EXHIBITING SPONSORS

TEKsystems — Own change

nexmo — The Vonage API Platform

orbis TECHNOLOGIES, INC.

Capital One

AUTOMATTIC

InterSystems — Health | Business | Government

techFrederick

SocketLabs — the science of hitting the inbox

dcv digital creative visionaries

MySQL

PREMIER SPONSORS

twilio

RingCentral®

Red Hat OpenShift

WIFI

SSID: CCMIT

MID ATLANTIC DEV CON ★ 2019 ★

Cloud – Mobile – Web – Dev

Strategically Reusing Code

Brad Griffith

Code Wrangler, Automattic

brad.griffith@automattic.com

Strategy

RICHARD P. RUMELT
"A giant in the field of strategy." —McKinsey Quarterly
GOOD STRATEGY
BAD STRATEGY
The Difference and Why It Matters

"

The core of strategy work is always the same: discovering the critical factors in a situation and designing a way of coordinating and focusing actions to deal with those factors.

RICHARD P. RUMELT
"A giant in the field of strategy." —McKinsey Quarterly
GOOD STRATEGY
BAD STRATEGY
The Difference and Why It Matters

“

A good strategy honestly acknowledges the challenges being faced and provides an approach to overcoming them. And the greater the challenge, the more a good strategy focuses and coordinates efforts to achieve a powerful competitive punch or problem–solving effect.

RICHARD P. RUMELT

"A giant in the field of strategy." —McKinsey Quarterly

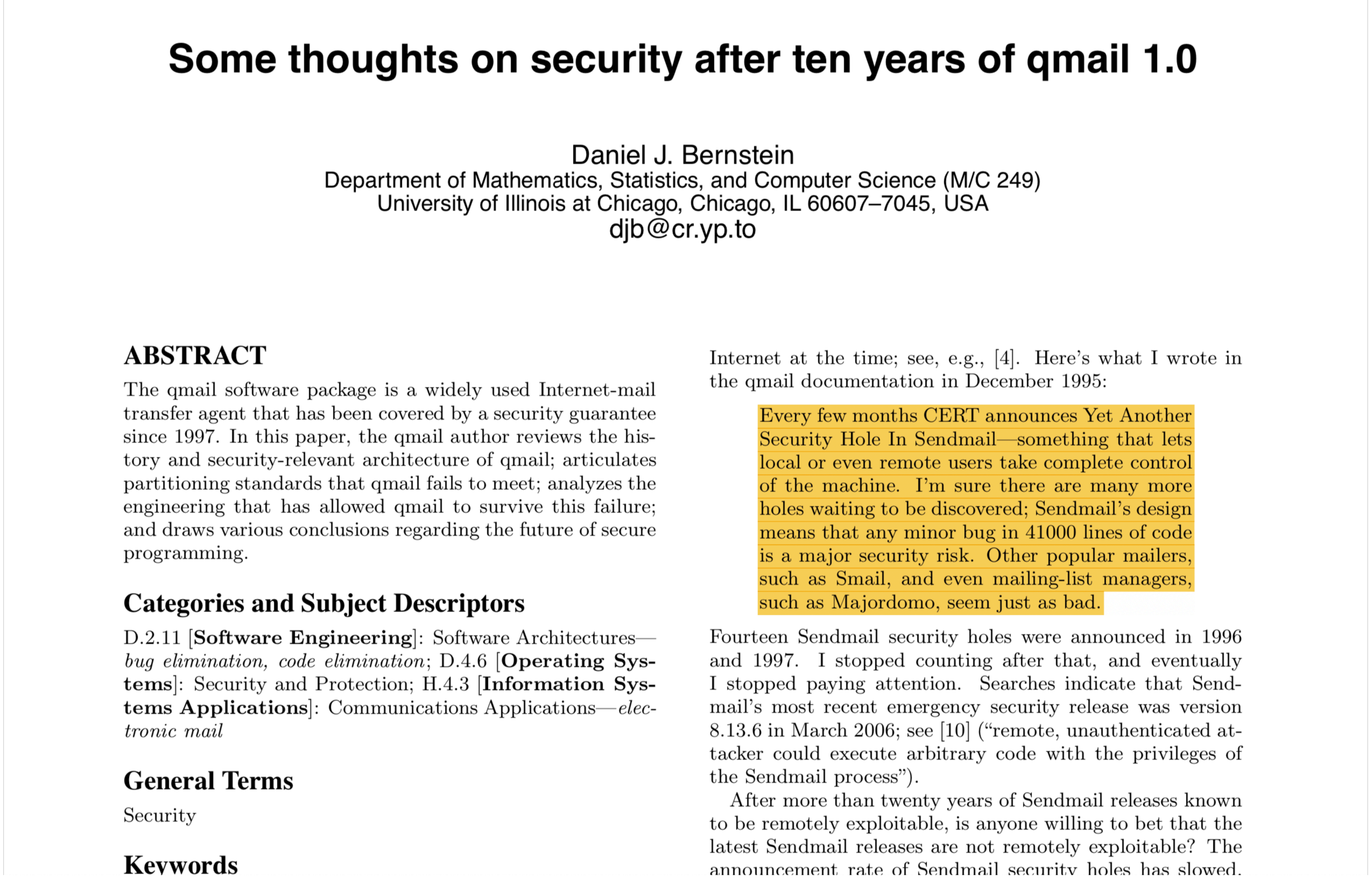
GOOD STRATEGY BAD STRATEGY

The Difference and Why It Matters

“

The potential gains to coordination do not mean that more [...] coordination is always a good thing. Coordination is costly, because it fights against the gains to specialization. To specialize in something is, roughly speaking, to be left alone [...] and not be bothered with other tasks, interruptions, and other agents’ agendas.

Go It Alone



Some thoughts on security after ten years of qmail 1.0

Daniel J. Bernstein
Department of Mathematics, Statistics, and Computer Science (M/C 249)
University of Illinois at Chicago, Chicago, IL 60607–7045, USA
djb@cr.yp.to

ABSTRACT

The qmail software package is a widely used Internet-mail transfer agent that has been covered by a security guarantee since 1997. In this paper, the qmail author reviews the history and security-relevant architecture of qmail; articulates partitioning standards that qmail fails to meet; analyzes the engineering that has allowed qmail to survive this failure; and draws various conclusions regarding the future of secure programming.

