A SPECIAL FREE WEB EVENT:

One Click, Universal Protection: Implementing Centralized Security Policies on Linux Systems

WEDNESDAY, AUGUST 27TH
at 11:00AM

As Linux continues to advance in its take-over of corporate data centers and institutions, ensuring the integrity and protection of Linux systems becomes a more critical priority every day. With 60% of the world's websites and an increasing share of most organizations' mission-critical workloads running on Linux, failing to stop malware and other advanced threats on Linux can severely negatively impact an organization's reputation and bottom line.

While Linux contains several features that can provide enhanced system security and integrity, most organizations either do not implement or only occasionally ensure proper enforcement. Why? Because they lack 1) real-time visibility into their environments and 2) the ability to centrally enforce policy across their entire system.

JOIN THE *LINUX JOURNAL* SECURITY EXPERTS AND BIT9'S PAUL RIEGLE FOR A DISCUSSION COVERING:

› The rise of Linux advanced attacks—why signature-based security measures are ineffective against advanced attacks.

› A brief overview of **Bit9 + Carbon Black**'s application control and ETDR solutions.

› How to help your organization centrally manage and enforce security policies across large Linux environments.

>> Reserve your spot NOW for this valuable FREE webinar!

Android Candy: Oyster—Netflix for Books!

LINUX JOURNAL EDITORS' CHOICE

For avid readers who can't find the time to visit their local library or struggle to carry giant tomes of awesomeness around with them, eBooks have become a convenient way to consume books. Whether it's on a Kindle, a Nook or in an app on your phone, eBooks are the ultimate in portability. Sometimes, however, it can be difficult to find the book you want in a format you can read. Enter Oyster (**http://www.oysterbooks.com**).

Netflix has taken movie streaming to the point where it's more common to stream a movie than it is to rent a DVD. Heck, many of us have abandoned all

---

LINUX JOURNAL

now available for the **iPad** and **iPhone** at the **App Store.**

Available on the App Store

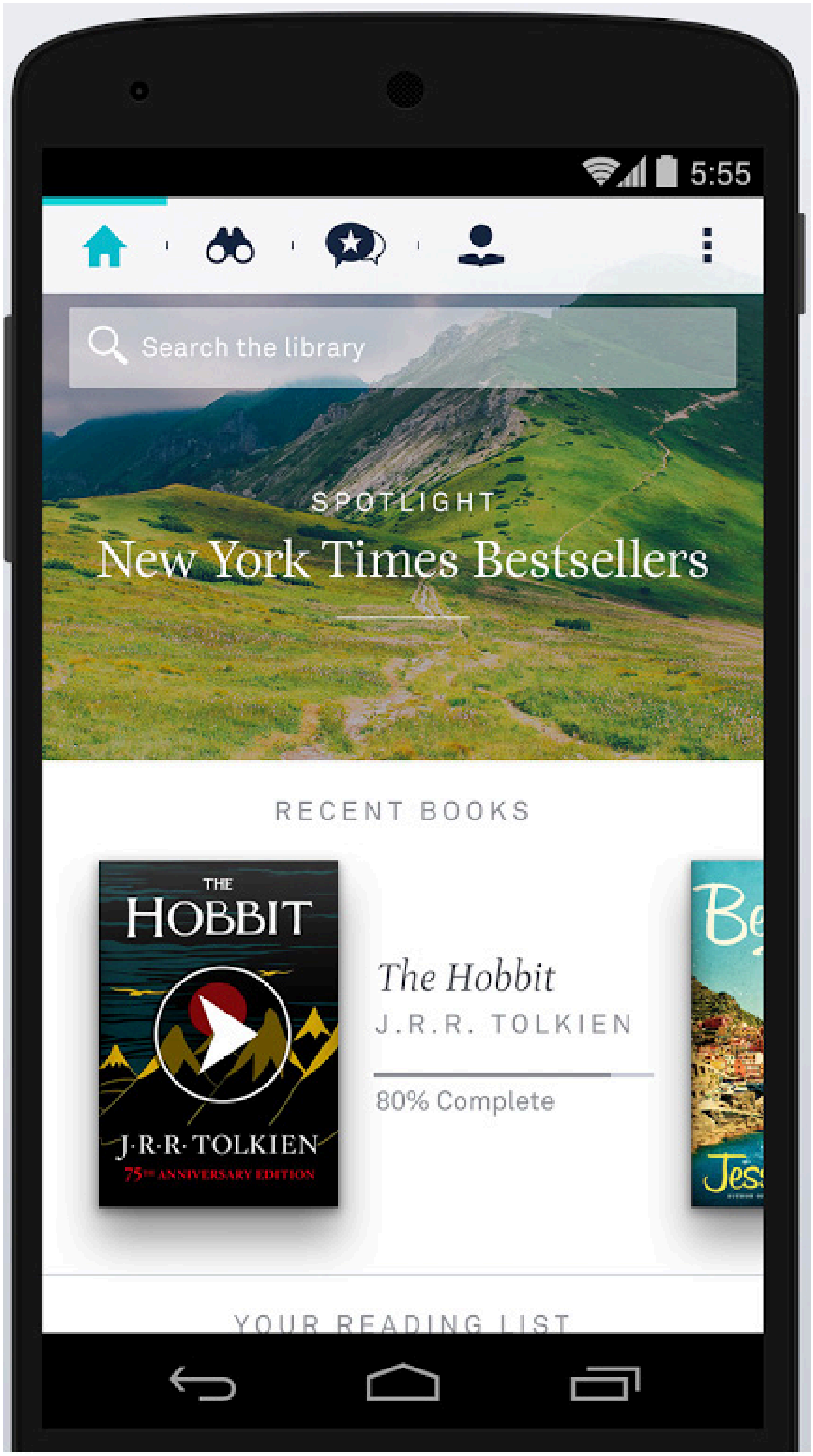
linuxjournal.com/ios

For more information about advertising opportunities within *Linux Journal* iPhone, iPad and Android apps, contact John Grogan at +1-713-344-1956 x2 or ads@linuxjournal.com.

television entertainment options *except* Netflix. Oyster is an app designed to do the same for books. For $9.95 a month, you get access to more than a half-million books and enjoy unlimited reading on your Android-powered device.

The app features cross-device (and cross-platform) location sync, off-line reading and a 30-day free trial with unlimited book reading. Like Netflix, Oyster does use DRM on its titles. Because the books aren't ever owned by me, I personally don't have a moral dilemma with DRM in this case, but for some it will be a showstopper. Nevertheless, due to its wide selection, awesome idea and cross-platform availability, Oyster takes this month's Editors' Choice award!

—SHAWN POWERS



REUVEN M. LERNER

First Steps with Web Development

Interested in becoming a Web developer? Here are some tips on where to start.

The Web is more than 20 years old, and a huge number of developers now work with it. It used to be enough to be a "Webmaster", meaning a jack of all trades: developer, designer, database administrator and system administrator. Today, we see increasingly specialized aspects of Web development—front-end developers, back-end developers, devops professionals and then "full-stack developers", who can do it all, or at least claim to.

This increased specialization reflects, in large part, the degree to which the Web has become huge and essential. Web sites are an important part of the world economy, and as befits an important part of the economy, they need to be designed, developed and maintained like any other type of engineering.

But with the specialization, and with the growth of the Web, also comes opportunity. During the past few months, my consulting work has increasingly involved coaching individuals who see the potential payoff (financial and personal) for becoming Web developers, and who want to improve their skills in order to do so. These people know if they manage to learn these skills, they'll be able to find a job, either with a startup or a large firm—or even better, starting their own on-line businesses.

I've been exposed to more and more people who know they want to work with "the Web", and they know they want to develop software, but they really don't understand the fundamentals or even where to begin.

So in this article, I describe some of the basics people should know if they're interested in doing Web development.

Not only are many traditional servers running Linux, but virtualization has become a stable and popular technology, using such systems as Vagrant, an open-source system that lets you develop on a virtual machine and then use that same virtual machine in production.

As a full-stack Web developer myself, I think it's important for anyone working with Web technologies to have at least a passing knowledge of each of these topics: Linux, databases, a server-side language and framework, and client-side technologies (HTML, CSS and JavaScript).

If you can learn at least a little of each of those, you'll be well on your way to being able to create Web applications. Moreover, you'll probably find that one or more parts of this technological constellation are particularly appealing to you, giving you a sense of the area in which you want to specialize. Or, if you find that you enjoy it all, you can be more of a generalist, applying your broad knowledge to many different aspects of Web applications.

Linux and Servers

Perhaps it won't surprise a reader of *Linux Journal* that being familiar with Linux is a good thing. However, it's not just good, but essential, given that Linux powers a surprisingly large proportion of the Web. Not only are many traditional servers running Linux, but virtualization has become a stable and popular technology, using such systems as Vagrant, an open-source system that lets you develop on a virtual machine and then use that same virtual machine in production. I must admit that I haven't yet taken the plunge and used Vagrant, but each time I have to wrestle with server configuration or get something to work on my employee's computer that worked fine on mine, Vagrant seems like an increasingly good option.

But really, if all you want to do is start with some Web development, a deep dive into Linux isn't necessary. Yes, you need to understand some basics, such as directories, users and permissions, as well as how to install and run programs. You also should know where your system's log files are—both general-purpose log files and also those produced by your HTTP server.

If you're new to the world of Linux administration, learning such systems as DNS, SMTP and SSL might well confuse more than help you; try to rely on third parties for as much of this as possible.

Knowing how to use a classic UNIX text editor, such as Emacs or vim, will help a great deal, even if you're more of a GUI kind of person, simply because you won't always have access to a GUI. And of course, learning the ins and outs of UNIX utilities, such as grep and find, will help tremendously.

I would say that you shouldn't worry too much about all of the virtualization and container systems that exist—not because they are irrelevant, but because they likely will confuse and complicate your progress, and those are things you can learn over time. If and when you decide to use Vagrant, Docker or Amazon Web Services, you can specialize and learn those particulars.

I should add that Linux is a deep and complex system, one that takes many years to understand and internalize fully. If it seems foreign and difficult at first, that's only natural, but over time, you'll customize your environment, learn more about how to work with it and feel more comfortable working with it.

Databases

There is no requirement that a Web application have a database. However, in practice, it's rare for a modern Web application not to use a database at all. (My company's Web site doesn't have a database, but that's because it's mostly for promotional purposes. However, my blog does have a database to store posts and comments, among other things.) I'm an old-school kind of guy, so I believe that traditional, relational databases with SQL are the right way to go for many Web applications. However, I do admit that the learning curve—of a client-server protocol, of the SQL query language and of the various tricks involved in a relational database—can be steep.

For this reason alone, many people starting off in Web development have begun to use so-called NoSQL databases, such as MongoDB, which allow you to store complex documents in a format that naturally maps to objects and modern programming languages, without the overhead of SQL queries. I can understand this perspective, and if you feel that learning SQL would be too complex for you, you might indeed want to go this way.

However, I think there's a reason why

relational databases continue to be so popular, namely because more than 40 years of research and practice have been so beneficial. I personally think it's worth the time and effort to work with a relational database, particularly if you'll be using a high-level language and object-relational manager (ORM) that handles most of the queries for you and, thus, hides many of the queries from you. The long-term benefits of understanding SQL and relational databases, and the flexibility of a normalized relational database, are still without peer, in my view.

This doesn't mean I think that all NoSQL is meaningless or worthless. However, I have found much more utility in key-value stores, such as Redis, than in NoSQL databases, such as MongoDB, particularly with the recent advances in PostgreSQL, which make storing and retrieving JSON objects easy and efficient.

If you're interested in database technology, a great book to try is *Seven Databases in Seven Weeks*, by Eric Redmond and Jim Wilson, and published by the Pragmatic Programmers. That book, as the title describes, is something of a survey course in modern database technology, looking at a number of different systems and comparing the advantages of each.

Programming Languages

Next comes the most important choice, but also the one that probably causes the greatest number of arguments among developers: the server-side programming language. Longtime readers of this column won't be surprised to hear that I think Python and Ruby are likely to be your best candidates, both because of the languages themselves, and because the Flask and Sinatra frameworks are simple, powerful and well documented, with large communities. You can create great Web applications with these frameworks—or, if you find that you're cobbling together too many separate packages, you can move up to Django and Rails, frameworks that undoubtedly can handle whatever you throw at them.

Although I personally prefer Ruby for Web development, there's no doubt that Python is not only capable, but that it's also easier for newcomers to programming to understand. (I say this after having taught two to three Python courses every month for the last five years.)

An increasingly popular option is JavaScript, often using node.js, on the assumption that it's a good idea to use the same language on both the client (that is, in the browser) and on the server. I haven't yet found node.js to be a compelling framework, partly because of its use of the "reactor" pattern, and partly because I'm used to the comforts of other, more mature frameworks. But given the huge number of people working in JavaScript nowadays, and the understandable interest in reducing the number of languages needed to engage in Web programming, I would expect JavaScript to become an increasingly viable option for Web development in general and for newcomers to the field in particular.

There are other options, as well, such as PHP, which is easy to learn and has a huge community. However, PHP has never impressed or excited me as a language. That said, the fact that WordPress now powers about 10% of the Web indicates that PHP is not going away any time in the near future. Moreover, there are many opportunities to extend, expand and work within the WordPress ecosystem. One friend of mine has a very successful business as a WordPress e-commerce consultant. Given the size of that ecosystem, I expect that he will have clients and work for about as long as he would like.

Depending on your development background, experience and taste, you might want to try something like Go, Scala or Clojure, all of which can be used for Web applications and have active, strong communities that continue to grow.

The bottom line is, if you already know a programming language, you should try to leverage that knowledge and use it to dip your toes into Web development. If you don't already feel comfortable with a language, I suggest trying Python or Ruby. They're easy to learn, have large communities and ecosystems, and have multiple frameworks that can cater to needs large and small. Of course, you may decide to go in a different direction—and if you do, the odds are that you'll be just fine. Learning to be a Web developer is generally more difficult than learning to develop Web applications in a new language or with a new framework.

Front End

The whole idea of the "front end" in Web development has been increasingly critical in the last decade. Whereas it was once enough to know about HTML, and then CSS, the combination of HTML5, CSS, JavaScript and numerous APIs has made front-end development a complex but vital part of any Web application stack.

That said, don't feel that you need to understand everything at once when working on the front end. You will need to learn some HTML, as well as some basic CSS. And the odds are good that you will need to learn some JavaScript as well, although just how much you will learn depends on how much client-side work you will need to do and how sophisticated it will need to be.

Perhaps the biggest boon to front-end development in the last few years has been the creation of front-end frameworks, such as Zurb Foundation and Twitter Bootstrap. These frameworks make it possible to focus on the design you want, taking advantage of the fact that others have had to struggle with the same issues before you. I have used Bootstrap on a number of projects and have found that it saves me time, while allowing the design to look far better than otherwise would have been the case. Bootstrap and Foundation don't mean that you don't have to learn CSS, but they dramatically reduce the amount of CSS you need in order to get a site up and running.

JavaScript is also enjoying a surge in popularity, partly because Web applications are becoming more sophisticated, and partly because everyone has a JavaScript development environment—namely a Web browser. The number of modules and packages for JavaScript is extraordinarily large and shows no signs of stopping. The growth of Ember.js and Angular.js is encouraging and interesting. At the same time, I think these front-end frameworks are too complex for beginning programmers. jQuery, although somewhat old-fashioned, is still a great JavaScript library that makes it easier to manipulate the DOM in a cross-platform way.

JavaScript is a vital skill every Web developer should have. The book *Eloquent JavaScript*, by Marijn Haverbeke and at **http://eloquentjavascript.net**, is a great introduction to the JavaScript language, although it focuses on the language, rather than its use in Web development.

So, Where Should You Start?

The good news is that Web development is so diverse, it's hard to make a bad or wrong choice. Almost no matter what you choose, there is a community using those same technologies.

If you're a complete newbie, I'd suggest trying Python, along with the Flask framework, the PostgreSQL database and the Bootstrap front-end framework along with jQuery.

If you feel very comfortable with object-oriented programming, or feel like you want a bit more flexibility in your code style, you might take a look at Ruby with the Sinatra framework—and also PostgreSQL, Bootstrap and jQuery.

Finally, WordPress development is a good way to get your feet wet in its own ecosystem. I'm not sure if I'd say that WordPress is the paradigm of beautiful software development, but it's hard to argue with the market share and plugin ecosystem that has grown up around this very successful, very user-friendly system.

Conclusion

It sometimes seems as if all software development is becoming Web development. That's obviously an exaggeration, but the demand for Web developers, and the number of interesting projects for such developers, continues to surge.

It doesn't matter very much what language or technology you learn; the important thing is to learn the Web mindset and try to figure out what niche of this growing industry you want to be a part of or benefit from.■

Reuven M. Lerner is a Web developer, consultant and trainer. He recently completed his PhD in Learning Sciences from Northwestern University. You can read his blog, Twitter feed and newsletter at http://lerner.co.il. Reuven lives with his wife and three children in Modi'in, Israel.

Send comments or feedback via http://www.linuxjournal.com/contact or to ljeditor@linuxjournal.com.

Resources

The Python language home page is at **http://python.org**. The site includes not only software downloads, but also extensive documentation and tutorials.

The Flask home page is at **http://flask.pocoo.org**. This includes links to the code, documentation, tutorials, examples and an official list of Flask extensions. The Django framework is at **https://www.djangoproject.com**, also with extensive documentation and code samples.

The home page for PostgreSQL is at **http://postgresql.org**, and the home page for jQuery is at **http://jquery.org**.

If you're interested in the Ruby programming language, go to **http://ruby-lang.org**. The Sinatra framework is at **http://www.sinatrarb.com**. The Ruby on Rails framework is at **http://rubyonrails.org**, including code, tutorials and resources.

```
/code business "success" do
  needs :speed
  needs :scale
  needs :consistency
  provider 'getchef.com'
end
```

getchef.com

CHEF™
CODE CAN

DAVE TAYLOR

Days Between Dates: the Counting

Dave continues developing a script to calculate how many days ago a specific date was—and finds that there are bugs!

In my last article, we began an exploration of date math by validating a given date specified by the user, then explored how GNU date offers some slick math capabilities, but has some inherent limitations, the most notable of which is that it isn't on 100% of all Linux and UNIX systems.

So here, let's continue the development by adding the date math itself. For this script, the goal is to answer the question of how many days have transpired between the specified date and the current date. The necessary algorithm has obvious applications in the broader "date A to date B" puzzle, but let's defer that for another time.

Remember also that we're talking about dates, not time, so it's not counting 24-hour increments but simply the number of days. So running the completed script at 11:59pm will offer a different result from at 12:01am, just a few seconds later.

To start, let's grab the current month, day and year. There's a neat `eval` trick we can use to do this:

```
eval $(date "+thismon=%m;thisday=%d;thisyear=%Y;dayofyear=%j")
```

This instantiates `thismon` to the current month, `thisday` to the current day of the month, `thisyear` to the current year and `dayofyear` to the numeric day number into the year of the current day, all in a single line—neat.

The user-specified starting date already is broken down into day of month, month of year and year, so once the current month, day and year are identified, there are four parts to the equation:

1. How many years have transpired * 365.
2. How many days were left in the starting year.
3. How many days have transpired since the beginning of this year.
4. Compensate for leap years in the interim period.

The first test demonstrates the nuances in this calculation because if we're calculating the number of days since, say, Nov 15, 1996, and today, Jun 3, 2014, we don't want to count 365*(2014–1996) because both the start and end years will be wrong. In fact, better math is to use this formula:

365 * (thisyear – starting year – 2)

But, that's not right either. What happens if the starting date is Jun 1, 2014, and the end date is Jun 3, 2014? That should then produce a zero value, as it also would if the start date was Mar 1, 2013, even though 2014–2013=1. Here's my first stab at this chunk of code:

```
if [ $(( $thisyear - $startyear )) -gt 2 ] ; then
  elapsed=$(( $thisyear - $startyear - 2 ))
  basedays=$(( elapsed * 365 ))
else
  basedays=0
fi
echo "$basedays days transpired between end of $startyear \
     and beginning of this year"
```

Now that isn't taking into account leap years, is it? So instead, let's try doing it differently to tap into the `isleap` function too:

```
if [ $(( $thisyear - $startyear )) -gt 2 ] ; then
  # loop from year to year, counting years and adding +1
  # for leaps as needed
  theyear=$startyear
  while [ $theyear -ne $thisyear ] ; do
    isleap $theyear
    if [ -n "$leapyear" ] ; then
      elapsed=$(( $elapsed + 1 ))
      echo "(adding 1 day to account for $theyear being a leap)"
    fi
    elapsed=$(( $elapsed + 365 ))
    theyear=$(( $theyear + 1 ))
  done
fi
```

Fast and easy. When I run this block against 1993 as a starting year, it informs me:

```
(adding 1 day to account for 1996 being a leap year)
(adding 1 day to account for 2000 being a leap year)
(adding 1 day to account for 2004 being a leap year)
(adding 1 day to account for 2008 being a leap year)
(adding 1 day to account for 2012 being a leap year)
7670 days transpired between end of 1993 and beginning of this year
```

For the second step in the algorithm, calculating the number of days from the specified starting date to the end of that particular year, well, that too has an edge case of it being the current year.

For the second step in the algorithm, calculating the number of days from the specified starting date to the end of that particular year, well, that too has an edge case of it being the current year. If not, it's a calculation that can be done by summing up the days of each of the previous months plus the number of days into the month of the starting date, then subtracting that from the total days in that particular year (since we have to factor in leap years, which means we have to consider whether the date occurs before or after Feb 29), or we can do the same basic equation, but sum up the days after the specified date. In the latter case, we can be smart about the leap day by tracking whether February is included.

