EXPLORING THE NEW DRUPAL FRONT-END WITH JAVASCRIPT

Hey, I'm
DRIES BUYTAERT
Founder & Project Lead, Drupal
Co-Founder and CTO, Acquia
@Dries
https://dri.es

Hey, I'm
DRIES BUYTAERT

Disclaimer: *I'm not a front-end developer*

Founder & Project Lead, ***Drupal***

Co-Founder and CTO, ***Acquia***

@Dries
https://dri.es

WHAT I'LL TALK ABOUT

1
History of JavaScript

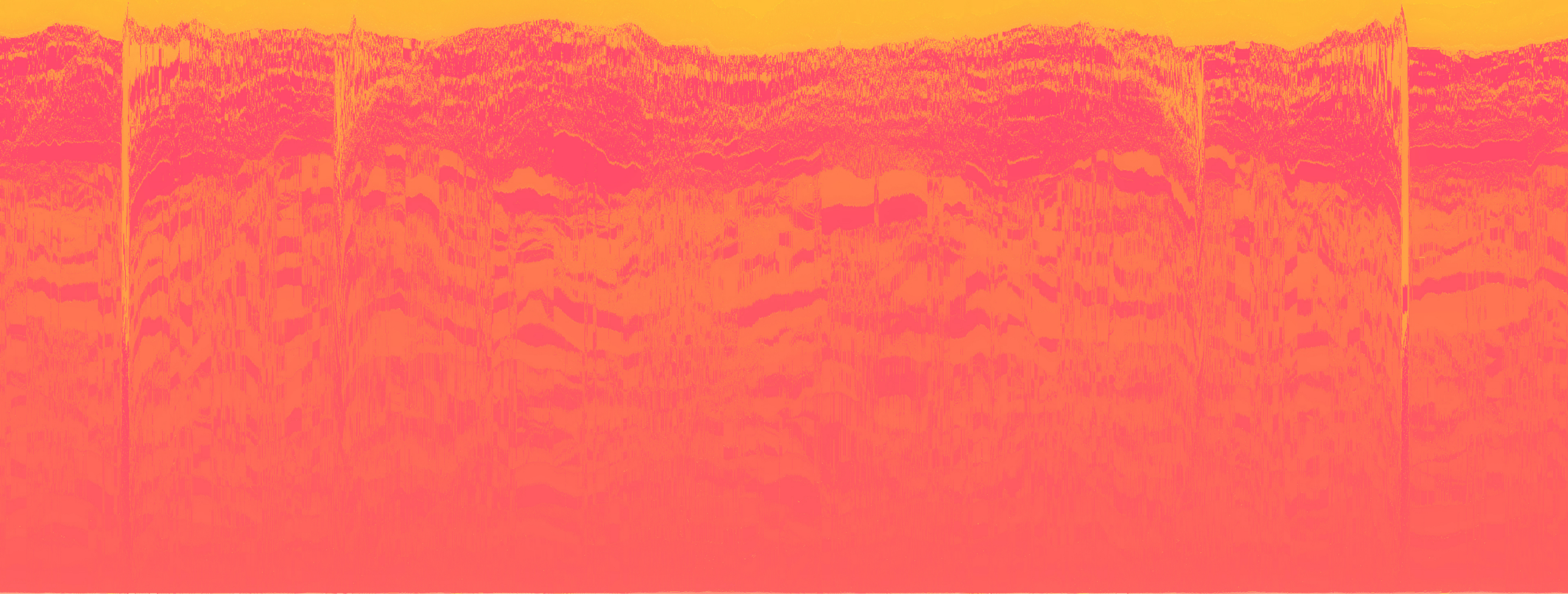
2
History of Drupal

3
API-First Drupal

4
JavaScript-driven Drupal admin interface

A brief history of JavaScript across the stack

*Did you know that JavaScript was created in just **10 days**?*



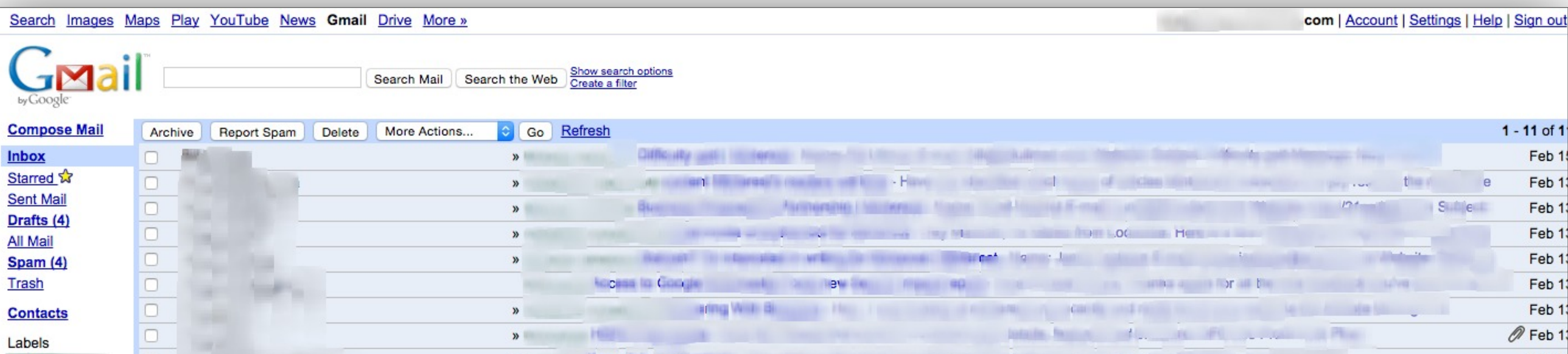
May 1995

Brendan Eich wrote the first version of JavaScript in 10 days while working at Netscape

For the first 10 years of JavaScript's life,
professional programmers denigrated JavaScript
because its target audience consisted of ***“amateurs"***

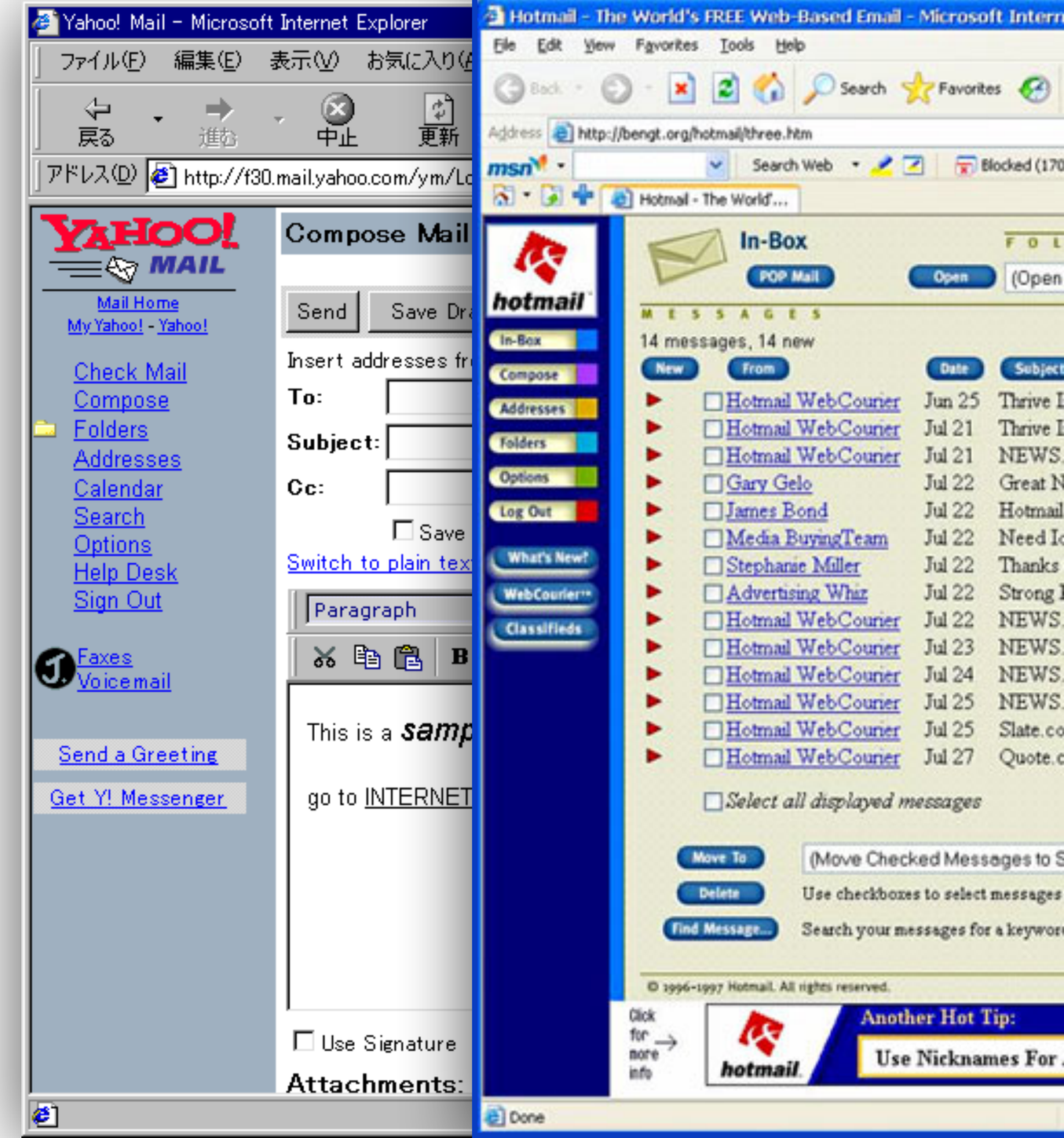
That changed in 2004 with the launch of Gmail

