



WHAT ARE YOU

TALKING ABOUT



JavaScript

/'dʒɑ:vəskrɪpt/

1. an object-oriented computer programming language commonly used to create interactive effects within web browsers.

Web

/wɛb/

1. the World Wide Web (*www*, *W3*) is an information system of interlinked hypertext documents that are accessed via the Internet and built on top of the Domain Name System. It has also commonly become known simply as the *Web*.



Framework

/'freɪmwə:k/

1. an essential supporting structure of a building, vehicle, or object.
2. a basic structure underlying a system, concept, or text.





The present



Modern day JS web app

+You **Search** Images Maps Play YouTube News Gmail More ▾ Sign in | 

A faster way to browse the web  [Install Google Chrome](#) 

Google

Advanced search Language tools

[Google Search](#) [I'm Feeling Lucky](#)

 Google gives you the tools to control your privacy and security. [Visit My Account](#)

[Advertising Programs](#) [Business Solutions](#) [+Google](#) [About Google](#)

© 2015 - [Privacy](#) - [Terms](#)

... under the hood

```
<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lan
window.google.vel.lu&&window.google.vel.lu(a),d.src=a,google.li=g+1});goog
google.j.b=(!!location.hash&&!!location.hash.match('[#&]((q|fp)=|tbs=rimg|
||(google.j.qbp==1);(function(){window.google.sn='webhp';google.timers={};
(function(){'use strict';var g=this,aa=function(a){var d=typeof a;if("obj
a.originalTarget!=b)return!0;(a=b.tagName in ea)|| (a=b.getAttributeNode("t
b.key)&&"CHECKBOX"!=k)|| (q=u(b),k=(q.getAttribute("role")||q.tagName).toUp
p.createEventObject(b)}catch(va){y=b}else y=b;e.event=y;a.v.push(e})if("to
(window['gbar']=window['gbar']||{})._CONFIG=[[[_0,"www.gstatic.com","og.og2
try{
var ga,ha;_.aa=_._aa||{};_.m=this;_.n=function(a){return void 0!==a};_.p=fu
_.ca=function(a){var c=typeof a;if("object"==c)if(a){if(a instanceof Array
else if("function"==c&&"undefined"==typeof a.call)return"object";return c}
ha=function(a,c,d){if(!a)throw Error();if(2<arguments.length){var e=Array.
_.v=function(a,c){var d=a.split("."),e=_.m;d[0]in e||!e.execScript||e.exec.
```



The past



Beginnings at Netscape

JavaScript was originally developed by *Brendan Eich* in *1995*, while he was working for *Netscape Communications Corporation*.





20 Things that just turned 20



Show Me
CABLES





Old school JS web app





Welcome to the IntraNet...

InGen

Building tomorrow out of yesterday...

Copyright © 1997, Universal City Studios, Inc.
IntraNet designed by Spin Cycle Entertainment.



Page source

```
<!--This file created 3/4/97 4:39 PM by Claris Home Page version 2.0-->
<HTML>
<HEAD>
  <TITLE>International Genetics Technologies</TITLE>
  <META NAME=GENERATOR CONTENT="Claris Home Page 2.0">
  <X-SAS-WINDOW TOP=57 BOTTOM=861 LEFT=284 RIGHT=814>
</HEAD>
<BODY background="graphics/butn_bg.gif">

<P ALIGN= CENTER><CENTER><TABLE BORDER=0>
<TR>
  <TD WIDTH=100>
    <P ALIGN= CENTER><A HREF="hr/hwelcome.htm"
      TARGET="files"><IMG SRC="graphics/butn_hum.gif" ALT="Humans"
      WIDTH=60 HEIGHT=66 X-SAS-UseImageWidth X-SAS-UseImageHeight
```



S_wrapper.js

```
s_linkInternalFilters="javascript:,.,/";  
  
var s_account="nbcuglobal,nbcufilmd,nbcuhomeentbu";  
var s_prop8 ="Film";  
var s_prop9 ="Home Entertainment";  
var s_prop10="JP: Lost World";  
  
document.write('<s'+'cript src="http://homevideo.universalstudios.com/s_cod
```



s_code.js

```
/* SiteCatalyst code version: H.2.  
Copyright 1997-2005 Omniture, Inc. More info available at  
http://www.omniture.com */  
/* Specify the Report Suite ID(s) to track here */  
var s=s_gi(s_account)  
***** CONFIG SECTION *****  
/* You may add or alter any code config here. */  
/* E-commerce Config */  
s.currencyCode="USD"  
/* Link Tracking Config */  
s.trackDownloadLinks=true  
s.trackExternalLinks=true  
s.trackInlineStats=true  
s.linkDownloadFileTypes="exe,zip,wav,mp3,mov,mpg,avi,wmv,doc,pdf,xls"  
s.linkLeaveQueryString=true
```