The basic code assumes we have an array of days-per-month for each of the 12 months or some other way to calculate that value. In fact, the original script included this snippet:

```
case $mon in
  1|3|5|7|8|10|12 ) dim=31 ;; # most common value
  4|6|9|11        ) dim=30 ;;
  2               ) dim=29 ;;  # is it a leap year?
  *               ) dim=-1 ;;  # unknown month
esac
```

A logical approach would be to turn this into a short function that can do double duty. That's easily done by wrapping it in `function daysInMonth {` and `}`.

With that, it's a matter of stepping through the remaining months, although be alert for the leap year calculation we need to do if month = 2 (Feb). The program has February always having 29 days, so if it isn't a leap year, we need to subtract one day to compensate:

```
if [ $thisyear -ne $startyear ] ; then
  monthsleft=$(( $startmon + 1 ))
  daysleftinyear=0
  while [ $monthsleft -le 12 ] ; do
    if [ $monthsleft -eq 2 ] ; then  # February. leapyear?
      isleap $startyear
      if [ -n "$leapyear" ] ; then
        daysleftinyear=$(( $daysleftinyear + 1 )) # feb 29!
      fi
    fi
    daysInMonth $monthsleft
    daysleftinyear=$(( $daysleftinyear + $dim ))
    monthsleft=$(( $monthsleft + 1 ))
  done
else
  daysleftinyear=0    # same year so no calculation needed
fi
```

The last part is to calculate how many days are left in the starting date's month, again worrying about those pesky leap years. This is only necessary if the start year is different from the current year and the start month is different from the current month. In the case that we're in the same year, as you'll see, we can use "day of year" and calculate things differently.

Here's the block of code:

```
if [ $startyear -ne $thisyear -a $startmon -ne $thismon ] ; then
  daysInMonth $startmon
  if [ $startmon -eq 2 ] ; then   # edge case: February
    isleap $startyear
    if [ -z "$leapyear" ] ; then
      dim=$(( $dim - 1 ))  # dim = days in month
    fi
  fi
  daysleftinmon=$(( $dim - $startday ))
  echo "calculated $daysleftinmon days left in the startmon"
fi
```

Powerful: Rhino

Rhino M4800/M6800
- Dell Precision M6800 w/ Core i7 Quad (8 core)
- 15.6"-17.3" QHD+ LED w/ X@3200x1800
- NVidia Quadro K5100M
- 750 GB - 1 TB hard drive
- Up to 32 GB RAM (1866 MHz)
- DVD±RW or Blu-ray
- 802.11a/b/g/n
- Starts at $1375
- E6230, E6330, E6440, E6540 also available

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

Tablet: Raven

Raven X240
- ThinkPad X240 by Lenovo
- 12.5" FHD LED w/ X@1920x1080
- 2.6-2.9 GHz Core i7
- Up to 16 GB RAM
- 180-256 GB SSD
- Starts at $1910
- W540, T440, T540 also available

Rugged: Tarantula

Tarantula CF-31
- Panasonic Toughbook CF-31
- Fully rugged MIL-SPEC-810G tested: drops, dust, moisture & more
- 13.1" XGA TouchScreen
- 2.4-2.8 GHz Core i5
- Up to 16 GB RAM
- 320-750 GB hard drive / 512 GB SSD
- CF-19, CF-52, CF-H2, FZ-G1 available

EmperorLinux
...where Linux & laptops converge

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

Model specifications and availability may vary.

Hmm...365*10 = 3650. Add a few days for the leap year, and that seems wrong, doesn't it?

We have a few useful variables that now need to be added to the "elapsed" variable: `daysleftinyear` is how many days were left in the start year, and `dayofyear` is the current day number in the current year (June 3, for example, is day 154). For clarity, I add it like this:

```
echo calculated $daysleftinyear days left in the specified year
elapsed=$(( $elapsed + daysleftinyear ))
# and, finally, the number of days into the current year
elapsed=$(( $elapsed + $dayofyear ))
echo "Calculated that $startmon/$startday/$startyear \
   was $elapsed days ago."
```

With that, let's test the script with a few different inputs:

```
$ sh daysago.sh 8 3 1980
The date you specified -- 8-3-1980 -- is valid. Continuing...
12419 days transpired between end of 1980 and beginning of this year
calculated 28 days left in the startmon
calculated 122 days left in the specified year
Calculated that 8/3/1980 was 12695 days ago.
$ sh daysago.sh 6 3 2004
The date you specified -- 6-3-2004 -- is valid. Continuing...
3653 days transpired between end of 2004 and beginning of this year
calculated 184 days left in the specified year
Calculated that 6/3/2004 was 3991 days ago.
```

Hmm...365*10 = 3650. Add a few days for the leap year, and that seems wrong, doesn't it? Like it's one year too many or something? Worse, look what happens if I go back exactly two years ago:

```
$ sh daysago.sh 6 3 2012
The date you specified -- 6-3-2012 -- is valid. Continuing...
0 days transpired between end of 2012 and beginning of this year
calculated 184 days left in the specified year
Calculated that 6/3/2012 was 338 days ago.
```

Something is definitely wrong. That should be 2*365. But it's not. Bah. Phooey. In my next article, we'll dig in and try to figure out what's wrong! ■

Dave Taylor has been hacking shell scripts for more than 30 years. Really. He's the author of the popular *Wicked Cool Shell Scripts* and can be found on Twitter as @DaveTaylor and more generally at his tech site http://www.AskDaveTaylor.com.

Send comments or feedback via http://www.linuxjournal.com/contact or to ljeditor@linuxjournal.com.

The Best SharePoint Training in the World returns to Boston!

Choose from more than 80 classes and tutorials!

"I really enjoyed it. I can hardly wait to get back to work and start using what I learned. I will encourage employees and co-workers to attend future SPTechCons. The conference had great speakers with relevant subjects, and the whole thing was well organized."
—Greg Long, Infrastructure Development Manager, ITG, Inc.

"I prefer SPTechCon over Microsoft's SharePoint Conference in Vegas. I'm definitely going to tell others to go."
—Ray Ranson, Senior Architect, RSUI

SPTechCon
The SharePoint Technology Conference

September 16-19, 2014
The Boston Park Plaza Hotel & Towers

Bolster your career by becoming a SharePoint Master!

- Learn from SharePoint experts, including dozens of SharePoint MVPs and Certified SharePoint Professionals
- Master document management
- Study SharePoint governance
- Find out about SharePoint 2013
- Learn how to create applications for SharePoint that solve real business problems
- Exchange SharePoint tips and tricks with colleagues
- Test-drive SharePoint solutions in the Exhibit Hall

If you or your team needs Microsoft SharePoint training, come to SPTechCon Boston!

Register Early and SAVE! → www.sptechcon.com

A BZ Media Event @SPTechCon Central Contractor Registration

KYLE RANKIN

Not So Dynamic Updates

Kyle covers some of the headaches and gotchas when deploying servers to cloud environments that hand out completely random IPs.

Typically when a network is under my control, I like my servers to have static IPs. Whether the IPs are truly static (hard-coded into network configuration files on the host) or whether I configure a DHCP server to make static assignments, it's far more convenient when you know a server always will have the same IP. Unfortunately, in the default Amazon EC2 environment, you don't have any say over your IP address. When you spawn a server in EC2, Amazon's DHCP server hands out a somewhat random IP address. The server will maintain that IP address even through reboots as long as the host isn't shut down. If you halt the machine, the next time it comes up, it likely will be on a different piece of hardware and will have a new IP.

dhclient Overrides

To deal with this unpredictable IP address situation, through the years, I've leaned heavily on dynamic DNS updates within my EC2 environments. When a host starts for the first time and gets configured, or any time the IP changes, the host will update internal DNS servers with the new IP. Generally this approach has worked well for me, but it has one complication. If I controlled the DHCP server, I would configure it with the IP addresses of my DNS servers. Since Amazon controls DHCP, I have to configure my hosts to override the DNS servers they get from DHCP with mine. I use the ISC DHCP client, so that means adding three lines to the /etc/dhcp/dhclient.conf file on a Debian-based system:

```
supersede domain-name "example.com";
supersede domain-search "dev.example.com", "example.com";
supersede domain-name-servers 10.34.56.78, 10.34.56.79;
```

With those options, once the

network has been restarted (or the machine reboots), these settings will end up in my /etc/resolv.conf:

```
domain example.com
search dev.example.com. example.com
nameserver 10.34.56.78
nameserver 10.34.56.79
```

I've even gone so far as to add a bash script under /etc/dhcp/dhclient-exit-hooks.d/ that fires off after I get a new lease. For fault tolerance, I have multiple puppetmasters, and if you were to perform a DNS query for the puppet hostname, you would get back multiple IPs. These exit hook scripts perform a DNS query to try to identify the puppetmaster that is closest to it and adds a little-known setting to resolv.conf called sortlist. The sortlist setting tells your resolver that in the case when a query returns multiple IPs to favor the specific IP or subnets in this line. So for instance, if the puppetmaster I want to use has an IP of 10.72.52.100, I would add the following line to my resolv.conf:

```
sortlist 10.72.52.100/255.255.255.255
```

The next time I query the hostname that returns multiple A records, it always will favor this IP first even though it returns multiple IPs. If you use ping, you can test this and see that it always pings the host you specify in sortlist, even if a dig or nslookup returns multiple IPs in random order. In the event that the first host goes down, if your client has proper support for multiple A records, it will fail over to the next host in the list.

dhclient Is Not So Dynamic

This method of wrangling a bit of order into such a dynamic environment as EC2 has worked well for me, overall. That said, it isn't without a few complications. The main challenge with a system like this is that the IPs of my DNS servers themselves might change. No problem, you might say. Since I control my dhclient.conf with a configuration management system, I can just push out the new dhclient.conf. The only problem with this approach is that dhclient does not offer any way that I have been able to find to reload the dhclient.conf configuration file without restarting dhclient itself (which means bouncing the network). See, if you controlled the DHCP server, you could update the DHCP server's DNS settings, and it would push out to clients when they ask for their next lease. In my case, a DNS server IP change meant generating a network blip throughout the entire environment.

I discovered this requirement the hard way. I had respawned a DNS server and pushed out the new IP to the dhclient.conf on all of my servers. As a belt-and-suspenders approach, I also made sure that the /etc/resolv.conf file was updated by my configuration management system to show the new IP. The change pushed out and everything looked great, so I shut down the DNS server. Shortly after that, disaster struck.

I started noticing that a host would have internal health checks time out; the host became sluggish and unresponsive, and long after my resolv.conf change should have made it to the host, it seemed it was updating the file again. When I examined the resolv.conf on faulty systems, I noticed it had the old IP scheme configured even though the DNS servers with that information were long gone. What I eventually realized was that even though I updated dhclient.conf, the dhclient script itself never grabbed those changes, so after a few hours when it renewed its lease, it overwrote resolv.conf with the old DNS IPs it had configured!

The Blip Heard Round the World

I realized that basically every host in this environment was going to renew its lease within the next few hours, so the network needed to be bounced on every single host to accept the new dhclient.conf. My team scrambled to stage the change on groups of servers at a time. The real problem was less that dhclient changes required a network blip, but more that the network blip was more like an outage that lasted a few seconds. We have database clusters that don't take kindly to the network being removed. At least, they view it (rightly so) as a failure on the host and immediately trigger failover and recovery of the database cluster. The hosts seemed to be taking way longer than they should to bounce their network, were triggering cluster failovers, and in some cases, required some manual intervention to fix things.

Fortunately for us, this issue affected only the development environment, but we needed to respawn DNS servers in production as well and definitely couldn't handle that kind of disruption there. I started researching the problem and after confirming that there was no way to update dhclient.conf without bouncing the network, I turned to why it took so long to restart the network. My dhclient-exit-hook script was the smoking gun. Inside the script, I slept for five seconds to make sure the network was up and then performed a dig request. This meant that when restarting the network, the DNS queries configured for the old IPs would time out and cause the host to pause before the network was up. The fix was for me to replace the sleep and dig query with a template containing a simple echo to append my sortlist entry to resolv.conf. My configuration management system would do the DNS query itself and update the template. With the new, faster script in place, I saw that my network restarts barely caused a network blip at all. Once I deployed that new exit hook, I was able to bounce the network on any host without any ill effects. The final proof was when I pushed changes to DNS server IPs in production with no issues.■

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

Send comments or feedback via http://www.linuxjournal.com/contact or to ljeditor@linuxjournal.com.

SHAWN POWERS

First Health, Now Recreation

What's more fun than working with Linux? Playing with Linux!

In my last article, I talked about the various techie ways I personally manage the health and exercise issues in my life. This time, I want to talk about nerdy entertainment, and how technology influences the fun parts of my day-to-day existence. Not that health and exercise aren't fun—but, you know what I mean. Let's consider these two articles a "summer vacation" from the server room and command line. For those of you in the southern hemisphere, it can be an extended winter break. For my next article, I'll put my virtual work clothes on and teach some Linux, but here it's about making technology serve our entertainment whims.

Games

Let's face it, sometimes it rains and you can't go outside. Sometimes it's too sunny, and if you go outside you'll burn. And sometimes, you're an insomniac, up all night with nothing to comfort you but the pale blue glow of a monitor. Whatever excuse you need to tell yourself, gaming on Linux is better than it's ever been (Figure 1).

Steam: Few things have helped the Linux gaming scene as much as the folks at Valve providing a native Linux version of the Steam platform. No, not all games are available for Linux, but there is a surprisingly large number that is. Whether you like to shoot zombies with the *Left-4-Dead* franchise or prefer to play Indie games, which often have incredible gameplay while sacrificing big-budget special effects, Steam

Few things have helped the Linux gaming scene as much as the folks at Valve providing a native Linux version of the Steam platform.

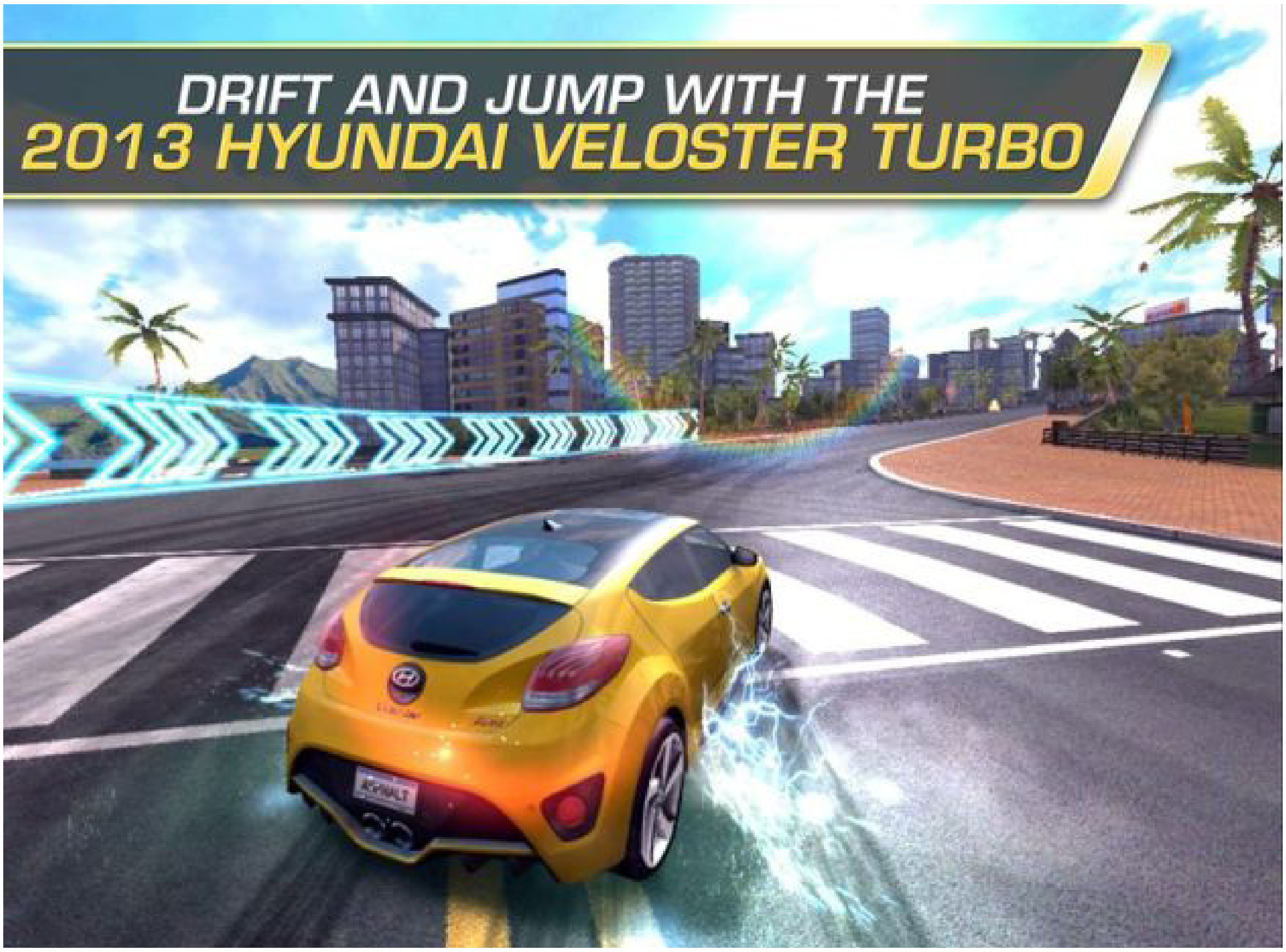


Figure 1. *GT Racing* has awesome graphics, incredible gameplay and is fairly easy to play.

is a resource for tons of awesome games. Download the client and start building your library now: **http://www.steampowered.com**. (If you don't like Steam, check out Desura, which is a similar concept: **http://www.desura.com**.)

Android: Just saying "Android" isn't really specific enough anymore. Google's mobile operating system is getting embedded into more and more products every day. In this particular case, however, I'm referring to phones and tablets. Although there certainly are epic games available for Android, I find the quick and challenging games are the most fun. While I personally never understood the draw of games like *Flappy Bird*, I will admit to hours <strike>lost</strike> invested into *Angry Birds* and *Candy Crush* (Figure 2). In fact, Android is becoming such

Figure 2. Perhaps the birds are angry that we keep forcing them to hurl themselves at pigs.

a gaming platform, many games are being cross-compiled for PCs and Android (**https://play.google.com/store/apps/category/GAME**).

Ouya, Fire TV and Roku: Remember plugging your Atari 2600 in to the black-and-white television and playing *Pac-Man* with a single-button joystick? No? Fine. Do you remember plugging your Nintendo in to the living-room TV and playing *Legend of Zelda* with a boxy controller? Well, even if your earliest console gaming memories are of *Wii Bowling*, the latest Android-powered gaming devices should be reminiscent of those old days, with a modern twist. Gaming and streaming entertainment are starting to merge together as the platforms mature, and devices like the Roku are proof. Whether you want to stream old

You even can make individual videos available off-line for those rainy camping trips out of cell tower range.

episodes of *Star Trek* or fling angry birds at naughty pigs, these new console devices are easy to use and fun to play with (**http://www.ouya.com**, **http://amazon.com/firetv** and **http://www.roku.com**).

Humble Bundle: I've mentioned the Humble Bundle folks before. It's a great organization that sells Indie products and raises money for charities. As it happens, it also supports Linux and Android platforms with most of its game releases. Be sure to read closely, however, because although the Humble Bundle folks try to find developers who support cross-platform gaming, sometimes the "bundles" are Windows-only. Because there's typically a new bundle every few weeks, it's not usually a big deal, however. Just be careful before you click.

Video

I already used the rain excuse, and the insomnia excuse is just depressing, but whatever your reason for being inside, watching the television is a time-honored way to kill the time. Healthy? Probably not, but that was last article's topic; here it's all about fun. And personally, I love to watch the TV!

Plex: I've written in length about Plex and Plexmediaserver in past issues, so I won't belabor the point much more. It's impossible to talk about Linux-based video-watching without bringing up one of my favorite consumption methods though, so here it is. Plex is awesome. You can watch your personal video collection on a plane, in a car, in a box, with a fox—anywhere you happen to be, your video collection is available to you. You even can make individual videos available off-line for those rainy camping trips out of cell tower range. My only remaining frustration with Plex? It doesn't support playlists. Thankfully, that's in the works for future versions (**http://www.plex.tv**). For playback on your own network, it's hard to beat the XBMC interface, which is what I still use on our TVs (**http://www.xbmc.org**).