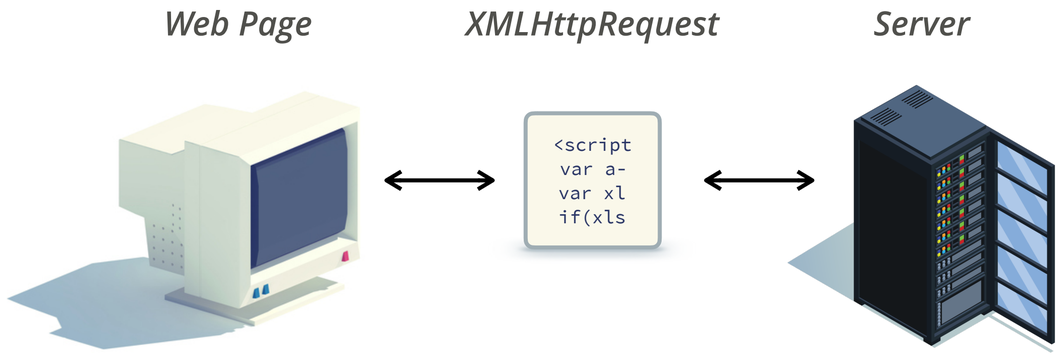
Gmail was the first popular web application that really showed off what was possible with client-side JavaScript



Competing e-mail services such as Yahoo! Mail and Hotmail featured extremely slow interfaces that used server-side rendering almost exclusively

Almost every action by the user requiring the server to ***reload the entire web page***





Gmail began to work around these limitations by using XMLHttpRequest; it allowed Gmail to load data in the background (asynchronously), avoiding a page refresh

Gmail's use of JavaScript ***caught the attention*** *of developers around the world*

2005

Google launched Google Maps, which used the same technology as Gmail to transform online maps into an interactive experience

Google's XMLHttpRequest approach in Gmail and Google Maps ultimately came to be called Ajax
(originally "Asynchronous JavaScript and XML")

This resulted in a renaissance period of JavaScript usage spearheaded by open source libraries and the communities that formed around them

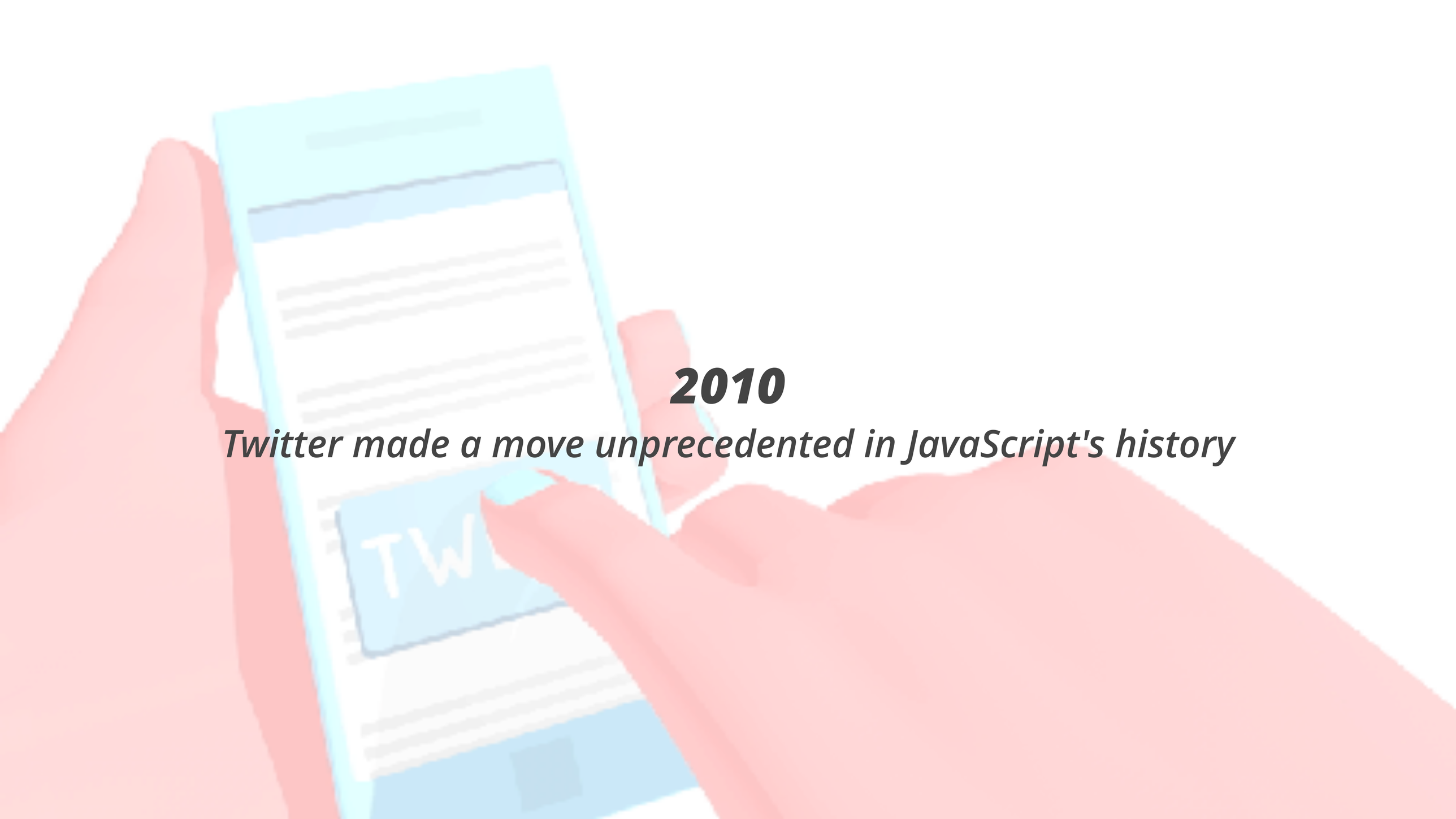
Did you know that Drupal helped put jQuery on the map?

(Drupal added jQuery to Drupal core as early as 2006)

*In 2008, Google launched Chrome with a faster JavaScript engine called **V8**. The release announcement read:*

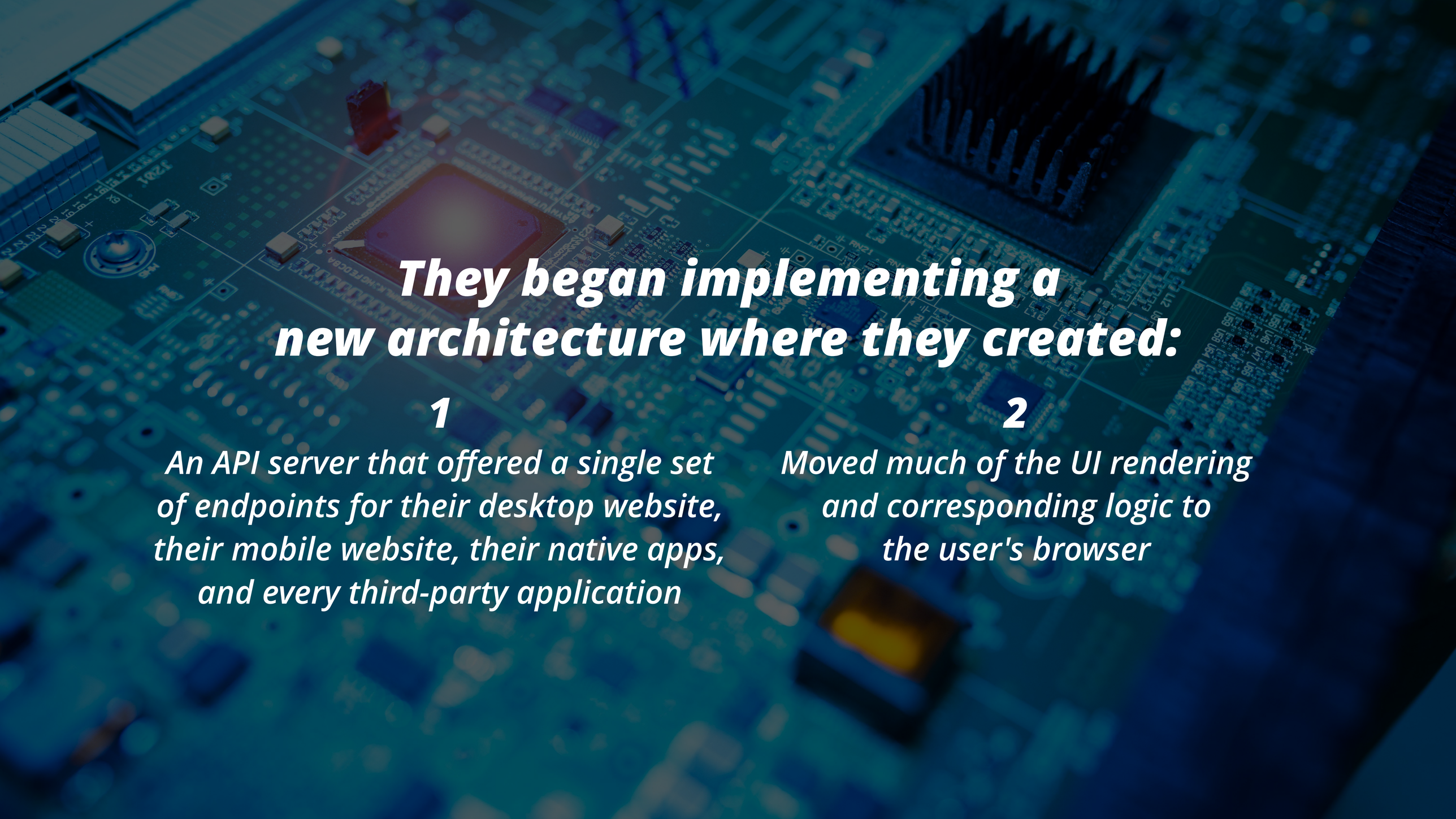
"We also built a more powerful JavaScript engine, to power the next generation of web applications that aren't even possible in today's browsers"

