flossUK

news@UK

The newsletter of FLOSS UK, the new name for the UK's oldest Open Systems User Group, UKUUG

Published electronically at `http://www.flossuk.org/Newsletter`

Volume 24, Number 4 **ISSN 0965-9412** **December 2015**

Contents

From the Secretariat

Jane Morrison

The minutes from the UKUUG Ltd. Annual General Meeting held on 23rd September have once again not been circulated to members via hard copy. All members were emailed on 8th October and advised that the minutes and associated documents could be found on the web site at: **`http://www.flossuk.org/Events/AGM2015`**

Of course anyone who wants a printed copy should contact me here at the office.

Current Council members are: Kimball Johnson, Ian Norton, Mark Keating, Quentin Wright, Gavin Atkinson and Kenny MacDonald

We thank retired Council member, Tim Fletcher for his past commitment.

We are currently putting in place a full schedule of events for 2016 - to date the following have been agreed:

Perl Tutorials - 8th, 9th, 10th, 11th & 12th February - see page 5 for more details

Un-conference - possibly London - Saturday 13th February

DEVOPS Spring 2016 - 15th, 16th & 17th March

- Tuesday 15th March - Workshops
- Wednesday 16th & Thursday 17th - Conference - see call for papers on page 6

This event is the UK's only conference aimed specifically at systems and network administrators. It always attracts a large number of professionals from sites of all shapes and sizes. As well as technical talks, the conference provides a friendly environment for delegates to meet, learn, and enjoy lively debate on a host of talks.

Bookings will open mid December, for more information see:

`http://www.flossuk.org/Events/Spring2016`

BarCamp - Birmingham - June

Dynamic Languages Conference - Autumn - Manchester

The annual membership subscription invoices will be sent out in January, please look out for your invoice and as always prompt payment will be gratefully received!

We would like to thank the continued support of our Sponsor members, SUSE and 2nd Quadrant. Their involvement does help us to keep event fees down.

I would like to take this opportunity to wish you all a very Happy Christmas and a Peaceful New Year. The Secretariat will close on Thursday 17th December and re-open on Monday January 4th 2016.

As you have already been notified in the September Newsletter for 2016 Council have decided to print just one issue Newsletter per year (around Spring) - future news, book reviews, articles etc. will be on our web site.

Chairman's Report

Kimball Johnson

Sponsor Members

I would like to take the opportunity to thank our two silver sponsors, 2ndQuadrant and SUSE. Their generous support helps us to put on the free events throughout the year.

Events

The Summer is always a quiet time for events, but in mid-November we hosted the annual LDAPCon conference in Edinburgh. The event was a success, and I would like to thank Kenny MacDonald for his assistance during the event. I hope all those that attended enjoyed the conference.

Coming up in Feburary we are running our popular Perl Training with Dave Cross once again. Dave has reworked the program and there are a series of 4 courses: 3 one day and 1 two day course, that can be taken individually or as a complete one week course. More details are on our website.

Furthermore the Spring Conference returns in March, this time hosted at Mary Ward House in London. Mandi Walls from Chef Software will be giving the opening Key Note, and limited ticket sales have opened. We also still have the call for papers open, and would encourage members to submit a talk on anything related to Free and Open systems. More details on the website.

We are always looking for ideas for tutorials to run for our members, so if you have any topic you wish to hear about, please contact me at **kimball.johnson@flossuk.org** or Jane on **office@flossuk.org**.

Support for local user groups

The budget is still available for FLOSS UK to assist local user groups by helping them to obtain speakers for their events and assist with travel expenses. In addition we have a small budget for assisting with projects that would benefit the Free Software or Free Hardware communities in some way.

If you have an idea and wish some support, please contact Jane on **office@flossuk.org** and it will be discussed by Council.

From the Editor

Paul Waring

As reported in the September newsletter, FLOSS UK will be moving to an annual printed newsletter – produced shortly after the Spring conference in March – supplemented by book reviews and event reports which will be published on the website throughout the year. This is therefore the last quarterly black and white newsletter which will be produced and sent to members. Archives of old newsletters will remain available on **www.flossuk.org** (from 2010) and **www.ukuug.org** (from 1992).

If you have any ideas or comments on the proposed newsletter changes, please do get in touch via email: **newsletter@flossuk.org**.

Perl Tutorials

Perl Tutorials - February 2016

Day 1 - Monday 8th February - Object Oriented Programming with Perl and Moose

Day 2 - Tuesday 9th February - Database Programming with Perl and DBIx::Class

Day 3 - Wednesday 10th February - An Introduction to Testing Perl Programs

Day 4 & 5 - Thursday 11th & Friday 12th February - Modern Web Programming with Perl

Monday 8th February - Object Oriented Programming with Perl and Moose

Moose is fast becoming the standard way to write object oriented code in Perl. Moose is powerful and flexible. It makes writing object oriented Perl far easier than it has ever been before.

This one-day training course explains what Moose is and how it will help you write better object oriented Perl. It is aimed at programmers who know Perl but who might not have kept up to date with some of the most recent Perl tools.

The course includes practical sessions where the attendees will have the chance to practise the skills that they have been taught.

By the end of the day, attendees will be understand the power of Moose and will be able to use it to build complex object systems.

Tuesday 9th February - Database Programming with Perl and DBIx::Class

Most applications need to store persistent data in some form. And many of those applications will use a database as that storage. For many years Perl has had a powerful database interface layer called DBI, but your Perl database programming will be even simpler if you use DBIx::Class on top of DBI.

This one-day course introduces DBIx::Class and explains how it can help you get your job done more quickly.

The course includes practical sessions where the attendees will have the chance to practise the skills that they have been taught.

By the end of the day, attendees will have a good understanding of the power of DBIx::Class and will be able to use it to write Perl programs that interact with databases.