Categories and Subject Descriptors

D.2.11 [**Software Engineering**]: Software Architectures—*bug elimination, code elimination*; D.4.6 [**Operating Systems**]: Security and Protection; H.4.3 [**Information Systems Applications**]: Communications Applications—*electronic mail*

General Terms

Security

Keywords

Internet at the time; see, e.g., [4]. Here's what I wrote in the qmail documentation in December 1995:

> Every few months CERT announces Yet Another Security Hole In Sendmail—something that lets local or even remote users take complete control of the machine. I'm sure there are many more holes waiting to be discovered; Sendmail's design means that any minor bug in 41000 lines of code is a major security risk. Other popular mailers, such as Smail, and even mailing-list managers, such as Majordomo, seem just as bad.

Fourteen Sendmail security holes were announced in 1996 and 1997. I stopped counting after that, and eventually I stopped paying attention. Searches indicate that Sendmail's most recent emergency security release was version 8.13.6 in March 2006; see [10] ("remote, unauthenticated attacker could execute arbitrary code with the privileges of the Sendmail process").

After more than twenty years of Sendmail releases known to be remotely exploitable, is anyone willing to bet that the latest Sendmail releases are not remotely exploitable? The announcement rate of Sendmail security holes has slowed.

University of Illinois at Chicago, Chicago, IL 60607–7045, USA
djb@cr.yp.to

ware package is a widely used Internet-mail
hat has been covered by a security guarantee
his paper, the qmail author reviews the his-
y-relevant architecture of qmail; articulates
ndards that qmail fails to meet; analyzes the
t has allowed qmail to survive this failure;
us conclusions regarding the future of secure

nd Subject Descriptors

re Engineering]: Software Architectures—
, *code elimination*; D.4.6 [**Operating Sys-**
 and Protection; H.4.3 [**Information Sys-**
ions]: Communications Applications—*elec-*

Internet at the time; see, e.g., [4]. Here's what I wrote in the qmail documentation in December 1995:

> Every few months CERT announces Yet Another Security Hole In Sendmail—something that lets local or even remote users take complete control of the machine. I'm sure there are many more holes waiting to be discovered; Sendmail's design means that any minor bug in 41000 lines of code is a major security risk. Other popular mailers, such as Smail, and even mailing-list managers, such as Majordomo, seem just as bad.

Fourteen Sendmail security holes were announced in 1996 and 1997. I stopped counting after that, and eventually I stopped paying attention. Searches indicate that Sendmail's most recent emergency security release was version 8.13.6 in March 2006; see [10] ("remote, unauthenticated at-qmail 1.00, 117685 words. On 15 June 1998 I released the current version, qmail 1.03, 124540 words. A slight derivative created by the community, netqmail 1.05, has 124911 words. I am aware of four bugs in the qmail 1.0 releases.

For comparison: Sendmail 8.7.5, released in March 1996, had 178375 words; Sendmail 8.8.5, released in January 1997, had 209955 words; Sendmail 8.9.0, released in May 1998, had 232188 words. The Sendmail release notes report hundreds of bugs in these releases. There are some user-visible feature differences between Sendmail and qmail, such as qmail's POP support, and Sendmail's UUCP support, and qmail's user-controlled mailing lists, and Sendmail's "remote root exploit" feature—just kidding!—but these don't explain the complexity gap; most of the code in each package is devoted to core MTA features needed at typical Internet sites.

Fingerprinting indicates that more than a million of the Internet's SMTP servers run either qmail 1.03 or netqmail 1.05. The third-party qmail.org site says

> A number of large Internet sites are using qmail: USA.net's outgoing email, Address.com, Rediffmail.com, Colonize.com, Yahoo! mail, Network Solutions, Verio, MessageLabs (searching 100M emails/week for malware), listserv.acsu.buffalo.edu (a big listserv hub, using qmail since 1996), Ohio State (biggest US University), Yahoo! Groups, Listbot, USWest.net (Western US ISP), Telenordia,

can't exploit qmail to steal or corrupt other users' data. If other programs met the same standard, and if our network links were cryptographically protected, then the only remaining security problems on the Internet would be denial-of-service attacks.

1.4 Contents of this paper

How was qmail engineered to achieve its unprecedented level of security? What did qmail do well from a security perspective, and what could it have done better? How can we build other software projects with enough confidence to issue comparable security guarantees?

My views of security have become increasingly ruthless over the years. I see a huge amount of money and effort being invested in security, and I have become convinced that most of that money and effort is being wasted. Most "security" efforts are designed to stop yesterday's attacks but fail completely to stop tomorrow's attacks and are of no use in building invulnerable software. These efforts are a distraction from work that *does* have long-term value.

In retrospect, some of qmail's "security" mechanisms were half-baked ideas that didn't actually accomplish anything and that could have been omitted with no loss of security. Other mechanisms have been responsible for qmail's successful security track record. My main goal in this paper is to explain how this difference could have been recognized in advance—how software-engineering techniques can be *measured* for their long-term security impact.

bly more code than were spent writing the function
e function is also a natural target for tests.)
he benefit scales to larger systems and to a huge va-
nctions; `fd_move()` is just one example. In many
utomated scan for common operation sequences
t helpful new functions, but even without automa-
quently find myself thinking "Haven't I seen this
nd extracting a new function out of existing code.

tomatically handling temporary errors

r the following excerpt from `qmail-local`:

```
stralloc_cats(&dtline,"\n")) temp_nomem();
```

`loc_cats` function changes a dynamically resized
iable `dtline` to contain the previous contents of
lowed by a linefeed. Unfortunately, this concate-
run out of memory. The `stralloc_cats` function
rns 0, and `qmail-local` exits via `temp_nomem()`,
the rest of the qmail system to try again later.
re thousands of conditional branches in qmail.
f of them—I haven't tried to count exactly—are
ing other than checking for temporary errors.
cases I built functions such as

```
uts(s)
s;
```

code in a better language and then using an automated translator to convert the code into C as a distribution language. Stroustrup's `cfront`, the original compiler from C++ to C, is an inspirational example, although as far as I know it has never acquired exception-handling support.