At the launch, ***V8 improved JavaScript performance by 10x over Internet Explorer*** *by compiling JavaScript code to native machine code before executing it*



2010

Twitter made a move unprecedented in JavaScript's history



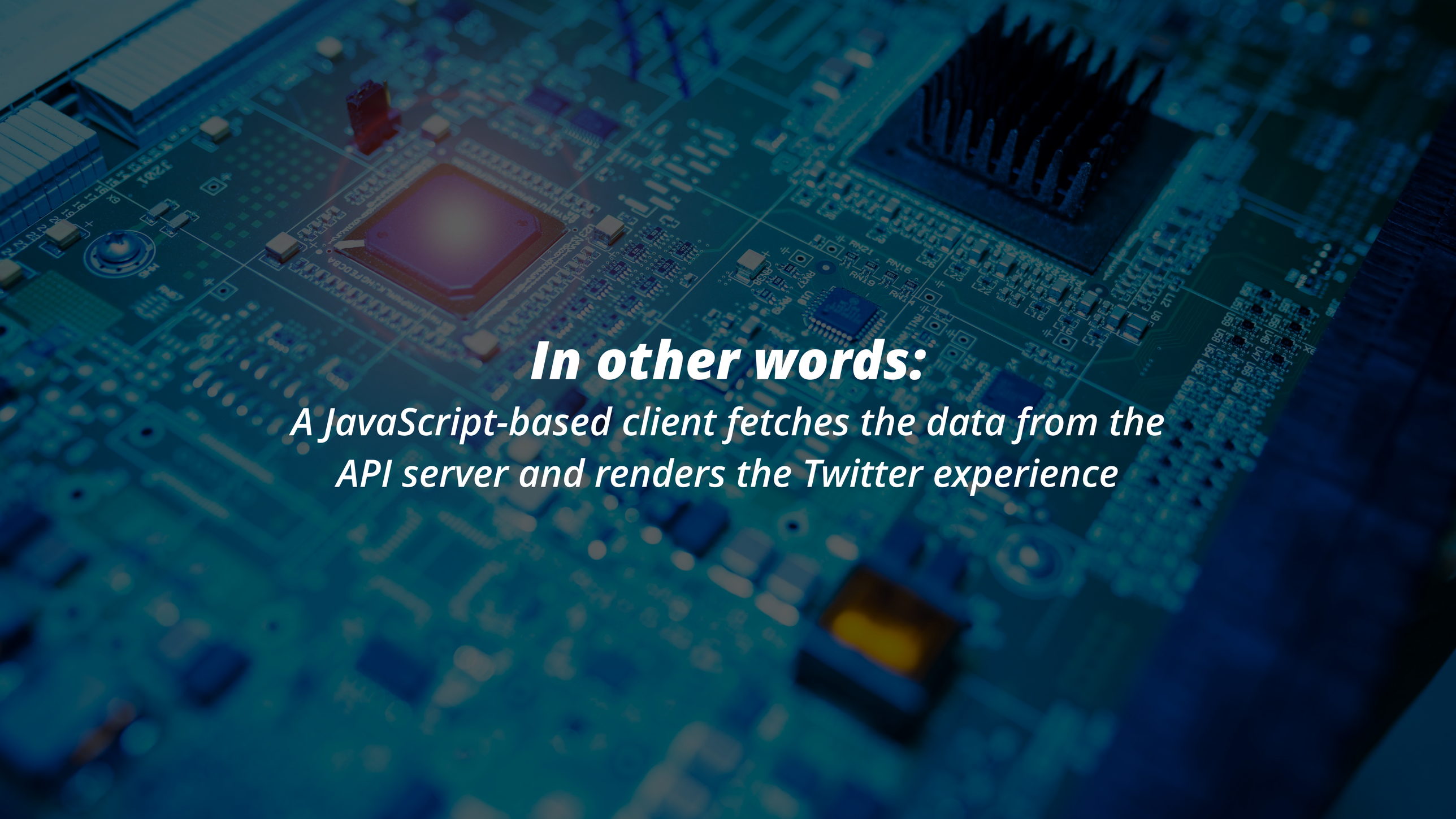
They began implementing a new architecture where they created:

1

An API server that offered a single set of endpoints for their desktop website, their mobile website, their native apps, and every third-party application

2

Moved much of the UI rendering and corresponding logic to the user's browser



In other words:

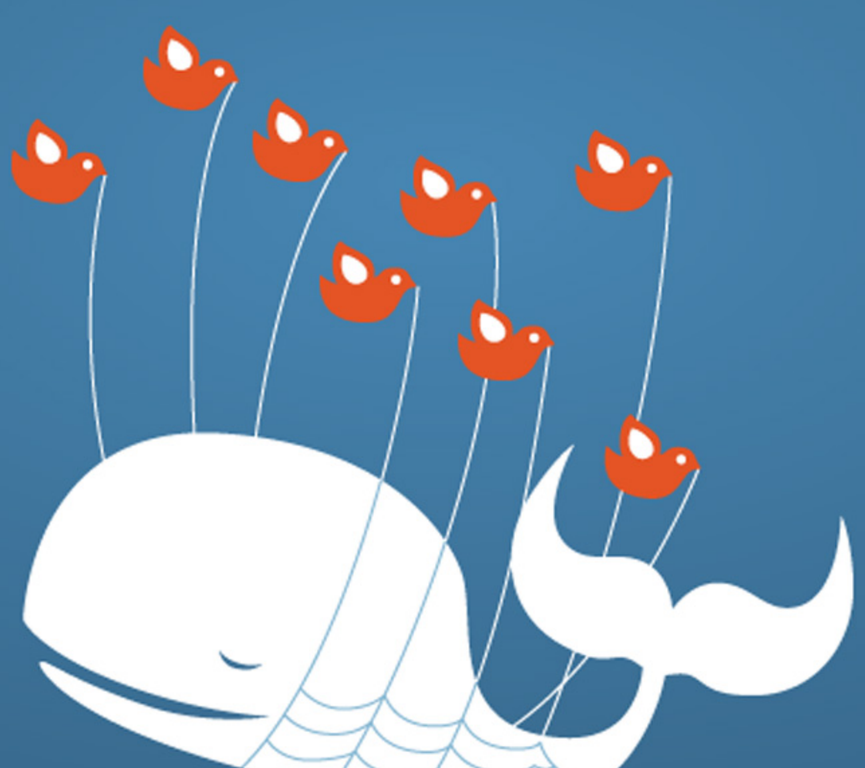
A JavaScript-based client fetches the data from the API server and renders the Twitter experience