Wednesday 10th February - An Introduction to Testing Perl Programs

For a very long time, Perl has had a strong testing culture. It is expected that CPAN modules will come with a comprehensive test suite which is run as part of the installation process. Many tools and frameworks are available to make testing Perl code as easy as possible.

This one-day course introduces various techniques and tools for testing Perl programs. It also shows attendees how to write their own testing tools or extend existing ones.

The course includes practical sessions where the attendees will have the chance to practise the skills that they have been taught.

By the end of the day, attendees will have a good understanding of the power of various Perl testing tools and will know how to use them to make their Perl code more robust.

Thursday & Friday - 11th & 12th February - Modern Web Programming with Perl

Perl first became really popular when it was used to write dynamic web pages in the 1990s. Perl still has many flexible and powerful tools which can be used to write modern web applications, but they have moved on a long way since the CGI programs of the last millennium.

This two-day course introduces PSGI the current standard for Perl web development and covers modern web frameworks like Dancer and Catalyst which build on top of PSGI and make it easier than ever to write web applications using Perl.

The course includes practical sessions where the attendees will have the chance to practise the skills that they have been taught.

By the end of the day, attendees will have a good understanding of various modern Perl web programming frameworks and will be able to write web applications which make use of them.

Delegate fees and on-line booking can be found on the web site at:

`http://www.flossuk.org/Events/Perl2016`

Early booking is essential!

Spring 2016 Call for Papers

The FLOSS UK Council are now accepting submissions for presentations and workshops for our main annual conference event. In recent years we have evolved into the UK's primary support organisation for Free/Libre Open Source Systems (FLOSS) supporting Free and Open Technology.

The Spring conference is the UK's longest running event supporting systems and network administrators. But as FLOSS UK has evolved to support open networks, software, hardware and data so our primary conference has become open to submissions on these areas.

If you would like to offer a forum or half-day workshop that might interest our target audience, please submit a proposal. If you have a novel solution to a problem, experience of a particular application or hardware platform, tips and tricks for fellow open source delegates, or a favourite tool you could talk about, use of open data, please submit a paper for consideration by the programme committee.

There will be a prize for the best conference talk. Bursaries may be made available for accepted speakers who are unable to afford travel costs, subject to Council approval with all bursaries being fixed in advance. All accepted speakers receive a complimentary conference place and a ticket for the conference dinner.

We are also seeking lightning talks for the event – please contact us if you have something you want to schedule in the programme. Lightning Talk speakers who offer no other presentation are not eligible for free conference places or dinner.

Significant Dates

- Initial closing date for abstracts: 30th December 2015
- Speakers notified by: 20th January 2016
- Conference: 15th-17th March 2016

To submit a paper, please email: **`spring2016@flossuk.org`**

LDAPCon 2015

Andrew Findlay

The fifth International Conference on LDAP, Directory Services and Identity Management: 14 countries represented, 45 people attending tutorials, 75 attending conference sessions, 22 peer-reviewed papers, 7 tutorials, and 8 sponsoring companies. 1600 messages in my email log!

Edinburgh did us proud: the Informatics Forum at the University is a great venue for this sort of event, with flexible spaces for sessions and plenty of circulation space so breaks were productive without being crowded. The food was excellent, and there are lots of hotels within walking distance.

David Goodman's keynote, LDAP 2020: Paradise Lost or Regained? reminded us that LDAP has its roots in the 1980s but is evolving to support future requirements. Christian Hollstein showed that an unfortunate clash between the TCP Nagle Algorithm and higher-level protocols would degrade performance, and proposed new controls to mitigate the problems. Mark Bannister presented DBIS, hoping to clean up a batch of long-standing issues with the LDAP representation of Unix NIS data, and Michael Stroder presented AE-DIR which sets out to provide a NIS-like service on a strict need-to-know basis.

Several papers addressed IAM (Identity and Access Management) and a new alliance was announced to promote the use of open-source IAM components. Peter Gietz described a project using an LDAP data store to record and analyse cemeteries, and Boyd Duffee explained how Google Apps can be driven from an organisation's existing directory service.

Directory services need high availability so data-replication is essential. Damy Mahl and Ludwig Krispenz each described different approaches to the problem. Several papers covered developments in server and database technology, made necessary by increasing demands for speed and by changes in the relative performance of CPU, RAM, networks, and mass-storage. Howard Chu ended the conference on a musical note with a performance on the fiddle before describing performance improvements in OpenLDAP.

All the presentations will soon be available on the conference website:

`http://ldapcon.org/2015/`

Many thanks to our sponsors, to FLOSS UK, and to the School of Informatics at Edinburgh for making it all possible. The conference runs every other year and we will soon be looking for our 2017 venue: somewhere further east perhaps!

Redecentralize.org conference

Irina Bolychevsky

Last month we organised the first Redecentralize.org conference![1] A full weekend, held in London with invited speakers and projects, all scheduled and planned unconference style to get discussions, collaborations and ideas going.

The topic? Technology that shapes society.

The challenge? Alternatives to monopolies. Mass privacy, security and an equal open web which represents the interests of its users.

We had a fantastic venue and breakfast, courtesy of ThoughtWorks along with delicious ethical food and Saturday drinks sponsored by the excellent FLOSS UK and Ethereum folks.

The conference was attended by a really great group of people and mix of projects. Everything from protocols to user friendly self hosting apps and products to community movements to data ownership platforms to standards.

Have a look at the storify of the tweets over the weekend![2]

I was interested to see the progress made in the space of self hosting, data ownership & easy encryption. A number of projects were in this space - I particularly liked the idea of CloudFleet[3], which has yet to launch, and is a small simple device you plug in at home that comes with MailPile[4] (simple web email client with encryption build it) and a calendar & contacts app installed. It automatically generates your private/public keys and handles encryption and syncing / backup between it and your devices.

