The Open Pitt

What's cooking in Linux and Open Source in Western Pennsylvania

Issue 36 | July 2007 | www.wplug.org

Book Review: *Everyday Scripting with Ruby* *by Patrick Wagstrom*

Author: Brian Marick
Publisher: The Pragmatic Bookshelf
ISBN: 0-9776166-1-4
$29.95, 301 pages, 2007

Ruby is a mature language with a sizable history behind it. It has been only recently, largely thanks to Ruby on Rails, that the Ruby community has exploded and become mainstream. Many of the hottest "Web 2.0" applications like Basecamp, Backpack, and Twitter are built on Rails. In particular, many people have seen the 20-minute screencast of creating a full weblog system with Ruby on Rails. Unlike PHP, Ruby is a general purpose scripting language first, and a web development language second. In *Everyday Scripting with Ruby*, Brian Marick attempts to enlighten developers on the ways that Ruby can be used for everyday tasks and highlight some of the best practices for Ruby development.

As a disclaimer, I approached this book with relatively little knowledge of Ruby—my only previous experience was installation and some very minor customization of a Ruby on Rails application. For the last six years, my primary programming language has been Python, but I also know and have moderate experience with most other common programming languages. In that sense, it's difficult to comment on how good of an introduction to basic programming concepts the book is. Instead, I'll focus on how it conveyed the Ruby language.

The book is designed as a roughly linear guide through the different elements of the Ruby programming language, starting out with basic concepts such as messages in Ruby (think of methods or functions in other languages), and progressing to topics like basic control structures, file I/O, test harnesses, scraping web pages, and object oriented programming. Chapters are broken up into two different categories, either a general chapter that helps explain a topic, or "Ruby Facts" that provide a quick and easy view of topics such as booleans, classes, hashes, and argument lists.

The back of the book states that the text will help developers: learn to automate simple tasks, gain an understanding of utilizing finished Ruby scripts, understand programming terminology, benefit from existing code libraries, and communicating more effectively with teammates.

For the beginning programmer and the developer new to Ruby, the book provides a good introduction into the semantics of Ruby and how to utilize code and packages from other developers through *irb*, an interactive interpreter for Ruby, and Ruby Gems, a shared library consisting of hundreds of Ruby modules. The presence and explanation of these tools makes it much easier for beginning programmers to understand what is going on and provides a very forgiving interface. For example, within *irb*, errors in code do not cause the interpreter to exit; rather, the exceptions are caught and the developer can continue to proceed, learning in a stepwise fashion how to interact with objects in the environment.

However, while the book may be a linear guide to picking up Ruby, and it starts out very gracefully, it quickly becomes apparent that it much of the book seems pasted on top of an underlying structure. That structure in this case is that the book was originally entitled *Scripting for Testers*, and sadly the change of the book to a more general focus did not come out as nicely as it could have. For a beginning developer, a target group for this book, the frequent discussions of testing may be very confusing and seem unneeded for someone who wants to just dive in. Most of the evidence that is provided for the needs for testing harnesses seems to be anecdotal. Perhaps, however, that's just the scientist in me looking at the work in the wrong way.

See **RUBY**, *p. 2*

June Roundup

Jun. 9 General User Meeting: **Brian Seklecki** spoke about load balancers, hardware devices used mainly in data centers to distribute requests to multiple servers, such as for a web site which receives too much traffic to be handled by a single machine. Brian began with a discussion of the situations in which you might need a load balancer. He then moved into an overview of specific types of applications like database and e-mail servers. Additional time was dedicated to managing persistent sessions on Web servers. Brian finished by explaining domain name system-related issues.

Coming Events

Aug. 5: 6th Annual WPLUG Open Source Picnic. 1PM to 6PM, Snyder Park, Whitehall
Aug. 11: Installfest. 10AM to 1PM, 1507 Newell-Simon Hall, CMU
Sep. 8: General User Meeting. *(Time and location TBA, see web site for details)*

The public is welcome at all events

From the Editor: Watch That License

Many developers in the Free and Open Source world seem averse to thinking about licensing, considering it too much of a political issue. But a quick glance at the recent examples of proprietary extensions for Joomla! and the Broadcom wireless adapter driver for OpenBSD should convince you that thinking about licensing ahead of time can help you avoid much acrimony and pain.

