Linux for PHP

A DOCKER CONTAINER MADE WITH PHP IN MIND

By Andrew Caya, ZCE, ZCA

Who am I?

- I am Andrew Caya
- CEO, CTO and Founder of Foreach Code Factory (https://etista.com)
- Instructor at Concordia University
- Started out with GW-BASIC and QBASIC in 1991
- C, C++ (Qt), Perl programmer
- Linux system administrator
- PHP developer since 2009
  - Zend Certified Engineer since 2015
  - Zend Certified Architect since 2016
- « Mercenary » developer since 2010 (thanks to Tim Lytle for the term)
- Author and Technical Reviewer for Packt Publishing since 2016
- Upcoming projects :
  - Author of « Mastering the Faster Web: PHP, MySQL and JavaScript » (due to be published within the next two weeks)
  - Creator of Linux for PHP (https://linuxforphp.net)

What is Linux for PHP?

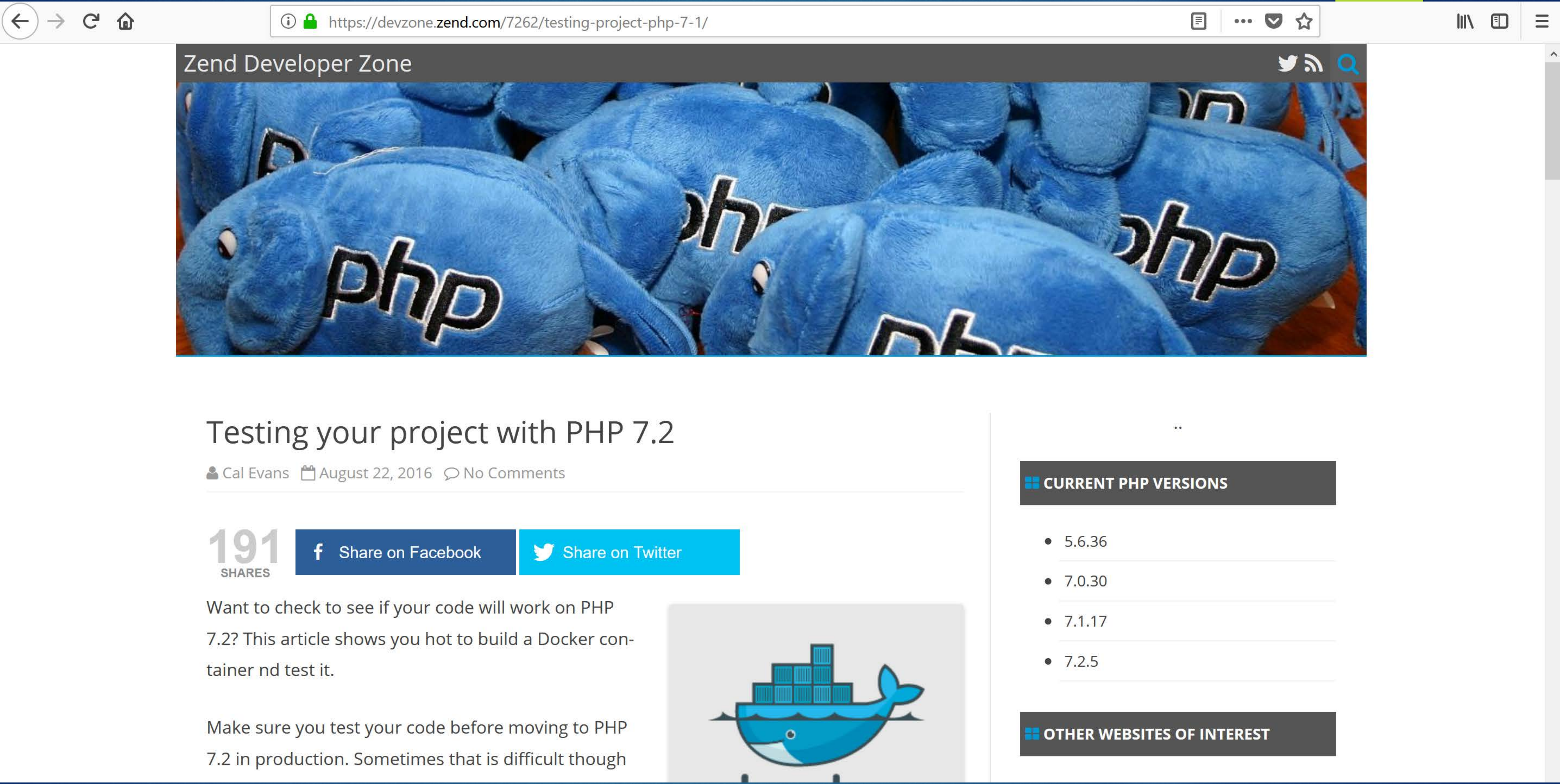
- Lightweight LAMPP Server
  - Linux for PHP is a lightweight version of Linux with all the software needed to easily compile and/or use any version of PHP.
- Cutting-Edge Technology
  - Linux for PHP is built with recent versions of all the underlying software needed to run PHP, making it a cutting-edge software infrastructure for your PHP code.
- Easily Customizable
  - Linux for PHP can easily be configured to suit your every needs. From running a CMS to debugging and profiling a script, Linux for PHP offers it all.