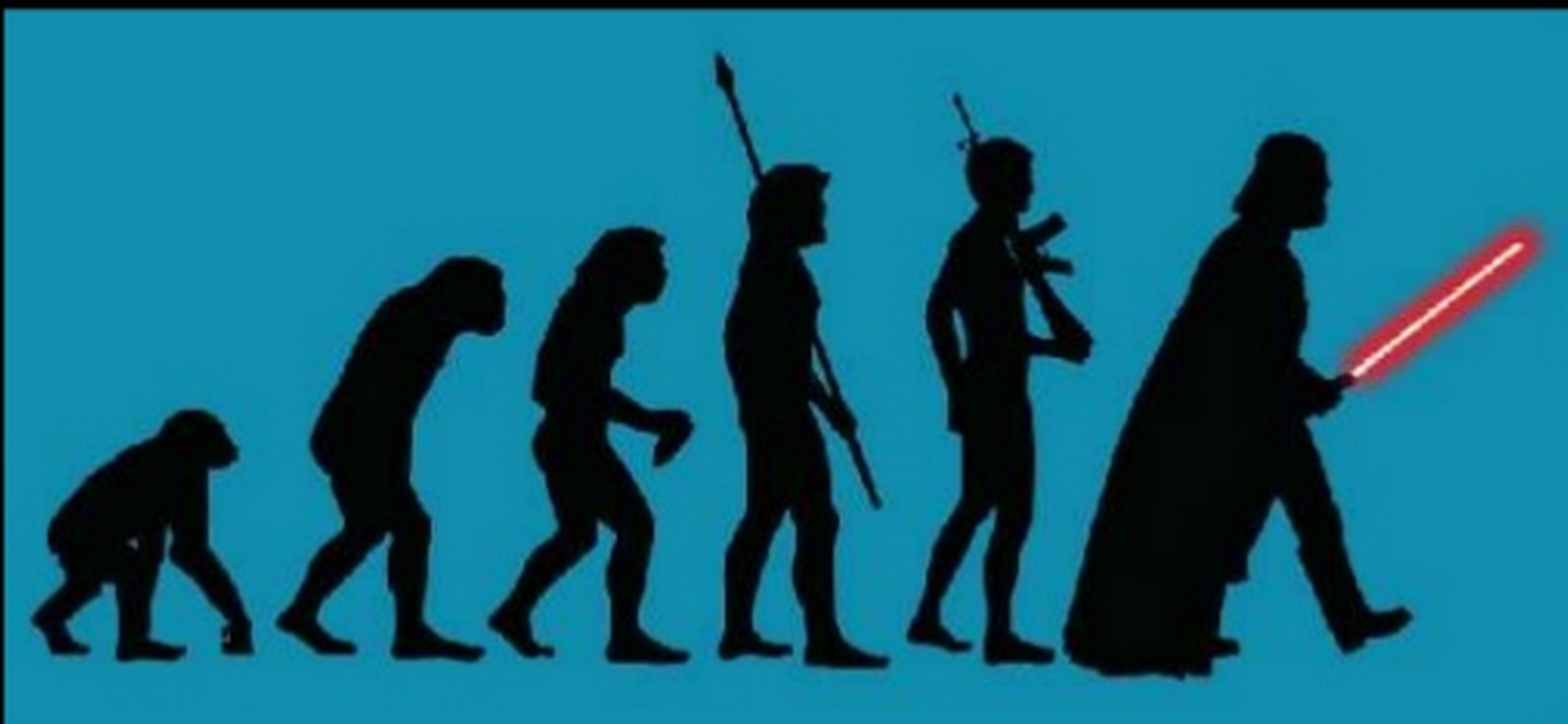


How was a JS web app built then?

- Lot's of JavaScript in separate files loaded via `<script>` tags.
- All variables, functions, etc. stored in the global `window` object.
- Cross-browser support almost non-existent - developers wrote in `Vanilla JS`.



EVOLUTION OF JS



5 key technologies **drastically** changed the
world of **JavaScript** and the **JS web app**





At the heart of **AJAX** is XMLHttpRequest

```
// This is the client-side script.  
  
// Initialize the Ajax request.  
var xhr = new XMLHttpRequest();  
xhr.open('get', 'send-ajax-data.php');  
  
// Track the state changes of the request.  
xhr.onreadystatechange = function () {  
    var DONE = 4; // readyState 4 means the request is done.  
    var OK = 200; // status 200 is a successful return.  
    if (xhr.readyState === DONE) {  
        if (xhr.status === OK) {  
            alert(xhr.responseText); // 'This is the returned text.'  
        } else {  
            alert('Error: ' + xhr.status); // An error occurred during the request.  
        }  
    }  
};
```



AJAX browser support

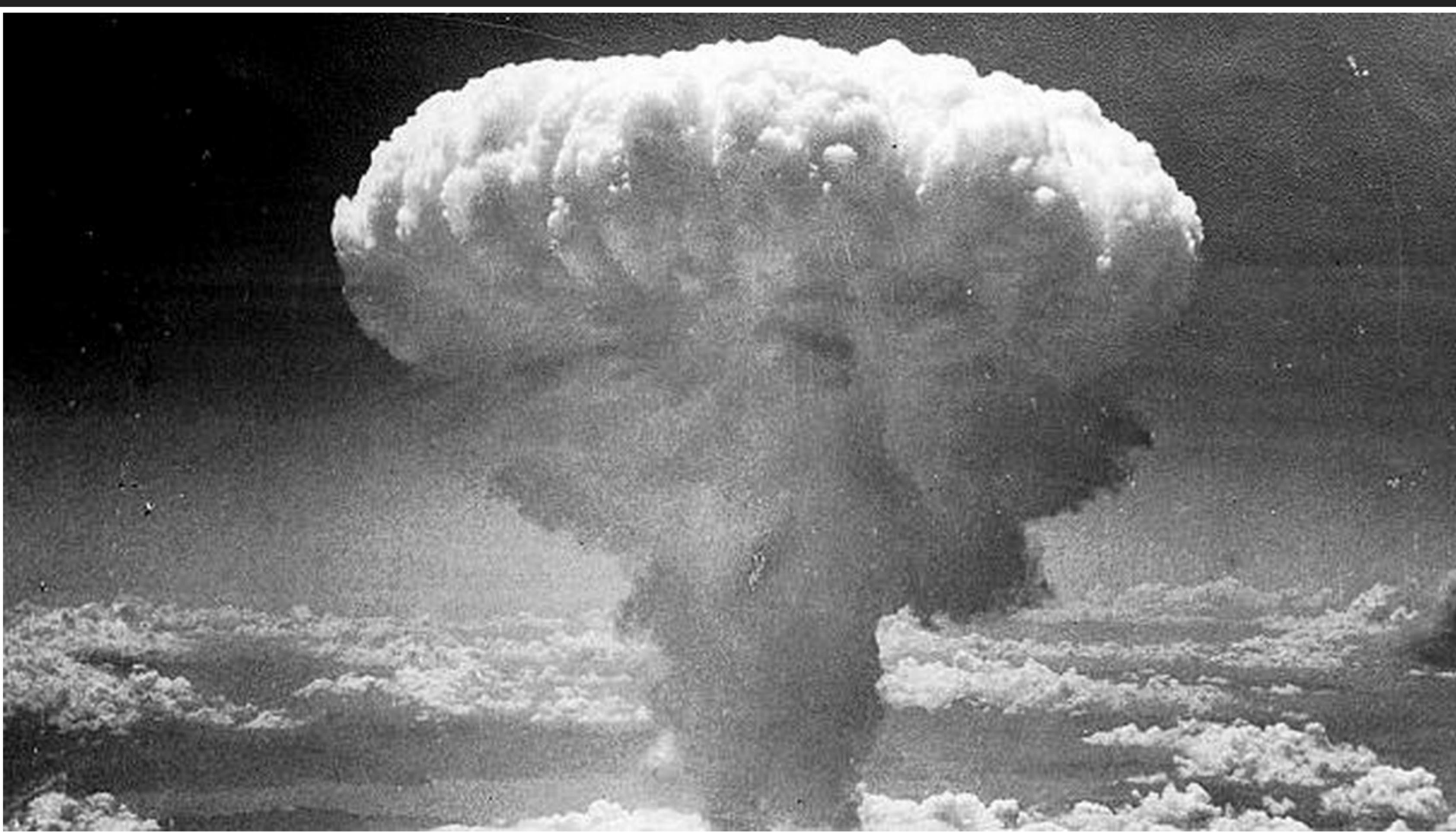
- As early as 1999, Microsoft's Internet Explorer 5 could do AJAX (back then it was implemented as XMLHTTP ActiveX control).
- In 2004, Google made a wide deployment of standards-compliant, cross browser Ajax with Gmail.
- However, the term AJAX was publicly stated on 18 February 2005 by Jesse James Garrett in an article titled [Ajax: A New Approach to Web Applications](#).
- So in reality, the **AJAX** technology became publicly available 2005+.