4.3 Reusing network tools

UNIX has a general-purpose tool, `inetd`, that listens for network connections. When a connection is made, `inetd` runs another program to handle the connection. For example, `inetd` can run `qmail-smtpd` to handle an incoming SMTP connection. The `qmail-smtpd` program doesn't have to worry about networking, multitasking, etc.; it receives SMTP commands from one client on its standard input, and sends the responses to its standard output.

Sendmail includes its own code to listen for network connections. The code is more complicated than `inetd`, in large part because it monitors the system's load average and reduces service when there is heavy competition for the CPU.

Why does Sendmail not want to handle mail when the CPU is busy? The basic problem is that, as soon as Sendmail accepts a new message, it immediately goes to a lot of effort to figure out where the message should be delivered and to try delivering the message. If many messages show up at the same time then Sendmail tries to deliver all of them at the same time—usually running out of memory and failing at most of the deliveries.

Sendmail tries to recognize this situation by checking the

4.4 Reusing access controls

I started using UNIX, specifically Ultrix, twenty years ago. I remember setting up my `.forward` to run a program that created a file in `/tmp`. I remember inspecting thousands of the resulting files and noticing in amazement that Sendmail had occasionally run the program under a uid other than mine.

Sendmail handles a user's `.forward` as follows. It first checks whether the user is allowed to read `.forward`—maybe the user has set up `.forward` as a symbolic link to a secret file owned by another user. It then extracts delivery instructions from `.forward`, and makes a note of them (possibly in a queue file to be handled later), along with a note of the user responsible for those instructions—in particular, the user who specified a program to run. This is a considerable chunk of code (for example, all of `safefile.c`, plus several scattered segments of code copying the notes around), and it has contained quite a few bugs.

Of course, the operating system already has its own code to check whether a user is allowed to read a file, and its own code to keep track of users. Why write the same code again?

When qmail wants to deliver a message to a user, it simply starts a delivery program, `qmail-local`, under the right uid. When `qmail-local` reads the user's delivery instructions, the operating system automatically checks whether the user is allowed to read the instructions. When `qmail-local` runs a program specified by the user, the operating system automatically assigns the right uid to that program.

I paid a small price in CPU time for this code reuse: qmail

I should have, similarly, put the `nsa.gov` configuration into `/var/qmail/control/domains/nsa.gov`, producing the same simplicity of code. I instead did something requiring slightly more complicated code: `nsa.gov` is a line in a file rather than a file in a directory. I was worried about efficiency: most UNIX filesystems use naive linear-time algorithms to access directories, and I didn't want qmail to slow down on computers handling thousands of domains. Most UNIX filesystems also consume something on the scale of a kilobyte to store a tiny file.

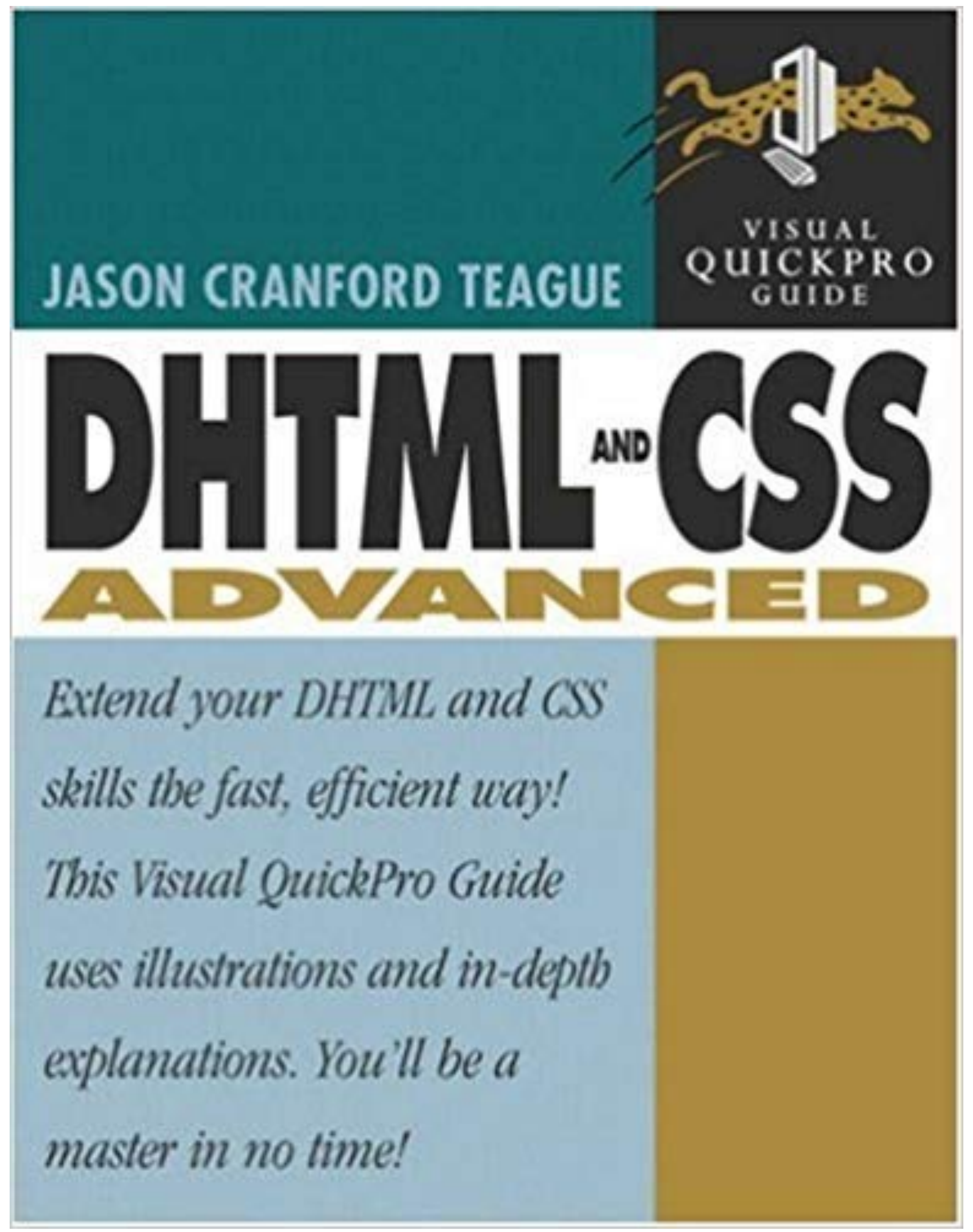
In retrospect, it was stupid of me to spend code—not just this file-parsing code, but also code to distribute message files across directories—dealing with a purely hypothetical performance problem that I had not measured as a bottleneck. Furthermore, to the extent that measurements indicated a bottleneck (as they eventually did for the message files on busy sites), I should have addressed that problem at its source, fixing the filesystem rather than complicating every program that uses the filesystem.