But, what about Alpine?

- Compiler is built with the musl libc (https://www.musl-libc.org)
  - Fully compatible with glibc? NO.
    - https://www.musl-libc.org/faq.html
    - https://github.com/gliderlabs/docker-alpine/blob/master/docs/caveats.md
- Alpine uses Busybox tools
  - Comprehensive set of programs to run a Linux system, but each tool remains impaired in functionality
    - https://wiki.alpinelinux.org/wiki/How_to_get_regular_stuff_working
- Compiler switches and non-standard libraries can cause issues
  - Some switches (or missing ones) can create strange SEGFAULTs on some processors
  - The use of some non-standard libraries in the toolchain can prove to be problematic (https://bugs.alpinelinux.org/issues/4986)

The origins of Linux for PHP…

HTTPS://DEVZONE.ZEND.COM/7262/TESTING-PROJECT-PHP-7-1/


https://devzone.zend.com/7262/testing-project-php-7-1/

Zend Developer Zone

Testing your project with PHP 7.2

Cal Evans August 22, 2016 No Comments

191 SHARES Share on Facebook Share on Twitter

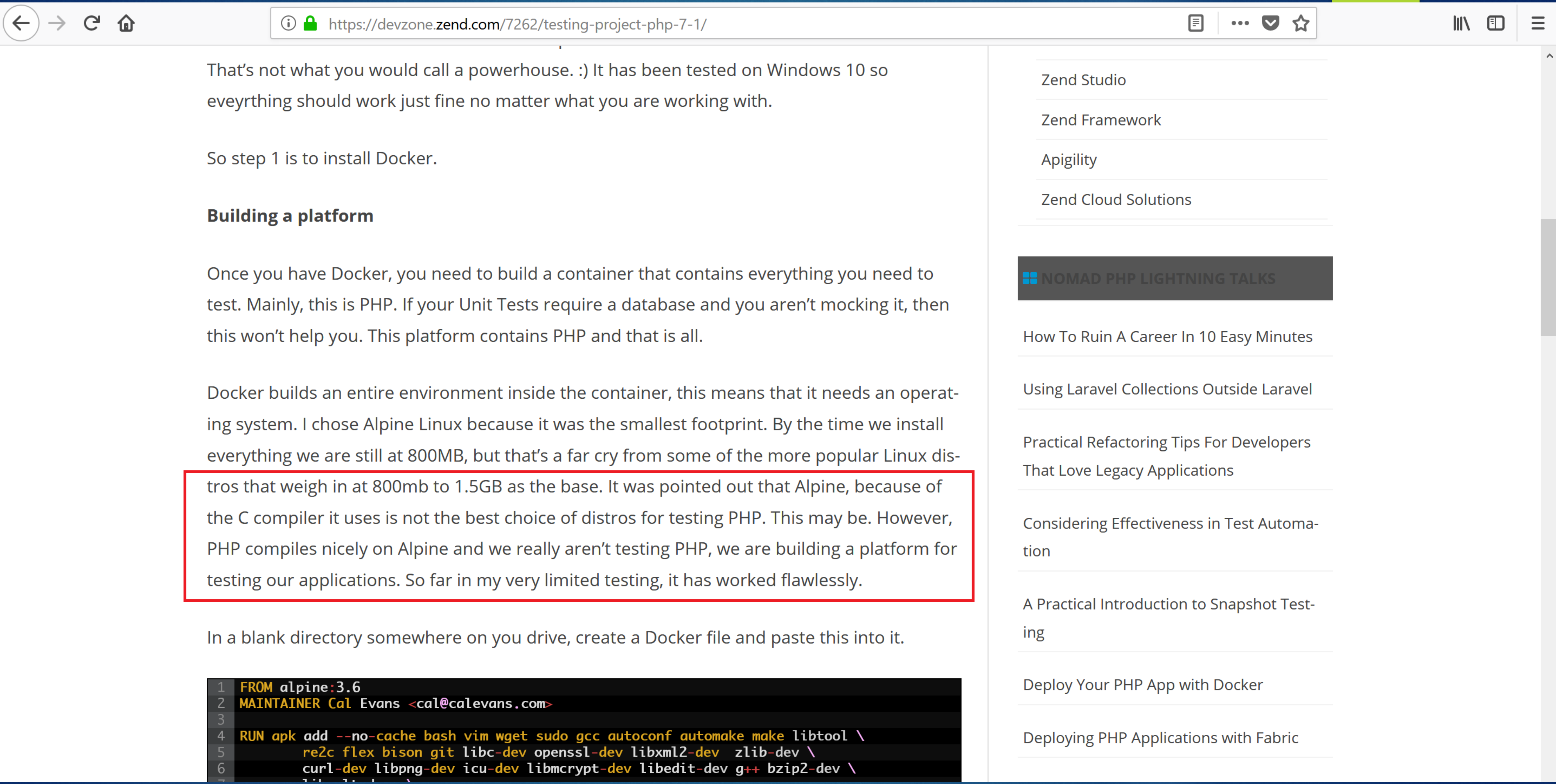
Want to check to see if your code will work on PHP 7.2? This article shows you hot to build a Docker container nd test it.