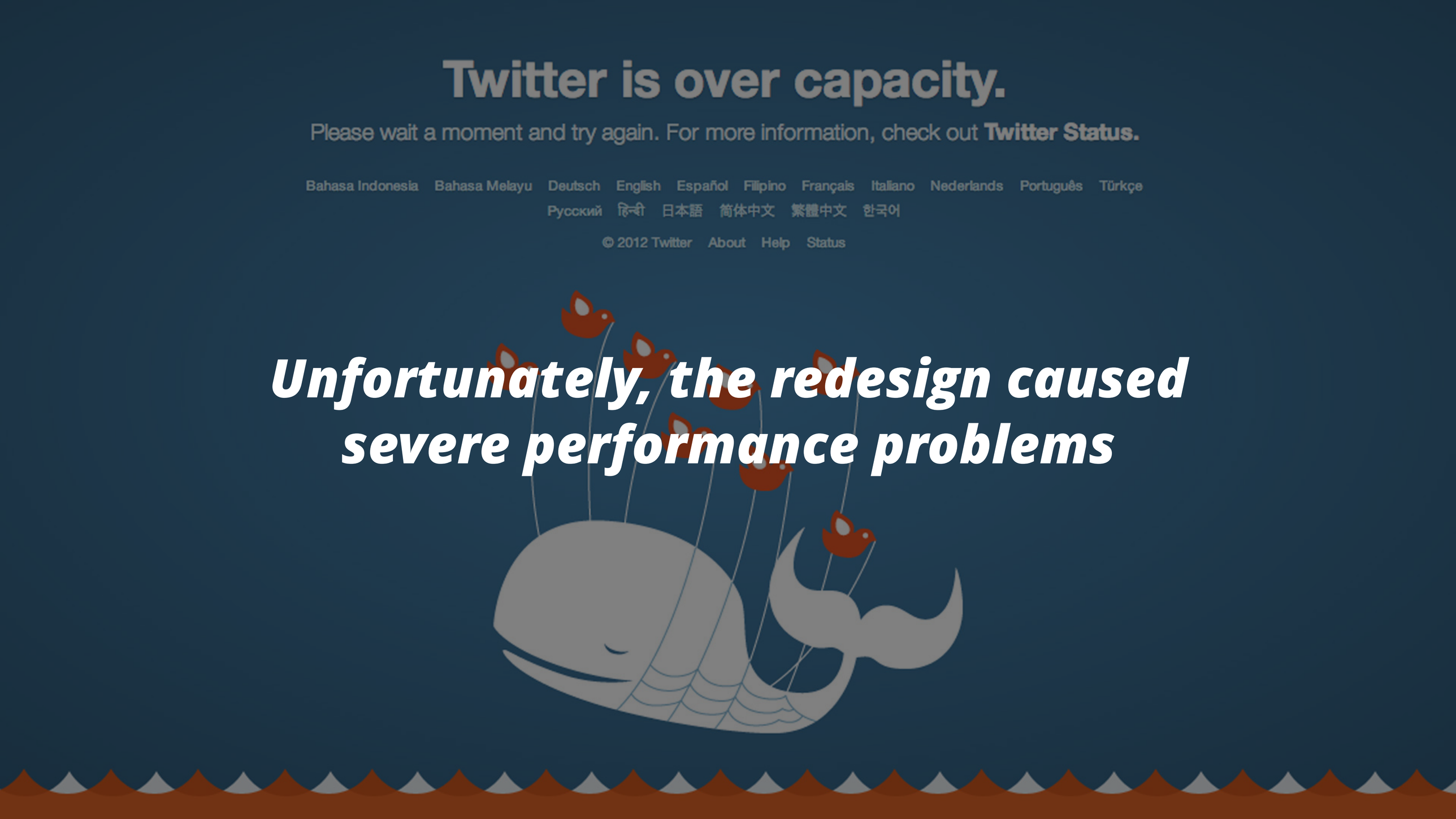
Twitter is over capacity.

Please wait a moment and try again. For more information, check out **Twitter Status.**

Bahasa Indonesia Bahasa Melayu Deutsch English Español Filipino Français Italiano Nederlands Português Türkçe
Русский हिन्दी 日本語 简体中文 繁體中文 한국어

© 2012 Twitter About Help Status





Unfortunately, the redesign caused severe performance problems

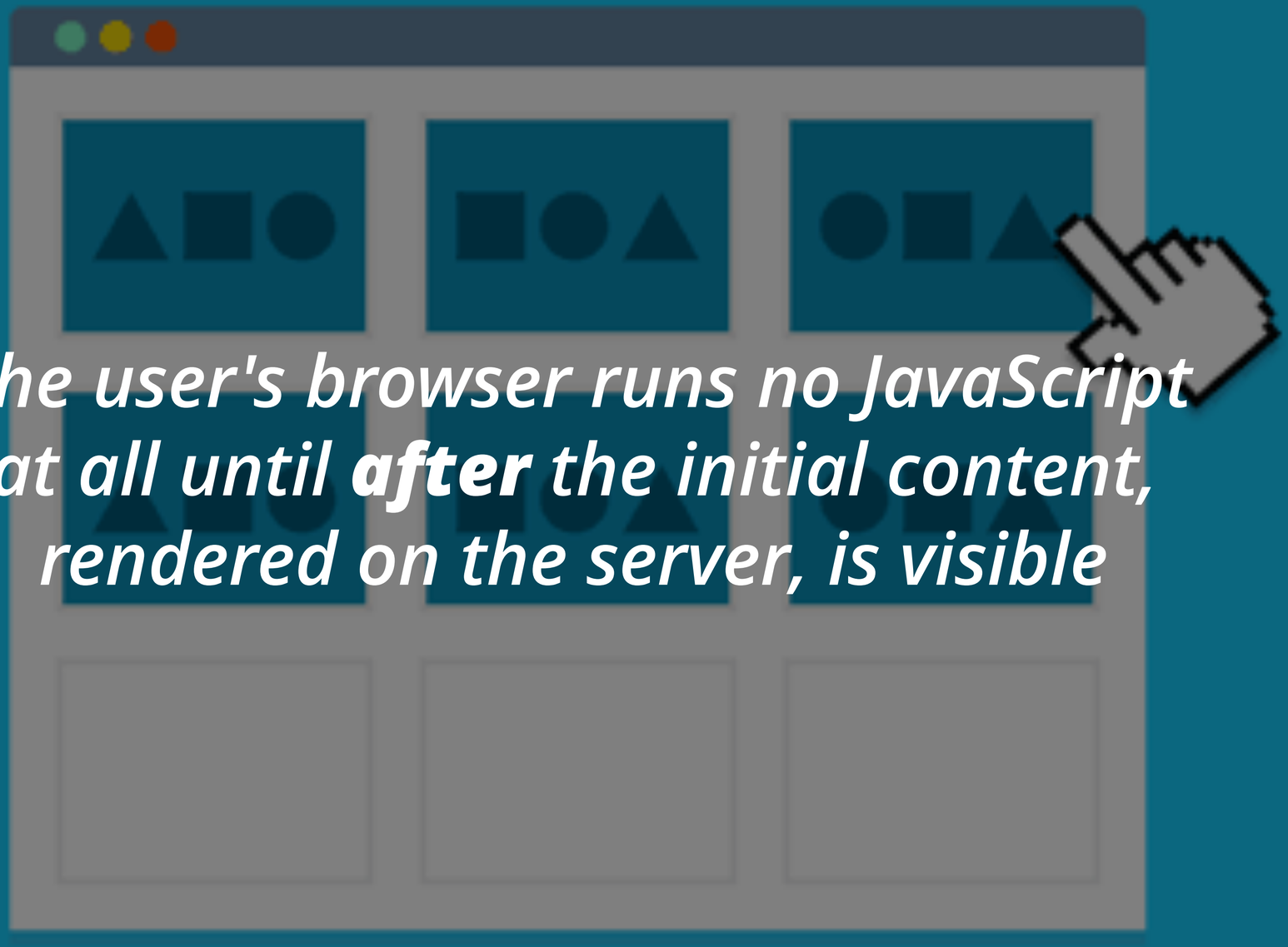
The "time to first interaction" was poor

Lots of JavaScript had to be downloaded, parsed and executed by the user's browser before anything of substance was visible

*Twitter suffered from these performance problems for almost **two years***

*Finally in 2012, **Twitter reversed course** by passing more of the rendering from the client back to the server*

The revised architecture ***renders the initial pages on the server***
but uses ***client-side JavaScript to add the interactive experience***



*The user's browser runs no JavaScript at all until **after** the initial content, rendered on the server, is visible*