Roku and Fire TV: I know, I

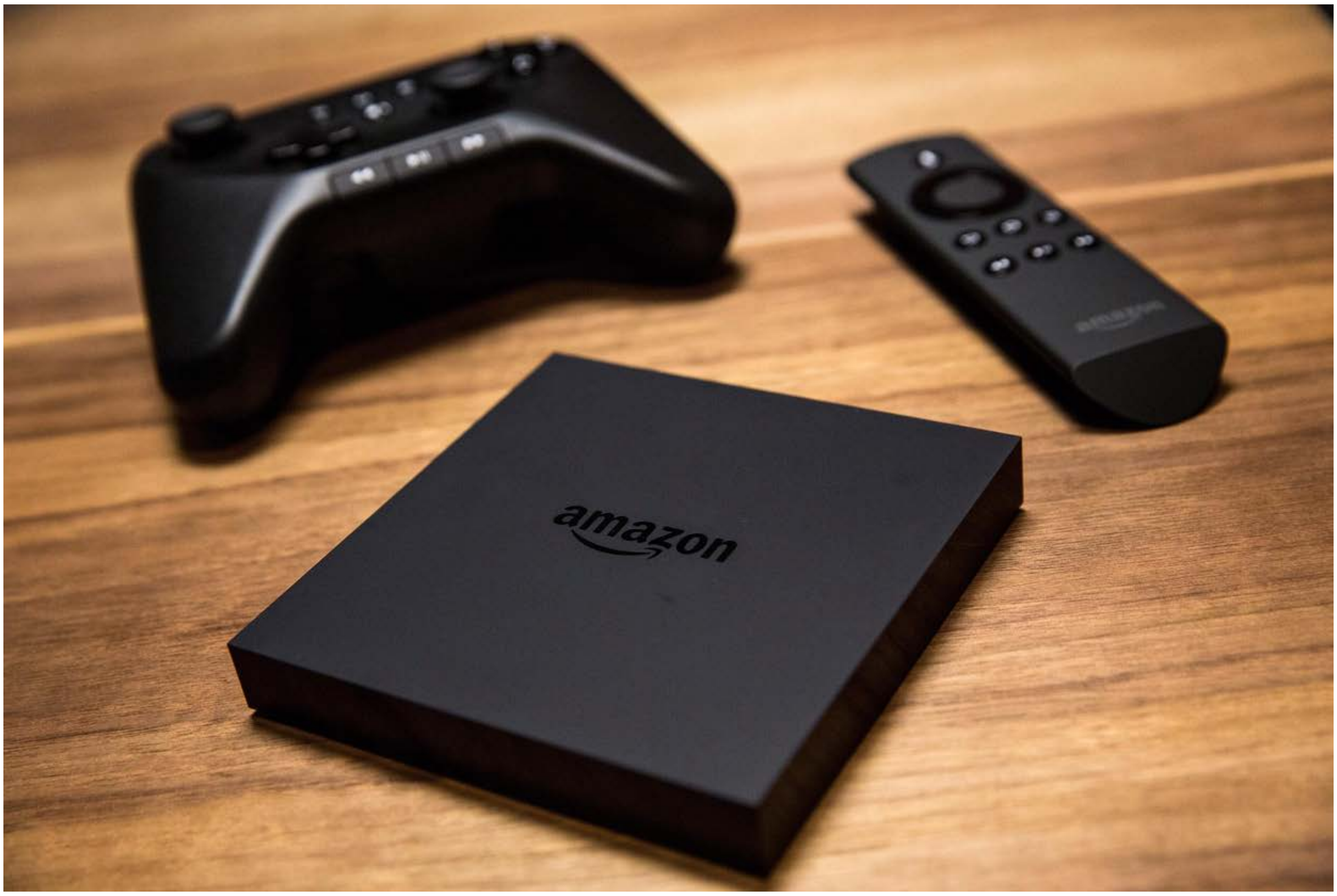


Figure 3. The Amazon Fire TV is a media streamer and a low-end gaming system. The interface is sleek, and if you're an Amazon user, it will be convenient.

mentioned these in the gaming section, but gaming is really a secondary function. These are both Android-based streaming devices. The Roku has a huge market share, which is interesting because it doesn't actually produce or stream content of its own. I find that refreshing, because it means the platform doesn't favor one video source over another. The Amazon Fire TV, on the other hand, does favor Amazon products. This isn't shocking, but it does admittedly allow for a number of third-party apps. Both devices support Netflix and Hulu, and have a native Plex app! If you want Plex on a television, it's hard to beat a Roku or FireTV (Figure 3).

Backyard Drive-in: It might sound cheesy, but if you're a little adventurous, an outdoor movie screening is incredibly fun. Make sure there are blankets for the ground, or some chairs, and provide bug spray along with the popcorn. I actually recommend using a yard fogger to

Figure 4. Outdoor theaters are just cool! (Photo courtesy of kelemvor: http://backyardtheater.com/forums/index.php?action=profile;u=181.)

keep the bugs at bay. The tech side is fairly simple. Find a light-colored building or hang up a white bed sheet, and use a projector to create your outdoor "theater". It helps to have some good speakers, and tons of popcorn. I find 1980s action movies or cheesy B-rate horror films to be the best choices, but your tastes will likely vary. The fun part for the kids is waiting until after dark to *start* the movie. They almost never last through the whole movie, but it's fun for the whole family (Figure 4).

Music

Streaming Music: We've gotten to the point with music that finding ways to *get* songs is super simple. Tools like Google Music's matching program (or iTunes "Match" subscription service) mean our music is always available to us from the cloud. Oddly, I don't own very much music, because I'm fickle and switch tastes often. For me, the streaming services like Pandora are ideal, because based on my likes and dislikes, they pick songs I'll probably enjoy. And, they're usually correct. Like I mentioned, there are tons of platforms for music. Rhapsody, Spotify, Last.FM,

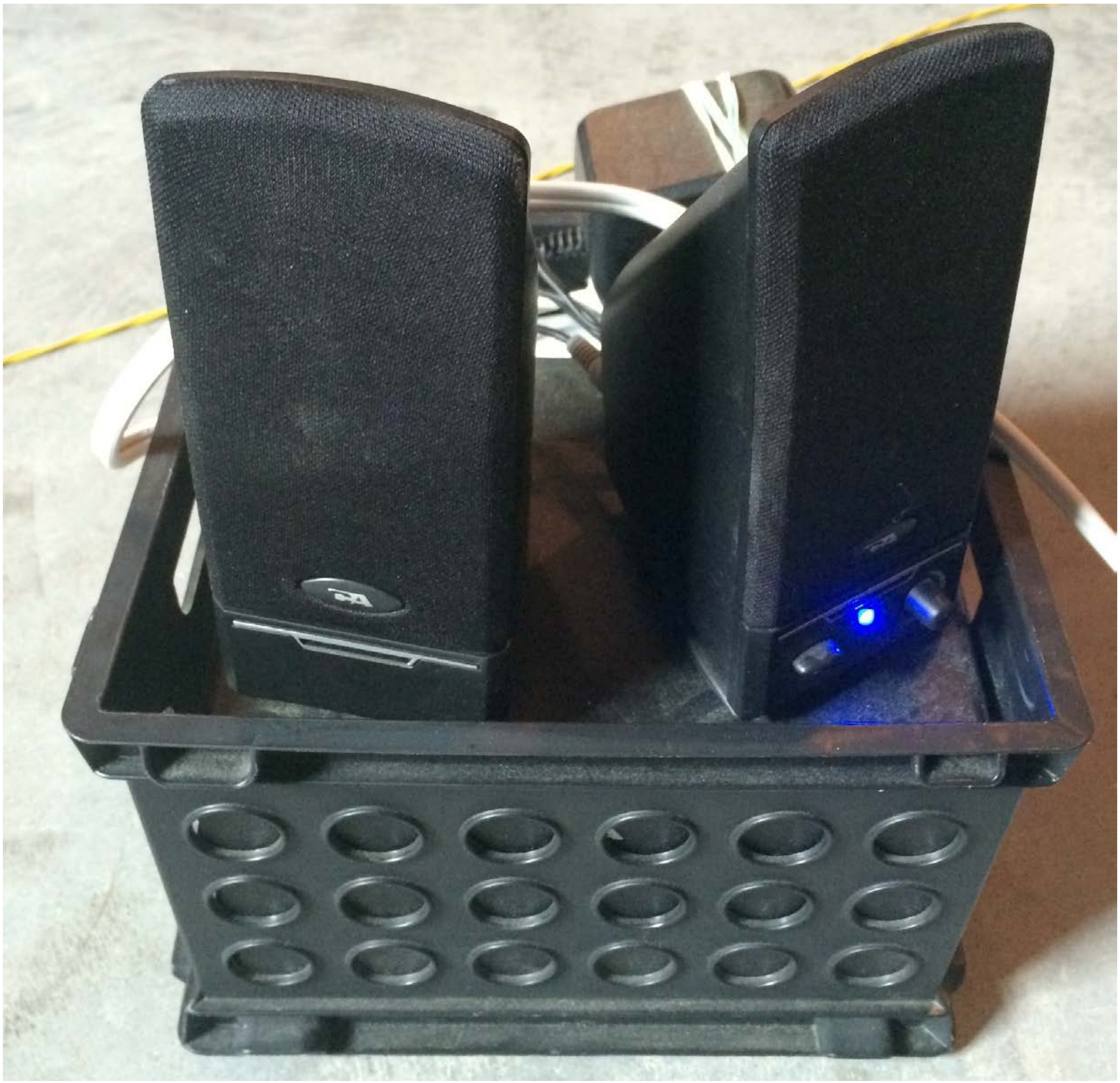


Figure 5. My first mobile speaker attempt is a bit sad. Still, it works in a pinch. The next version will be *much* nicer!

Pandora, Google Music—whatever style of consumption appeals to your particular taste is available.

Music Hardware: I know, you're thinking, "Shawn, 'music hardware' is called 'speakers'", but I mean more than just speakers. For instance, I like to listen to audiobooks in the shower. It means that a good book will sometimes result in running out of hot water, but I still enjoy listening to them nevertheless. I tried a wide variety of Bluetooth-enabled shower speakers, and quite frankly, they all stink. I'm hoping someone has found one that is loud enough for old deaf people like me, but so far, I have a drawer full of tinny-sounding Bluetooth devices with suction cups. If you know of a good Bluetooth speaker designed for the shower, please let me know! I'll be sure to write about it in a future issue and give you praise and credit!

Something I've been working on more recently is a portable speaker system for the backyard. I could open the window and jam a speaker into it, but I really want to control the music from my phone. My current solution is to cram some powered speakers into a laundry basket, and connect it to something like the D-Link WiFi Audio Extender so I can stream wirelessly to wherever I set the basket. In fact, the folks at D-Link sent me their newest model (the DCH-M225), and I'm going to try to make a proper weekend project out of building the perfect portable jam center. If it works, I'll write about it in a future issue. If it fails, I'll probably post sad pictures on our Twitter feed (Figure 5).

Books, Mmmm, Books

I love reading. This is probably not a surprise, since I'm a writer, a nerd, socially awkward and get a sunburn from thinking warm thoughts. I've spent a lot of time indoors, and a lot of that time has been in faraway places with aliens and trolls. When I was younger, I had to brave the outdoors long enough to walk the couple miles to the local library. Now, I have an endless supply of alternate realities at the literal tips of my fingers. And the eBook revolution has only just begun!

Audible: I must admit, my hectic lifestyle means that most of my reading is done while I'm driving my kids from one appointment or sporting event to the next. (My kids, unlike their father, are fairly athletic—I don't understand genetics.) Since reading a book while driving is illegal and a poor example for my kids, I tend to listen to books more than read them. Audible is a great way to get the most recent books delivered seamlessly. The Android app will sync your spot with a Kindle eBook if you buy both versions, so you can read at stoplights if you really prefer the written word. I don't suggest that

Figure 6. The Overdrive app is really slick, but is only as useful as the selection your library offers. (Photo courtesy of http://overdrive.com.)

though, as it seems dangerous!

Non-Audible Audiobooks: The only downside of audiobooks is the cost. Mind you, the cost is understandable, because the author, narrator and entire production crew deserve to be paid for their work, but it means that instead of $7 for a paperback, it's $25 for a performance. There are other audiobook resellers, but the selections are never as good as Audible, and the prices aren't much better. Thankfully, there's another option, depending on where you live—the library.

Unfortunately, the availability of audiobooks varies widely from library to library. When I lived in a city, my local library had more audiobooks than I ever could listen to, plus, the librarians almost always would buy a book if I requested they add it to their library! Now that we're in a small town, the local library doesn't have the budget to carry a large selection. Your library might be able to import physical copies of audiobooks for you, but the on-line digital stuff is really the nicest. If you have friends or relatives in a big city who don't use their local library, perhaps you could work out a deal with them. I'll leave the moral implications up to you to sort out.

Generally with a library, you get free access to a number of audiobooks at a time. The Overdrive app is the standard method used to access those books. Libraries still are sorting out DRM issues, and some books are available only on Windows-compatible listening devices. That really annoys me, so I tend to find ways to strip the DRM from the books and listen to them on whatever darn device I want (generally an Android device running the SmartAudiobook Player **http://www.linuxjournal.com/content/android-candy-smart-audiobook-player**). Again, there are moral implications. Nevertheless, the library is a great resource for audiobooks. Sometimes, the selection is amazing. Even if it isn't, contact the librarians—sometimes they have a budget for ordering more books, but they're just waiting for requests. Find the app at **http://omc.overdrive.com**, and ask at your local library about its digital options (Figure 6).

Book Books

I know, for many of you, nothing ever will replace dead tree pulp and ink. I still painfully remember the years of Letters to the Editor sharing everyone's frustration when *Linux Journal* switched to digital. Thankfully, eBooks and their reading devices continue to improve year after year.

Yes, the same folks who provide the awesome selection of games on a regular basis now also provide eBook bundles.

Now, like with music, there are so many ways to get and read eBooks, it can be overwhelming.

The Standard Stores: Amazon (Kindle), Barnes & Noble (Nook), Apple (iDevices), Google Play (Android devices), plus a ton of smaller companies sell eBooks. With most companies, the idea is to lock you in to their ecosystem, so you always will buy from them instead of bargain shopping for the best price. This is both good and bad. Companies like Amazon make the purchasing and reading experience so seamless and pain-free that it's hard to go elsewhere, even if a book might be less expensive from a competitor. You're paying for convenience along with the book itself.

It's certainly possible to install all the major apps on your Android device and read a book on whatever platform you could buy it for the cheapest, but most of us tend to like a consistent interface and don't want our book selections splintered across various cloud accounts. I'm horribly guilty of this, and I tend to buy all my books from one particular vendor. In fact, it wasn't until I started buying eBook bundles from Humble Bundle that I had to rethink my eBook collection.

Humble Bundle: Yes, the same folks who provide the awesome selection of games on a regular basis now also provide eBook bundles. The bundles usually contain books from authors I love, and I really like to support the charities by purchasing books. You pay what you want, and the money gets split among various charities along with the author. (As it happens, Linux users are always the top contributors per capita. Go freedom!)

These DRM-free eBooks are just downloads and aren't part of any cloud ecosystem. That meant I had to come up with some way to consolidate my eBook library in one place. Thankfully, Calibre (**http://www.linuxjournal.com/content/calibre-cloud**) is a great tool for managing an eBook collection. Yes, it means stripping some DRM off my Amazon books, but having all my books in one "library" is like a salve for my OCD.

The Overdrive Library Again: Remember the Overdrive program for listening to audiobooks from the library? Well it also does eBooks and includes an app for reading the library-loaned eBooks you check out. I do think it allows for an unfortunate side effect in that going to the library is no longer necessary for reading books, but that's just nostalgia on my part. I'm thrilled libraries are evolving, and I urge you to check out your local library's digital collection!

Oyster: Finally, I leave you with Oyster. You may have read about it in the UpFront section of this issue, as it won this month's Editors' Choice award. Oyster is basically the Netflix of the book world. With a $9.95 subscription and an app on your phone and/or tablet, you get unlimited access to more than a half-million books! Read about it in the UpFront section, or surf over to http://www.oysterbooks.com for more details and a free 30-day trial.

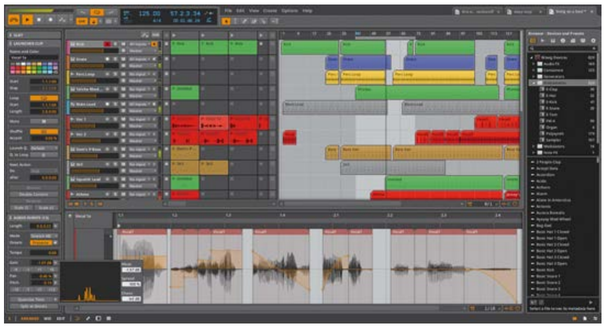
Build Your Own Adventure

Honestly, after all the years of working as a sysadmin, writing for *Linux Journal* and just being a geek, my favorite Linux pastime is learning. I love to build projects like BirdCam (http://birds.brainofshawn.com). I'm excited every time I can solve a household problem with a Raspberry Pi (the digital fireplace in my family room, for instance). And it's all I can do not to quit writing this article so I can go build my outdoor, mobile, wireless speaker system for streaming music in the backyard.

Whatever your favorite Linux adventure might be, hopefully some of the things I discussed here make those adventures more fun. At the very least, I hope you check out your local library to see what sort of digital offerings it provides. It's awesome to see an establishment as old as the library move into the next phase of free knowledge sharing. Libraries understood the idea of open source before the term even meant something. Enjoy the rest of your summer (or winter!) everyone, and get ready for next issue, when we're back in the classroom.■

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.

Send comments or feedback via http://www.linuxjournal.com/contact or to ljeditor@linuxjournal.com.

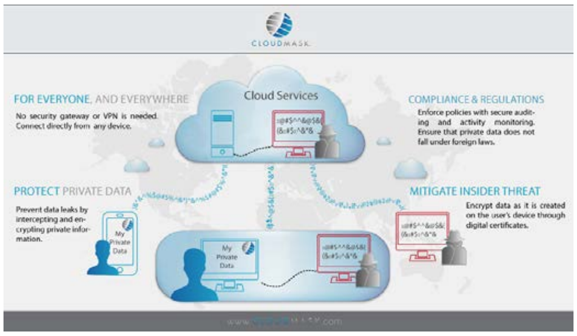


Bitwig Studio

Bitwig recently announced innovations complementary to Bitwig Studio, a dynamic software solution for music creation and performance to realize one's musical ideas at every stage of production. The first innovation is the new Community Controllers section of the application's Web site, which is dedicated to Bitwig Studio's capabilities for deep integration with just about any hardware controller. This integration is possible due to the Bitwig Open Conroller API, an open-source JavaScript platform allowing virtually anyone to create, share and collaborate on powerful controller scripts. A wide range of controllers already are available, ranging from those containing only a few knobs or buttons to elaborate keyboards with motorized faders, colored pads, rotary encoders with LEDs and color displays. A related innovation is the new Community Controllers Library, an archive of the latest controller scripts from the Bitwig Community. Bitwig users already have begun hacking, modifying and creating control scripts from scratch, collaborating with others and sharing their work. The two repositories are destined to open a wide range of new integration possibilities with Bitwig Studio.
http://www.bitwig.com

Glacier Computer Fusion 7

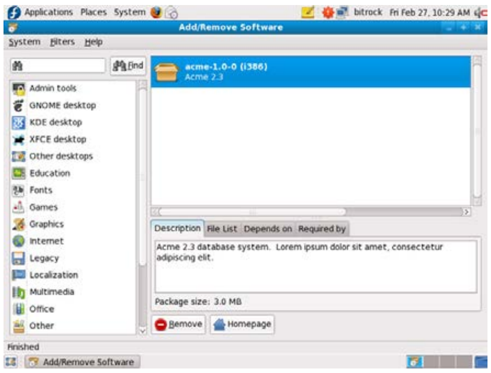
Glacier Computer believes that its new Fusion 7 Rugged Android Tablet is the device that will cause the industrial marketplace to embrace the Android platform and all of its advantages fully. The Fusion 7 extends the Glacier Computer product line of rugged industrial PCs—including rugged tablets, portable hand-held devices and fixed mounted data collection computers—that typically are used in harsh industrial environments where a standard PC cannot survive. Glacier states that the Fusion 7 is compact, light and feature-rich, enabling businesses and their mobile users to add efficiencies throughout the operation while providing a capital asset with an extended useful life and a superior ROI. The Fusion 7 combines the advanced power management and memory management capabilities of Android enabling more than one million applications to run on the typically rugged Glacier device. To ease transition to the new OS, Glacier is collaborating with The Allen Group and SOTI, which will enable broader adoption of the Android OS within the traditional Windows environments of the industrial marketplace.
http://www.glaciercomputer.com


CLOUDMASK
FOR EVERYONE, AND EVERYWHERE
No security gateway or VPN is needed. Connect directly from any device.
Cloud Services
COMPLIANCE & REGULATIONS
Enforce policies with secure auditing and activity monitoring. Ensure that private data does not fall under foreign laws.
PROTECT PRIVATE DATA
Prevent data leaks by intercepting and encrypting private information.
MITIGATE INSIDER THREAT
Encrypt data as it is created on the user's device through digital certificates.
My Private Data
www.cloudmask.com

CloudMask's Secure Data Masking for Governments

Governments have unique needs and security requirements, and any loss of confidentiality in government affairs can have far-reaching consequences. To meet these needs, CloudMask has unveiled a Secure Data Masking for Governments solution. The new offering brings the security, performance and scalability of the enterprise-class SaaS CloudMask solution to fulfill the requirements of governments. CloudMask elaborates that its advanced masking technology "neutralizes the risks of data breaches by enabling users to apply end-to-end (E2E) encryption and tokenization to their data. This occurs at the moment of creation, and data remains protected throughout its lifecycle. This is a very different approach from existing solutions that encrypt data while it is in transit and/or at-storage, while leaving it exposed during server processing." No VPN, encryption gateway or any other hardware encryption devices are required. The CloudMask solution has been evaluated by the Canadian Federal Government for integration with its security infrastructure and currently is going through Common Criteria Certification to meet the security certification required to operate with governments in 26 countries.
http://www.cloudmask.com

BitRock's InstallBuilder

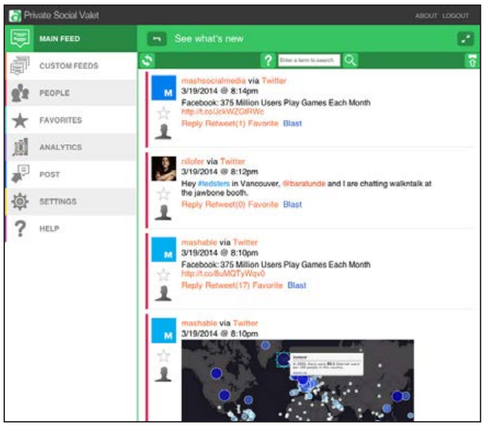


Acme 2.3 database system. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Package size: 3.0 MB

No matter what OS you throw at it, the BitRock's InstallBuilder packaging and deployment platform can help you build a cross-platform installer for it. InstallBuilder, now in version 9, is a development tool for creating cross-platform installers, native packages, DVDs and CD-ROMs. It provides an intuitive graphical development environment, making it easy to learn and use, and an XML-based project format that allows for collaborative development, version control and integration with automated build and test environments. From one project file and build environment, developers can build installers for Linux, Windows, Mac OS X, Solaris, FreeBSD, HP-UX, AIX, IRIX, IBM iSeries and pSeries and other platforms. The latest version enables the creation of cross-platform installers up to 25% faster and 20% smaller than previous versions. HTML license support, a feature previously available only for Qt, is now available for all InstallBuilder users.
http://installbuilder.bitrock.com