jQuery was released on August 26, 2006 by John Resig. The world has never been the same.





What jQuery did to the world of JavaScript

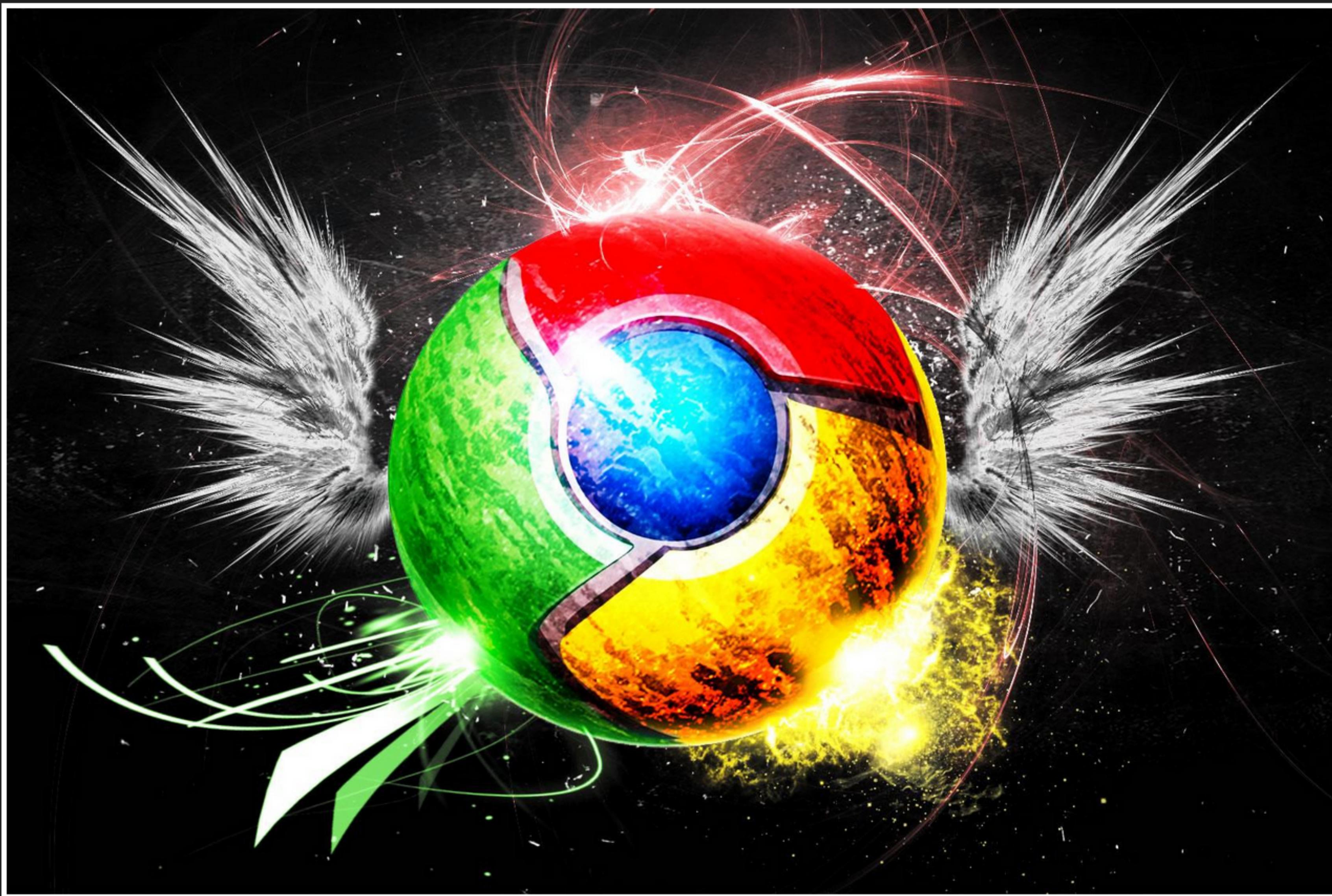
- Encourages separation of JavaScript and HTML - very easy to add **event handlers** to the **DOM**.
- Eliminates **cross-browser** incompatibilities - write once, run in any browser.
- **AJAX** helper methods - `$.get()`, `$.post()`, etc.
- Effects and animation.