Cozy[5] have been hard at work since we interviewed them in January last year[6] and now have a comprehensive google import that lets you transfer all your data to your Cozycloud instance (you need a server or hosting provider) and then connect it to bank feeds and other apps. This is a great platform from which to control your own data and see what's possible - although it's currently all unencrypted.

Incidentally, when it comes to self hosting, I recently got myself a Sandstorm[7] instance and have been playing around with installing apps. It's actually pretty user friendly and fun (once you get over needing to run your own server) - but still missing cool apps like shout-irc[8] which is the one I'm really keen to run. And crucially: usable email, calendar & contact list apps which encrypt by default. For most, it's still too hard to move off Google.

Back to the conference: the new cool kids in town are of course Ethereum[9] and IPFS[10]. I'll also add Backfeed[11] to the group and all three are really interesting, have a huge following, potentially work well together and were all represented at the conference.

Ethereum is a platform for building decentralised application that are run by all ethereum nodes. They've invented their own programming language for running general scripts ('smart contracts') on the blockchain and corresponding crypto-currency - ether. They had a great triple bill of talks, covering everything from the historical backdrop by the inimitable Vinay Gupta, a friendly overview by Gavin Wood[12] (who we'd interviewed about Ethereum a year ago![13]) and a great talk from Jutta Steiner on practical applications and Provenance.[14]

A version of all the videos with slides will be coming over the next few days on the redecentralize website and YouTube channel![15]

Backfeed is also blockchain based. It's a protocol for encoding human consensus to get decentralised ownership and governance (decision making, remuneration and collaboration) on the blockchain through reputation. You can think of it as adding a social reputation based layer to the decentralised applications on ethereum.

This creates the potential for massive truly decentralised organisations where hundreds of people take actions and are rewarded financially (with cryptocurrency payment) and with ownership / greater decision making by the actions they take. Contracts on the blockchain allow for decentralised obligations and agreements to be dispensed automatically in a trusted fashion that cannot be gamed or abused. This could eliminate many overhead costs: infrastructure, security, management layers as well as creating true collaboration.

IPFS is a new hypermedia distribution protocol & peer-to-peer distributed file system, which can be used as the storage layer in the model above. It uses content addressing (a cryptographic hash of the content as its address instead of location as in the case of http) which allows for faster downloading (bit-torrent style) of content from numerous peers and permanent versioning of content and data that does not depend on the initial uploader's server / computer to remain available. Ian gave a great overview[16] of the issues with the current http based system and how IPFS resolves these.

My instinct is that to succeed in some kind of decentralisation or distribution of power, we not only need novel capabilities, but network power. We must challenge the walled garden business model by giving users what they want - the ability to choose their application and still be able connect together and keep their network independently of any provider. For that, we need open standards. In addition to creating an equal open playing field that empowers individuals, standards can help smaller market shares and networks pool together to challenge larger monopolies.

At the conference, Matrix[17] participated in both the speed geeking, lightning talks, held a session and wrote a blog post on the event![18] It's designed as an open standard for decentralised communication. They've also built bridges to existing popular apps like slack which you can experience from their pretty new app: vector.im.

SoLID[19] - a standard and conventions for building privacy preserving apps that leave data in control of the user on a linked data platform, is also aiming to provide an open API standard that it hopes will fuel the whole ecosystem. Andrei and Nicola from the team talked about the importance of data ownership[20] and showed off what they've been working on.

In general, I remain somewhat skeptical since I'm really looking for an adoption strategy and usable apps that have shipped. Standards and protocols only work when they have mass adoption. You can enforce this if you hold a majority market share of users, at which point a strategy isn't necessary, but if you don't enjoy such privilege you cannot simply rely on your solution being 'right' - it needs to be simple while providing enough value to get over the switching cost inertia or people's tendency to hack together their own solution.

As a community we still need to devote more focus on user research, marketing, business models, design and product management. Even at protocol or standards levels having a 'killer app' which uses it that really takes off is key. Many of our problems at this stage are social not technical - I've seen so many similar started and abandoned projects. Hundreds of them. Each time a new one begins I wonder: is it just 'not-invented-here' syndrome or perhaps previous solutions never had enough users or brand recognition to make them worth building on? I don't know, but one thing I'm trying to do is make the work of other people a lot more visible and pose these challenges to them.

Lastly, we had lots of great sessions that were not about specific technologies or protocols. We heard from CyberSalon[21] who are developing a digital bill of rights[22], Max with brilliant anecdotes for why you should decentralise and on keeping data for 80 years[23], Francis led a session to discover the best file syncing and backup storage solution[24] (the happiest person was copying files by hand), exploring privacy, compliance and usability[25] by the new non-profit Hi:project[26] and a lovely round table discussion on how to spread important, but unpopular ideas - learning from the vegan movement.[27]

There was more, much more and this is just my snapshot. Watch the #redecentralize tag on twitter, sign up to the newsletter[28] and discussion list[29] and get involved in taking back the net!