Omnibond Systems' acilos



mashsocialmedia via Twitter
3/19/2014 @ 8:14pm
Facebook: 375 Million Users Play Games Each Month
http://t.co/UckWZGtRWc
Reply Retweet(1) Favorite Blast

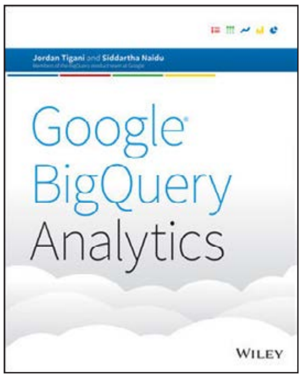
nilofer via Twitter
3/19/2014 @ 8:12pm
Hey #tedsters in Vancouver, @baratunde and I are chatting walkntalk at the jawbone booth.
Reply Retweet(0) Favorite Blast

mashable via Twitter
3/19/2014 @ 8:10pm
Facebook: 375 Million Users Play Games Each Month
http://t.co/8uMQTyWqv0
Reply Retweet(17) Favorite Blast

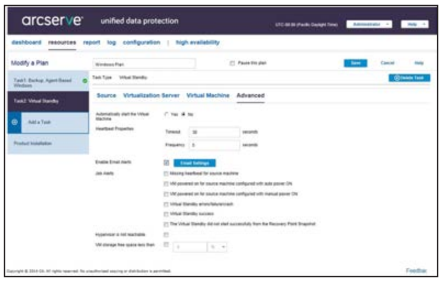
mashable via Twitter
3/19/2014 @ 8:10pm

What is the most logical anagram of "acilos"? It's "social", of course, and social media is what Omnibond Systems' acilos is all about. More precisely, acilos is a so-called personal social-media valet, a personal Web application that helps users aggregate, filter, customize and send updates to social-media feeds from Twitter, Facebook, LinkedIn and Instagram—all in a single, simple interface. The free and open-source acilos grew out of an internal need at Omnibond to deal with the sheer volume of social-media content. The resulting tool was acilos, and due to Omnibond's commitment to the Open Source community, the company decided to share acilos with the rest of the world. Now acilos' source is fully accessible and open for contributors to help develop the next levels of personal social interaction and to add their preferred social-media accounts. acilos is available at GitHub, Amazon Web Services and the application's Web site.
http://www.acilos.com

Jordan Tigani and Siddartha Naidu's *Google BigQuery Analytics* (Wiley)



Do you want to be an expert in Google BigQuery and the BigQuery API? Your tutors to reach that goal are authors Jordan Tigani and Siddartha Naidu, who have penned the new Wiley book title *Google BigQuery Analytics*. The book's contents demonstrate how to use BigQuery effectively, avoid common mistakes and execute sophisticated queries against large data sets. Targeted at business and data analysts who want the latest tips on running complex queries and writing code to communicate with the BigQuery API, the book uses real-world examples to demonstrate current best practices and techniques. Other covered topics include explaination and demonstration of streaming ingestion, transformation via Hadoop in Google Compute engine, AppEngine datastore integration and using GViz with Tableau to generate charts of query results. In addition to the mechanics of BigQuery, the book also covers the architecture of the underlying Dremel query engine, providing a thorough understanding that leads to better query results. A companion Web site includes all code and data sets from the book.
http://www.wiley.com



CA Technologies' CA arcserve Unified Data Protection

The upgraded CA arcserve Unified Data Protection (UDP) is so transformative, says IT solution provider CA Technologies, that it renders "point solutions" for backup and recovery obsolete. The company describes CA arcserve UDP as an easy-to-use solution providing cost-effective backup and "Assured Recovery" across mixed IT environments, helping businesses boost the availability of critical systems. The key selling point for the new version is simplicity, reducing the cost and complexity of deploying multiple point solutions. A new unified architecture and recovery point server drive a full range of highly efficient data protection capabilities for physical and virtual environments through a simple, Web-based user console. CA arcserve UDP provides more than 30 new marquee features to give businesses and service providers precise and flexible functionality to meet stringent service-level and recovery objectives. The solution comes in five editions so that customers buy just the right level of protection, all of which support Linux, UNIX and Microsoft Windows systems.
http://www.arcserve.com

Hazelcast Enterprise

hazelcast

On the same day that Larry Ellison announced Oracle's In-Memory Computing strategy, the company Hazelcast announced Hazelcast Enterprise, the company's own commercial In-Memory Computing solution. Hazelcast provides a radically different vision of In-Memory Computing driven by open-source, commodity hardware and open standards. Hazelcast is a leading open-source In-Memory Data Grid and is available under the Apache software license. Meanwhile, Hazelcast Enterprise extends the open-source Hazelcast core with commercial features and is a commercially licensed software offering. The features in the enterprise offering that extend the Hazelcase open core include .NET and C++ clients, off-heap memory, improved Web sessions, Tomcat 6 and 7 Web sessions clustering, WAN copy, WAN replication to enterprise and improved security. Hazelcast itself also features the "Stabilizer", a sophisticated stress and load-testing system for ensuring the reliability of production-scale grid systems.
http://hazelcast.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.

VAGRANT

Revolutionize your development work flow in a programmatic way using the powers of Vagrant.

RICHARD DELANEY

How many times you have been hit by unit tests failing because of environment differences between you and other team members? How easy is it to build your project and have it ready for development? Vagrant provides a method for creating repeatable development environments across a range of operating systems for solving these problems. It is a thin layer that sits on top of existing technologies that allows people working on a project to reproduce development environments with a single command: `vagrant up`.

Vagrant is open-source software, and it sees lots of contributions from its community in every release. It was first released in 2010. Its founder Mitchell Hashimoto founded the company HashiCorp to work on Vagrant and a number of other DevOps tools in November 2012. HashiCorp has since released a number of other tools that are worth checking out, such as Packer, Consul and Serf. These are for different areas of the development lifecycle.

Vagrant uses virtual machines from various providers. It allows for easy configuration using a Vagrantfile. This allows development teams to set up the virtual machine the same way every time in a programmable fashion. This repeatability encourages developers to tear down and bring up environments more often. The development environment that all team members are developing against now can be "stored" in the version control system in such a way that anyone (new or otherwise) can have an environment set up after issuing a single command.

Continuous integration systems can use the same method. Building an environment and setting up a project in this way allows the developer to be sure that once other environments are provisioned in the same way, the project will run as it did in development. Environment-specific unit test failures is a class of problems completely removed by Vagrant.

Vagrant takes the building and provisioning of your software from tribal knowledge within your team to a reproducible provisioning script that is stored alongside your code. Others can read your Vagrantfile and provisioning scripts and have your project working on their system in minutes. This helps integrate new developers into your team quickly. It is easier to enable cross-team development within a larger company as it's easier to get up and running with a project. It also allows designers or technical writers to get quick access to your project without having to know any technical details about how to set up your project.

Vagrant Technologies

Originally, Vagrant supported only VirtualBox as a provider. A provider is the underlying virtualization technology that Vagrant will interact with. At the time of this writing, Vagrant supports a wide variety of providers, VMware, hyper-v and, most recently, Docker. Vagrant only ships with VirtualBox support by default. Other providers can be added as plugins using Vagrant's plugin system. For the purpose of this article, I use VirtualBox as the provider, as it is the default, and is likely the most accessible to everyone. It is not the most performant provider; VMware in most cases out-performs it, and Docker (although it is not a drop-in replacement) is a very exciting technology in terms of speed and definitely is worth checking out.

The next configurable part of Vagrant is provisioner support. After a virtual machine has been started, Vagrant uses "provisioners" to provision the box in a repeatable way. This is done as part of bringing the virtual machine up in most cases. The simplest type of provisioner is "shell", which runs a defined shell script at boot of the VM as the root user. More advanced provisioners, such as Salt, Chef and Puppet, also are available. In my workplace, we use Puppet to provision our virtual machines so as to better match our development systems with machines running in the production environments.

Depending on the provider, Vagrant also allows for "Synced Folders"; this allows very easy sharing of folders between the host and guest machine. This means every developer on your team can use a different IDE and tooling that runs on the host, and yet develop on the same guest machine. Vagrant makes these things incredibly easy to set up as you will see later in the article.

How to Install Vagrant

Vagrant has installers for all of the main operating systems: Windows, Linux (deb and rpm) and Mac OS X. This portability across host operating systems allows developers in a team to get the same local development experience on any environment.

For this article, I used Ubuntu 12.04 LTS, and I used the deb file from the downloads section of the Vagrant home page. For Linux distributions that don't accept deb or rpm package types, there is still the ability to run older versions of Vagrant, provided that the distribution has support for Ruby and Ruby gems, but this has been discontinued for newer releases of Vagrant.

If you have not already installed VirtualBox, you will need to install it. This can be done through the package manager, although I used the packages provided on the Oracle Web site.

This article uses Vagrant 1.6 and VirtualBox 4.3.

Using Vagrant

In this article, I run through setting up a project using Vagrant, including how to write the Vagrantfile, how to boot the virtual machine using Vagrant and interacting with that VM. The first thing you need is the base box on which you are going to build your project. A base box is a provider-specific bare-bones operating system disk image with some basic things installed. Vagrant uses SSH to connect to all its boxes, so an SSH server must be enabled, and a specific user must have SSH enabled. There are a number of other requirements to making a Vagrant-compatible box. The process is too involved for this article, but more specific instructions can be found on the Vagrant or specific providers' Web sites. Vagrant, by default, expects a vagrant user on the box and the password (by convention) should be "vagrant".

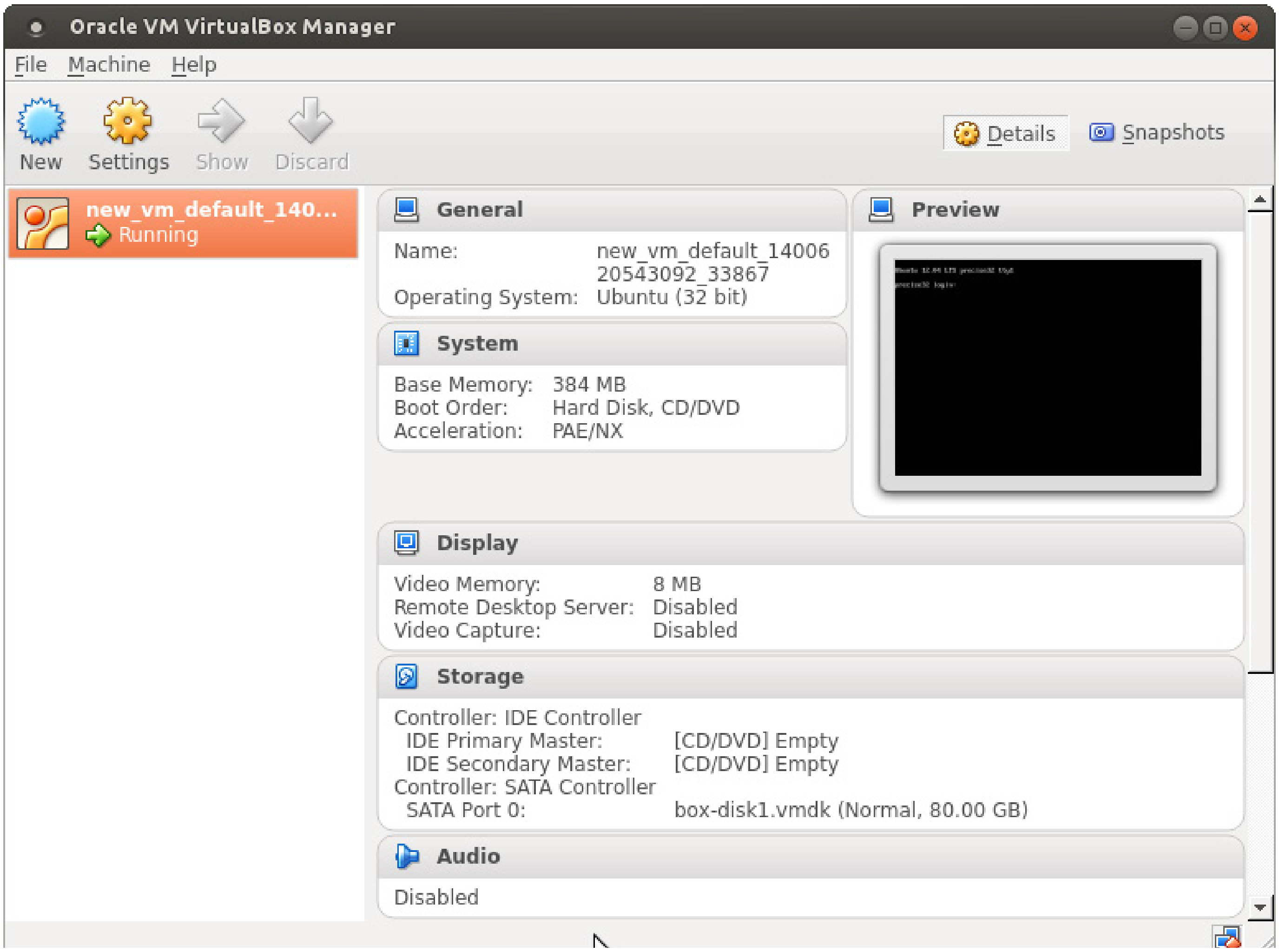
For the purpose of this article, you will use a publicly available base box. HashiCorp provide a base box for Ubuntu 12.04 LTS. To add a box to Vagrant:

```
$ vagrant box add "hashicorp/precise32"
```

This command downloads the box from Vagrant cloud. Vagrant cloud is a service HashiCorp provides, which among other things offers box discovery. Community base boxes are uploaded and can be used. Vagrant knows that you are accessing a base box from the Vagrant cloud and will download it directly from there. Once downloaded, you will not need to download the base box again to use in other Vagrant projects on the same host. You can see available boxes by using the list command:

```
$ vagrant box list
hashicorp/precise32 (virtualbox, 1.0.0)
```

Now that the box has been added, you will craft a basic Vagrantfile. A Vagrantfile is a Ruby file that sets up certain variables for your box. Here's



new_vm_default_140...
Running

General
Name: new_vm_default_14006 20543092_33867
Operating System: Ubuntu (32 bit)

System
Base Memory: 384 MB
Boot Order: Hard Disk, CD/DVD
Acceleration: PAE/NX

Preview

Display
Video Memory: 8 MB
Remote Desktop Server: Disabled
Video Capture: Disabled

Storage
Controller: IDE Controller
IDE Primary Master: [CD/DVD] Empty
IDE Secondary Master: [CD/DVD] Empty
Controller: SATA Controller
SATA Port 0: box-disk1.vmdk (Normal, 80.00 GB)

Audio
Disabled

Figure 1. VirtualBox Screenshot

a basic Vagrantfile:

```
Vagrant.configure("2") do |config|
  config.vm.box = "hashicorp/precise32"
end
```

This shows the power of Vagrant—with this simple file, an Ubuntu 12.04 LTS virtual guest machine can be created and started on any system Vagrant supports. To start the Vagrant VM, simply do the following:

```
$ vagrant up
```

It may take some time for the virtual machine to boot, but when the command returns to the prompt, you successfully have started your virtual machine. At this time, it might be worth opening the VirtualBox application so you can see the new virtual machine that has been created and ensure that it is marked as running.

As I mentioned previously, base boxes for use with Vagrant must be accessible via SSH. This allows you to attach to your new virtual machine simply by doing the following in the same directory as the Vagrantfile:

```
$ vagrant ssh
```

You will be left in a bash prompt in the virtual machine. In two commands, you have booted the VM and `ssh`'d in to the VM. One more thing to note before continuing on to using Vagrant more powerfully is that Vagrant, by default, shares the directory that the Vagrantfile is located in under the /vagrant folder. To demo this capability, while still in the SSH session, do the following:

Guest machine:

```
$  cd /vagrant
$ echo "test" > test
```

Host machine:

```
$ ls
test  Vagrantfile
```

As should be clear, you created a new file "test" on the guest, and the file appears in the same directory on the host. Vagrant uses VirtualBox shared folders to allow you to keep files in sync between the host and the guest very easily. This is especially useful when developing software projects using Vagrant when you want to be able to edit files locally and have those changes automatically appear in your development VM for testing purposes. It allows developers to run their choice of editor while developing on the same platform as anybody else in the team.

Provisioning

You've seen how to boot a simple virtual machine using Vagrant, so next, let's look at how to provision that VM in a repeatable way. As already mentioned, Vagrant supports a wide range of provisioners, but for the sake of this article, bash scripts will be used due to their familiarity. Adding a provisioning step is a matter of adding a single line inside the configure block in the Vagrantfile. The bootstrap also can be an inline script, but for most cases, it makes sense to supply a script file that contains the provisioning steps.

Add the following bootstrap.sh script to the same directory as the Vagrantfile:

```
#!/usr/bin/env bash
# provisioning is run as root by default


# make sure system sources are up to date.
apt-get update
```

```
# install a web server
echo "==== Installing NGINX ===="
apt-get install -y nginx

echo "==== Start the NGINX Server ===="
service nginx start
```

This bootstrap script makes sure the system sources are up to date first. It then installs the nginx Web server. The provisioning is added to the Vagrantfile by adding the following line to the configure block so that your Vagrantfile now looks like this:

```
Vagrant.configure("2") do |config|
  config.vm.box = "hashicorp/precise32"
  config.vm.provision :shell, :path => "bootstrap.sh"
end
```

To use the new provisioning, destroy the running VM. This will stop and remove the virtual machine. To destroy the VM, simply use the command `vagrant destroy`. The base box still will be available for use after running the destroy command. Now to test the new configuration, simply run `vagrant up`. When a provisioning step is present in the Vagrantfile, Vagrant automatically will provision the box once it has booted. You can turn off the provisioning by using the `--no-provision` flag.

When `vagrant up` is complete, you will be able to `ssh` in to the machine as before. As shown below, let's make sure that the provisioning has been successful. Note that nginx is now installed on the system and running on the system:

```
$ vagrant ssh
$ which nginx
/usr/sbin/nginx
$ netstat -at | grep http
tcp        0      0 *:http      *:*        LISTEN
```

The virtual machine has been provisioned, and nginx is set up and started. The Web server can be tested to be working, but it's more difficult if you have to `ssh` in to the box to test this. Next, let's look at how to configure networking.

The first option to access ports on your virtual machine is port forwarding. For this example, nginx serves http traffic on port 80. To forward the guest machine port 80 to port 8080 on the host, add the following line to the Vagrantfile:

```
config.vm.network "forwarded_port", guest: 80, host: 8080
```

As before, destroy the VM and do another `vagrant up`. Once the virtual machine has been brought up and provisioned, the Web server can be accessed on the host port 8080. On your favorite browser, visit http://localhost:8080/ to be greeted with the nginx home page. This is one of the easier ways to be able to connect to the virtual machine.

For example, let's modify the bootstrap.sh to link the "html" directory in the Vagrant directory to the nginx root directory. This would be an example of doing simple static HTML Web site development. This could be distributed to anybody to modify the HTML on the host and immediately see the results on the guest.

The revised bootstrap looks like this:

```
#!/usr/bin/env bash
# provisioning is run as root by default

# make sure system sources are up to date.
apt-get update
# install a web server
echo "==== Installing NGINX ===="
apt-get install -y nginx

echo "==== Start the NGINX Server ===="
service nginx start

echo "==== Symlink Test Directory ===="
ln -s /vagrant/html/ /usr/share/nginx/www/html
```

You can access the html directory through the nginx Web server using the forwarded port once the VM has been booted and provisioned by Vagrant. For example, using curl:

```
$ curl "http://localhost:8080/html/tester.html"
```

This example assumes an html directory is in your Vagrant directory that has a tester.html file in it.

What if you want to provide a way to access the virtual machine as if it were another machine on the network? Vagrant provides a very easy way to use VirtualBox networking to set up the virtual machine with a static IP on the same LAN. Remove the port forwarding line from the Vagrantfile and replace it with the following:

```
config.vm.network "private_network", ip: "192.168.56.20"
```

Once you have re-ran `vagrant up`, your virtual machine will be accessible from your host using the IP address specified above. In my own projects, I use this private networking to assign my static IP a domain by modifying the /etc/hosts file. This allows testing of complicated http redirect flows that rely on certain domains.

It also allows multiple Vagrant virtual machines to contact one another, as it is a shared network between the host and all other virtual machines that are on the network. For distributed projects, being able to control the networking access of multiple standalone components all of which have their own Vagrantfile (or shared) is very powerful.

Again, in my work environment, it's not uncommon to spin up multiple virtual machines, one for a database, one for an app server and one for a squid proxy. Running functional tests against these tests a production-like setup. Vagrant allows you to do this in a simple, easy-to-repeat way, and the private network means there is no hassle in setting up intercommunication between the various components.

The last part of Vagrant that I want to cover is having multiple machines in the one Vagrantfile. I already have given an example of why you might want to have multiple Vagrantfiles for different components, so that they can be tested/developed on their own or as a collective. Having multiple machines in the one Vagrantfile is powerful when a single project can have multiple parts that don't belong on the one machine. Often in a Web development project, you will run the app server and database on the same machine for testing but will have multiple app servers and databases when that system gets into the production environment. This is the cause of a lot of production bugs due to untested component configuration.

Vagrant allows you to manage this all through a single Vagrantfile. If the idea of multiple machines is one that interests you, I also recommend looking closely at the new Docker container option for Vagrant. Docker is built on top of Linux containers and allows you to spin up machines much faster than VirtualBox. Multiple machine support in Vagrant is all the more powerful if the provider level is much faster to spin up machines.