Google Chrome took us by surprise in 2008.
Before that, JavaScript developers didn't really
believe in JavaScript.





The JavaScript engine V8 in Google Chrome
was carefully crafted to optimize the language
execution.



Chrome Experiments

Celebrating creative code for the web.

About Submit Mobile

Search



Featured Experiments

1-3 of 31 [Prev](#) | [Next](#)



[Ball Pool](#)

Mr.doob



[Z-Type](#)

Dominic Szablewski



[100,000 Stars](#)

Google Data Arts Team



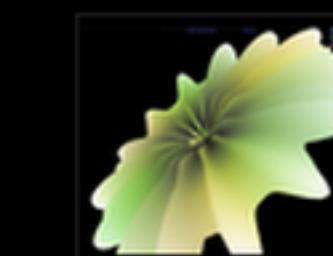
All Experiments 1-18 of 719

Show only ▾



[The Scrolling Man](#)

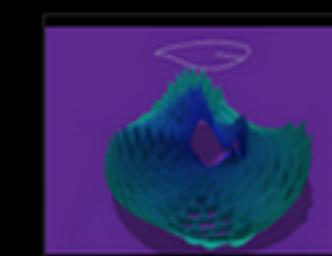
Aron Sommer



[Spherical Harmonics](#)

Distortion

Stefan H Singer



[Lilypad](#)

Manny Tan



[NessBots](#)

Ronen Ness



[HelloRun™](#)

HelloEnjoy



[Interactive 3D Skeleton](#)

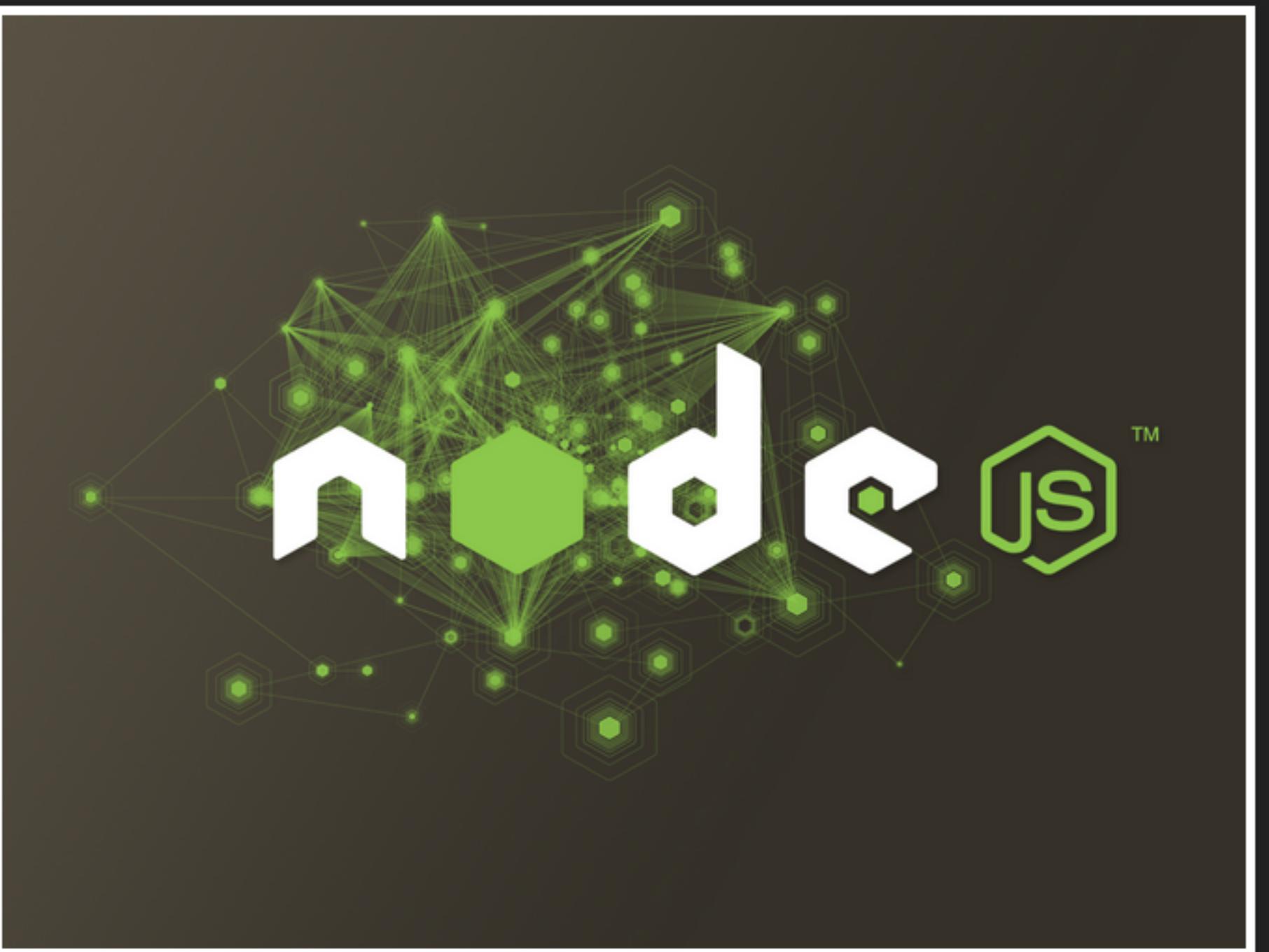
Andy Barber





Get ready

It's coming



In 2009 Ryan Dahl released Node.js. This was a very serious achievement. JavaScript spread beyond the browser.