5. ELIMINATING TRUSTED CODE

5.1 Accurately measuring the TCB

"Even if all of these programs are completely compromised, so that an intruder has control over the `qmaild`, `qmails`, and `qmailr` accounts and the mail queue, he still can't take over your system," I wrote in the qmail documentation. "None of the other programs trust the results from these five." I continued in the same vein for a while, talk-

JavaScript


JASON CRANFORD TEAGUE
VISUAL QUICKPRO GUIDE
DHTML AND CSS
ADVANCED
Extend your DHTML and CSS skills the fast, efficient way! This Visual QuickPro Guide uses illustrations and in-depth explanations. You'll be a master in no time!

Internet Explorer 5

```
elementReference.attachEvent("event", functionReference);
```

Netscape Navigator

```
elementReference.addEventListener("eventType", functionReference, captureSwitch);
```

Are you fed up with your web browser?

You're not alone. We want you to know that there *is* an alternative...

Introducing Mozilla

Firefox 1.0

"I installed Firefox on my laptop today. It's so fast — I never knew there could be that much of a difference."

— [illegible], New Zealand

"I was tired of my browser crashing everyday, so I tried Firefox. Now I can't live without it. Pop-up blocking, secure browsing and no spyware. Best of all... not a crash since I switched."

— [illegible], USA

"I thought changing and learning a new web browser would be difficult, but with Firefox I had no problems at all. Browsing is now smooth."

— [illegible], Finland

Firefox is the free, open source web browser from the Mozilla Foundation that lets you surf faster and more efficiently and helps avoid annoying pop-ups and spyware. Join us and make the switch today — Firefox imports your favorites, settings and other information, so there's nothing to lose.

Find out what more than 10 million users from around the world already know: there is an alternative!

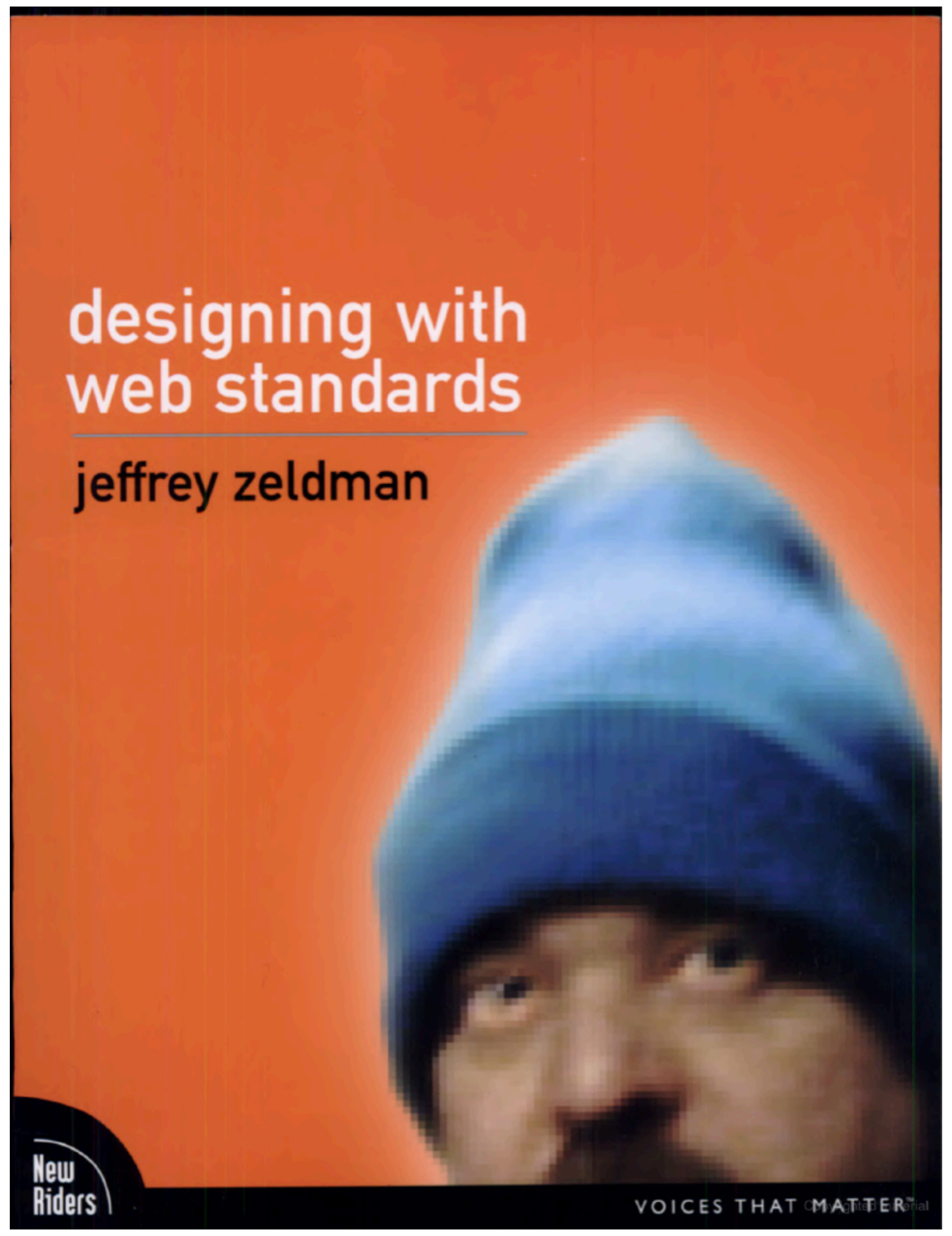
Download today from

GetFirefox.com

This message has been brought to you by [illegible] Mozilla Foundation, a non-profit organization dedicated to promoting choice and innovation on the Internet. Special thanks to [illegible] for helping to make this possible.