[1] `http://redecentralize.org/conference`

[2] http://ow.ly/V3siN

[3] https://cloudfleet.io/

[4] https://www.mailpile.is/

[5] https://cozy.io/

[6] http://ow.ly/V3stO

[7] http://sandstorm.io/

[8] http://shout-irc.com/

[9] https://ethereum.org/

[10] http://ipfs.io/

[11] http://backfeed.cc/

[12] https://youtu.be/1uiwMPabR5o

[13] http://redecentralize.org/interviews/2014/09/23/18-gavin-ethereum.html

[14] https://youtu.be/sNukHAmSbFY

[15] https://www.youtube.com/user/redecentralize

[16] https://youtu.be/TiCUIh7tNtU

[17] http://matrix.org/

[18] http://bit.ly/21flubL

[19] https://github.com/solid

[20] https://youtu.be/yi4SgNyDJ9w

[21] http://www.cybersalon.org/

[22] https://youtu.be/tNAZJ3kVgqY

[23] https://youtu.be/Y8EdwQ_6onE

[24] http://bit.ly/1lJfOWZ

[25] https://youtu.be/JV8znCbdGjY

[26] http://hi-project.org/

[27] https://youtu.be/PrGcXZRirpI

[28] http://t.co/z1zZ16C57Z

[29] https://github.com/redecentralize/swarm/wiki/Email-list

Helping teachers to hack

Les Pounder

As I write this, I am travelling to Leeds to take part in the Raspberry Pi Foundation's Picademy[1] training scheme. Created by Carrie Anne Philbin, Picademy is a two day training course for teachers of all abilities who wish to incorporate the Raspberry Pi into their scheme of work. It covers the use of the Raspbery Pi. From setting up to coding the single board computer and many other factors in between. Too often we hear that teachers have bought a Raspberry Pi and left it to languish on a dusty shelf. So how can we reduce the number of dusty Pis? Well Picademy is the answer.

Picademy is two days of training that offers a unique twist. During day one teachers experience a hands on series of lessons that focus on the core values of the Raspberry Pi. Learning to hack and using the term not in the same way as the media, rather using it for its original intention. Picademy is also key to introducing skills such as coding, electronics, problem solving and team work. At the end of day one the cohort are set a challenge for day two. Create a project using their Raspberry Pi. So for six hours on day two we see the class working with code, electronics and arts materials to produce some quite frankly fantastic projects.

At Picademy teachers are exposed to coding using Python, Ruby and Scratch but they are also introduced to Linux, in the form of the Raspbian operating system. In fact they are shown the basics of navigating and using the BASH shell and how to use sudo correctly. So let's stop there for one moment, we are training school teachers to use Linux and they are in turn instructing their class to replicate their actions and understand the consequences. Linux is making progress into our schools and there is even a controlled assessment for GCSE students that covers the use of Linux for a multi user system. Could this be the start of Linux entering the classroom?

Picademy also provides a network of support for their alumni which goes much further than simple handouts and PDF. Social media has helped Picademy grow and teachers are now using Twitter as a new form of training and development. The conversations and networks created by Twitter have seen adoption of the Raspberry Pi in schools increase rapidly. The Raspberry Pi and Picademy are also the subject for many of the evening Twitter discussions such as Tweachcode[2] and CASchat[3] which focus on the educational use of technology. For Picademy alumni there is also a closed Google + community where they can discuss education related projects and post curriculum focused content.

Picademy also provides an insight into education focused projects one of which is a Raspberry Pi thin client network. PiNet[4] is a project by Andrew Mulholland and it initially started life as an A Level project to facilitate the use of a thin client setup for Raspberry Pi. This project caught the eye of the Foundation who have since supported Andrew. PiNet requires an older computer with at least a dual core 2Ghz CPU to act a a server on a LAN. PiNet runs on top of a standard install of Ubuntu[5]. During the install process an SD card image is created for the Raspberry Pi. This is image is then used to network boot each Raspberry Pi which then requests and writes information to a shared network drive on the PiNet server. This project is demonstrated to the group and they are invited to test drive PiNet with a view to possible adoption in their classroom.

Teachers leave Picademy with a strong commitment to using their Raspberry Pi in class and just browsing the Picademy[6] hash tag shows the diverse range of projects that have been born from Picademy. My particular favourite is a game of Twister which uses two motors to pick a colour and body part at random. Totally reproducible with very little cost.

Picademy is now taking place around the UK, within centres located in Leeds, Birmingham that form part of Google's Digital Garage scheme, and at the head quarters of the Raspberry Pi Foundation, lovingly referred to as "Pi Towers" in Cambridge.

If you are a teacher, educator or member of a coding group then Picademy is the place for you to learn new skills and show others how versatile the Raspberry Pi can be.

[1] **https://www.raspberrypi.org/picademy/**

[2] **https://twitter.com/TweachCode**

[3] **https://twitter.com/hashtag/caschat**

[4] **http://pinet.org.uk/**

[5] **http://www.ubuntu.com/**

[6] **https://twitter.com/hashtag/picademy**

Of Matlab and Python

Ian Hopkinson

I've been a scientist and data analyst for nearly 25 years. Originally as an academic physicist, then as a research scientist in a large fast moving consumer goods company and now at a small technology company in Liverpool. In common to many scientists of my age I came to programming in the early eighties when a whole variety of home computers briefly flourished. My first formal training in programming was FORTRAN after which I have made my own way.

I came to Matlab in the late nineties, frustrated by the complexities of producing a smooth workflow with FORTRAN involving interaction, analysis and graphical output.

Matlab is widely used in academic circles and a number of industries because it provides a great deal of analytical power in a user-friendly environment. Its notation for handling matrix (array) calculations is slick. Its functionality is extended by a range of toolboxes, and there is a community of scientists sharing new functionality. It shares this feature set with systems such as IDL and PV-WAVE.

However, there are a number of issues with Matlab:

- as a programming language it has the air of new things being botched onto a creaking frame. Support for unit testing is an afterthought, there is some integration of source control into the Matlab environment but it is with Source Safe. It doesn't support namespaces. It doesn't support common data structures such as dictionaries, lists and sets.
- The toolbox ecosystem is heavily focused on scientific applications, generally in the physical sciences. So there is no support for natural language processing, for example, or building a web application based on the powerful analysis you can do elsewhere in the ecosystem;
- the licensing is a nightmare. Once you've got core Matlab additional toolboxes containing really useful functionality (statistics, database connections, a 'compiler') are all at an additional cost [1]. In my experience you often find yourself needing a toolbox

for just a couple of functions. For an academic things are a bit rosier, universities get lower price licenses and the process by which this is achieved is opaque to end-users. As an industrial user, involved in the licensing process, it is as bad as line management and sticking needles in your eyes in the 'not much fun thing to do' stakes;

- running Matlab with network licenses means that your code may stop running part way through because you've made a call to a function to which you can't currently get the license. It is difficult to describe the level of frustration and rage this brings. Now of course one answer is to buy individual licenses for all, or at least a significant surplus of network licenses. But tell that to the budget holder particularly when you wanted to run the analysis today. The alternative is to find one of the license holders of the required toolbox and discover if they are actually using it or whether they've gone off for a three hour meeting leaving Matlab open;
- deployment to users who do not have Matlab is painful. They need to download a more than 500MB runtime, of exactly the right version and the likelihood is they will be installing it just for your code;

I started programming in Python at much the same time as I started on Matlab. At the time I scarcely used it for analysis but even then when I wanted to parse the HTML table of contents for Physical Review E, Python was the obvious choice. I have written scrapers in Matlab but it involved interfering with the Java underpinnings of the language.