For multiple machine support, the Vagrantfile allows you to configure multiple machines, each having their own configuration. The Vagrantfile below shows an example of what a MySQL master and slave Vagrantfile might look like:

```
Vagrant.configure("2") do |config|
  config.vm.box = "hashicorp/precise32"


  config.vm.define "master" do |master|
    master.vm.provision :shell, :path => "master_bootstrap.sh"
    master.vm.network "private_network", ip: "192.168.56.20"
  end
  config.vm.define "slave" do |slave|
    slave.vm.provision :shell, :path => "slave_bootstrap.sh"
    slave.vm.network "private_network", ip: "192.168.56.21"
  end
end
```

Each machine is defined with the first config block as separate blocks. They support all of the same options as the normal one-machine Vagrant config blocks. For example, for the master here, you define the shell provisioning script to be run as well as the static IP address to give this virtual machine. This allows you to test distributed systems in the same easy repeatable way as you could with a single machine.

When using a multiple machine configuration like the above, there are a few changes to the way you have to run Vagrant commands. Running `vagrant up` will bring up both boxes automatically. The master will be provisioned first and then the slave. You can choose to bring up only one of the machines by supplying the machine name:

```
$ vagrant up master
```

When the boxes have booted and been provisioned, you also have to specify the machine name when using `vagrant ssh`.

Conclusion

Let's reflect on what you have achieved here: multiple virtual machines booting, being provisioned automatically and consistent IP addresses using one command—`vagrant up`. This is repeatable across the three major operating systems and a number of different providers including the most popular virtualization technologies and cloud offerings.

This removes so many of the pain points from working on systems in local development. There are a lot of advantages to making the switch to using Vagrant for both local development and your continuous integration systems. All developers' test machines are running the same code and are provisioned in the same fashion. Test failures are not tied to a specific developer's setup and should be reproducable on anyone's machine by running `vagrant up`. This article covered only some of the amazing capabilities of Vagrant, so I encourage you also to read over the brilliant documentation at **http://docs.vagrantup.com**.■

Richard Delaney is a software engineer with Demonware Ireland. Richard works on back-end Web services using Python and the Django Web framework. He has been an avid Linux user and evangelist for the past five years.

Send comments or feedback via http://www.linuxjournal.com/contact or to ljeditor@linuxjournal.com.

An Introduction to OpenGL Programming

Learn how to create a 3-D cube using triangles.

MIHALIS TSOUKALOS

OpenGL is a well-known standard for generating 3-D as well as 2-D graphics that is extremely powerful and has many capabilities. OpenGL is defined and released by the OpenGL Architecture Review Board (ARB).

This article is a gentle introduction to OpenGL that will help you understand drawing using OpenGL.

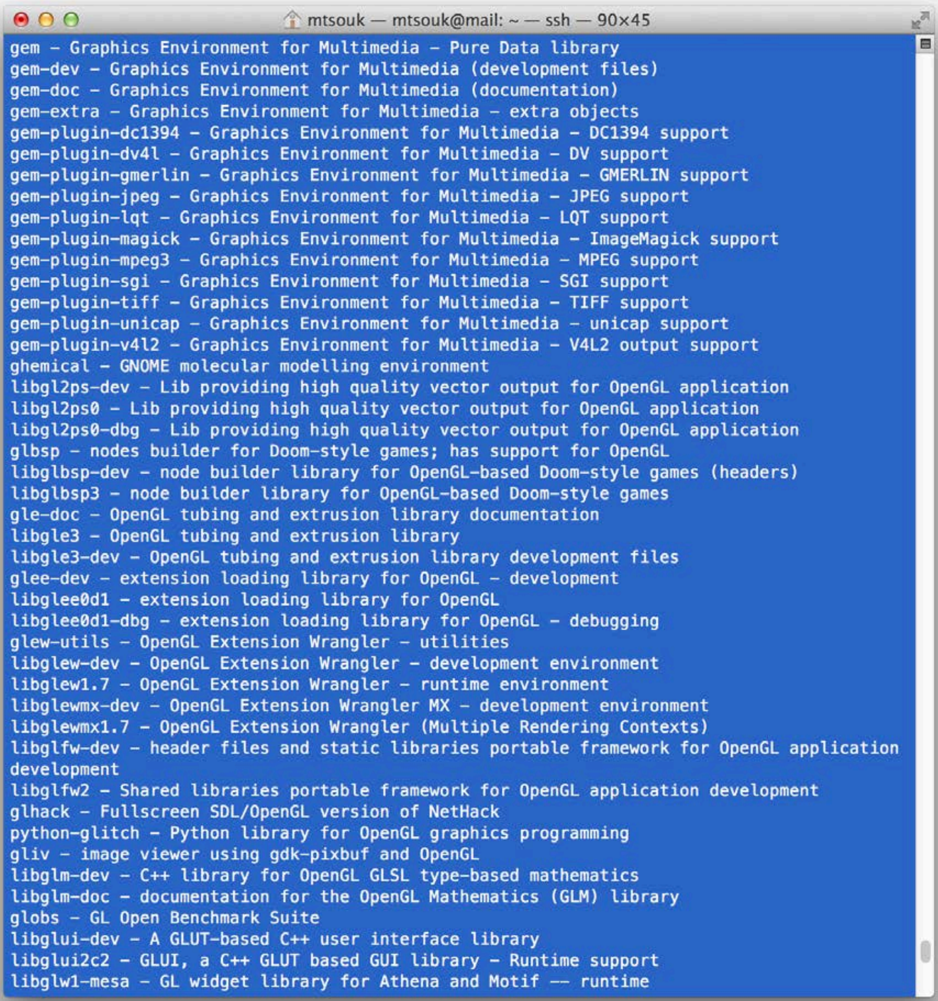
The latest version of OpenGL at the time of this writing is 4.4, which uses a different technique for drawing from the one presented here. Nevertheless, the purpose of this article is to help you understand OpenGL philosophy, not teach you how to code using the latest OpenGL version. Therefore, the presented source code can be run on Linux machines that have an older OpenGL version installed.

Installing OpenGL

If you run the following command on a Debian 7 system to find all packages that contain the word "opengl", you'll get plenty of output (Figure 1):

```
$ apt-cache search opengl
```

There are many free OpenGL implementations for Linux, but you need only one. I installed FreeGLUT, because it is the most up



```
gem - Graphics Environment for Multimedia - Pure Data library
gem-dev - Graphics Environment for Multimedia (development files)
gem-doc - Graphics Environment for Multimedia (documentation)
gem-extra - Graphics Environment for Multimedia - extra objects
gem-plugin-dc1394 - Graphics Environment for Multimedia - DC1394 support
gem-plugin-dv4l - Graphics Environment for Multimedia - DV support
gem-plugin-gmerlin - Graphics Environment for Multimedia - GMERLIN support
gem-plugin-jpeg - Graphics Environment for Multimedia - JPEG support
gem-plugin-lqt - Graphics Environment for Multimedia - LQT support
gem-plugin-magick - Graphics Environment for Multimedia - ImageMagick support
gem-plugin-mpeg3 - Graphics Environment for Multimedia - MPEG support
gem-plugin-sgi - Graphics Environment for Multimedia - SGI support
gem-plugin-tiff - Graphics Environment for Multimedia - TIFF support
gem-plugin-unicap - Graphics Environment for Multimedia - unicap support
gem-plugin-v4l2 - Graphics Environment for Multimedia - V4L2 output support
ghemical - GNOME molecular modelling environment
libgl2ps-dev - Lib providing high quality vector output for OpenGL application
libgl2ps0 - Lib providing high quality vector output for OpenGL application
libgl2ps0-dbg - Lib providing high quality vector output for OpenGL application
glbsp - nodes builder for Doom-style games; has support for OpenGL
libglbsp-dev - node builder library for OpenGL-based Doom-style games (headers)
libglbsp3 - node builder library for OpenGL-based Doom-style games
gle-doc - OpenGL tubing and extrusion library documentation
libgle3 - OpenGL tubing and extrusion library
libgle3-dev - OpenGL tubing and extrusion library development files
glee-dev - extension loading library for OpenGL - development
libglee0d1 - extension loading library for OpenGL
libglee0d1-dbg - extension loading library for OpenGL - debugging
glew-utils - OpenGL Extension Wrangler - utilities
libglew-dev - OpenGL Extension Wrangler - development environment
libglew1.7 - OpenGL Extension Wrangler - runtime environment
libglewmx-dev - OpenGL Extension Wrangler MX - development environment
libglewmx1.7 - OpenGL Extension Wrangler (Multiple Rendering Contexts)
libglfw-dev - header files and static libraries portable framework for OpenGL application
development
libglfw2 - Shared libraries portable framework for OpenGL application development
glhack - Fullscreen SDL/OpenGL version of NetHack
python-glitch - Python library for OpenGL graphics programming
gliv - image viewer using gdk-pixbuf and OpenGL
libglm-dev - C++ library for OpenGL GLSL type-based mathematics
libglm-doc - documentation for the OpenGL Mathematics (GLM) library
globs - GL Open Benchmark Suite
libglui-dev - A GLUT-based C++ user interface library
libglui2c2 - GLUI, a C++ GLUT based GUI library - Runtime support
libglw1-mesa - GL widget library for Athena and Motif -- runtime
```

Figure 1. Running `apt-cache search opengl`

to date. FreeGLUT is an open-source alternative to the OpenGL Utility Toolkit (GLUT) library:

```
root@mail:~# apt-get install freeglut3 freeglut3-dev libglew-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed
 and is no longer required:
  fonts-liberation
Use 'apt-get autoremove' to remove it.
The following extra packages will be installed:
  libgl1-mesa-dev libglu1-mesa-dev libice-dev libpthread-stubs0
  libpthread-stubs0-dev libsm-dev libx11-dev libx11-doc
  libxau-dev libxcb1-dev libxdmcp-dev libxext-dev libxt-dev
  mesa-common-dev x11proto-core-dev x11proto-input-dev
  x11proto-kb-dev x11proto-xext-dev xorg-sgml-doctools xtrans-dev
Suggested packages:
  libice-doc libsm-doc libxcb-doc libxext-doc libxt-doc
The following NEW packages will be installed:
  freeglut3 freeglut3-dev libgl1-mesa-dev libglu1-mesa-dev
  libice-dev libpthread-stubs0 libpthread-stubs0-dev libsm-dev
  libx11-dev libx11-doc libxau-dev libxcb1-dev libxdmcp-dev
  libxext-dev libxt-dev mesa-common-dev x11proto-core-dev
  x11proto-input-dev x11proto-kb-dev x11proto-xext-dev
  xorg-sgml-doctools xtrans-dev
0 upgraded, 22 newly installed, 0 to remove and 0 not upgraded.
Need to get 7,651 kB of archives.
After this operation, 24.9 MB of additional disk space
 will be used.
Do you want to continue [Y/n]?
```

You also will need a C++ compiler to compile the code.

Finally, you may need to install the mesa-utils package in order to be able to use the glxinfo command:

```
# apt-get install mesa-utils
```

The glxinfo command displays useful information about your OpenGL installation, as you can see in the following output:

```
...
GLX version: 1.4
GLX extensions:
    GLX_ARB_get_proc_address, GLX_ARB_multisample,
    GLX_EXT_import_context, GLX_EXT_texture_from_pixmap,
    GLX_EXT_visual_info, GLX_EXT_visual_rating,
    GLX_MESA_copy_sub_buffer, GLX_MESA_multithread_makecurrent,
    GLX_OML_swap_method, GLX_SGIS_multisample, GLX_SGIX_fbconfig,
    GLX_SGIX_pbuffer, GLX_SGI_make_current_read
OpenGL vendor string: VMware, Inc.
OpenGL renderer string: Gallium 0.4 on llvmpipe
  (LLVM 3.4, 128 bits)
OpenGL version string: 2.1 Mesa 10.1.3
OpenGL shading language version string: 1.30
OpenGL extensions:
...
```

Mesa is a 3-D graphics library with an API that is so very similar to OpenGL's, it is pretty much indistinguishable.

OpenGL Pipeline

Figure 2—taken from the OpenGL

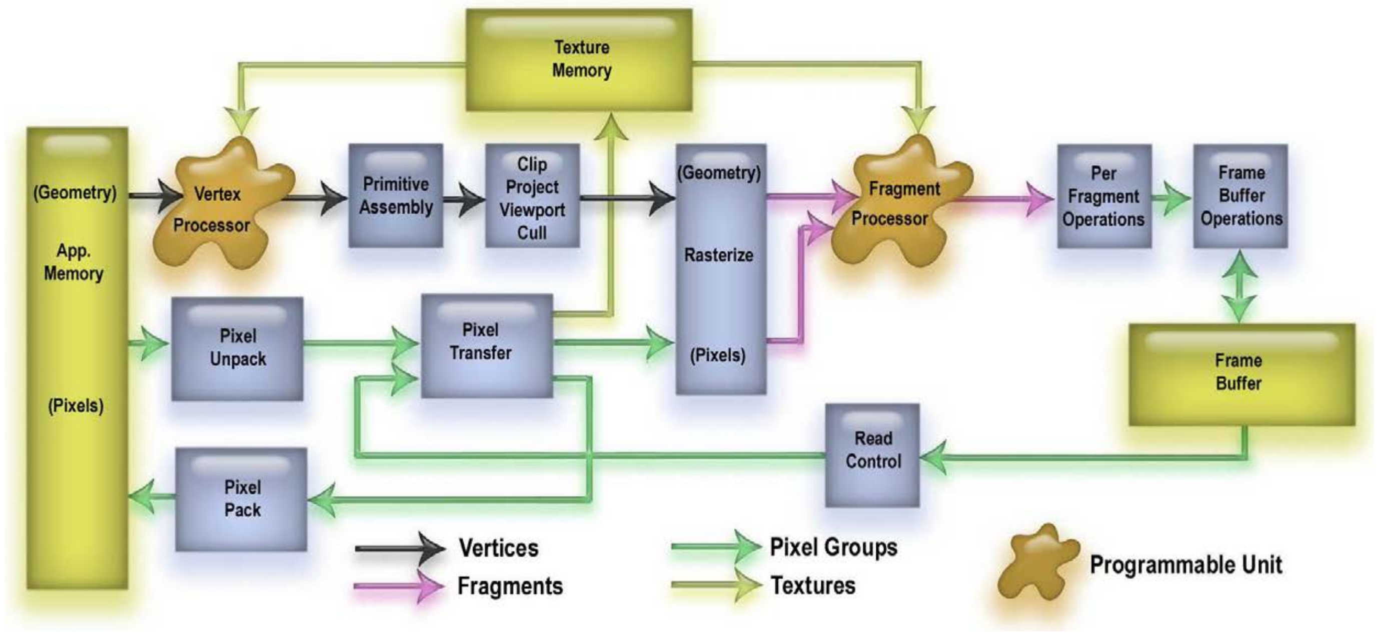


Figure 2. OpenGL Architecture

Shading Language book (aka "The Orange Book")—shows the programmable OpenGL pipeline with the vertex and fragment processors. As you can imagine, the OpenGL pipeline is complex, but you do not have to understand it fully in order to be able to use OpenGL. The Pipeline shows how OpenGL operates in the background. Newer versions of the OpenGL Pipeline are even more complex!

OpenGL is a big state machine. Most calls to OpenGL functions modify a global state that you cannot access directly.

The OpenGL Shading Language code that is intended for execution on one of the OpenGL programmable processors is called a Shader. The OpenGL Shading Language has its roots in C (presenting the OpenGL Shading Language is beyond the scope of this article).

OpenGL does not define a windowing layer, because it tries to be platform-neutral and leaves this functionality to the operating system. The OS must provide a rendering context that accepts commands as well as a framebuffer that keeps the results of the drawing commands.

Matrix Algebra is extensively used in 3-D graphics, so it is good for you to know how to add, multiply, subtract and divide matrices, although you will not need to code such operations yourself. It also is useful to become familiar with 3-D coordinates and be able to sketch on paper the 3-D scene you are trying to draw.

Drawing a Triangle

Now it's time for some real OpenGL code. The code in Listing 1, when executed, draws a triangle on-screen using OpenGL.

The code in Listing 1 for setting up

Listing 1. triangle.cc

```
// Programmer: Mihalis Tsoukalos
// Date: Wednesday 04 June 2014
//
// A simple OpenGL program that draws a triangle.

#include "GL/freeglut.h"
#include "GL/gl.h"

void drawTriangle()
{
    glClearColor(0.4, 0.4, 0.4, 0.4);
    glClear(GL_COLOR_BUFFER_BIT);

    glColor3f(1.0, 1.0, 1.0);
    glOrtho(-1.0, 1.0, -1.0, 1.0, -1.0, 1.0);

        glBegin(GL_TRIANGLES);
                glVertex3f(-0.7, 0.7, 0);
                glVertex3f(0.7, 0.7, 0);
                glVertex3f(0, -1, 0);
        glEnd();

    glFlush();
}

int main(int argc, char **argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE);
    glutInitWindowSize(500, 500);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("OpenGL - Creating a triangle");
    glutDisplayFunc(drawTriangle);
    glutMainLoop();
    return 0;
}
```

OpenGL is large, but you will have to learn it only once.

On a Debian 7 system, the following command compiled the triangle.cc OpenGL program without any error messages:

```
$ g++ triangle.cc -lglut -o triangle
```

On an Ubuntu Linux system, the same command produced the following error messages:

```
/usr/bin/ld: /tmp/ccnBP5li.o: undefined reference to symbol
➥'glOrtho'
//usr/lib/x86_64-linux-gnu/mesa/libGL.so.1: error adding
//symbols: DSO missing from command line
collect2: error: ld returned 1 exit status
```

The solution is to compile the triangle.cc program by linking the executable to an additional library (-lGL):

```
mtsouk@mtsouk-VirtualBox:/media/sf_OpenGL.LJ/code$ g++
➥triangle.cc -lglut -lGL -o triangle
```

The libGL.so library accepts OpenGL commands and makes sure that they are put on the screen in some way. If your graphics card does not have 3-D acceleration, libGL contains a software renderer that gives a 2-D image as output to the X Window System. This is the case with Mesa.

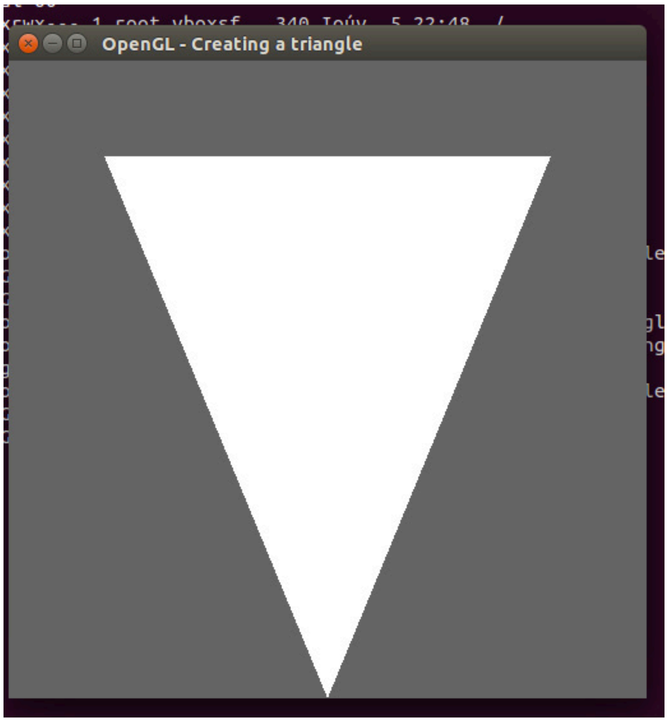


Figure 3. Drawing a Triangle Using OpenGL

libGL also can pass the OpenGL information to the X Window System, if the GLX extension is present. Then, the X Window System can do software rendering with the help of Mesa or use hardware acceleration.

The output from the executable will produce the triangle shown in Figure 3. The correct compilation of triangle.cc is proof that your Linux system can be used for developing OpenGL applications.

There is more than one way of drawing a triangle using OpenGL, mostly depending on the OpenGL version you are using. Although the presented method is an older way of programming

OpenGL applications, I find it straightforward and easy to understand. Remember, whichever method you use, the coordinates of the triangle will be the same.

Note: keep in mind that the most important part of this article is the code!