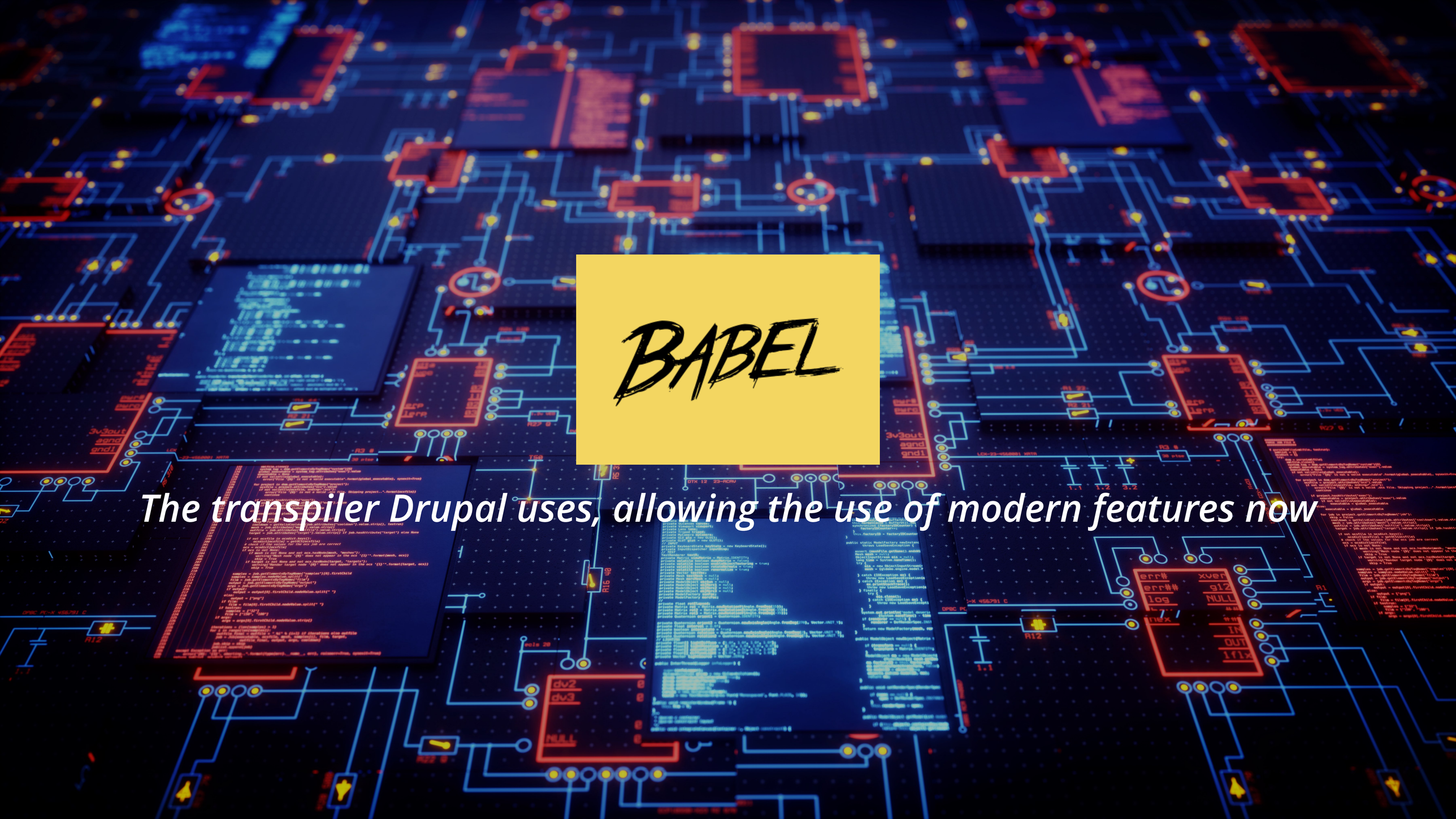
WIN!

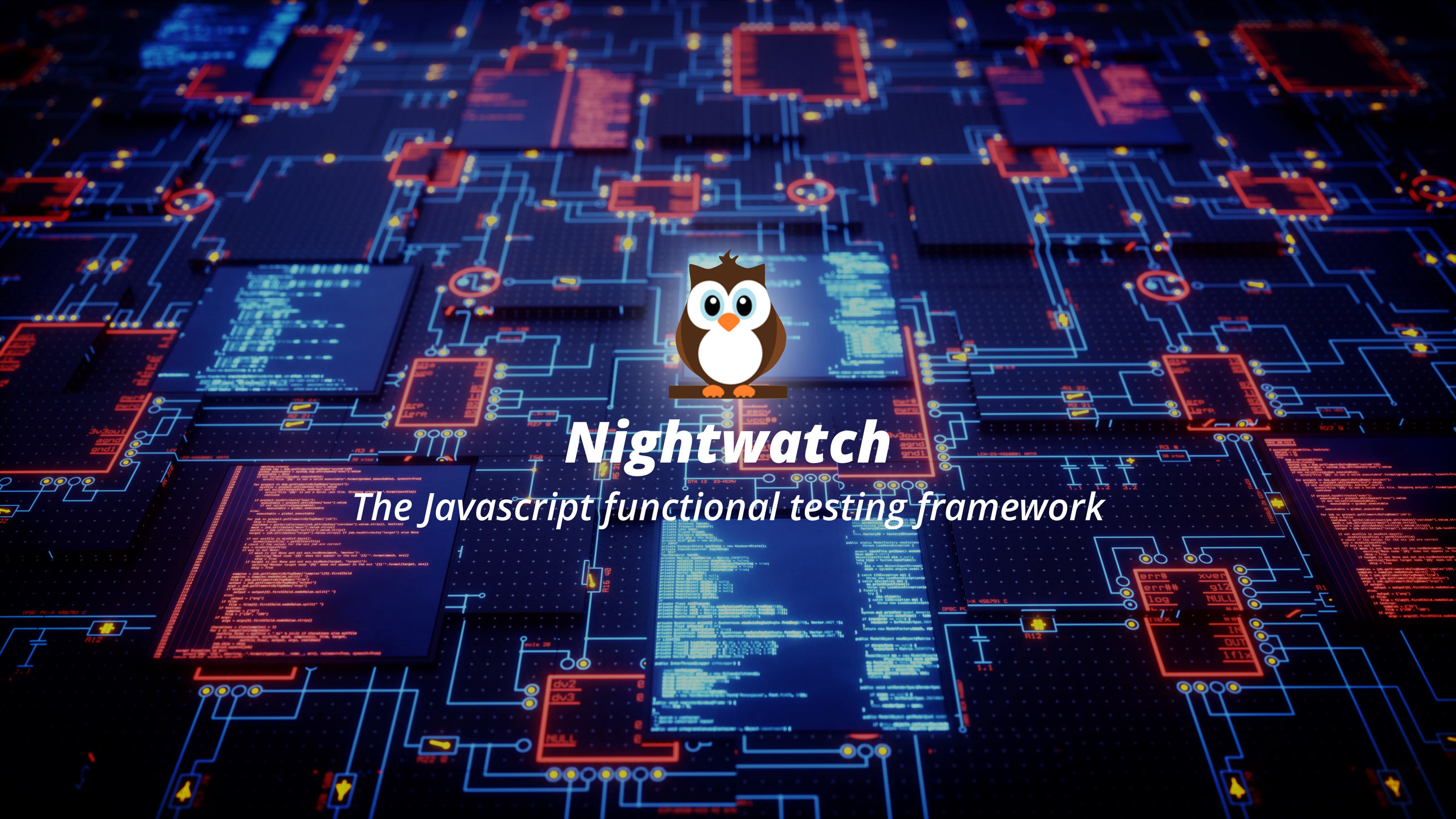
This new hybrid architecture reduced Twitter's page load time by ***80%***

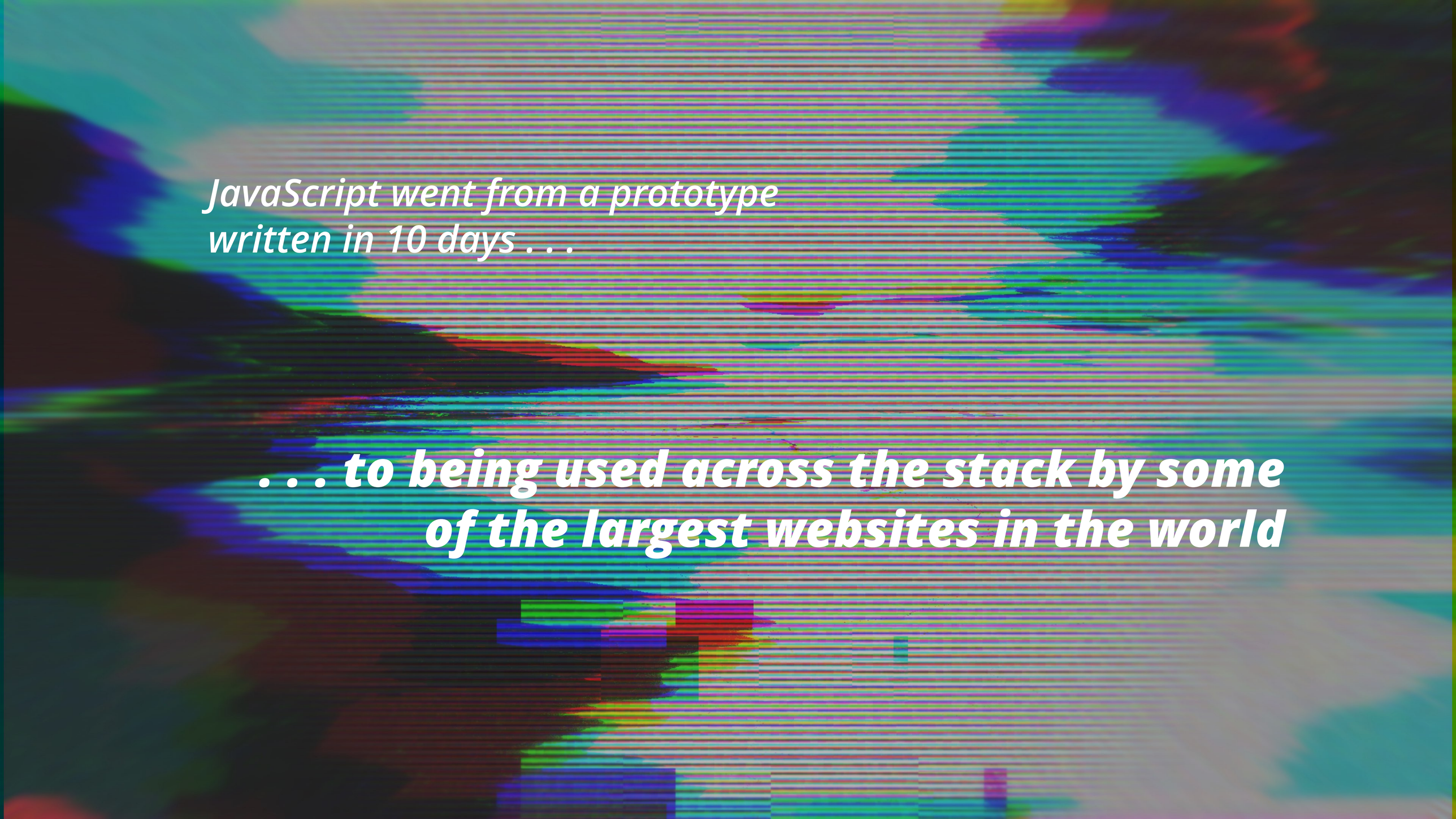
In 2013,
Airbnb was the first
to use Node.js to provide
isomorphic (also called universal
or simply shared) JavaScript