Make sure you test your code before moving to PHP 7.2 in production. Sometimes that is difficult though

CURRENT PHP VERSIONS

- 5.6.36
- 7.0.30
- 7.1.17
- 7.2.5

OTHER WEBSITES OF INTEREST


https://devzone.zend.com/7262/testing-project-php-7-1/

That's not what you would call a powerhouse. :) It has been tested on Windows 10 so eveyrthing should work just fine no matter what you are working with.

So step 1 is to install Docker.

Building a platform

Once you have Docker, you need to build a container that contains everything you need to test. Mainly, this is PHP. If your Unit Tests require a database and you aren't mocking it, then this won't help you. This platform contains PHP and that is all.

Docker builds an entire environment inside the container, this means that it needs an operating system. I chose Alpine Linux because it was the smallest footprint. By the time we install everything we are still at 800MB, but that's a far cry from some of the more popular Linux distros that weigh in at 800mb to 1.5GB as the base. It was pointed out that Alpine, because of the C compiler it uses is not the best choice of distros for testing PHP. This may be. However, PHP compiles nicely on Alpine and we really aren't testing PHP, we are building a platform for testing our applications. So far in my very limited testing, it has worked flawlessly.

In a blank directory somewhere on you drive, create a Docker file and paste this into it.

```
FROM alpine:3.6
MAINTAINER Cal Evans <cal@calevans.com>

RUN apk add --no-cache bash vim wget sudo gcc autoconf automake make libtool \
        re2c flex bison git libc-dev openssl-dev libxml2-dev  zlib-dev \
        curl-dev libpng-dev icu-dev libmcrypt-dev libedit-dev g++ bzip2-dev \
```

Zend Studio

Zend Framework

Apigility

Zend Cloud Solutions

NOMAD PHP LIGHTNING TALKS

How To Ruin A Career In 10 Easy Minutes

Using Laravel Collections Outside Laravel

Practical Refactoring Tips For Developers That Love Legacy Applications

Considering Effectiveness in Test Automation

A Practical Introduction to Snapshot Testing

Deploy Your PHP App with Docker

Deploying PHP Applications with Fabric

Linux for PHP features

- Standard toolchain compiled from scratch
  - ~100 packages built with ~5500 lines of Bash code
- Compilation decisions are taken according to what suits PHP best
- Compiler switches for all software and libraries are chosen with hardware compatibilty in mind
- Most downloadable images weigh ~500Mb
- Very short release cycles (~6 months)
- Up to date libraries (https://hub.docker.com/r/asclinux/linuxforphp/)
- Cross-platform compatibility thanks to Docker (Windows, Mac, Linux/Unix)

Linux for PHP use cases

- Easily compile a recent version of PHP
- Easily test any PHP script against any PHP version
- Easily profile any script with Blackfire.io
- Easily test any CMS/framework
- COMING SOON: Composer integration (sneak peek!)

You are only one command away from PHP anywhere, anytime!

Thank you!

https://joind.in/talk/23ec3

@linuxforphp
https://linuxforphp.net

Andrew Caya, ZCE, ZCA

@AndrewSCaya

https://ca.linkedin.com/in/andrewscaya