Movable Type

Drupal™



designing with web standards

jeffrey zeldman

New Riders

VOICES THAT MATTER

mootools

```
var Animal = new Class({

    initialize: function(name) {
        this.name = name;
    }

});

var Cat = new Class({
    Extends: Animal,

    talk: function() {
        return 'Meow!';
    }

});
```

jQuery
write less, do more.

```
$('div.test')
  .on('click', handleTestClick)
  .addClass('foo');
```

yui

```
YAHOO.myProject.myModule = function () {


	//"private" variables:
	var myPrivateVar = "I can be accessed only from within YAHOO.myProject.myModule.";

	//"private" method:
	var myPrivateMethod = function () {
		YAHOO.log("I can be accessed only from within YAHOO.myProject.myModule");
	}


	return  {
		myPublicProperty: "I'm accessible as YAHOO.myProject.myModule.myPublicProperty.",
		myPublicMethod: function () {
			YAHOO.log("I'm accessible as YAHOO.myProject.myModule.myPublicMethod.");

			//Within myProject, I can access "private" vars and methods:
			YAHOO.log(myPrivateVar);
			YAHOO.log(myPrivateMethod());

			//The native scope of myPublicMethod is myProject; we can
			//access public members using "this":
			YAHOO.log(this.myPublicProperty);
		}
	};

}(); // the parens here cause the anonymous function to execute and return
```

```
YAHOO.myProject.myModule = function () {


        //"private" variables:
        var myPrivateVar = "I can be accessed only from within YAHOO.myProject.myModule.";

        //"private" method:
        var myPrivateMethod = function () {
                YAHOO.log("I can be accessed only from within YAHOO.myProject.myModule");
        }


        return  {
                myPublicProperty: "I'm accessible as YAHOO.myProject.myModule.myPublicProperty.",
                myPublicMethod: function () {
                        YAHOO.log("I'm accessible as YAHOO.myProject.myModule.myPublicMethod.");

                        //Within myProject, I can access "private" vars and methods:
                        YAHOO.log(myPrivateVar);
                        YAHOO.log(myPrivateMethod());

                        //The native scope of myPublicMethod is myProject; we can
                        //access public members using "this":
                        YAHOO.log(this.myPublicProperty);
                }
        };

}(); // the parens here cause the anonymous function to execute and return
```

“

JavaScript is a language that most people don’t bother to learn before they use it. You can’t do that with any other language, and you shouldn’t want to, and you shouldn’t do that with this language either.