Python has matured since my early use. It now has a really great system of libraries which can be installed pretty much trivially, they extend far beyond those offered by Matlab. And in my view they are of very good quality. Innovation like IPython notebooks take the Matlab interactive style of analysis and extend it to be natively web-based. If you want a great example of this, take a look at the examples provided by Matthew Russell for his book, Mining the Social Web.

Python is a modern language undergoing slow, considered improvement. That's to say it doesn't carry a legacy stretching back decades and changes are small, and directed towards providing a more consistent language. Its used by many software developers who provide a source of help, support and an impetus for an decent infrastructure.

Ubuntu users will find Python pre-installed. For Windows users, such as myself, there are a number of distributions which bundle up a whole bunch of libraries useful for scientists and sometimes an IDE. I like python(x,y) [2]. New libraries can generally be installed almost trivially using the pip package management system. I actually use Python in Ubuntu and Windows almost equally often. There are a small number of libraries which are a bit more tricky to install in Windows – experienced users turn to Christoph Gohlke's fantastic collection of precompiled binaries [3].

In summary, Matlab brought much to data analysis for scientists but its time is past. An analysis environment built around Python brings wider functionality, a better coding infrastructure and freedom from licensing hell.

[1] **http://uk.mathworks.com/pricing-licensing/**

]2] **http://python-xy.github.io/**

[3] **http://www.lfd.uci.edu/˜gohlke/pythonlibs/**

This article originally appeared on the author's blog and is reproduced with permission: http://ow.ly/UWdTb

Exercises for Programmers

Brian P. Hogan

Pragmatic Bookshelf
ISBN: 978-1-68050-122-3
110pp.
£ 15.99
Published: September 2015

reviewed by Nigel Barker

The subtitle '57 Challenges to Develop Your Coding Skills' makes it clear that this short book from The Pragmatic Programmers is not a guide to help desk-bound coders improve their physical fitness. Instead the book consists entirely of those end of chapter problems found in most 'Learning' a computer language books.

An initial chapter covers design and testing, after that it's just the challenges, most of which get a page or two outlining the problem, defining constraints and adding further challenges to extend the problem.

Starting with 'Hello, World' the book advances through text handling, handling quoted text, calculations, flow control, functions, data, files, interfacing with external APIs and ends with five ideas for full programs.

Some of the ideas you might have seen before, like currency converters and word counters, but some were certainly new to me, like using the Open Notify API to display the names of all the people in space right now.

For each challenge the non IT bits are briefly but carefully explained, but sometimes there is some knowledge assumed of the current IT landscape, e.g. JSON, Firebase.

I liked this book, the length is right (97 pages of content) and the price is right ($24). When learning a new programming language I'm sure it will be helpful to at least think about how you could solve some of these 57 challenges.

Jump Start Git

Shaumik Daityari

SitePoint
ISBN: 978-0-9941826-5-4
150pp.
£ 19.99
Published: September 2015

reviewed by James Roberts

I actually looked up the US definitions of 'jump-start' after the first 30 pages or so of this book to see if it meant something other than that I, in the UK, expected, but it doesn't. I have perforce had to use Git to get fresh versions of some software, but don't work in a team code-wise and find Git just as awkward as the name suggests, so am not an enthusiastic Git user: I'd hoped this book might change that.

I do expect a book called 'Jump-start-something' to at least cause something to start quickly. Perhaps even to give new energy to my git-ing going. Sadly this book does neither (nor does it start my car). I suspect this is the author's first published book, and he has not been served well by the editor(s) or publisher, see later.

What's wrong with it? Well, what's Git? It's a version control system or VCS. Who uses it? Developers who write program code, particularly in teams. So, let's consider the likely audience that would buy a 'jump-start git' book as against, say, 'The Really Long Book of VCS Systems Innards Analysed, Part 2 (CS Year 2 coursebook)' (imaginary title)?

- They'd likely know what a VCS is and want a very rapid introduction to Git's version of it with no superfluities.
- This would perhaps be because they just got a new job starting tomorrow which requires them to use Git and they haven't used it before, or a parallel scenario requiring rapid familiarisation – a jump-start
- They would not at this time have any academic interest in Git's history, rationale or raison-d'etre; they just want to know how to use it immediately

What we get instead is nine pages of introduction to various VCS systems with historical references to SCCS and RCS which would have been used by the probable reader's grandfather (ie, me) including the most wrong-headed single sentence about the genesis of Git via Bitkeeper I have yet to read in print.

Once the actual Git material starts, after referring to installing Git, one of the first things the author does is to set the Git config to automatic colourization. This is completely reasonable and entirely sensible, but unfortunately from then on the author regularly refers to the colour of the output, like 'green lines starting with a + sign show what's been added, and the red line starting with a - sign shows what's been removed'

The editor and publisher have however set the book in monochrome, clearly without warning the author that this would be the case, so not only does the author refer to colours that are not in any of the screenshots, the actually intended colour-highlight is instead various shades of faded grey, with the red shade of grey barely distinguishable from the green shade of grey or any of the other would-be colour shades of grey: this could so easily have been avoided had the editor or publisher wished to do so by liaising with the author, and it would have been a very good idea to have done so, though I have now developed a better understanding of the difficulties faced by red-green colour-blind developers. This error really does ruin the point the author is trying to make each time in his introductions to the various aspects of Git.