The biggest advantage Airbnb's JavaScript isomorphism had over Twitter's approach is the notion of ***a completely reusable rendering system***

WHERE JAVASCRIPT
IS RIGHT NOW


BABEL
The transpiler Drupal uses, allowing the use of modern features now


Nightwatch
The Javascript functional testing framework


JavaScript went from a prototype written in 10 days . . .
. . . to being used across the stack by some of the largest websites in the world

A short history of Drupal

I started Drupal from my dorm room 18 years ago

*HTML 4 was **just released** and **CSS was all the rage***

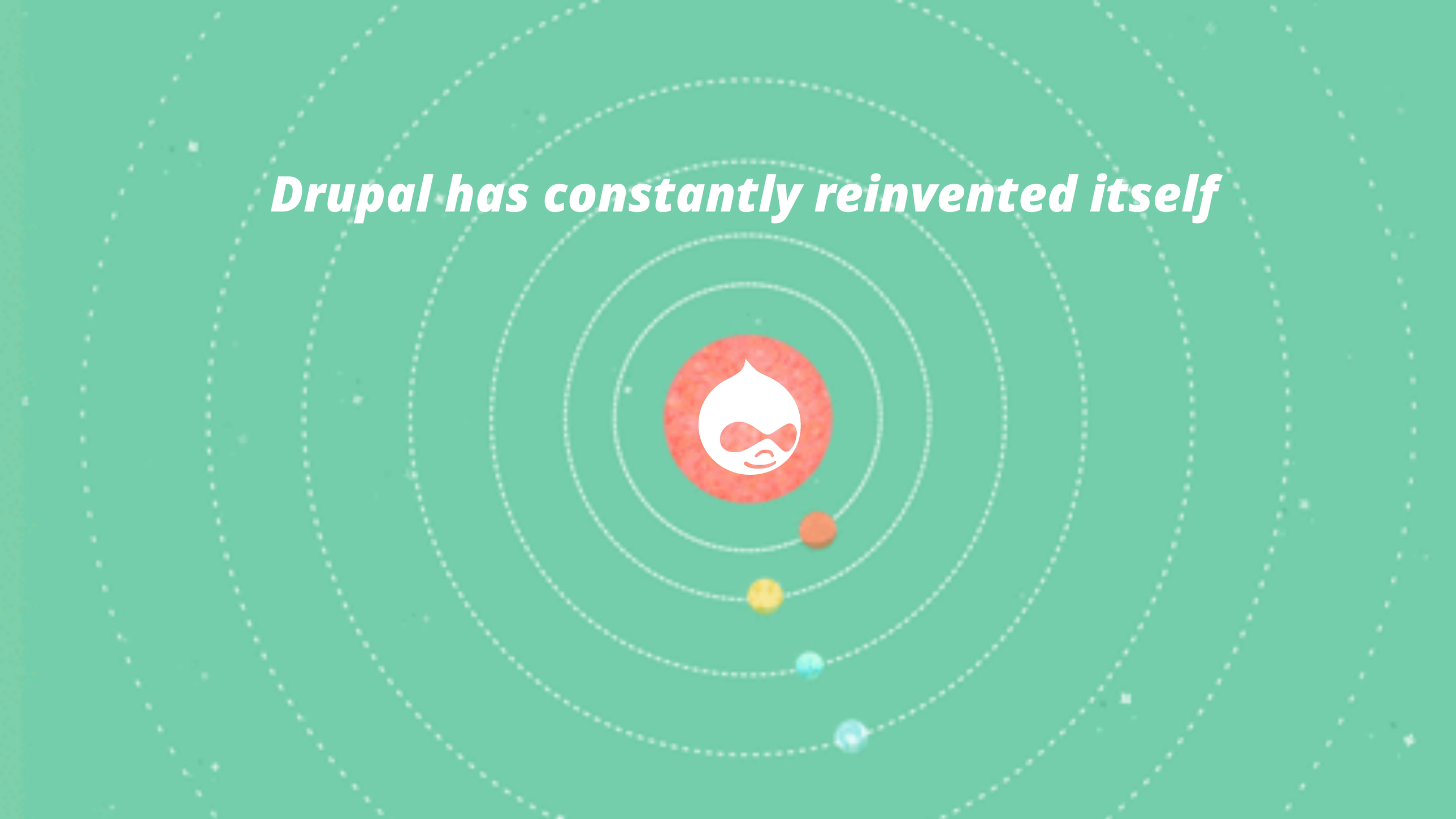
JavaScript was
for amateurs
A:\>dir
Volume in drive A has no label
Volume Serial Number is 2A87-6CE1
Directory of A:\
COMMAND COM 93,040 04-10-05 5:54p
DISPLAY SYS 17,175 04-10-05 5:54p
EGA CPI 58,870 04-10-05 5:54p
EGA2 CPI 58,870 04-10-05 5:54p
EGA3 CPI 58,753 04-10-05 5:54p
KEYB COM 21,607 04-10-05 3:04a
KEYBOARD SYS 34,566 04-10-05 5:54p
KEYBRD2 SYS 31,942 04-10-05 5:54p
KEYBRD3 SYS 31,633 04-10-05 5:54p
KEYBRD4 SYS 13,014 04-10-05 5:54p
MODE COM 29,239 04-10-05 5:54p
11 file(s) 448,709 bytes
0 dir(s) 885,344 bytes free
A:\>

Google was in its infancy

The mobile web didn’t exist

Social media wasn't invented until five years later

Drupal has constantly reinvented itself

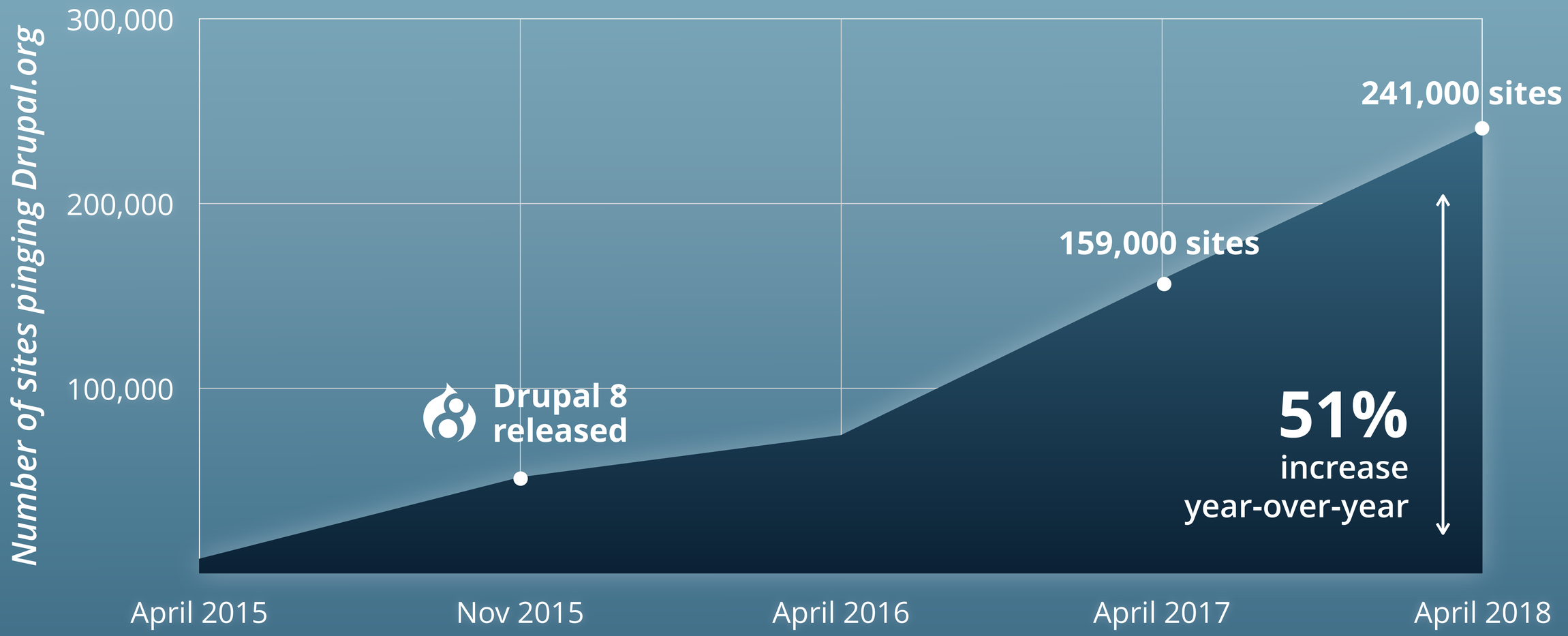


7,000+
code contributors
in the last year
3%
of web
1M+
registered users
on drupal.org
35,000
contributors
FREE
13M+
unique visitors
to drupal.org
in the last year
15,000+
modules
1 OUT OF 40
sites in the world run on Drupal

“Each morning we are born again.
What we do today is what matters most.”