DOUGLAS CROCKFORD

ember

node JS

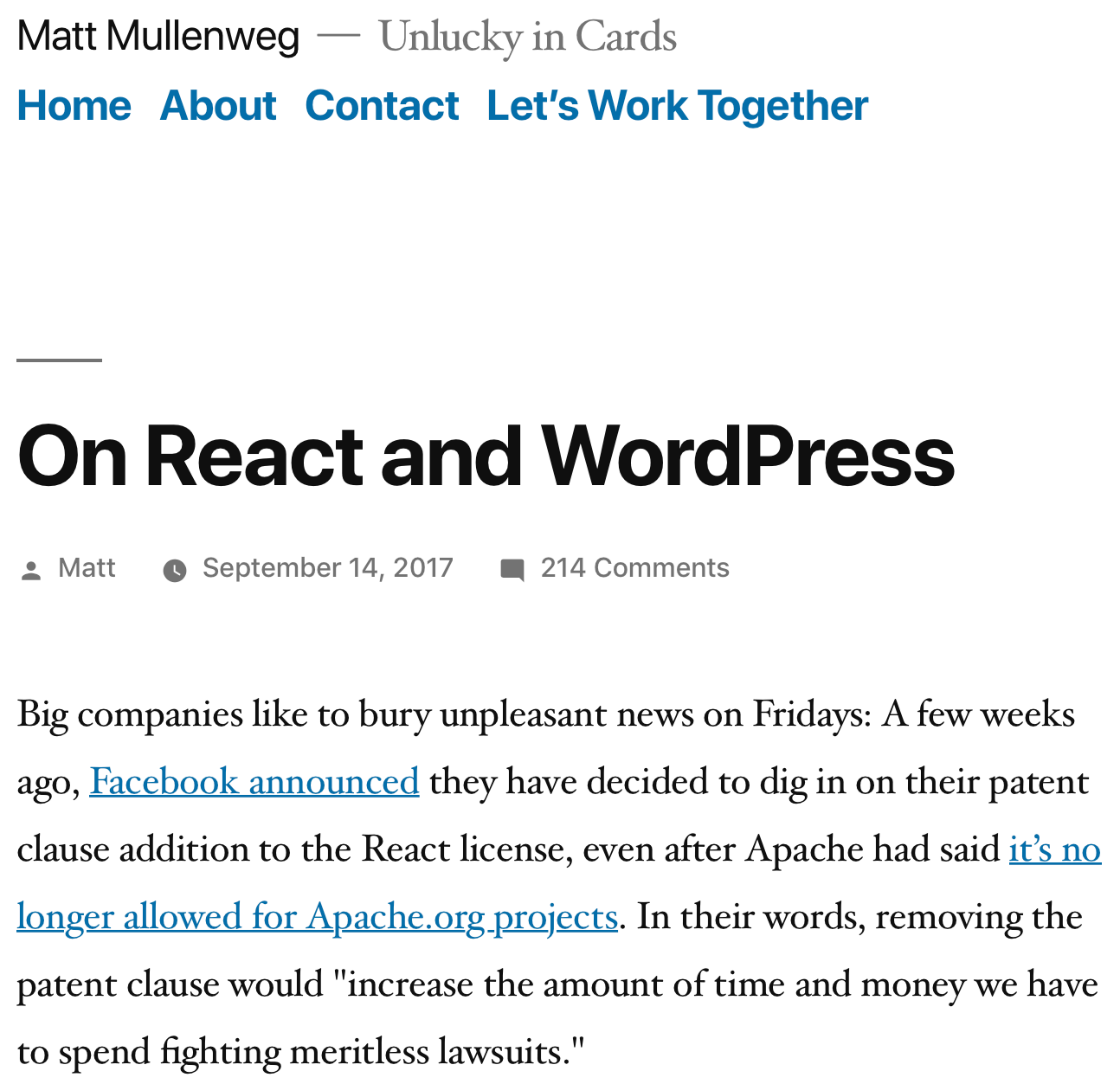
npm

BABEL

webpack
MODULE BUNDLER

Open Source

- The freedom to run the program as you wish, for any purpose (freedom 0).
- The freedom to study how the program works, and change it so it does your computing as you wish (freedom 1). Access to the source code is a precondition for this.
- The freedom to redistribute copies so you can help others (freedom 2).
- The freedom to distribute copies of your modified versions to others (freedom 3). By doing this you can give the whole community a chance to benefit from your changes. Access to the source code is a precondition for this.



Matt Mullenweg — Unlucky in Cards

Home About Contact Let's Work Together

On React and WordPress

Matt September 14, 2017 214 Comments

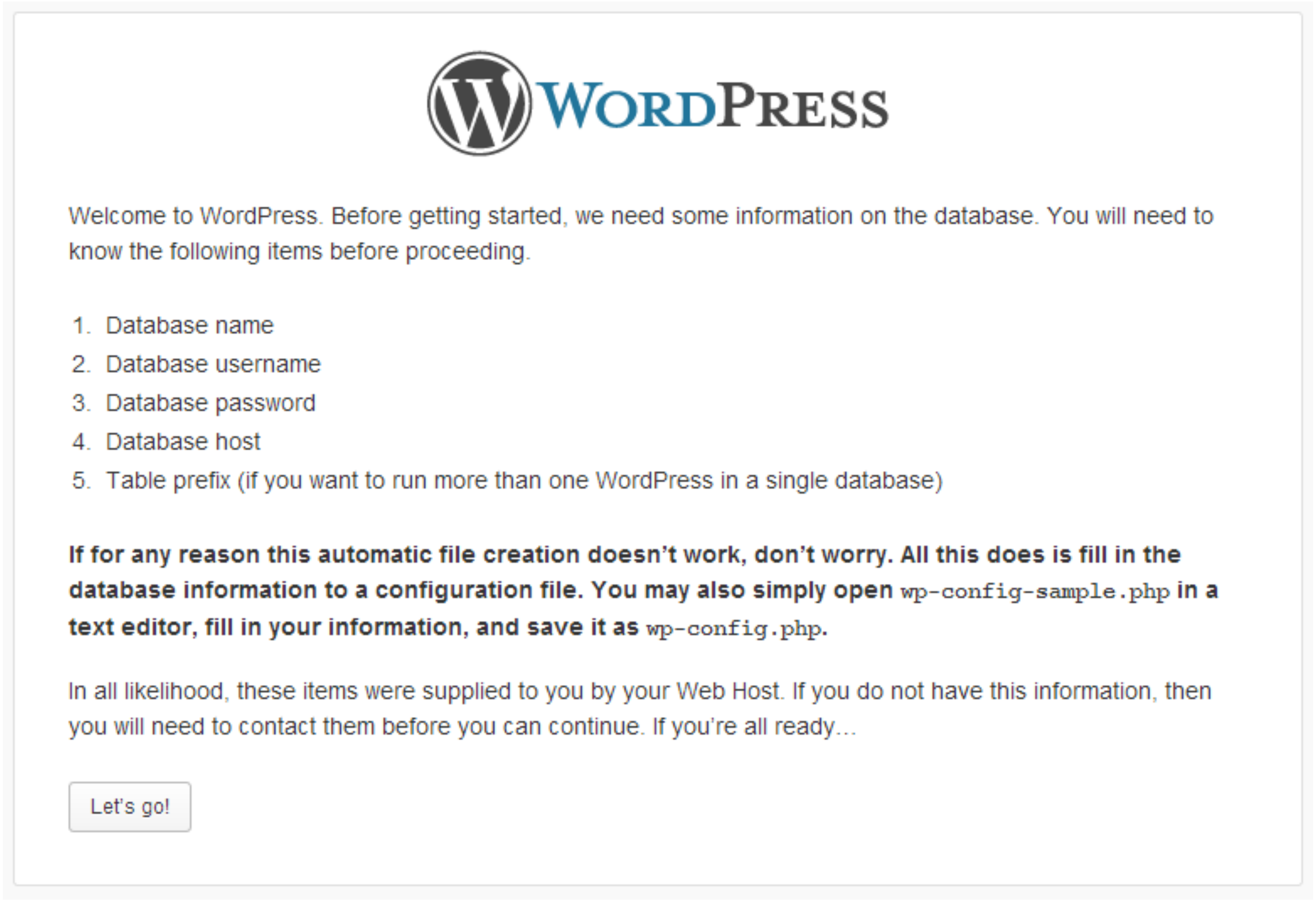
Big companies like to bury unpleasant news on Fridays: A few weeks ago, Facebook announced they have decided to dig in on their patent clause addition to the React license, even after Apache had said it's no longer allowed for Apache.org projects. In their words, removing the patent clause would "increase the amount of time and money we have to spend fighting meritless lawsuits."