If you are working through the book on your terminal, as you should be, then of course you will be able to see the described effects; but if you are doing that why not use one of the on-line guides to Git? The point of a book is that many people (me included) find it easier to learn by reviewing the activities on the three levels, first the book and the idea, then try it in the terminal, then perhaps some exercises or direct to a real-life trial. Much of that point is here removed.

I worked through the first two practical chapters to see how it goes and it's all clear enough with no errors (but no colours). It's a bit discursive for my taste (as a rapid introduction book) but gets there in the end.

I am not expert in Git, but I did not find anything obviously wrong with the content I tested, I can't speak for the advanced topics. I did note that the early configuration step of choosing the editor that Git will use, emphasised in 'Pro-Git' with the warning 'You may find, if you don't setup an editor like this, you will likely get into a really confusing state when they are launched. Such example on a Windows system may include a prematurely terminated Git operation during a Git initiated edit' is entirely skipped.

I have one other minor bone to pick: the preface states '…(the) book is suitable for anyone interested in managing multiple revisions of code, data and documents…'.

The book has in fact a total of five paragraphs concerning data and documents in Chapter 8, which includes one paragraph on managing Photoshop documents.

My consultancy has tried to get graphics designers using VCS of any sort, and the results have not been good. Graphics designers have visual minds and text-based tools just won't fly, even with a GUI. Moreover most of their output is big blobs. Git does not handle this sort of thing well (there's git-annex and git-media - but the issue is doing diffs on big binaries, not easy). For the same reasons few version control systems make any stab at versioning big binary files adequately (Subversion and Boar may be exceptions), and Git is no better at this than most: graphics designers will get on better with the Adobe Cloud. I see this section as just padding.

To summarise: this is not a dreadful book, it's OK I suppose, and if you end up with it I expect you will learn most of what you need to know about using Git. However, while the author is as far as I can tell (I'm no Git expert) accurate it does not have that air of authority that comes with the book by the developers (which is available to read online free). The book is also rather short for the price, compared with other offerings.

The book should have been much better had the publisher told the editor and author that all colour references would be removed, and the screenshots had been redone accordingly; a good deal of editorial tightening could also have been applied. It's not the author's fault that this has not been done.

For a real jump-start to Git try the online one-page guide 'git - the simple guide', q.v. For an authoritative guide try 'Pro-Git' whose authors include one of the developers. And as a final note, SitePoint continuously fills my inbox with invitations to subscribe. However, based on this book, I think I'll continue to abstain.

Git for Teams

Emma Jane Hogbin Westby
O'Reilly Media
ISBN: 978-1-4919-1118-1
356pp.
£ 33.50
Published: September 2015

reviewed by Kenneth MacDonald

The two forewords, one written by the 'Git for Windows' maintainer (Johannes Schindelin) and the other by Hewlett Packard's Director of Open Source Engagement (Mark Atwood), summarise the book's content and style very well. Emma's teaching experience shines through as she guides the reader through a series of workflow and team dynamic models before introducing Git as a supporting tool. This is what sets this book apart from other guides to Git - this is a guide to the development process, helping you formalise your own practices or to adopt any of the well known ones in common use. She provides links to further reading and relevant online videos when appropriate.

The book starts off with a section on team structures and dynamics. You will probably find aspects of your own team's idiosynchrosies described here.

The project governance section is a welcome addition to the book, with a clear and frank discussion of issues and a range of solutions covering authorship, copyright and distribution licences, leadership models and finally codes of conduct.

She then moves on to consider access models, whether your team is made up of dispersed contributors, uses collocated contributor repositories or practises shared maintenance.

Every book introducing Git has to have a discussion on branching strategies. This book does so without invoking any actual Git commands, instead concentrating on the pros and cons of each, encouraging you to choose the right convention for your team - there being no limits to how you can use branches. There are short sections on mainline development, branch per feature, state branching and scheduled deployment strategies.

In the workflow discussion, she helpfully references the GitHub Flow, GitLab Flow and Git-Flow methodologies. Continuous delivery and deployment are mentioned in passing - this book is not a guide to setting up any infrastructure to support it, nor does it pretend to be.

It is only in the second part of three that Git as a tool is explored and she suggests the reader sits at a computer and follows along. I found the initial stages of these a bit disjointed, although I perhaps hadn't entered the spirit quite enough and was too tempted to jump around the material. This section of the book requires time set aside to work through it, not dip in and out.

The initial work is targeted at teams of one, the reader. This allows you to become familiar with Git as your revision control system. More advanced features supporting teams of more than one introduced later, including publishing your work and undertaking code reviews.

A particularly useful 'Rrrrgh!' chapter on rolling back, reverting, resetting and rebasing - all ways to recover from a variety of past mistakes. Her teaching experience comes to the fore again, with instructions to redraw and expand a flow chart for helping you recover such situations as they arise.

The final section of this book consists of three different Git hosting options - Open Source projects on GitHub, private team working on Bitbucket and self hosted collaboration with GitLab. This is the part of the book that could become out of date over time as these services and tools change, or even get replaced. But the text is as easy and engaging as always, and the worked examples really help the reader get a genuine feel for each of these options.