Drupal 8 released
November 2015
8.0

Drupal 8 sites have grown **51% in the last year**



Two main focus areas

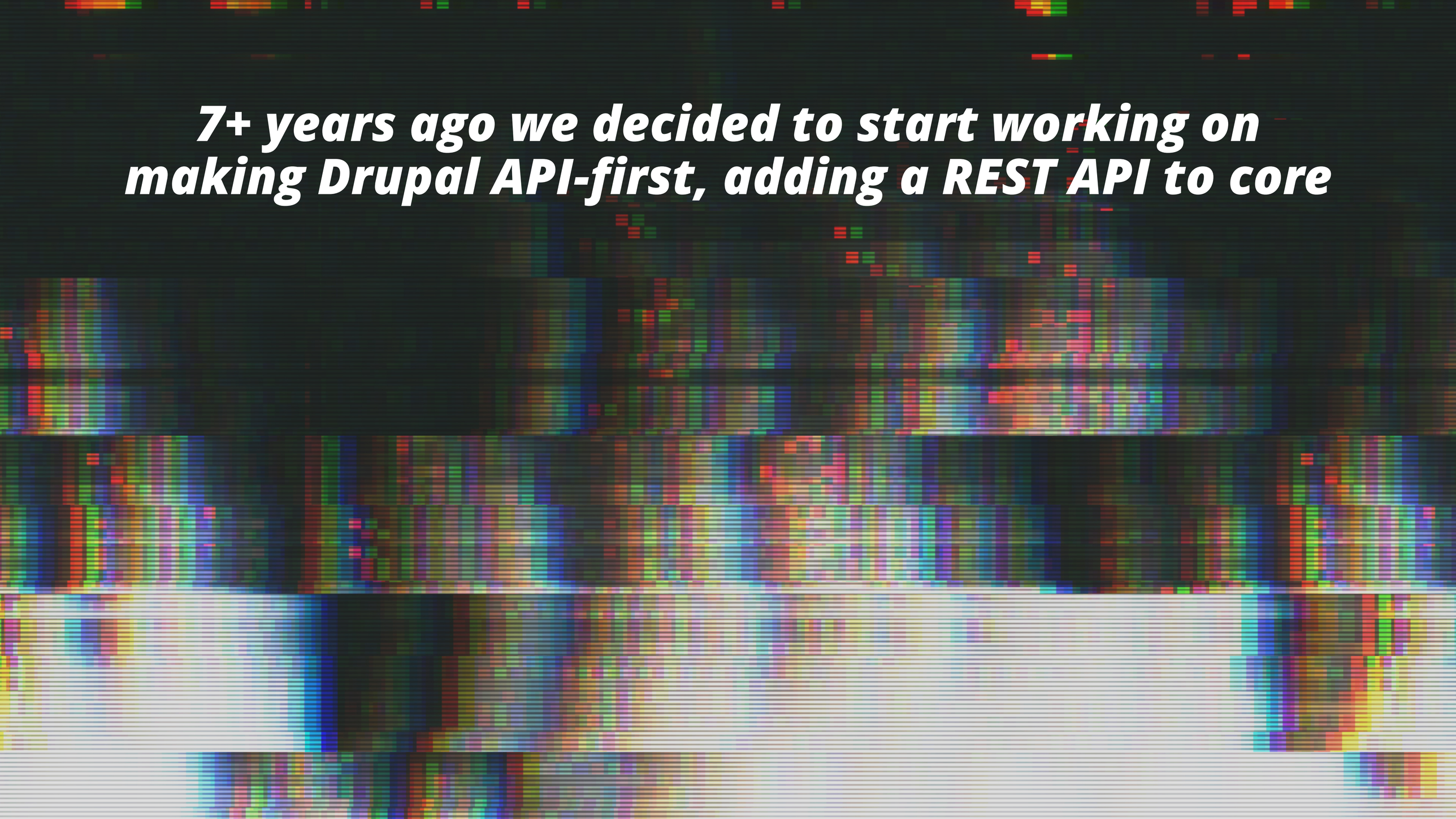
1

Drupal 8 is becoming easier to use for content creators, and site builders

2

Drupal 8 marks the start of Drupal's evolution to an API-first platform

Working toward an API-first Drupal

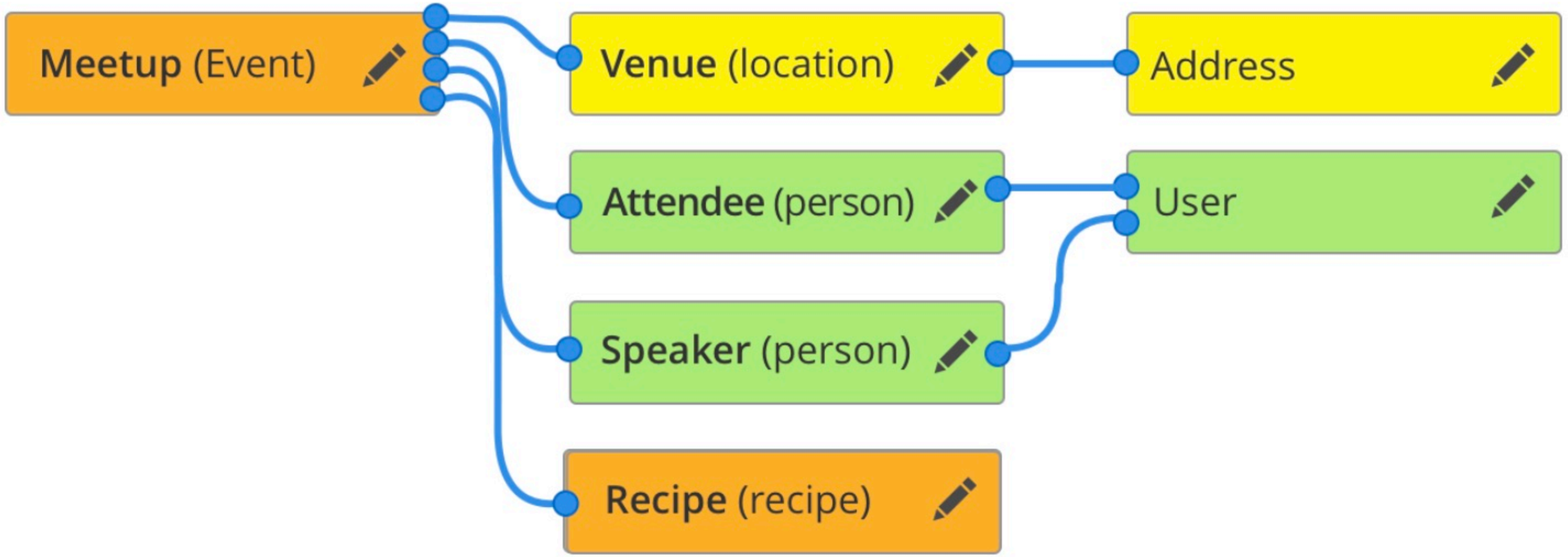

7+ years ago we decided to start working on
making Drupal API-first, adding a REST API to core

Today, Drupal's REST support is rich and mature

We've been focused on support for all entity types, configuration entity support, user login/logout/registration support, custom REST plugins, and most recently, file upload support!