Drawing a Cube Using OpenGL

Next, let's program an application that draws a cube using OpenGL. You need to construct the cube using triangles. A cube has six faces, and each face needs at least two triangles to be defined. The reason I say "at least" is that, generally

Listing 2. cube.cc

```
// Programmer: Mihalis Tsoukalos
// Date: Wednesday 04 June 2014
//
// A simple OpenGL program that draws a colorful cube
// that rotates as you move the arrow keys!
//
// g++ cube.cc -lm -lglut -lGL -lGLU -o cube

#define GL_GLEXT_PROTOTYPES
#ifdef __APPLE__
#include <GLUT/glut.h>
#else
#include <GL/glut.h>
#endif
#include <math.h>

// Rotate X
double rX=0;
// Rotate Y
double rY=0;

// The coordinates for the vertices of the cube
double x = 0.6;
double y = 0.6;
double z = 0.6;

void drawCube()
{
        // Set Background Color
    glClearColor(0.4, 0.4, 0.4, 1.0);
        // Clear screen
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    // Reset transformations
    glLoadIdentity();

    // Rotate when user changes rX and rY
    glRotatef( rX, 1.0, 0.0, 0.0 );
    glRotatef( rY, 0.0, 1.0, 0.0 );≠

    // BACK
        glBegin(GL_TRIANGLES);
            glColor3f(0.4, 0.3, 0.5);
                glVertex3f(x, y, z);
                glVertex3f(x, -y, z);
                glVertex3f(-x, y, z);
        glEnd();

        glBegin(GL_TRIANGLES);
            glColor3f(0.5, 0.3, 0.2);
                glVertex3f(-x, -y, z);
                glVertex3f(x, -y, z);
                glVertex3f(-x, y, z);
        glEnd();

        // FRONT
        // Using 4 trianges!
        glBegin(GL_TRIANGLES);
            glColor3f(0.1, 0.5, 0.3);
                glVertex3f(-x, y, -z);
                glVertex3f(0, 0, -z);
                glVertex3f(-x, -y, -z);
        glEnd();

        glBegin(GL_TRIANGLES);
                glColor3f(0.0, 0.5, 0.0);
                glVertex3f(-x, -y, -z);
                glVertex3f(0, 0, -z);
                glVertex3f(x, -y, -z);
        glEnd();

        glBegin(GL_TRIANGLES);
            glColor3f(0.1, 0.3, 0.3);
                glVertex3f(-x, y, -z);
                glVertex3f(x, y, -z);
                glVertex3f(0, 0, -z);
        glEnd();

        glBegin(GL_TRIANGLES);
                glColor3f(0.2, 0.2, 0.2);
                glVertex3f(0, 0, -z);
                glVertex3f(x, y, -z);
                glVertex3f(x, -y, -z);
        glEnd();

        // LEFT
        glBegin(GL_TRIANGLES);
        glColor3f(0.3, 0.5, 0.6);
                glVertex3f(-x, -y, -z);
                glVertex3f(-x, -y, z);
                glVertex3f(-x, y, -z);
        glEnd();

        glBegin(GL_TRIANGLES);
                glColor3f(0.5, 0.5, 0.5);
                glVertex3f(-x, y, z);
                glVertex3f(-x, -y, z);
                glVertex3f(-x, y, -z);
        glEnd();
```

speaking, you can use more triangles if you want in order to have a smoother and more accurate shape, but when drawing a cube, that's not necessary. As I'm sure you've realized, you are going to need 12 triangles in total.

A cube also has eight vertices. Each triangle requires three different vertices to be defined.

Listing 2 shows the full source code of the cube.cc file.

Figure 4 shows two different screenshots from the cube.cc application. The left screenshot of the program just shows a square,

```
        // RIGHT
        glBegin(GL_TRIANGLES);
        glColor3f(0.2, 0.2, 0.2);
                glVertex3f(x, y, z);
                glVertex3f(x, y, -z);
                glVertex3f(x, -y, z);
        glEnd();

        glBegin(GL_TRIANGLES);
        glColor3f(0.0, 0.0, 0.0);
                glVertex3f(x, -y, -z);
                glVertex3f(x, y, -z);
                glVertex3f(x, -y, z);
        glEnd();

        // TOP
        glBegin(GL_TRIANGLES);
        glColor3f(0.6, 0.0, 0.0);
                glVertex3f(x, y, z);
                glVertex3f(x, y, -z);
                glVertex3f(-x, y, -z);
        glEnd();

        glBegin(GL_TRIANGLES);
        glColor3f(0.6, 0.1, 0.2);
                glVertex3f(-x, y, z);
                glVertex3f(x, y, z);
                glVertex3f(-x, y, -z);
        glEnd();

        // BOTTOM
        glBegin(GL_TRIANGLES);
        glColor3f(0.4, 0.0, 0.4);
                glVertex3f(-x, -y, -z);
                glVertex3f(-x, -y, z);
                glVertex3f(x, -y, z);
        glEnd();

        glBegin(GL_TRIANGLES);
                glColor3f(0.3, 0.0, 0.3);
                glVertex3f(x, -y, -z);
                glVertex3f(-x, -y, -z);
                glVertex3f(x, -y, z);
        glEnd();

    glFlush();
    glutSwapBuffers();
}

void keyboard(int key, int x, int y)
{
    if (key == GLUT_KEY_RIGHT)
        {
                rY += 15;
        }
    else if (key == GLUT_KEY_LEFT)
        {
                rY -= 15;
        }
    else if (key == GLUT_KEY_DOWN)
        {
                rX -= 15;
        }
    else if (key == GLUT_KEY_UP)
        {
                rX += 15;
        }

    // Request display update
    glutPostRedisplay();
}


int main(int argc, char **argv)
{
        // Initialize GLUT and process user parameters
        glutInit(&argc, argv);

        // Request double buffered true color window with Z-buffer
        glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);

        glutInitWindowSize(700,700);
        glutInitWindowPosition(100, 100);

        // Create window
        glutCreateWindow("Linux Journal OpenGL Cube");

        // Enable Z-buffer depth test
        glEnable(GL_DEPTH_TEST);

        // Callback functions
        glutDisplayFunc(drawCube);
        glutSpecialFunc(keyboard);

        // Pass control to GLUT for events
        glutMainLoop();

        return 0;
}
```

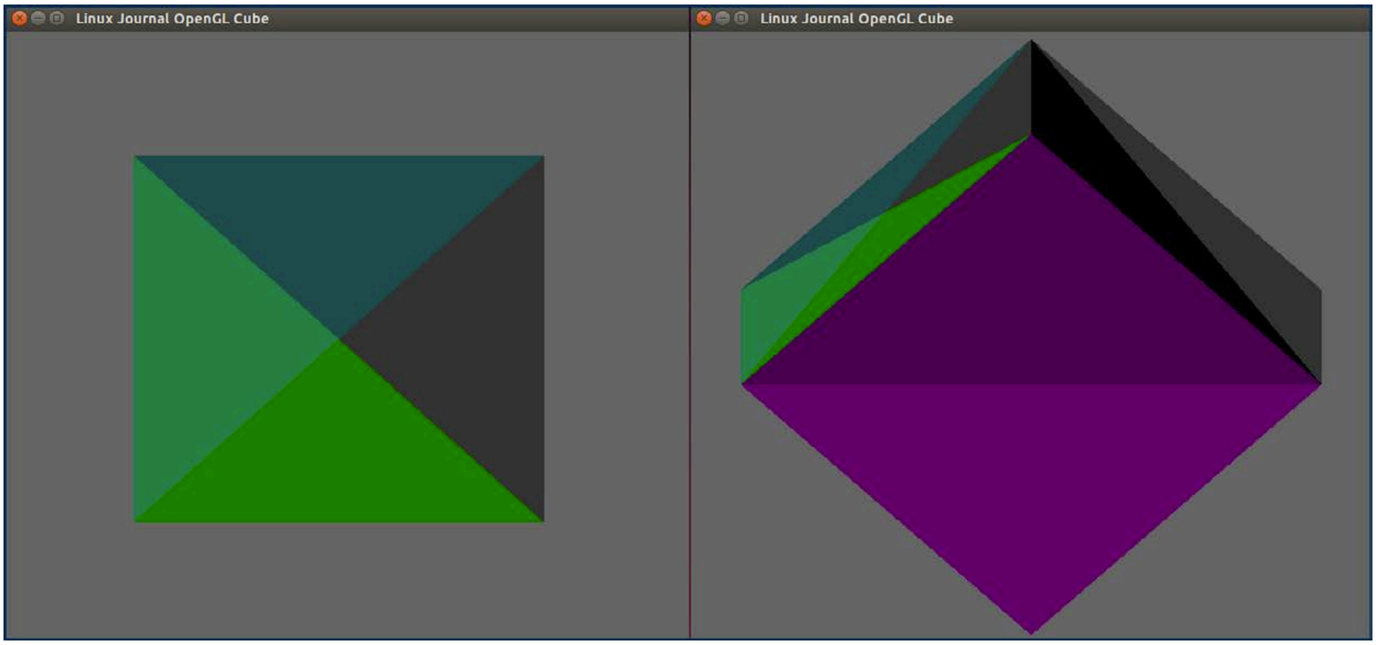


Figure 4. The Graphical Output of cube.cc

because you are looking straight at the front face of the cube and can't see any of the other five faces of the cube. Therefore, the shape you see on screen looks like it is in 2-D. When you start rotating the cube using the arrow keys, you can tell that it is a 3-D shape.

Explaining the Code

Each triangle is defined by three vertices. Each vertex is defined with the help of a single point (x,y,z). Each point comes with three numbers (coordinates), because OpenGL uses a 3-D space. A cube needs eight vertices to be defined.

As an exercise, the front face of the cube is made using four triangles. Figure 5 shows the coordinates of the front face of the cube when four triangles are used to define it. The point (0,0,-0.6) has been chosen arbitrarily. You just need a point that belongs to the front face of the cube.

Defining the Vertices: Figure 6 shows the coordinates of the vertices of the cube for x=0.6, y=0.6 and z=0.6. Note that the vertices that define an edge have two out of their three coordinates exactly the same.

As you can see in Figure 6, the coordinates of the four vertices that define the front face and the respective coordinates of the four vertices that define the back face differ only in the value of the

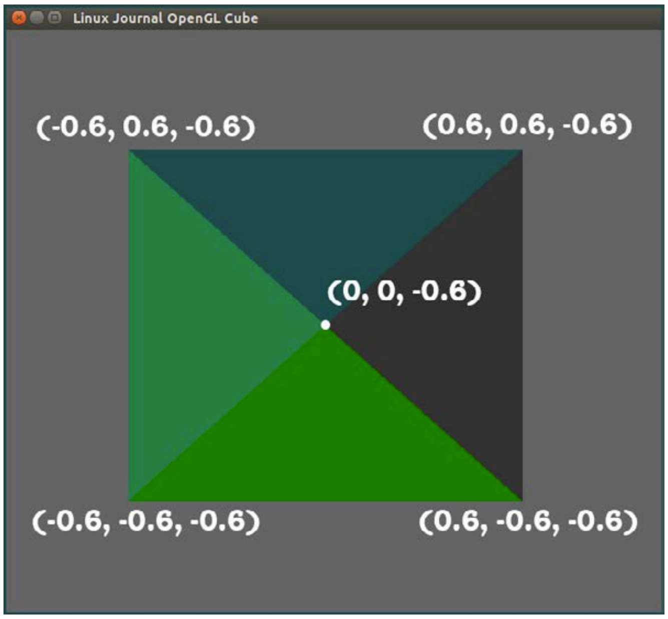


Figure 5. The Coordinates of the Front Face of the Cube

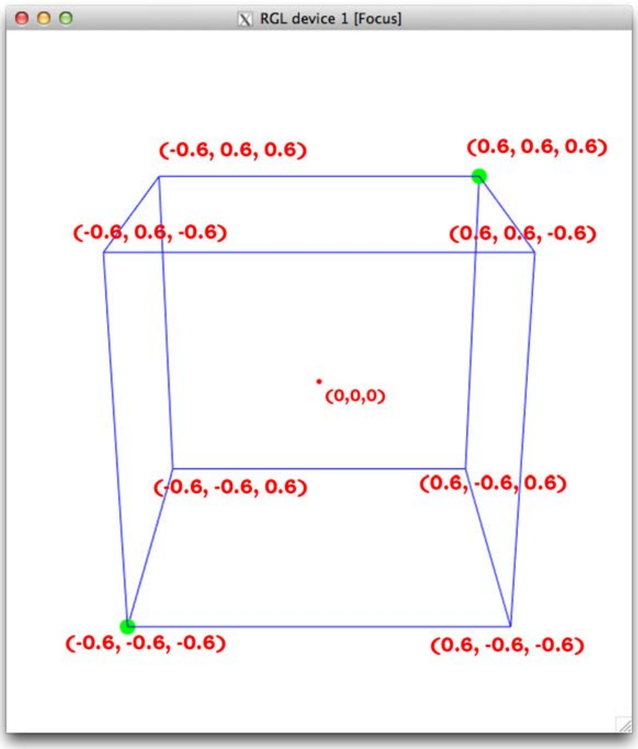


Figure 6. The Vertices of the Cube

z-axis coordinate.

Defining the Triangles: The triangles are based on vertices. Each triangle needs three vertices to be defined. Two triangles are needed for defining each face of the cube, except for the front face where four triangles are used. The following commands create a color triangle based on the values of x, y and z:

```
glBegin(GL_TRIANGLES);
glColor3f(0.4, 0.0, 0.4);
        glVertex3f(-x, -y, -z);
        glVertex3f(-x, -y, z);
        glVertex3f(x, -y, z);
glEnd();
```

Changing Color: You can change the color of a shape with the `glColor3f(...)` command. The `glColor3f(...)` command takes three parameters, which represent the RGB values of the desired color.

Changing Perspective: You can change the perspective of the scene with the following commands:

```
glRotatef(rX, 1.0, 0.0, 0.0);
glRotatef(rY, 0.0, 1.0, 0.0);
```

As the user presses one of the four arrow keys, the perspective angle changes accordingly.

Drawing the Cube: Drawing the cube is done face by face. Each face needs two triangles to be drawn except for the front face where four triangles are used.

Once you get the coordinates of the triangles correctly, drawing is pretty easy. The drawing of each triangle starts with the `glBegin(GL_TRIANGLES)` command and finishes with the `glEnd()` command. The `GL_TRIANGLES` is an OpenGL primitive. Other primitive types are `GL_POINTS`, `GL_LINES`, `GL_LINE_STRIP`, `GL_LINE_LOOP`, `GL_TRIANGLE_STRIP`, `GL_TRIANGLE_FAN`, `GL_QUADS`, `GL_QUAD_STRIP` and `GL_POLYGON`. Eventually, every primitive shape in OpenGL is represented by one or more triangles.

If you are drawing using triangles (`GL_TRIANGLES`), the order in which you put the vertices is not important. If you are drawing using rectangles (`GL_POLYGON`), the four vertices of the rectangle must be drawn in the right order, although it does not matter if you draw CW or CCW. If the drawing order is wrong, the rectangle you are trying to draw will have a big hole in it.

Using the Arrow Keys: Using the arrow keys is pretty simple. The following code checks for the arrow keys and acts accordingly:

```
void keyboard(int key, int x, int y)
{
    if (key == GLUT_KEY_RIGHT)
        {
                rY += 15;
        }
    else if (key == GLUT_KEY_LEFT)
        {
                rY -= 15;
        }
    else if (key == GLUT_KEY_DOWN)
        {
                rX -= 15;
        }
    else if (key == GLUT_KEY_UP)
        {
                rX += 15;
        }

    // Request display update
    glutPostRedisplay();
}
```

The `keyboard(...)` function is registered inside the `main(...)` function using the following line of code:

```
glutSpecialFunc(keyboard);
```

Rotating a Cube Automatically

As a bonus, let's look at how to make a cube rotate automatically

Drawing using rectangles is easier and requires less code, but complex OpenGL code runs faster when triangles are used.

(Figure 7).

This time, the cube is drawn using rectangles. As the cube has six faces, only six rectangles are needed. Drawing using rectangles is easier and requires less code, but complex OpenGL code runs faster when triangles are used.

Note: every object can be split into triangles, but a triangle cannot be split into anything other than triangles.

Listing 3 shows the source code for rotateCube.cc.

Note that the `main(...)` functions of triangle.cc, cube.cc and rotateCube.cc are pretty similar, although the three programs perform different tasks.

The crucial thing here is the usage of the `glutTimerFunc(...)` function. It registers a timer callback for the `update(...)` function that is going to be triggered in a specified number of milliseconds. The `update(...)` function changes the angle of the scene every time it's called.

Conclusion

OpenGL programming is not easy, but this article should help you start

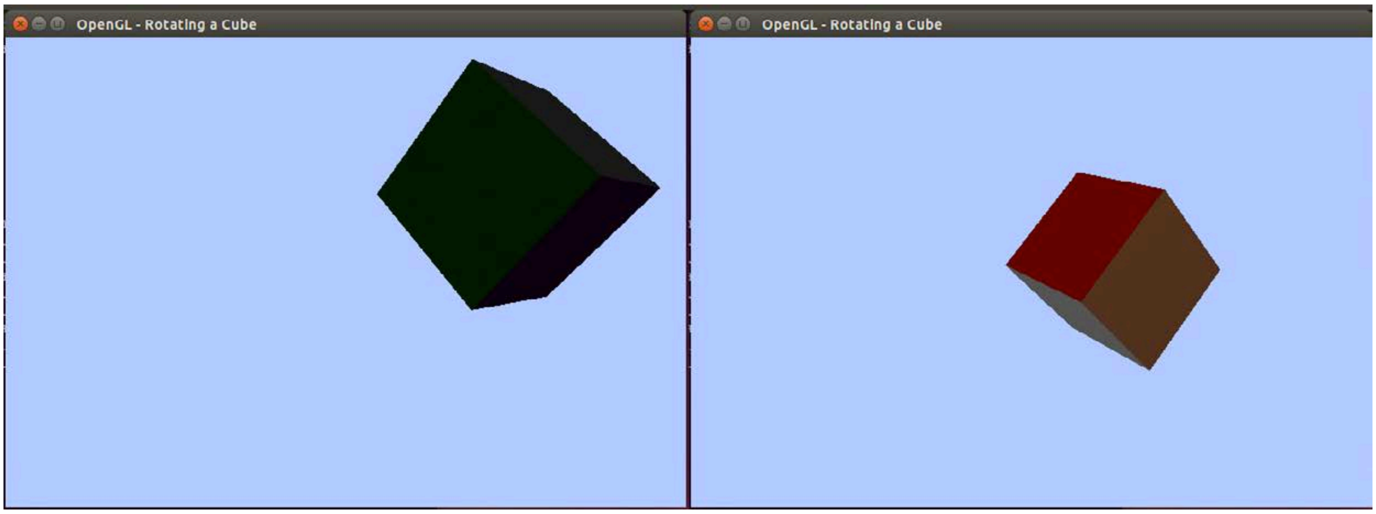


Figure 7. The Output of rotateCube.cc

Listing 3. rotateCube.cc

```
// Programmer: Mihalis Tsoukalos
// Date: Wednesday 04 June 2014
//
// A simple OpenGL program that draws a triangle
// and automatically rotates it.
//
// g++ rotateCube.cc -lm -lglut -lGL -lGLU -o rotateCube

#include <iostream>
#include <stdlib.h>

// the GLUT and OpenGL libraries have to be linked correctly
#ifdef __APPLE__
#include <OpenGL/OpenGL.h>
#include <GLUT/glut.h>
#else
#include <GL/glut.h>
#endif

using namespace std;

// The coordinates for the vertices of the cube
double x = 0.6;
double y = 0.6;
double z = 0.6;

float angle = 0.0;

void drawCube()
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    // Reset transformations
        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity();

        glTranslatef(0.0, 0.0, -5.0);

        // Add an ambient light
        GLfloat ambientColor[] = {0.2, 0.2, 0.2, 1.0};
        glLightModelfv(GL_LIGHT_MODEL_AMBIENT, ambientColor);

        // Add a positioned light
        GLfloat lightColor0[] = {0.5, 0.5, 0.5, 1.0};
        GLfloat lightPos0[] = {4.0, 0.0, 8.0, 1.0};
        glLightfv(GL_LIGHT0, GL_DIFFUSE, lightColor0);
        glLightfv(GL_LIGHT0, GL_POSITION, lightPos0);

        glTranslatef(0.5, 1.0, 0.0);
        glRotatef(angle, 1.0, 1.0, 1.0);

    glRotatef( angle, 1.0, 0.0, 1.0 );
    glRotatef( angle, 0.0, 1.0, 1.0 );
        glTranslatef(-0.5, -1.0, 0.0);

        // Create the 3D cube

    // BACK
    glBegin(GL_POLYGON);
    glColor3f(0.5, 0.3, 0.2);
    glVertex3f(x, -y, z);
    glVertex3f(x, y, z);
    glVertex3f(-x, y, z);
    glVertex3f(-x, -y, z);
    glEnd();

        // FRONT
        glBegin(GL_POLYGON);
        glColor3f(0.0, 0.5, 0.0);
        glVertex3f(-x, y, -z);
        glVertex3f(-x, -y, -z);
        glVertex3f(x, -y, -z);
        glVertex3f(x, y, -z);
        glEnd();

        // LEFT
        glBegin(GL_POLYGON);
        glColor3f(0.5, 0.5, 0.5);
        glVertex3f(-x, -y, -z);
        glVertex3f(-x, -y, z);
        glVertex3f(-x, y, z);
        glVertex3f(-x, y, -z);
        glEnd();


        // RIGHT
        glBegin(GL_POLYGON);
        glColor3f(0.0, 0.0, 0.0);
```

writing OpenGL applications quickly. If you want to become proficient in OpenGL, keep practicing writing more OpenGL programs. OpenGL is easy to start with but difficult to master.