Node.js pushed JS in 2 new directions

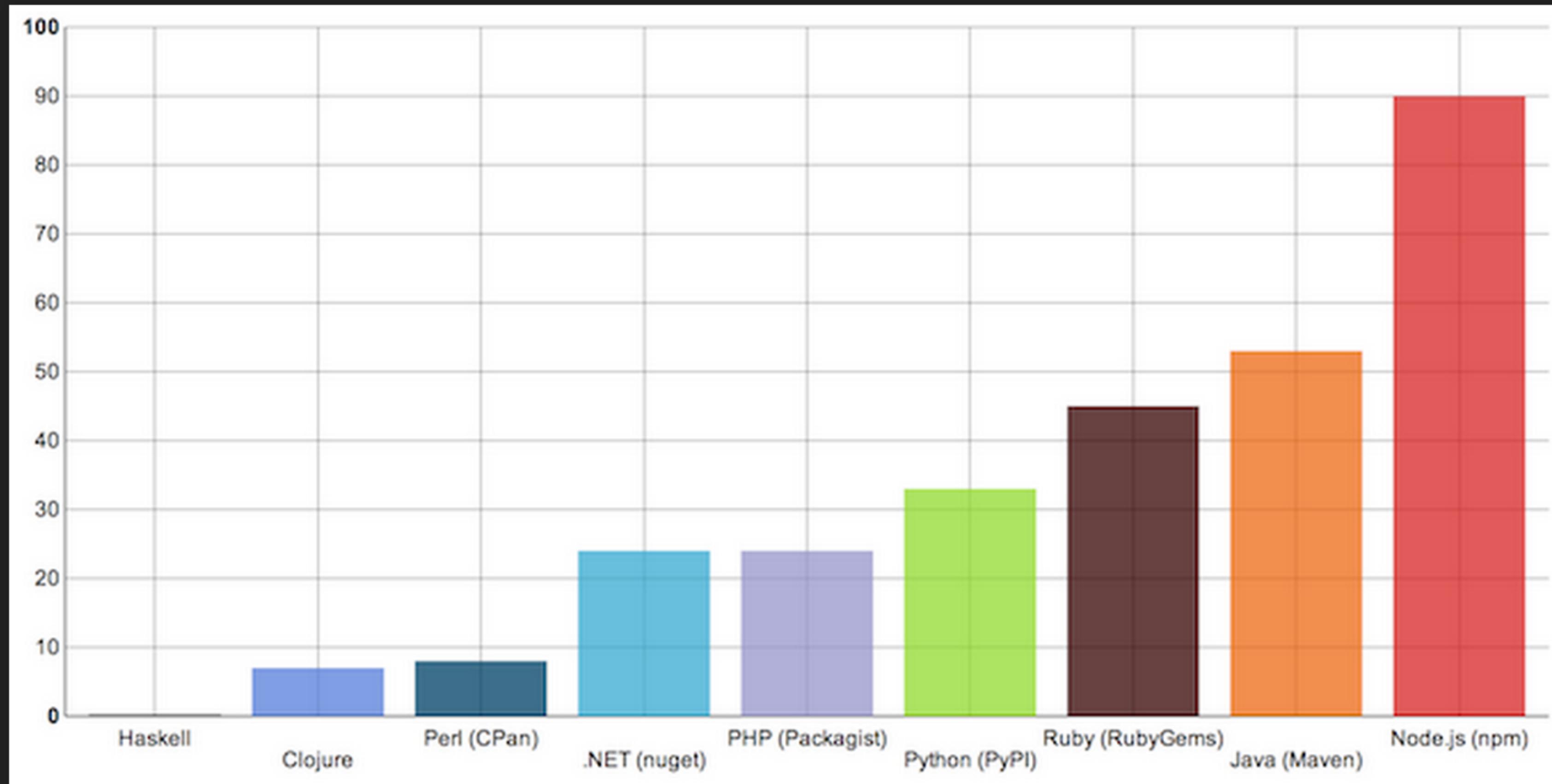
- Server logic development now became possible in JavaScript. Previously this was the area of only PHP, Java, Python, etc.
- It became possible for JavaScript developers to develop tools in JavaScript. Previously, they had to use another language (for example C, C++, etc.).



With Node.js came NPM - package manager for
JavaScript



Package growth - in the thousands





Started in 2009 Kevin Dangoor, the CommonJS project aimed at specifying APIs for a JavaScript ecosystem outside the browser.



Most importantly, the **Asynchronous module definition** emerged.





What RequireJS achieved

- Asynchronous loading of JS files - no need to have multiple `<script>` tags.
- Front-end code organization by modules. A module can be required when necessary.
- Ultimately RequireJS allowed for a sane project structure for large front-end code bases.