REST API + JSON API + GraphQL

Working with Drupal's entity graph



JSON API and GraphQL can query linked data
(e.g. Drupal entities and entity references) in a single query

Drupal as a content service for building decoupled applications

As a result, more and more organizations are
building decoupled applications served by Drupal

Chupachups UK

Powdr

Princess Cruises

yha
Youth Hostel Association

New Balance

Gault & Millau

Concern Worldwide

Legacy

The Weather Channel

Warner Music Group

Crossfit Games

Thrillist

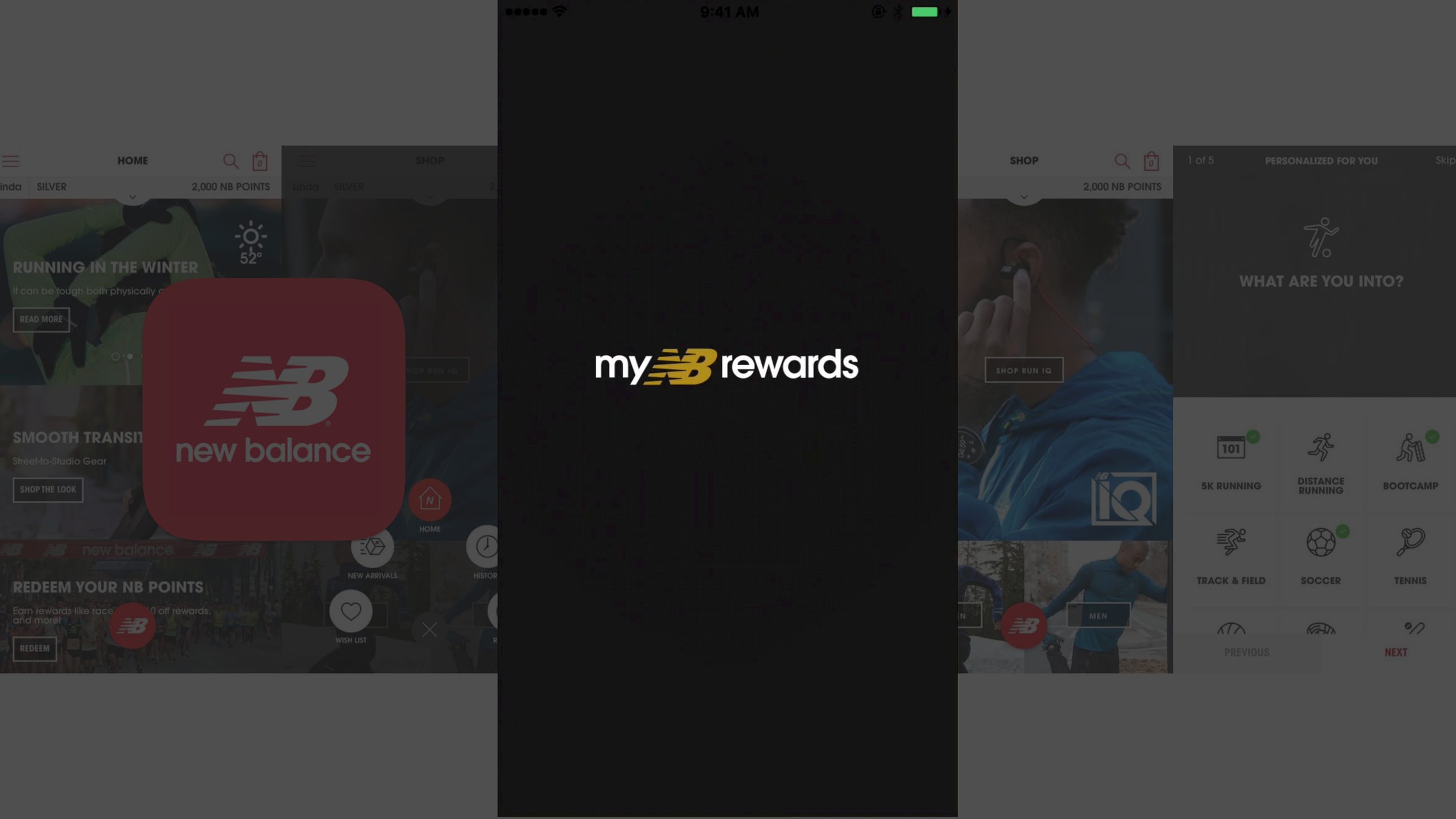
The Dodo

French Fédération
of Athletics

Casa Modelo

teach.org

New Balance


9:41 AM
HOME
SILVER
2,000 NB POINTS
RUNNING IN THE WINTER
READ MORE
52°
SMOOTH TRANSIT
Street-to-Studio Gear
SHOP THE LOOK
new balance
REDEEM YOUR NB POINTS
REDEEM
SHOP
SHOP RUN IQ
HOME
NEW ARRIVALS
WISH LIST
MEN
1 of 5
PERSONALIZED FOR YOU
Skip
WHAT ARE YOU INTO?
5K RUNNING
DISTANCE RUNNING
BOOTCAMP
TRACK & FIELD
SOCCER
TENNIS
PREVIOUS
NEXT
my NB rewards

Working toward a JavaScript-driven Drupal admin interface

While organizations use JavaScript frameworks to create customer-facing experiences using Drupal, Drupal's own administration interface has not yet embraced a modern JavaScript framework

Drupal needs to provide a cutting-edge experience for its own users, the content creators and site builders

*We decided to start working on **an alternative Drupal administrative UI using React** to experiment with ways we could innovate on Drupal's behind-the-scenes interfaces*

6 STEPS TOWARDS A REACT UI FOR DRUPAL'S ADMINISTRATION BACKEND

1) Stabilize the JSON API module

2) Improve our JavaScript testing infrastructure

3) Create designs for the administration UI

4. Allow contributed modules
to use React or Twig

5) Implement missing web services APIs

6) Make the React UI extensible and configurable

This work benefits
ALL decoupled builds

WHAT I TALKED ABOUT

1	2	3	4
History of JavaScript	*History of Drupal*	*API-First Drupal*	*JavaScript-driven Drupal admin interface*

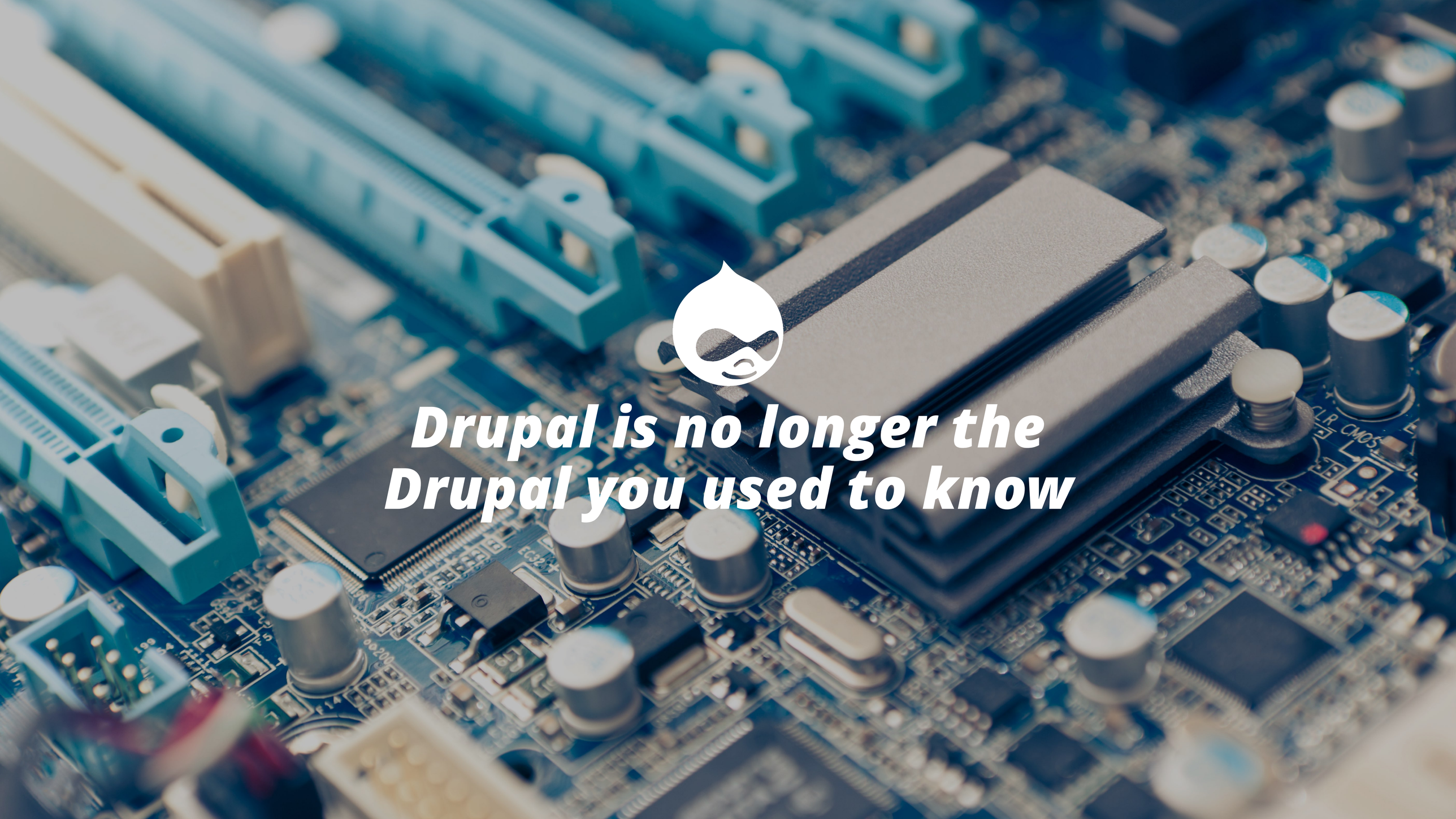
Thank you to those who are involved in this initiative from the Drupal Community

Sally Young, Lauri Eskola, Tim Plunkett, Matt Davis, Angie Byron, Daniel Wehner, Wim Leers, Gabe Sullice, Alex Pott, Cristina Chumillas, Matt Grill, Mateu Aguiló Bosch, Ted Bowman, and many more!

*Why should **you** help?*

Conclusion

We've been making steady progress the past years to move Drupal to a more API-first and JavaScript-centric world


Drupal is no longer the
Drupal you used to know

Q & A*

Find these slides at https://dri.es

*(*I'm not a front-end developer)*