Acknowledgement

I would like to thank Dr Nikos Platis for sharing a small part of his OpenGL knowledge with me.■

Mihalis Tsoukalos is a UNIX administrator, a programmer (UNIX and iOS), a DBA and mathematician. You can reach him at http://www.mtsoukalos.eu and @mactsouk (Twitter). Contact him if you are looking for a UNIX system administrator.

```
        glVertex3f(x, -y, -z);
        glVertex3f(x, -y, z);
        glVertex3f(x, y, z);
        glVertex3f(x, y, -z);
        glEnd();

        // TOP
        glBegin(GL_POLYGON);
        glColor3f(0.6, 0.0, 0.0);
        glVertex3f(x, y, z);
        glVertex3f(-x, y, z);
        glVertex3f(-x, y, -z);
        glVertex3f(x, y, -z);
        glEnd();


        // BOTTOM
        glBegin(GL_POLYGON);
        glColor3f(0.3, 0.0, 0.3);
        glVertex3f(-x, -y, -z);
        glVertex3f(-x, -y, z);
        glVertex3f(x, -y, z);
        glVertex3f(x, -y, -z);
        glEnd();

        glFlush();
    glutSwapBuffers();
}

// Function for increasing the angle variable smoothly,
// keeps it <=360
// It can also be implemented using the modulo operator.
void update(int value)
{
        angle += 1.0f;
        if (angle > 360)
                {
                        angle -= 360;
        }

        glutPostRedisplay();
        glutTimerFunc(25, update, 0);
}

// Initializes 3D rendering
void initRendering()
{
        glEnable(GL_DEPTH_TEST);
        glEnable(GL_COLOR_MATERIAL);

        // Set the color of the background
        glClearColor(0.7f, 0.8f, 1.0f, 1.0f);
        glEnable(GL_LIGHTING);
        glEnable(GL_LIGHT0);
        glEnable(GL_NORMALIZE);
}


// Called when the window is resized
void handleResize(int w, int h)
{
        glViewport(0, 0, w, h);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        gluPerspective(45.0, (double)w / (double)h, 1.0, 200.0);
}


int main(int argc, char **argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
    glutInitWindowSize(700, 700);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("OpenGL - Rotating a Cube");
        initRendering();

    glutDisplayFunc(drawCube);
        glutReshapeFunc(handleResize);

        // Add a timer for the update(...) function
    glutTimerFunc(25, update, 0);

    glutMainLoop();
    return 0;
}
```

Resources

OpenGL: **http://www.opengl.org**

Learning Modern 3D Graphics Programming: **http://www.arcsynthesis.org/gltut**

OpenGL Superbible, 6th edition, Graham Sellers, Richard S. Wright and Nicholas Haemel, Addison Wesley, ISBN: 0321902947

GLEW: **http://glew.sourceforge.net**

The Mesa 3D Graphics Library: **http://www.mesa3d.org**

WEBCASTS

Learn the 5 Critical Success Factors to Accelerate IT Service Delivery in a Cloud-Enabled Data Center

Today's organizations face an unparalleled rate of change. Cloud-enabled data centers are increasingly seen as a way to accelerate IT service delivery and increase utilization of resources while reducing operating expenses. Building a cloud starts with virtualizing your IT environment, but an end-to-end cloud orchestration solution is key to optimizing the cloud to drive real productivity gains.

> **http://lnxjr.nl/IBM5factors**

Modernizing SAP Environments with Minimum Risk—a Path to Big Data

Sponsor: SAP | Topic: Big Data

Is the data explosion in today's world a liability or a competitive advantage for your business? Exploiting massive amounts of data to make sound business decisions is a business imperative for success and a high priority for many firms. With rapid advances in x86 processing power and storage, enterprise application and database workloads are increasingly being moved from UNIX to Linux as part of IT modernization efforts. Modernizing application environments has numerous TCO and ROI benefits but the transformation needs to be managed carefully and performed with minimal downtime. Join this webinar to hear from top IDC analyst, Richard Villars, about the path you can start taking now to enable your organization to get the benefits of turning data into actionable insights with exciting x86 technology.

> **http://lnxjr.nl/modsap**

WHITE PAPERS

White Paper: JBoss Enterprise Application Platform for OpenShift Enterprise

Sponsor: DLT Solutions

Red Hat's® JBoss Enterprise Application Platform for OpenShift Enterprise offering provides IT organizations with a simple and straightforward way to deploy and manage Java applications. This optional OpenShift Enterprise component further extends the developer and manageability benefits inherent in JBoss Enterprise Application Platform for on-premise cloud environments.

Unlike other multi-product offerings, this is not a bundling of two separate products. JBoss Enterprise Middleware has been hosted on the OpenShift public offering for more than 18 months. And many capabilities and features of JBoss Enterprise Application Platform 6 and JBoss Developer Studio 5 (which is also included in this offering) are based upon that experience.

This real-world understanding of how application servers operate and function in cloud environments is now available in this single on-premise offering, JBoss Enterprise Application Platform for OpenShift Enterprise, for enterprises looking for cloud benefits within their own datacenters.

> **http://lnxjr.nl/jbossapp**

WHITE PAPERS

Linux Management with Red Hat Satellite: Measuring Business Impact and ROI

Sponsor: Red Hat | **Topic:** Linux Management

Linux has become a key foundation for supporting today's rapidly growing IT environments. Linux is being used to deploy business applications and databases, trading on its reputation as a low-cost operating environment. For many IT organizations, Linux is a mainstay for deploying Web servers and has evolved from handling basic file, print, and utility workloads to running mission-critical applications and databases, physically, virtually, and in the cloud. As Linux grows in importance in terms of value to the business, managing Linux environments to high standards of service quality — availability, security, and performance — becomes an essential requirement for business success.

> **http://lnxjr.nl/RHS-ROI**

Standardized Operating Environments for IT Efficiency

Sponsor: Red Hat

The Red Hat® Standard Operating Environment SOE helps you define, deploy, and maintain Red Hat Enterprise Linux® and third-party applications as an SOE. The SOE is fully aligned with your requirements as an effective and managed process, and fully integrated with your IT environment and processes.

Benefits of an SOE:

SOE is a specification for a tested, standard selection of computer hardware, software, and their configuration for use on computers within an organization. The modular nature of the Red Hat SOE lets you select the most appropriate solutions to address your business' IT needs.

SOE leads to:

- Dramatically reduced deployment time.
- Software deployed and configured in a standardized manner.
- Simplified maintenance due to standardization.
- Increased stability and reduced support and management costs.
- There are many benefits to having an SOE within larger environments, such as:
  - Less total cost of ownership (TCO) for the IT environment.
  - More effective support.
  - Faster deployment times.
  - Standardization.

> **http://lnxjr.nl/RH-SOE**

Promise Theory—What Is It?

Mark Burgess describes the idea behind Promise Theory—the framework developed between Oslo University College and the University of Amsterdam during the past decade to solve the problem of configuring distributed systems and services.

MARK BURGESS

During the past 20 years, there has been a growing sense of inadequacy about the "command and control" model for managing IT systems. Years in front of the television with a remote control have left us hard pressed to think of any other way of making machines work for us. But, the truth is that point-and-click, imperative scripting and remote execution do not scale very well when you are trying to govern the behavior of a large number of things.

IT installations grow to massive size in data centers, and the idea of remote command and control, by an external manager, struggles to keep pace, because it is an essentially manual human-centric activity. Thankfully, a simple way out of this dilemma was proposed in 2005 and has acquired a growing band of disciples in computing and networking. This involves the harnessing of autonomous distributed agents.

From Imposing to Obtaining

Imagine that we want to implement a software design or enact a business process. For cultural reasons, our first thought is to want to issue a list of instructions to all the parts involved to carry out some kind of an algorithm, but that might not be the best choice at scale. An algorithm is just a story we tell ourselves about how we envisage the process in our minds. Often we try to tell the same story to a computer or to a team of people, in the form of a flow chart or scripted language. This is one way of describing, building or processing the design. The algorithm might involve a lot of detail and take the form of step-by-step instructions, but other approaches use parallelism or non-linear stories to lay out the job more efficiently. In either case, the instructor assumes that all of the parts involved simply will do exactly as they are instructed. Alas, this is not necessarily true. Remote parts of a system might be unable to comply with these instructions, or even be unwilling to comply, depending on their nature and their circumstances.

This command and control model is called an obligation model in computer science. It has many problems. One of those problems is that it separates intent from implementation, creating uncertainty of outcome. The place where the instructions are determined is not the place where they must be carried out, so the instructor does not have the information about conditions where the local work needs to take place. If two different instructors start issuing conflicting instructions to the same set of parts, there would start a conflict, which no part of the system would be able to resolve, because none of the parts would have the information in one place: intent is not local to the point of action. The instructors might not even be aware of each other to resolve the conflict.

Luckily, there is a complementary

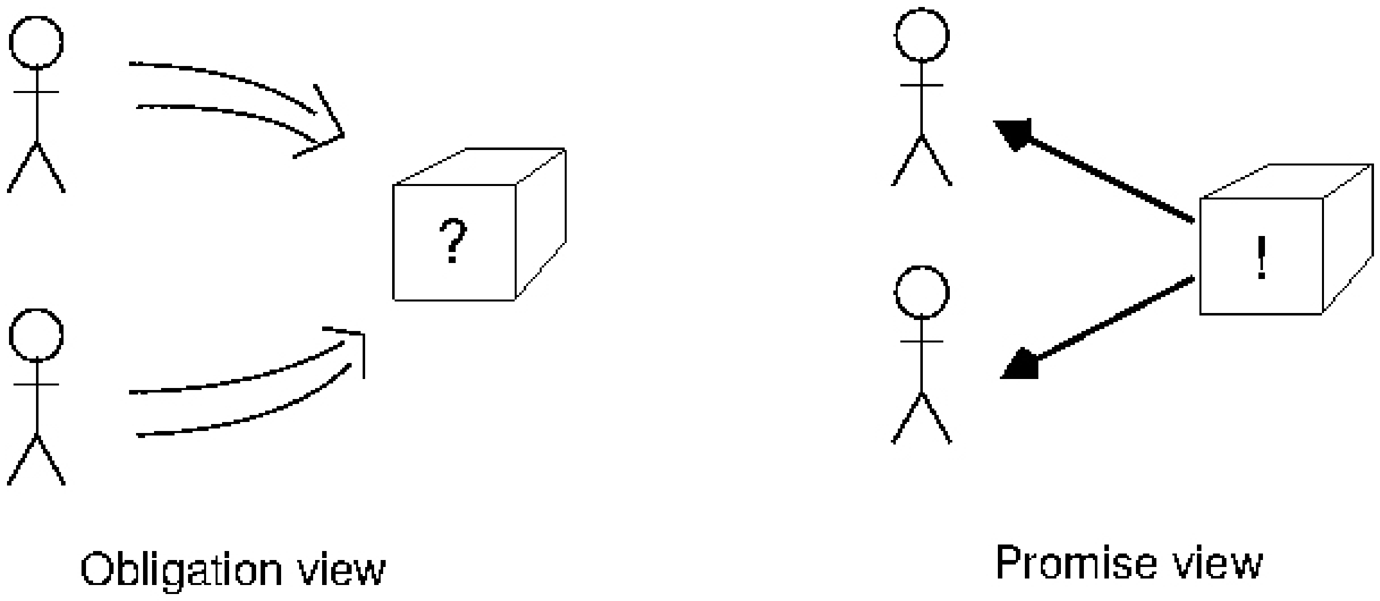


Figure 1. Obligation Model vs. Promise Theory

approach to looking at design that fixes these deficiencies, not in terms of obligations, but in terms of promises.

In a promise-based design, each part behaves only according to the promises it makes to others. Instead of instructions from without, we have behavior promised from within. Since the promises are made by "self" (human self or machine self), it means that the decision is always made with knowledge of the same circumstances under which implementation will take place. Moreover, if two promises conflict with one another, the agent has complete information about those circumstances and conflicting intentions to be able to resolve them without having to ask for external help.

A promise-oriented view is somewhat like a service view. Instead of trying to remote-control things with strings and levers, one makes use of an ecosystem of promised services that advertise intent and offer a basic level of certainty about how they will behave. Promises are about expectation management, and knowing the services and their properties that will help us to compose a working system. It doesn't matter here how we get components in a system to make the kinds of promises we need—that is a separate problem.

Electronics are built in this way, as is plumbing and other commoditized construction methods. You buy components (from a suitable supplier) that promise certain properties (resistance, capacitance, voltage-current relationships), and you combine them based on those expectations into a circuit that keeps a greater promise (like being a radio transmitter or a computer).

Example—CSS

To offer an example of a promise-oriented language, think of HTML and cascading style sheets on the Web. One can break up a Web page into labeled and tagged objects like titles and paragraphs, tables and images and so on. Each of these may be tagged with a specific identity, simply a type (like title, paragraph, link and so on). In a Promise Theory model, these regions could be viewed as "autonomous agents", which through CSS, make promises about what color, font and spacing they will use to render their content initially and over time. The HTML file has regions of the form:

```
<h1>Title...</h1>
<p>Text....</p>
```

Although these are regions of a text file, this is not substantially different from files on a disk. It is just a collection of containers. The CSS promises look like this:

```
h1.main {color: red; font-size: 12px; }
p.main  {text-align:justify;}
```

That is, a header h1 in section "main" would be red and use 12-point text. Compare this to files. The configuration tool CFEngine for building and maintaining computers, allows us to keep promises about what attributes system resources will have. For example, instead of colors and fonts, files can promise to have certain permissions, content and names. One would write:

```
files:
 debian::
   "/etc/passwd"
      perms => mo("root", "644");
```

The language looks superficially different, but it basically is the same kind of declarative association between patterns of objects and what they promise. The promise a region makes itself is the one it will render for all time, until the promise has changed. So this is not a fire-and-forget push-button command, but rather the description of a state to be continuously maintained. In CFEngine style, we could write this alternative form of the HTML style sheet:

```
html:
 main::
  "h1"
    font_size => "12px",
        color => "red";
  "p"
    text_align => "justify";
```

From Promise Theory, we see that these patterns are basically the same; thus, one could say that CFEngine is a kind of "style sheet for servers", in this sense.

Composition of Promises

Promise Theory deals with how to think, in this way, about a much wider range of problems. It was proposed (by myself) in 2005 as a way to formalize how the UNIX configuration engine CFEngine intuitively addressed the problem of managing distributed infrastructure. Such formal models are important in computer science to prove correctness. It has since been developed by myself and Jan Bergstra, and it is being adopted by an increasing number of others.

This complementary non-command way of thinking seems unnatural to us in the context of infrastructure, but more usual in the context of Web pages. Its strengths are that it applies equally to human and machine systems, and that it enforces a discipline of documenting the information necessary and sufficient to implement a design as promises. It is literally actionable (executable) documentation. One can easily convert to a complementary promise view by changing:

```
"YOU MUST (NOT)..." ---> "I PROMISE TO (NOT) ..."
```

A side effect of documenting the necessary and sufficient conditions for a purpose is that one sees all of the possible failure modes of a design enumerated as promises: "what if that promise is not kept?" Autonomy guarantees no hidden assumptions.

The main challenge faced in this view is how to see the desired effect emerge from these promises. What story do we tell about the system? In imperative programming languages, the linear story is the code itself. However, in a promise language, the human story is only implicit in a set of enabling promises. We have to tell the story of what happened differently. For some, this is a difficult transition to make, in the noble tradition of Prolog and Lisp and other functional languages. However, at scale and complexity, human stories are already so hard to tell that the promise approach becomes necessary.

Configuration Management

The way we view the world in Promise Theory is as a collection of agents or nested containers of things that can keep promises. These containers could be:

- A part of a file.
- A file.
- A directory (of files).
- A partition (of directories).
- A container (or logical partition).
- A virtual machine (or logical machine).
- A local area network (of machines), etc.

The reference software with this kind of thinking is still CFEngine (version 3), as it was designed explicitly in this model, though several software systems work in an implicitly promise-oriented fashion. Promise thinking permeates the world of networking too, where autonomy is built into the modus operandi of the devices. Other examples where promises are made to users include interfaces and APIs, microservices or Service Oriented Architecture (SOA).

Knowledge-Oriented Service Design

Promises turn design and configuration into a form of knowledge management, by shifting the attention away from what changes (or which algorithms to enact), onto what interfaces exist between components and what promises they keep and why. The service-oriented style of programming, made famous by Amazon and Netflix, uses this approach for scalability (not only machine scalability but for scaling human ownership). It is hailed as a cloud-friendly approach to designing systems, but Promise Theory also tells us that it is the route to very large scale. Applications have to be extensible by cooperation (sometimes called horizontal scaling through parallelism rather than vertical scaling through brute force). Databases like Cassandra illustrate how to deal with the issues of scale, redundancy and relativity.

Perhaps interestingly, biology has selected redundant services as its model for scaling tissue-based organisms. This offers a strong clue that we are on the right track. Avoiding strong dependencies is a way to avoid bottlenecks, so this shows the route to scalability.

Autonomy and standalone thinking seem to fly in the face of what we normally learn about programming—that is, to share resources, but this is not necessarily true. Security and scalability both thrive under autonomy, and complexity melts away when the dependencies between parts are removed and all control is from within. The near future world of mobile and embedded devices will not tolerate a remote-control model of management, and the sprawl of the cloud's server back end will force us to decentralize 3-D-printed micro-data centers, like the electricity substations of today. In this world, I think we can expect to see more of the autonomous thinking pervading society.■

Mark Burgess is the founder, CTO and original author of CFEngine. Formerly Professor of Network and System Administration at Oslo University College, he led the way in theory and practice of automation and policy-based management for 20 years. In the 1990s, he underlined the importance of idempotent, autonomous desired state management ("convergence") and formalised cooperative systems in the 2000s ("promise theory"). He is the author of numerous books and papers on Network and System Administration.

Send comments or feedback via http://www.linuxjournal.com/contact or to ljeditor@linuxjournal.com.

Integrating Trac, Jenkins and Cobbler—Customizing Linux Operating Systems for Organizational Needs

Developing packages for distributions has been entrusted to distribution maintainers for many years. However, organizations can support their own custom modifications to their favorite distribution. Using methods and tools from the distribution and the broader Open Source community, EMSL (Environmental Molecular Sciences Laboratory) has developed a process to build, develop and validate creation of RPMs to be included with the Scientific Linux distribution deployed on HPC systems. This process helps EMSL meet the HPC users' needs with rapid turnaround while managing communication between end user and package development.

DAVID BROWN

Organizations supporting Linux operating systems commonly have a need to build customized software to add or replace packages on production systems. This need comes from timing and policy differences between customers and the upstream distribution maintainers. In practice, bugs and security concerns reported by customers will be prioritized to appropriate levels for the distribution maintainers who are trying to support all their customers. This means that customers often need to support patches to fill the gap, especially for unique needs, until distribution maintainers resolve the bugs. </EndSoapBox>

Customers who desire to fill the support gap internally should choose tools that the distribution maintainers use to build packages whenever possible. However, third-party software packages often present challenges to integrate them into the distribution properly. Often these packages do not follow packaging guidelines and, as a result, do not support all distribution configurations or procedures for administration. These packages often require more generic processes to resolve the improper packaging. </EndSoapBoxAgain>

From this point on, the tools and methods discussed in this article are specific to Red Hat Enterprise Linux (RHEL). These tools and methods also work with derivative distributions like Scientific Linux or Community Enterprise OS (CentOS). Some of the tools do include support for distributions based on Debian. However, specifics on implementation of the process focus on integration with RHEL-based systems.

The build phase of the process (described in "A Process for Managing and Customizing HPC Operating Systems" in the April 2014 issue of *LJ*) requires three pieces of software that can be filled by Trac (**http://trac.edgewall.org**), Cobbler (**http://www.cobblerd.org**) and Jenkins (**http://www.jenkins-ci.org**). However, these pieces of software do not fill all the gaps present from downloading source code to creation of the overlay repository. Further tools and processes are gained by analysis of the upstream distribution's package management process and guidelines.

The application of the Fedora Packaging Guidelines (**https://fedoraproject.org/wiki/Packaging:Guidelines**) and its counterpart EPEL Packaging Guidelines (**https://fedoraproject.org/wiki/EPEL:Packaging**) are good references for how to package software for RHEL-based systems appropriately. These guidelines call out specifics that often are overlooked by first-time packagers. Also, tools used in the process, such as Mock (**https://fedoraproject.org/wiki/Projects/Mock**), work well with the software mentioned previously.

Fedora uses other tools to manage building packages and repositories. These tools are very specific to Fedora packaging needs and are not general enough for use in our organization. This is primarily due to technical reasons and features that I go into in the Jenkins section of this article.

The rest of this article focuses on implementing Trac, Cobbler, Jenkins, and the gaps between the three systems. Some of the gaps are filled using native plugins associated with the three systems. However, others are left to be implemented using scripts and processes requiring human interactions. There are points where human interaction is required to facilitate communication between groups, and other points are where the process is missing a well implemented piece of software. I discuss setup, configuration and integration of Trac, Cobbler and Jenkins, along with some requests for community support.

Trac

Trac consists of an issue-tracking system and wiki environment to support software development projects. However, Trac also works well for supporting the maintenance of administrative processes and managing change on production systems. I'm going to discuss the mapping to apply a software development process to the process by which one administers a production system.

I realize that talking about issue tracking and wiki software is a religious topic for some. Everyone has their favorite software, and these two kinds of systems have more than enough open-source options out there from which people can choose. I want to focus on the features that we have found useful at EMSL to support our HPC system and how we use them.

The ticket-tracking system works well for managing small changes on production systems. These small changes may include individual critical updates, configuration changes and requests from users. The purpose of these tickets is to record relevant technical information about the changes for administrators as well as management. This helps all stakeholders understand the cost and priority of the change. These small changes can be aggregated into milestones, which correspond to outage dates. This provides a starting framework to track what change happens and when on production systems.

Trac's wiki has features that are required for the process. The first is the ability to maintain a history of changes to individual pages. This is ideal for storing documents and procedures. Another feature is the ability to reference milestones from within pages. This feature is extremely useful, since by entering a single line in the wiki, it displays all tickets associated with the milestone in one simple line. These two features help maintain the procedures and outage pages in the wiki.

The administrative procedures are documented in the wiki, and they include but are not limited to software configuration, startup, shutdown and re-install. The time required to perform these administrative procedures also should be noted in the page. We also make sure to use the plain-text options for specifying commands that need to be run, as other fonts may confuse readers. In many cases, we have specified the specific command to run in these procedures. For complex systems, creating multiple pages for a particular procedure is prudent. However, cross links between pages should be added to note when one part of the procedure from each page should be followed.

Trac's plugin infrastructure does not have plugins to Jenkins or Cobbler. However, what would be the point of a plugin going from Trac to continuous integration or provisioning? Most software development models keep ticket systems limited to human interaction between the issuer of the ticket and the people resolving it. Some exceptions are when tickets are considered resolved but are waiting for integration testing. Automated tests could be triggered by the ticketing system when the ticket's state is changed. However, mapping these sorts of features to administrative procedures for managing production systems do not apply.

Cobbler

Cobbler works well for synchronizing RPM-based repositories and using those repositories to deploy systems. The RPMs are synchronized daily from Jenkins and distribution maintainers. The other important feature is to exclude certain packages from being synchronized locally. These features provide a platform to deploy systems that have specific customized packages for use in the enterprise.