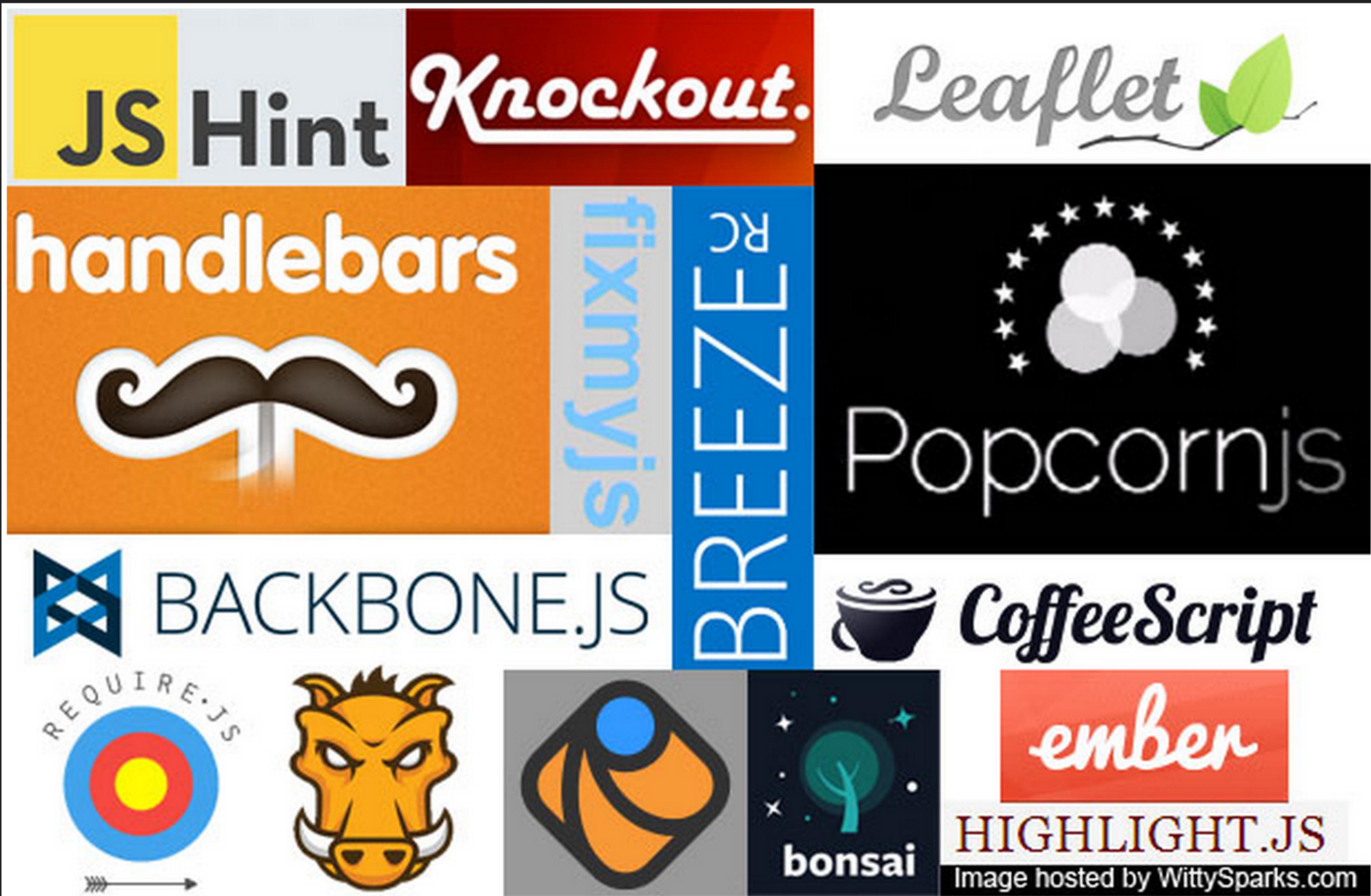




JavaScript **web** frameworks

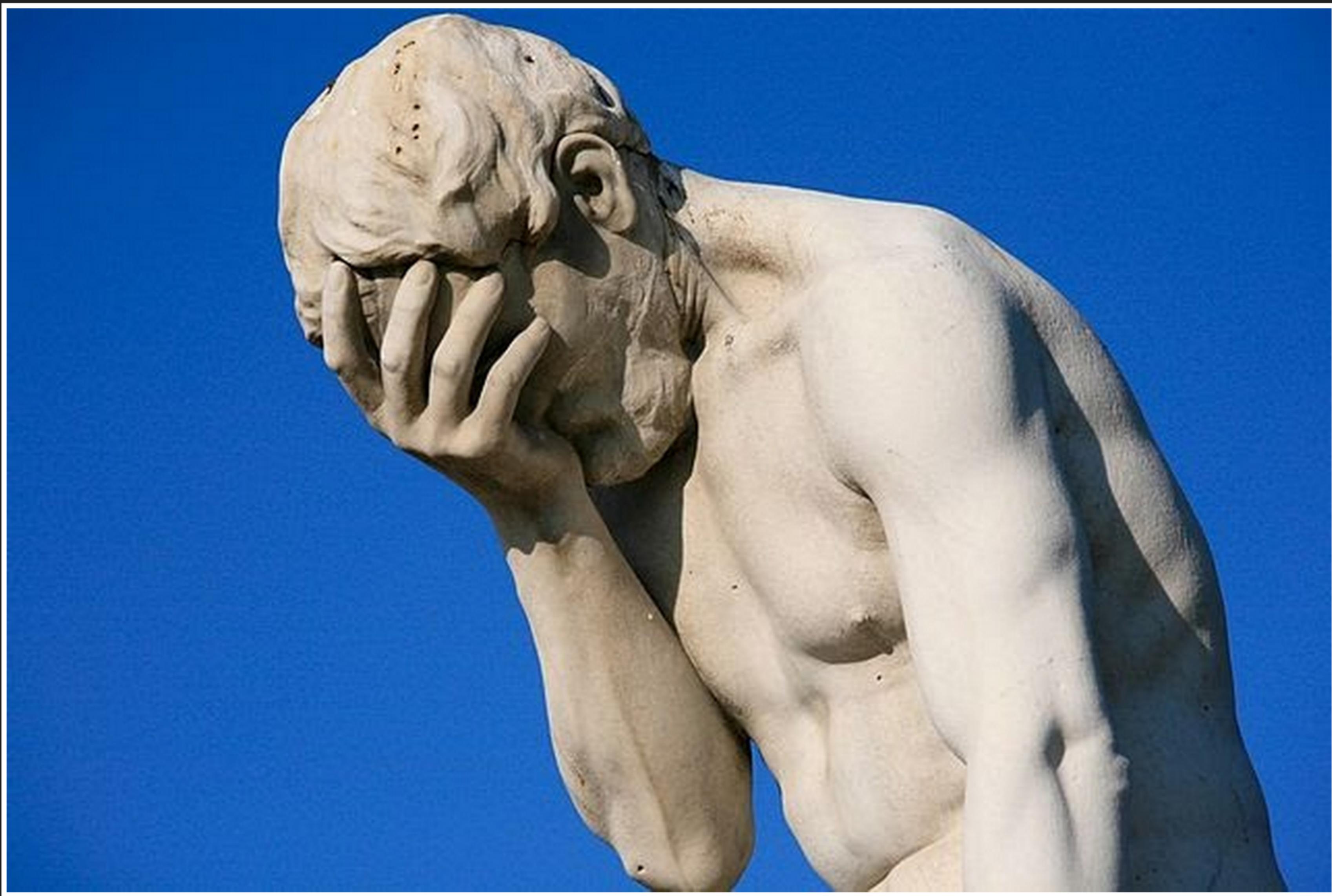