Modern PHP

Josh Lockhart
O'Reilly Media
ISBN: 978-1-4919-0501-2
268pp.
£ 12.99
Published: February 2015

reviewed by Lindsay Marshall

The first thing I have always said to any class to whom I am teaching PHP is 'all the PHP books available are terrible - do not buy any of them'. (The second is that almost all the examples of PHP coding that you see on the Internet are also terrible - please no sarcastic comments from PHP haters!) However, this year I changed that and said 'if you really want to buy a book about PHP, then buy Modern PHP (and don't even look at any of the others)'. Unlike many other books claiming to teach the latest features of PHP this book, really does take in it says on its cover 'New features and good practices'. Put simply, I learned a lot of useful inspiration from this book about packages I had not come across and useful tricks with PHP features. So, yes, if you want to buy a book about PHP, I can heartily recommend this one. I'm sure all this information is out there on the Internet but finding it, and having the will to find it, is the problem, and a collection of useful concepts is always good. Not too extravagantly priced either. Two thumbs up.

CSS Master

Tiffany B Brown
SitePoint
ISBN: 978-0-9941826-2-3
350pp.
£ 21.00
Published: September 2015

reviewed by Lindsay Marshall

A preface to book about CSS should not be ugly. This book is ugly. The typefaces chosen are unpleasant, there is too much white space everywhere making the pages look a little faint and blurry, and, most seriously, it is printed in greyscale - no colour anywhere in the text. Oh, and it uses footnotes - a personal bugbear of mine. YMMV.

The book starts with this gem 'CSS has grown from a language for formatting documents into a robust language for designing web applications. Its syntax is easy to learn, making CSS a great entry for those new to programming'. Seriously? Writing CSS is not programming. The syntax is horrible and confusing. And it really hasn't very much to do with designing web applications, other than matters of formatting. (Yes, I know about the clever things you can do with CSS.)

But what about the real content? It is really rather good. Lots of nice examples, clearly explained. Good coverage of features and their ins and outs: selectors, transforms, transitions etc. etc. But I have a hard time getting past the look and feel of the book. The lack of colour really makes some of the illustrations difficult to work out.

Again, all this material is available online, and probably in colour, but if you want a book, and you can get past its presentation then, this would not be a bad choice. But it exists in a crowded marketplace.

iOS 9 Programming Fundamentals with Swift

Matt Neuburg
O'Reilly Media
ISBN: 978-1-4919-3677-1
604pp.
£ 22.79
Published: October 2015

reviewed by Lindsay Marshall

The trouble with publishing is that it is cannot be as responsive to the real world as it needs to be. It takes time to get a book to press, even when the book is just a revamp of a previous edition. This book covers iOS9, Xcode 7 and Swift 2.0. It arrived with me about a week after iOS 9.1 was released along with Xcode 7.1 and Swift 2.1, and as I write, beta versions of iOS 9.2 are available. Of course, these are point releases and not major changes, but even so I am not quite sure why you would want to spend a fairly substantial amount of money on a book that you know is (in part) obsolete when it hits the shelves and will be guaranteed to be out of date within a year, or sooner. This is not a slim volume either - 578 pages: possible future use as a doorstop. Oh, and I just noticed that I have a copy of the previous edition in the 'waiting to be reviewed' pile (iOS 8.3, Xcode 6.3 and Swift 1.2. 558 pages). I don't think I need to do that now.

None of this means that the material in the book is bad: Neuburg is well known for the quality of his books and this is no exception. Lots of high quality material, good, practical examples. Lots of useful advice. My problem with the book though is about its targeting. I have no idea who this book is meant for. It starts by trying to explain what Object oriented programming, so is clearly aimed at readers with little actual programming knowledge. But Swift is moving target. A new language with slightly unusual features influenced by other languages that are not particularly well known. I like Swift a lot and am enjoying programming in it, but it is by no means an easy language. I would hesitate to use it as an introduction to Object Oriented programming for anyone, let alone a novice. For example there is material later on in the book about memory management and the language that I had to read several times to understand fully what was going on. Someone starting out in programming would really struggle, and probably give up.

Ultimately, I think there are two books here: an introductory book explaining the basics of Swift, Xcode and iOS development, and a more intermediate level volume that assumes a greater level of experience and knowledge. But you are still faced with the problem of built-in obsolesence. The only people I could see buying this book would be professional developers who needed to get up to speed on the basics of iOS9 etc. and could expense the purchase - this is really not a book for the casual reader or the learner.

The Architecture of Privacy

Courtney Bowman, Ari Gesher, John K Grant, Daniel Slate

O'Reilly Media
ISBN: 978-1-4919-0401-5
200pp.
£ 12.60
Published: September 2015

reviewed by Lindsay Marshall

'This book is not for privacy experts.' And that's probably a good thing. A slim volume giving a high level view of important aspects of privacy in electronic systems, but including a good selection of real world examples illustrating various issues. The writing is concise but clear and the fact that it has four authors is not glaringly obvious. For anyone wanting to get grips with privacy terminology and concepts, this is an excellent, practical introduction. It is slightly US focussed in places, though it does discuss EU policies and laws as well. Definitely recommended.

Ansible: Up and Running

Lorin Hochstein

Pragmatic Bookshelf
ISBN: 978-1-4919-1532-5
334pp.
£ 26.50
Published: May 2015

reviewed by Paul Waring

This year I've been consolidating several servers, and took this as an opportunity to transfer everything to this new-fangled configuration management thing which people have been talking about. Ansible seemed like a good choice for a small number of servers, and conveniently this book was released just as I was starting to look into the subject.

The first chapter covers the basics, not just of what Ansible is but how it works and some of the advantages, and by the end of it you will have a virtual machine setup and managed with Ansible. One important thing to note at this point is that the book assumes you are using at least version 1.8.4, which may not be available for your Linux distribution (e.g. Ubuntu prior to 15.10). There's also no discussion of using Ansible to manage Windows servers, which may be an issue for some, although the author points out that this is a relatively new feature and could form a book in itself.

The next three chapters introduce the core concepts in Ansible: playbooks, inventories and variables & facts. All the basics are covered, with plenty of examples and some useful diagrams. The author also warns readers against useful but potentially difficult to debug features such as handlers, and having tried to use them I regret not heeding the author's advice from the start!