The initial setup for Cobbler is to copy the primary repositories for the distribution of your choice to "repos" in Cobbler. The included repositories from Scientific Linux are the base operating system, fastbugs and security. Other distributions have similar repository configurations (see the Repositories and Locations sidebar). The other repository to include is EPEL, as it contains Mock

Repositories and Locations

- Extra Packages for Enterprise Linux: **http://dl.fedoraproject.org/pub/epel/6/x86_64**
- Scientific Linux 66 Base: **http://ftp1.scientificlinux.org/linux/scientific/6/x86_64/os**
- Scientific Linux 6 Security: **http://ftp1.scientificlinux.org/linux/scientific/6/x86_64/updates/security**
- Scientific Linux 6 Fastbugs: **http://ftp1.scientificlinux.org/linux/scientific/6/x86_64/updates/fastbugs**
- CentOS 6 Base: **http://mirror.centos.org/centos/6/os/x86_64**
- CentOS 6 FastTrack: **http://mirror.centos.org/centos/6/fasttrack/x86_64**
- CentOS 6 Updates: **http://mirror.centos.org/centos/6/updates/x86_64**
- RHEL 6 Server Base: rhel-x86_64-server-6 channel
- RHEL 6 Server FasTrack: rhel-x86_64-server-fastrack-6 channel
- RHEL 6 Server Optional: rhel-x86_64-server-optional-6 channel
- RHEL 6 Server Optional FasTrack: rhel-x86_64-server-optional-fastrack-6 channel
- RHEL 6 Server Supplementary: rhel-x86_64-server-supplementary-6 channel

and other tools used to build RPMs. There are other repositories that individual organizations should look into, although these four repositories are all that is needed.

The daily repositories either are downloaded from the Web on a daily basis or synchronized from the local filesystem. The daily repositories get the "keep updated" flag set, while the test and production repositories do not. For daily repositories that synchronize from a local filesystem, the "breed" should be set to rsync, while daily repositories that synchronize from the Web should set their "breed" to yum. This configuration, through experience, has been chosen because some RPMs do not upgrade well with new kernels nor do they have standard update processes normal to Red Hat or Fedora.

An example of a set of repositories

Jenkins is a very powerful continuous integration tool used in software development. However, from a system administration view, Jenkins is a mutant cron job on steroids.

would be as follows:

- phi-6-x86_64-daily — synchronizes automatically from the local filesystem using rsync once daily.
- epel-6-x86_64-daily — synchronizes automatically from the Web using reposync once daily.
- phi-6-x86_64-test — synchronizes manually from phi-6-x86_64-daily using rsync.
- epel-6-x86_64-test — synchronizes manually from epel-6-x86_64-daily using rsync.
- phi-6-x86_64-prod — synchronizes manually from phi-6-x86_64-test using rsync.
- epel-6-x86_64-prod — synchronizes manually from epel-6-x86_64-test using rsync.

To exclude critical packages from the upstream distribution, the "yum options" flags are set on the daily repository to remove them. For example, to exclude the kernel package from from being synchronized, add `exclude=kernel*`. It's important for administrators to consult both the Cobbler and yum.conf man pages to get the syntax right.

Setting up Cobbler in this way allows administrators to deploy systems using customized critical packages. Cobbler also is used in future phases where the repositories are used to deploy the test and production clusters. The repositories and their relationships are all Cobbler needs to support package building, the test cluster and the production cluster.

Jenkins

Jenkins is a very powerful continuous integration tool used in software development. However, from a system administration view, Jenkins is a mutant cron job on steroids. Jenkins handles periodic source code checkout from source code management (SCM) repositories and

Table 1. Packaging vs. Development

SOFTWARE DEVELOPMENT	RPM PACKAGING
Download source code from SCM.	Download released source, spec file and patches.
Run the build process.	Build the RPMs using Mock.
Run the testing suite.	Validate the RPMs using rpmlint.
Publish test results.	Save validation output for inspection.
Save source code package to repository.	Save built RPMs for later download.
Send notification to pertinent developers.	Send notification to pertinent packagers.

downloading of released source code, via HTTP or FTP. It then runs a series of generic jobs that build, test and deploy the resulting software. These generic interfaces work well for building and distributing RPMs to be included by Cobbler.

The use of Jenkins in a software development role is not all that different from building RPMs (see Table 1 for a comparison of the two processes). The first step in the two processes differs in that (hopefully) the software development code required for the build step is in one place. Package developers need to have, at a minimum, two locations to pull code from to continue with the build. The first location is for patches and spec files, normally kept in an SCM. The second is for released source code packages. Source code is released in a single file and usually in some container format (such as tar, rar or zip). These files do not normally belong in an SCM and are more suited to an S3 (**http://docs.aws.amazon.com/AmazonS3/latest/API/Welcome.html**), swift (**http://docs.openstack.org/api/openstack-object-storage/1.0/content**) or blob store-like interface.

Jenkins is built primarily for downloading code from one and only one SCM. However, you can work around this issue by adding another build step. This means that the SCM plugin is used to download the spec file and patches while the first step in the build process downloads the source code package. After these two steps are done, the source code, patches or spec file can be patched with site-specific customization.

The next step is to build RPMs using Mock. This involves several tasks that can be broken up into various build steps (see the Mock Build in Jenkins sidebar). All these steps are done using the Jenkins execute shell build steps. Some of the Jenkins jobs we use are multi-configuration jobs that

Listing 1. basic-mock-jenkins.sh

```
#!/bin/bash -xe

# keep in mind DIST is defined in multi-configuration axis
MOCK="/usr/bin/mock -r $DIST"
PKG=${JOB_NAME##*/}
# keep in mind VER could also be a multi-configuration axis
VER=${VER:-1.0}
# if you are ripping apart an RPM might have this one too
REL=${REL:-4.el6}

OUT=$PWD/output

wget -O $PKG-$VER.tar.gz
 ➥http://www.example.com/sources/$PKG-$VER.tar.gz
rm -f $OUT/*.src.rpm
if ! $MOCK --resultdir=$OUT --buildsrpm --spec=$PKG.spec
 ➥--sources=$PWD
then
    more $OUT/*.log | cat
    exit -1
fi

if ! $MOCK --resultdir=$OUT --rebuild $OUT/*.src.rpm
then
    more $OUT/*.log | cat
    exit -1
fi

rpmlint $OUT/*.rpm > rpmlint.log
```

contain one axis defining the Mock chroot configuration. That chroot configuration should be generated from the daily repositories defined in Cobbler. Following these tasks can get you started on using Mock in Jenkins (Listing 1).

Mock Build in Jenkins

1. Prepare the source and specs.
2. Run Mock source rpm build.
3. Run Mock rpm build.
4. Run rpm validation.

The number of warnings and errors should be tracked, counted and graphed over a series of builds. This gives a good indication whether bugs are being resolved or introduced over time.

Once the RPMs are built, it's important to run `rpmlint` on the resulting RPMs. This output gives useful advice for how to package RPMs properly for the targeted platform. This output should be handled like any other static code analysis tool. The number of warnings and errors should be tracked, counted and graphed over a series of builds. This gives a good indication whether bugs are being resolved or introduced over time.

The generated RPMs and `rpmlint` output need to be archived for future use. The archive artifacts plugin works well for capturing these files. There also is an artifact deployer plugin that can copy the artifacts to directories that Cobbler can be configured to synchronize from for its part of the process.

There is some room for improvement in this process, and I outline that in the conclusion. However, this is the basic framework to start using Jenkins to build RPMs using Mock and rpmlint. This part of the process needs constant care and attention as new updates are pushed by the distribution and package developers. Jenkins does have plugins to Trac and other issue-tracking systems. However, they are not

Table 2. Software

ROLE	SOFTWARE CHOICE
Continuous Integration	Jenkins
Repository Management	Cobbler
Provisioning	Cobbler
Ticket Tracking	Trac
Wiki	Trac
Package Building	Mock
Package Guidelines	Fedora Packaging Guidelines

included in this process, as we find e-mail to be a sufficient means of communication. The outlined process for building RPMs using Jenkins helps us track the hacks we use to manipulate important packages for our systems.

Conclusion

I have discussed a method for setting up tools to develop RPMs against a custom distribution managed by Cobbler. Along with Trac, package developers can maintain updated RPMs of critical applications while managing communication. However, this process is not without gaps. First, I'll go over the gaps present in Jenkins, discussing core and plugin gaps that were not found. Then I'll discuss the gaps in Cobbler regarding repository management. These two systems are lacking in integration, although that can be worked around.

MultiSCM is a functionality in Jenkins that would simplify the package building process. There is a MultiSCM plugin (**https://wiki.jenkins-ci.org/display/JENKINS/Multiple+SCMs+Plugin**); however, it is advertised as a proof-of-concept code. The hope is that the radio button selection for SCM would turn into a set of check boxes. There are related bugs, but they have not seen traction in years. Package development is another good example of the need to download and poll for updates on code from multiple places.

Here are links to information on the Jenkins Multiple SCMs Bugs:

- **https://issues.jenkins-ci.org/browse/JENKINS-7192**
- **https://issues.jenkins-ci.org/browse/JENKINS-9720**

Static code analysis tools are available as plugins for Jenkins (**https://wiki.jenkins-ci.org/display/JENKINS/Violations**), although these plugins do not include rpmlint. These plugins create graphs to track the number of warnings and errors in code over time. To perform the same task for packaging would be very helpful. However, you can work around this gap by using the generic plot plugin (**https://wiki.jenkins-ci.org/display/JENKINS/Plot+Plugin**) and another build step for each job.

Mock has a very well defined interface and workflow. A generic plugin to use Mock in Jenkins would be very useful. The plugin should include configuring the chroot configuration. Two kinds of build jobs also could be created, one using spec and source files, the other using source RPMs. A test also would need to be created to verify that Mock can be run without prompting for a user password. This plugin would be very helpful for automating this process, as we currently have to copy scripts between jobs.

There are some additions to Cobbler that would be useful for this process as well. There are no per-repo triggers. The ability to tell Trac that packages went from repo test to repo prod would be useful. Furthermore, the ability to tell Jenkins to build a package because a dependent package updated also would be useful.

The other useful addition to Cobbler would be the ability to remove older RPMs in the destination tree while synchronizing from the remote mirror. Cobbler repositories, if the "breed" is yum, build up in an append-only fashion. Processes for managing the space may be run periodically by removing the RPMs and then synchronizing the repository again. However, this leaves the repository in a broken state until the process is complete. This feature could be useful in any Cobbler deployment, as it would make sure repositories do not continue to take up space when RPMs are not needed.

Trac does not need any additional plugins to integrate better with Cobbler or Jenkins. We have found some usability issues with manipulating large tables in the wiki format. Some plugin to make editing large tables easier in the wiki format would be useful for us. Also, editing long pages becomes an issue if you cannot put comments throughout the page. We validate our procedures by having members of the group who are unfamiliar with the system read through the procedure. The reader should be able to comment on but not edit parts of the page. We have worked around or found plugins on the Trac Hacks page (**http://www.trac-hacks.org**) to resolve these issues.

The final request is for some level of certification from distribution maintainers to certify third-party packages. Many of the third-party packages we have applied to this process to do not support all distribution configurations. A certification from distribution maintainers validating that software distributed by third-party vendors have packaged their software appropriately for the distribution would help customers determine the cost of support.

This is by no means a complete solution for organizations to build customized critical applications.

There are still gaps in the system that we have to work around using scripts or manual intervention. We constantly are working on the process and tools to make them better, so any suggestions to improve it are welcome. However, these tools do fill the need to support customization of critical applications for HPC at EMSL.

Acknowledgement

The research was performed using EMSL, a national scientific user facility sponsored by the Department of Energy's Office of Biological and Environmental Research and located at Pacific Northwest National Laboratory.■

David Brown is a high-performance computing system administrator with a B.S. in Computer Science from Washington State University. He has worked at the Pacific Northwest National Laboratory (PNNL) in the Environmental and Molecular Sciences Laboratory (EMSL) since January 2007. He also is a Fedora Package Maintainer and supports several scientific and administrative packages that are used in HPC environments. He has experience in high-performance filesystems (Lustre) and cloud technology (OpenStack).

Send comments or feedback via http://www.linuxjournal.com/contact or to ljeditor@linuxjournal.com.

DOC SEARLS

Can We Stop Playing Card Games with Business?

Credit cards cause itches for all of us. Let's scratch them.

A friend who works in one of the big banks recently told me that any new-fangled approach to identity and payments is going to have a hard time getting traction while credit cards continue to work as well as they do. "Using credit cards is too easy, too normal, too entrenched in Business As Usual", he said. They used to say the same thing about Windows.

As it happens, I am in the middle of a credit-card mess right now that is certainly normal and Business As Usual, but far from easy.

See, all I'm trying to do is buy something on-line, from a retailer that's not Amazon. (I'm not going to name names, beyond saying that much.) First, the retailer's system for sorting out billing and shipping addresses was broken. Then, after I got over that hurdle (which I suppose is still there), my credit card was declined. In the course of dealing with both failures, I have pounded countless numbers and letters, over and over again, into my laptop's keyboard and my phone's screen. On top of that, I have had to give the last four digits of my social security number, plus a bunch of other information, over and over again, to too many different people at the credit-card company's call centers.

The credit-card company doesn't

know (or seem to care) why my card was declined. The attempted purchase was not unusual in any way. The error message from the retailer was also unhelpful. It said, "Something went wrong when trying to process your payment. Please review the error message and try again: Braintree payment for Order #5283 (Transaction ID gb3xx6m) was declined: Declined." For the last several hours, the credit-card company also has been unable to turn off the fraud protection flag that caused it to decline the purchase in the first place, because its system for doing that is down.

Several more hours have passed between the last paragraph and this one, which I'm writing after calling the credit-card company for the Nth time and learning at last that the payment went through. The retailer's system, however, still says "awaiting payment".

I don't know if any of the new ideas for solving this kind of mess are going to succeed. Most of them probably won't. But I do believe the only answers that will finally work are ones that give us ways to be known in the virtual world that resemble the ways we are known in the physical one.

For a base-level model of that, let's say I go into coffee shop and order a cortado (**http://en.wikipedia.org/wiki/Cortado**). A cortado has the ideal ratio of coffee and steamed milk. It's also known as a piccolo latte in some places, such as Australia (**http://www.cafeculture.com/general-interest/what-is-a-piccolo-latte**). Try it out. Here is what happens around identity and payment:

1. I am anonymous—literally, nameless. But...

2. They can see I am a person, with an obvious ability to pay.

3. They accept payment either via cash or credit card.

4. If I use cash, it's also anonymous, even though there is a unique number on every piece of paper currency. The shop is not interested in those numbers.

5. If I use a card, they do not care about the name on the card or the number. All they care about is whether the payment goes through. In other words, they are not burdened with any details about me. (I am purposely keeping loyalty cards and other marketing gimmicks off the table here. I'll explain why below.)

6. If they ask for a name, it's only so

they can call it out when the drink is ready. Not for any other reason.

7. That's it.

Now, let's say this is a coffee shop where I am already known to the people behind the counter. Here the only difference is that I might be extended some friendly courtesies. If the people are honorable, I can expect that they won't be out to screw me—for example, by giving or selling information about me to people or systems I don't know.

There are permissions and agreements, casual and formal, on both sides of this system, and they have been in the vernacular of every market since the dawn of commerce. When this system works well, there is also no power asymmetry—or at least not one being exploited. The customer and the shop meet as equals in the marketplace. They trust each other. This trust has a high degree of leverage toward future business between both parties. In terms of game theory (**http://www.beyondintractability.org/essay/prisoners-dilemma**), that leverage is toward positive sum (aka win-win) outcomes (**http://www.beyondintractability.org/essay/sum**).

On the whole, we have positive sum outcomes from the game played with credit cards. But there is a degree of entrapment to that game as well. We are not equals, and we do not enjoy full agency.

This is not the fault of the credit-card system, but of mass marketing's obsolete methods and assumptions, which date back to the time when Industry won the Industrial Revolution. In order to get scale—to market to masses—it was necessary for companies to treat many customers the same way, as populations rather than as individual human beings. Even in the Internet Age, attempts by business to "personalize" and "mass customize" relations with customers are made by business—not by the customer.

And, because business mostly talks only to itself about this stuff, it is easy for business to rationalize negative externalities (**http://economics.fundamentalfinance.com/negative-externality.php**). Among those externalities are:

1. Wastes heaped by others—for example, all the promotional crap that fills communication channels and brings no positive response (in other words, 99.x% of all promotional messages).

2. Inconveniences—such as the need by users and customers to authenticate in separate and arcane ways with every different company's different system.

3. Resentments and other bad feelings by users and customers toward both of the above, plus knowledge that they are being spied upon and manipulated in ways they cannot escape without giving up relationships they can't or won't do without.

4. Companies needing to maintain or hire the large and expensive server farms and "big data" crunching systems to process and make decisions on personal data gained by surveillance and fed back to #1, above.

All those negative externalities afflict the commercial Web today—and are spreading into brick-and-mortar retailing as well. We see this with loyalty programs (**http://en.wikipedia.org/wiki/Loyalty_program**) that give us additional cards and key tags to carry around.

These externalities exist to a far lesser degree with credit cards than they do with marketing gimmicks (because credit cards' purposes are

Advertiser Index

Thank you as always for supporting our advertisers by buying their products!

ADVERTISER	URL	PAGE #
Chef	http://getchef.com	2, 37
Drupalize.me	http://www.drupalize.me	114
EmperorLinux	http://www.emperorlinux.com	41
Fossetcon 2014	http://fossetcon.org	113
High Performance Computing Shows 2014	http://www.flaggmgmt.com/hpc/	7
Silicon Mechanics	http://www.siliconmechanics.com	3
SPTechCon	http://www.sptechcon.com	43

ATTENTION ADVERTISERS

The *Linux Journal* brand's following has grown to a monthly readership nearly one million strong. Encompassing the magazine, Web site, newsletters and much more, *Linux Journal* offers the ideal content environment to help you reach your marketing objectives. For more information, please visit http://www.linuxjournal.com/advertising.

mostly restricted to transactions), but are still present with credit cards, as we see with my own story, above.

Einstein (is said to have) said "The significant problems we face cannot be solved at the same level of thinking we were at when we created them" (**http://en.wikiquote.org/wiki/Albert_Einstein**). I can remember like it was yesterday the feeling of futility I had when I was doing work for Sun Microsystems, back in the 1980s, enduring the torture of sitting in meetings where the pointless topic was "fixing" UNIX, which had a zillion variants, most of which were owned by somebody, including Sun. There was no way UNIX's problems could be solved at that level. We needed Gnu and Linux to solve it. (And yes, the BSDs too.) That's how I feel today about trying to fix identity and payments inside the current mass-marketing system.

The level at which we need to solve identity and payments is the individual one. It needs to happen on our side. We're the ones who need to manage our identities, how we pay for stuff, what information we hand over when we pay for things, and how we control our side of relations with the retailers and service providers of the world.

Today there are already many new projects that start with the customer and move toward business. I'm involved in many of those, through ProjectVRM (**http://blogs.law.harvard.edu/vrm**). (Here is a partial list of developers: **http://cyber.law.harvard.edu/projectvrm/VRM_Development_Work**.) And, I consult some of those formally as well. But I won't name any, because I want clean new Linux geek thinking and code-writing applied to this problem.

Here's the main challenge: not using cards at all. Bitcoin (**http://en.wikipedia.org/wiki/Bitcoin**) is the biggest breakthrough in that direction, so far—or at least the one with the most geek interest.

What else have we got—or could we have? The answers will come, because there's a huge itch here.

So tell me who's scratching what, and how. I'd rather salute those than complain more about Business As Usual. ■

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.

Send comments or feedback via http://www.linuxjournal.com/contact or to ljeditor@linuxjournal.com.

FOSSETCON

FREE AND OPEN SOURCE SOFTWARE EXPO AND TECHNOLOGY CONFERENCE

FOSSETCON 2014

Come out and participate in the First Annual Fossetcon 2014 Florida's Only Free and Open Source Conference. With in 10 minutes of Disney Land, Universal Studios and Epcot Center.

DAY 0 FOOD, TRAINING, WORKSHOPS AND CERTIFICATIONS

DAY 1 FOOD, KEYNOTES, EXPO HALL, SPEAKER TRACKS

DAY 2 FOOD, KEYNOTES, EXPO HALL, SPEAKER TRACKS

BSD *Friendly*

FREE FOOD, TRAINING, CERTIFICATIONS AND GIVEAWAYS!!!

SEPT 11 - SEPT 13

ROSEN PLAZA HOTEL ORLANDO, FL

Fossetcon 2014: The Gateway To The Open Source Community

powered by:

CITRIX Open@Citrix

ORACLE

MySQL

Omnibond OrangeFS

ubuntu Supported by Canonical

NVIDIA

conservancy

More info at
www.fossetcon.org

drupalize.me

Instant Access to Premium Online Drupal Training

- *Instant access to hundreds of hours of Drupal training with new videos added every week!*
- *Learn from industry experts with real world experience building high profile sites*
- *Learn on the go wherever you are with apps for iOS, Android & Roku*
- *We also offer group accounts. Give your whole team access at a discounted rate!*

Learn about our latest video releases and offers first by following us on Facebook and Twitter (@drupalizeme)!

Go to http://drupalize.me and get Drupalized today!

```
 *
 * Usage: $(obj).insertAtCursor(text);
 *   obj = a textarea or textfield
 *   text = a string to insert
 */
$.fn.extend({
  insertAtCaret: function(myValue){
    return $(this).each(function(){
      // If target element is hidden, don't do it.
      if ($(this).is(':hidden') || $(this).parents(':hidde
        return;
      }
      if (document.selection) {
        this.fo
        sel = c
        sel.tex
        this.fo
      }
      else if (
        var sta
        var end
        var scr
        this.va
        this.fo
        // High
```