If you are part of a project that accepts code from others, you need to be sure the following points are covered:

- Ensure that the contributor has the rights to the code being contributed. If he isn't the original author, make sure you find out the actual origin.
- Get explicit terms for the contribution. Don't simply assume that a patch is covered by the same license as your project.
- Make sure the license for the contribution is compatible with your project's license. Some projects may want to go a step further and have all contributors assign copyright to the project to maintain a single point of ownership.

Be sure to do your part by including a clear copyright statement in each source file of the project. A note simply saying “GPL” or “BSD” is insufficient; there are multiple versions of both of these licenses. A good example of what a notice should look like can be found at the end of the GPL text.

You wouldn't drop a block of C code into a Java program and expect it to work; don't mix incompatible licenses and expect to avoid serious headaches down the road.

RUBY, *from p. 1*

Despite this issue, there still are some very strong points in the book that can help out most developers. In particular, the examples are very good and are real-world situations, providing some insight on interacting with remote websites and reading data out of a *subversion* repository. In both of these instances, however, the mechanism employed to obtain the data is rather fragile, relying on the text output from the program or website. Nonetheless, the explanation is well done and artfully introduces concepts as they are needed, such as the use of regular expressions, which is always a tricky concept.

Part of me wonders if some of the difficulty in providing explanations in the book is because of the structure of the Ruby programming language. Coming from a Python perspective, where clear code and a standard indentation style are typical, and having a good amount of experience with Perl, Ruby seems a little like it is a cross between Perl and Python, but just different enough that you can't quite understand everything that is going on in the language. In particular, the difference between utilizing the results of a message and examining the value of a variable can be difficult to tell in the code because of shared semantics between the two concepts. The book is laid out very much like a traditional programming book, but Ruby does not seem to be as much of a traditional language as something like C or Java.

While the book aimed to complete five goals, I cannot say it was excellent in any particular one of the areas. In particular, some areas such as improving communication with teammates are only barely touched on in the benefits of writing a test harness. Indeed, it appears that in this case the writer tried to make too broad of a book in too short of space, and ended up with book that doesn't do particularly well in any of the areas.

For someone with moderate programming experience who has a burning desire to learn Ruby, this book may suffice, but I can't help but think that for the average person there may be better introductory texts or better languages to start out with.

You can visit the book's Web site at <http://www.pragmaticprogrammer.com/titles/bmsft/>.

Patrick Wagstrom is a Ph.D. candidate at Carnegie Mellon University researching communication and collaboration in Open Source development. He has been using Linux since 1994.

The Open Pitt is published by the Western Pennsylvania Linux Users Group
<http://www.wplug.org/top/>

Editor: Vance Kochenderfer

What is Linux?

Linux is a *kernel*, the core of a computer operating system, created by Linus Torvalds. It is typically packaged as a *distribution*, which includes the extra programs necessary to make a computer functional and useful. Since 1991, it has grown from a one-man project which ran on one computer to one with thousands of contributors running on everything from personal organizers to million-dollar supercomputers.

What are Open Source and Free Software?

Open Source and Free Software provide you, the user, with the opportunity to see the source code of the programs you use. You are free to use it, share it with others, and even make changes to it if you wish. While the Free Software and Open Source communities differ in their philosophical approach, in practical terms they share nearly identical goals. Learn more at <http://www.opensource.org/> and <http://www.gnu.org/>.

This newsletter was produced using Open Source and Free Software.

Annual Open Source Picnic

WPLUG's 6th annual Open Source Picnic will be held on Sunday, August 5. As in recent years, we will be at Snyder Park in Whitehall (in the South Hills).

Please go to the WPLUG wiki <http://wplug.ece.cmu.edu/wiki/index.php/2007OpenSourcePicnic> to RSVP if you plan on coming. While you're there, why not sign up to bring food, supplies, or organize an activity?

Snyder Park features a covered pavilion, playground, basketball court, and baseball field. Count on bringing the whole family for an afternoon of friends and fun!

Complete details and directions can be found on the wiki page. See you there!