The rest of the book moves into more advanced features and specific use cases, such as deploying to Amazon EC2 and of course Docker. Most of these are self-contained and can be safely skipped (I have yet to use any of them), with the exception of the final chapter on debugging which is concise and extremely useful.

The appendices are handy for reference purposes, especially the one covering SSH which outlines how to generate keys, get them onto a server and deal with common error messages.

Overall, this book is a great introduction to Ansible, and I used it to get 'up and running' very quickly.

Pragmatic Unit Testing in Java 8 with JUnit

Jeff Langr, Andy Hunt, Dave Thomas

Pragmatic Bookshelf
ISBN: 978-1-941222-59-1
200pp.
£ 13.25
Published: March 2015

reviewed by Paul Waring

Like many software developers, I like to save myself time and effort wherever possible, and part of that involves having a good set of automated tests which catch mistakes as early as possible. For Java, this effectively means using JUnit, which is by far and away the most popular unit testing framework for the language. This book covers that topic and nothing else, and has the advantage of focusing on the latest version of Java.

The first part of the book covers the basics of testing, including why and when you should write unit tests. As introductions go, it is very gentle, and I had no trouble keeping up, although some of the language and 'conversations' between developers made me cringe at times.

Part II descends into buzzwords, including FIRST, BICEP and CORRECT. Painful those these are to read, many people use them and the authors do a reasonable job of explaining each one.

Part III moves up a level to consider the wider picture, something which technical books sometimes forget in favour of details. The bulk of the focus is on refactoring – both code and tests – but broader design issues are also covered. These chapters are especially useful when working with existing or legacy codebases, as it is often harder to add tests at this stage than from day one.

The final part of the book looks at Test Driven Development, or writing tests before code – a difficult concept to grasp initially but useful once you get your head round it. Advanced topics such as testing multithreaded code and databases are also covered, although in limited detail. Given that these two areas are becoming more important as time goes one, a dedicated chapter for each would be a good addition to a future edition.

Overall, this is a good introduction to testing in Java, starting from first principles and building up to Test Driven Development. It's definitely worth picking up if you do any development in Java.

Contributors

Nigel Barker lives in East Lothian and works at Dell SecureWorks. His first exposure to Unix was when the comic distributor he worked for bought a Texas Instruments TI-1500 mini computer in 1989.

Irina Bolychevsky is a freelance consultant currently working for the W3C and Open Data Institute. Previously Commercial Director at Open Knowledge and CKAN's product owner, she's also the co-founder of redecentralize.org. Loves tea and technology. Drinks coffee to look cool.

Andrew Findlay is an independent consultant and trainer specialising in Directory Services, data synchronisation, and e-mail systems. He has worked with X.500 and LDAP since 1988, and has designed directory schema for a number of large organisations. Andrew holds BSc and PhD degrees in Cybernetics from the University of Reading and is an active member of both the IET and FLOSSUK.

Ian Hopkinson is Senior Data Scientist at ScraperWiki following work as an academic and a research scientist at Unilever. Throughout this time he has rummaged through data with a wide variety of open source tools.

Kimball Johnson has been programming since a very early age, starting with BBC Micros, then MS DOS and Windows Systems, however was enlightened with a copy of Debian GNU/Linux Woody at university. From this developed an affinity to Systems Administration, but like all good admins he is lazy and so tries to automate as much of his job as possible, and now follows the way of the Infrastructure as Code.

Kenneth MacDonald is a senior computing officer at the University of Edinburgh and FLOSS UK Council member. He discovered Linux at a time when we had to swap the kernel and root floppy discs at boot time while coding for his PhD in Geophysics and has been managing systems at the university ever since, with a particular interest in configuration management.

Lindsay Marshall developed the Newcastle Connection distributed UNIX software and created the first Internet cemetery. He is a Senior Lecturer in the School of Computing Science at the Newcastle University. He also runs the RISKS digest website and the Bifurcated Rivets weblog.

Jane Morrison is Company Secretary and Administrator for FLOSS UK, and manages the FLOSS UK office at the Manor House in Buntingford. She has been involved with FLOSS UK administration since 1987. In addition to FLOSS UK, Jane is Company Secretary for a trade association (Fibreoptic Industry Association) that she also runs from the Manor House office.

Les Pounder works closely with North Western Linux and Free Software groups to promote the use of Open Source software as opposed to proprietary software. He is also the organiser of UCubed, a free Linux and open source event in Manchester, and an organiser of Barcamp Blackpool and Blackpool Geekup, and has been head of crew at Oggcamp. He writes for Linux Format magazine and contributes to Linux podcasts including Fullcircle, Ubuntu UK Podcast and Linux Outlaws.

James Roberts was one of the last people in the UK to learn to program on 80-column cards at Control Data, and ran into Linux SLS in 1992. He runs a small company/consultancy that sneaks Linux and *BSD into SME Windows shops wherever it fits. He has inter alia taught micro-computing and MIDI and audio recording to young adults.

Paul Waring works as a freelance IT consultant. Outside of work he can usually be found filing documentation bugs against various open source and free software projects. He also edits the FLOSS UK newsletter.

Contacts

Kimball Johnson
Chairman
London

Gavin Atkinson
Council member
York

Mark Keating
Council member
Lancaster

Kenny Macdonald
Council member
Edinburgh

Ian Norton
Council member
Morecambe

Quentin Wright
Treasurer
Warwick

Paul Waring
Newsletter Editor & System Administrator
Manchester

Alain Williams
System Administrator
Watford

Sam Smith
OpenTech
Cambridge

Jane Morrison
FLOSS UK Secretariat
The Manor House
Buntingford
Herts
SG9 9AB
Tel: 01763 273475
Fax: 01763 273255
office@flossuk.org