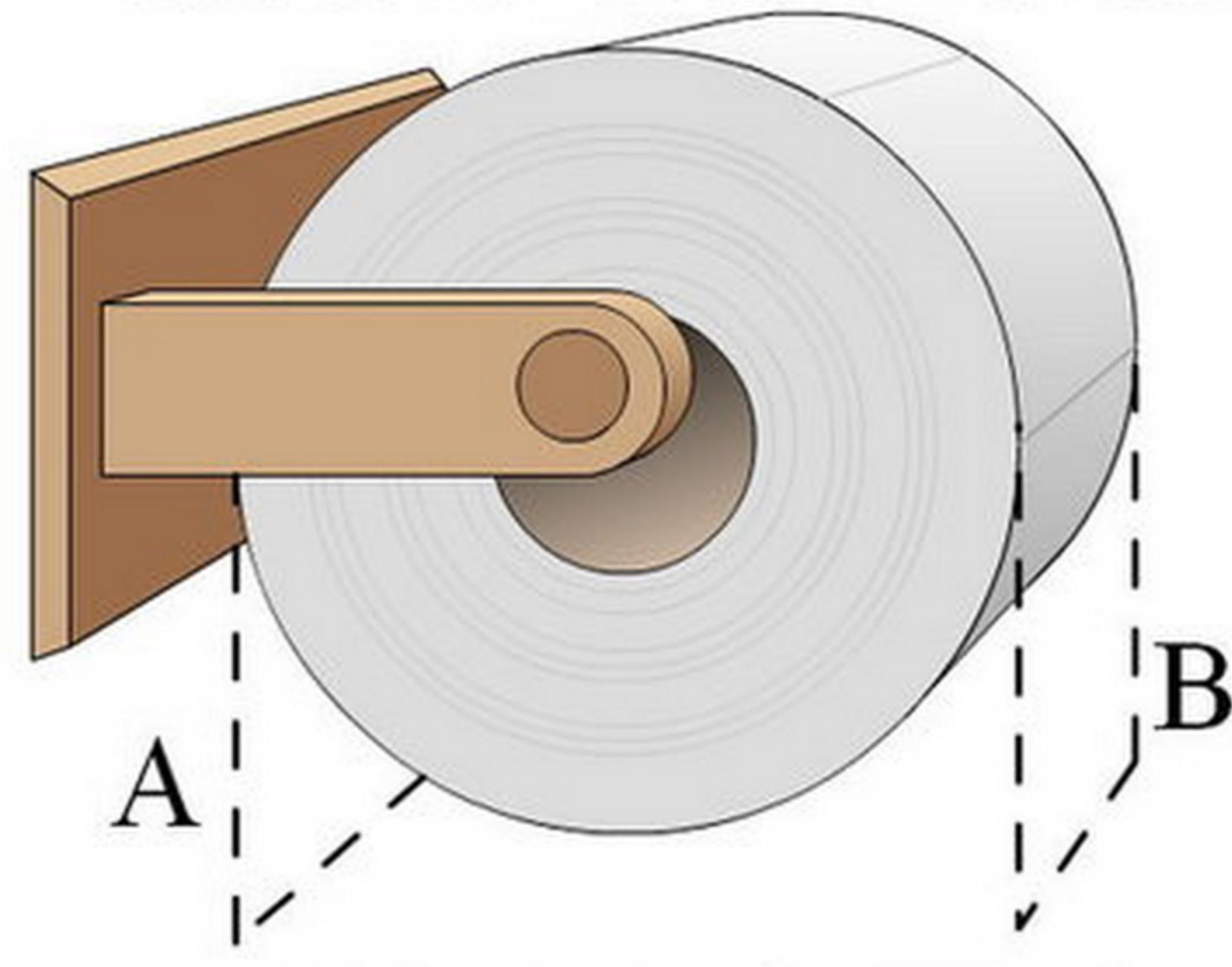


There are *over 50* JavaScript **web frameworks**
to choose from.