POSTED ON SEP 22, 2017 TO WEB

Relicensing React, Jest, Flow, and Immutable.js

THE APACHE™ SOFTWARE FOUNDATION

FREE SOFTWARE FOUNDATION



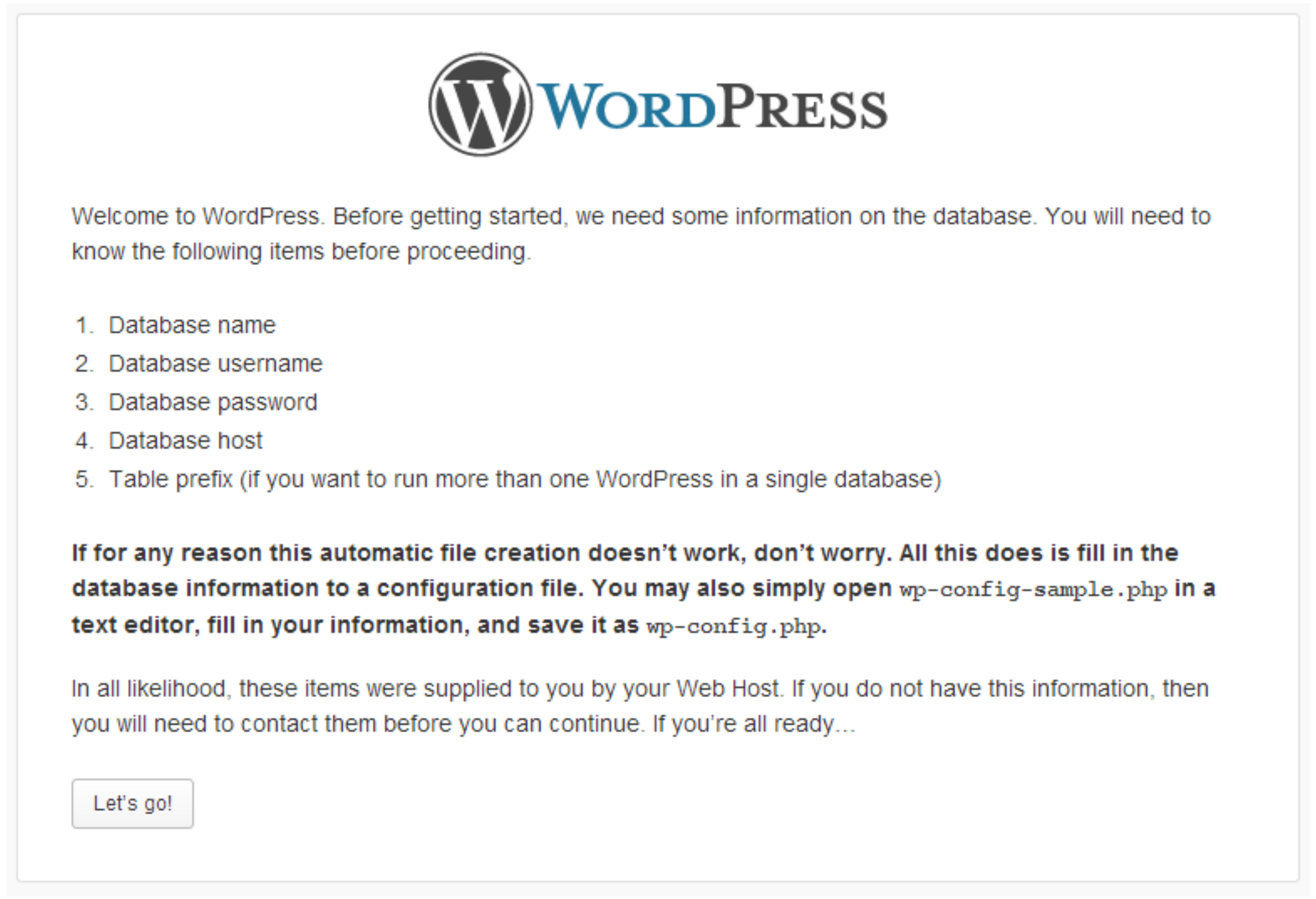
WORDPRESS

Welcome to WordPress. Before getting started, we need some information on the database. You will need to know the following items before proceeding.

1. Database name
2. Database username
3. Database password
4. Database host
5. Table prefix (if you want to run more than one WordPress in a single database)

If for any reason this automatic file creation doesn't work, don't worry. All this does is fill in the database information to a configuration file. You may also simply open `wp-config-sample.php` in a text editor, fill in your information, and save it as `wp-config.php`.

In all likelihood, these items were supplied to you by your Web Host. If you do not have this information, then you will need to contact them before you can continue. If you're all ready…


Welcome to WordPress. Before getting started, we need some information on the database. You will need to know the following items before proceeding.

1. Database name
2. Database username
3. Database password
4. Database host
5. Table prefix (if you want to run more than one WordPress in a single database)

If for any reason this automatic file creation doesn't work, don't worry. All this does is fill in the database information to a configuration file. You may also simply open `wp-config-sample.php` in a text editor, fill in your information, and save it as `wp-config.php`.

In all likelihood, these items were supplied to you by your Web Host. If you do not have this information, then you will need to contact them before you can continue. If you're all ready…

- Configure DNS
- Install and configure Apache
- Install and configure MySQL
- Install and configure PHP
- Create a database
- Provide the credentials
- Install and configure memcached
- ElasticSearch
- So many plugins

WordPress.com

PANTHEON

WP engine

GoDaddy

Amazon Lightsail

SAMSUNG

HUAWEI

verizon

AT&T

AWS, MongoDB, and the Economic Realities of Open Source

Monday, January 14, 2019

In 1999, music industry revenue in the United States peaked at $14.6 billion (all numbers are from the RIAA). It is important to be precise, though, about what was being sold:

- $12.8 billion was from the sale of CDs
- $1.1 billion was from the sale of cassettes
- $378 million was from the sale of music videos on physical media
- $222.4 million was from the sale of CD singles

In short, the music industry was primarily selling plastic discs in jewel cases; the music encoded on those discs was a means of differentiating those pieces of plastic from other ones, but music itself was not being sold.

STRATECHERY

The Open Source Conundrum

Thus we have arrived at a conundrum for open source companies:

- MongoDB leveraged open source to gain mindshare.
- MongoDB Inc. built a successful company selling additional tools for enterprises to run MongoDB.
- More and more enterprises don't want to run their own software: they want to hire AWS (or Microsoft ••• or Google) to run it for them, because they value performance, scalability, and availability.

This leaves MongoDB Inc. not unlike the record companies after the advent of downloads: what they sold was not software but rather the tools that made that software usable, but those tools are increasingly obsolete as computing moves to the cloud. And now AWS is selling what enterprises really want.

WIRED

BUSINESS

KLINT FINLEY BUSINESS 06.02.17 07:00 AM

DIVERSITY IN OPEN SOURCE IS EVEN WORSE THAN IN TECH OVERALL

TODAY'S WORLD RUNS on open source software. The web, smartphones, the Amazon Echo, your car—

The privilege of free time in Open Source

Open Source communities often incorrectly believe that everyone can contribute. Unfortunately, not everyone has equal amounts of free time to contribute.

In Open Source, there is a long-held belief in meritocracy, or the idea that the best work rises to the top, regardless of who contributes it. The problem is that a meritocracy assumes an equal distribution of time for everyone in a community.

Strategy

RICHARD P. RUMELT
"A giant in the field of strategy." —McKinsey Quarterly
GOOD STRATEGY
BAD STRATEGY
The Difference and Why It Matters

"

A good strategy has an essential logical structure that I call the kernel. The kernel of a strategy contains three elements: a diagnosis, a guiding policy, and coherent action. The guiding policy specifies the approach to dealing with obstacles called out in the diagnosis. Coherent actions are feasible coordinated policies, resource commitments, and actions designed to carry out the guiding policy.

SvN

Basecamp

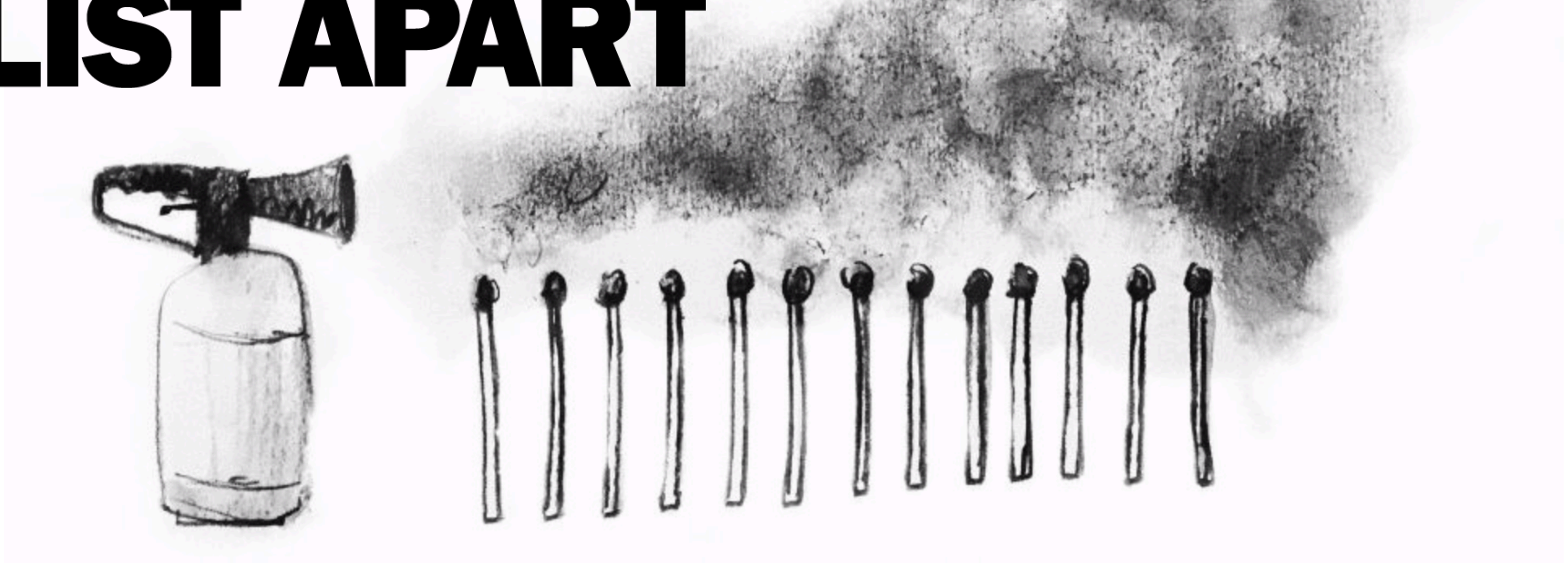
The Majestic Monolith

DHH / FEBRUARY 29, 2016 / 3 COMMENTS

Some patterns are just about the code. If your code looks like this, and you need it to do that, here's what to do. You'd do well to study such patterns, as they give you a deep repertoire of solutions ready to apply and make your code better every time you hit their context.

Then there are other patterns that are less about the code and more about how the code is being written, by whom, and within which organization. The

A LIST APART



Responsible JavaScript: Part I

by **Jeremy Wagner** · March 28, 2019

Published in Application Development, JavaScript

By the numbers, JavaScript is a performance liability. If the trend persists, the median page will be shipping at least 400 KB of it before too long, and that's merely what's *transferred*. Like other text-based resources, JavaScript is almost always served compressed—but that might be the only thing we're getting consistently

PUBLIC MONEY PUBLIC CODE

ABOUT REASONS TAKE ACTION RESOURCES SPREAD THE WORD LANGUAGE

Already **25815 SIGNATURES** – sign the open letter now!

PUBLIC MONEY PUBLIC CODE

Why is software created using taxpayers' money not released as Free Software?

We want legislation requiring that publicly financed software developed for the public sector be made publicly available under a Free and Open Source Software licence. If it is public money, it should be public code as well.

Code paid by the people should be available to the people!

Thank You

Brad Griffith

Code Wrangler, Automattic

brad.griffith@automattic.com