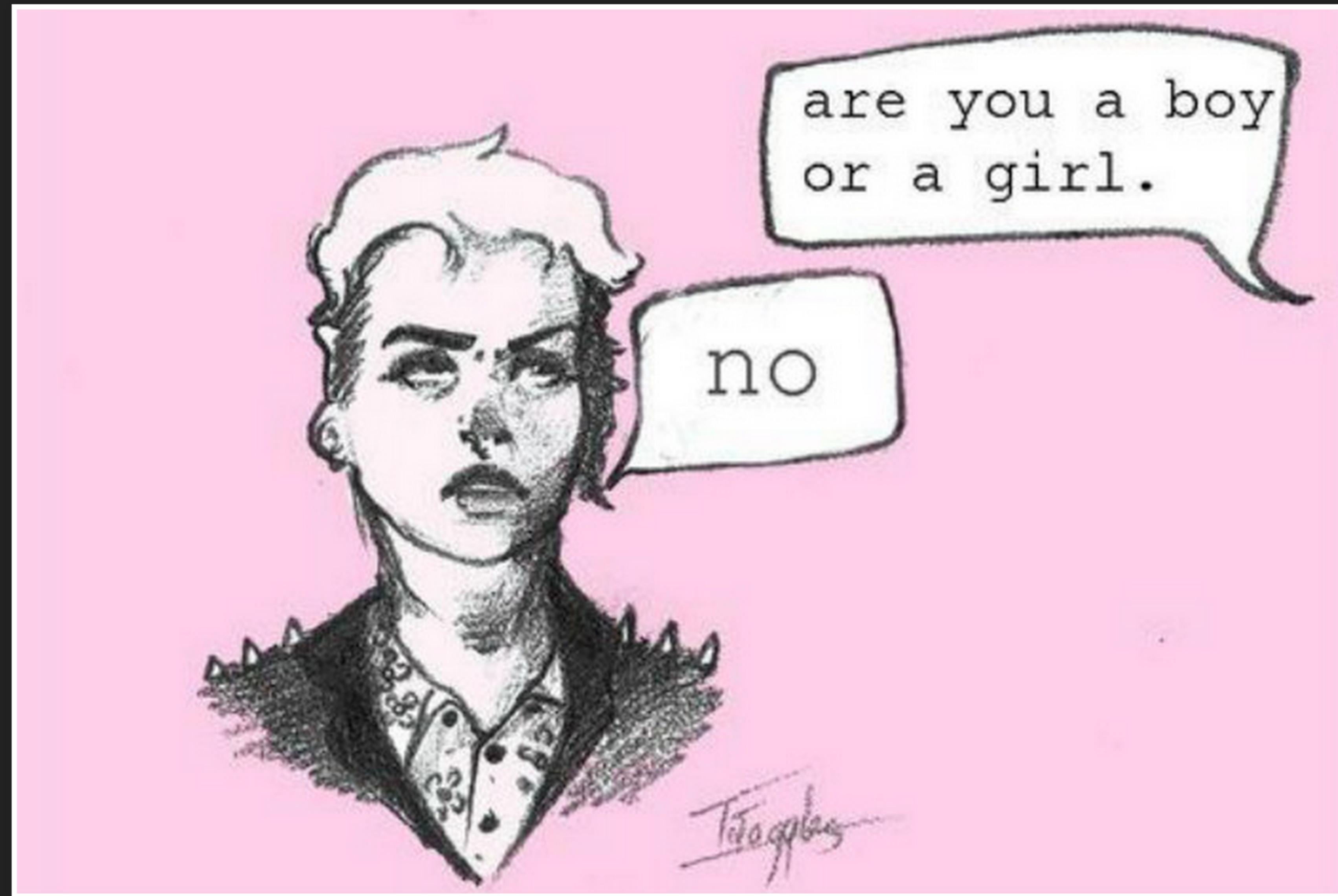


How to choose a right JavaScript web frameworks?

THE GREAT



DEBATE







**KEEP
CALM
AND
BE
PROFESSIONAL**

Rule #1: the client is always right

Most of the time the choice of the technology/framework to use has already been made for you. You come on a project, and it is a matter of reading the documentation on the framework used in the project.



Rule #2: go for open source solutions

If you have the choice, it is a safe bet to choose open source.
Not only is it free as in freedom, but also free as in beer.
Seriously, though, open source allows you to freely modify
the original source.



Rule #3: size of project's **community** is important

The more people contribute to the project, the more of a chance that this project will not die as time goes by. Also, a large community means lots of examples, and a big help to you on your specific issues.



Rule #4: maturity indicates stability

A general rule of thumb is to choose projects which have been in development for at least 2 or 3 years. Less than that, and you are risking to hit on serious bugs on production. You don't want that.



Rule #5: try using different frameworks for different projects

Until you try something, you can't know for sure whether something will work for you or not. Experience is what counts!



Battle-ready JS web frameworks: 2015





HTML enhanced for web apps!

[View on GitHub](#)[Download \(1.3.15 / 1.4.0\)](#)[Design Docs & Notes](#)[Follow +AngularJS on g+](#)[Follow @angularjs](#) 66.9K followers[Tweet](#) 4,937

Learn Angular in your browser for free!

Why AngularJS?

HTML is great for declaring static documents, but it falters when we try to use it for declaring dynamic views in web-applications. AngularJS lets you extend HTML vocabulary for your application. The resulting environment is extraordinarily expressive, readable, and quick to develop.

Alternatives

Other frameworks deal with HTML's shortcomings by either abstracting away HTML, CSS, and/or JavaScript or by providing an imperative way for manipulating the DOM. Neither of these address the root problem that HTML was not designed for dynamic views.

Extensibility

AngularJS is a toolset for building the framework most suited to your application development. It is fully extensible and works well with other libraries. Every feature can be modified or replaced to suit your unique development workflow and feature needs. Read on to find out how.



Angular U

June 22nd - 25th 2015
San Francisco, California



ng-conf

March 5th-6th 2015
Salt Lake City, Utah



[Backbone.js \(1.2.0\)](#)

» GitHub Repository
» Annotated Source

[Getting Started](#)

- Introduction
- Models and Views
- Collections
- API Integration
- Rendering
- Routing

[Events](#)

- on
- off
- trigger
- once
- listenTo
- stopListening
- listenToOnce
- Catalog of Built-in Events

[Model](#)

- extend
- constructor / initialize
- get
- set
- escape
- has
- unset
- clear
- id
- idAttribute
- cid
- attributes
- changed
- defaults
- toJSON
- sync
- fetch
- save
- destroy

- Underscore Methods (9)
- validate
- validationError
- isValid
- url
- urlRoot
- parse
- clone
- isNew
- hasChanged
- changedAttributes
- previous
- previousAttributes

[Collection](#)

- extend
- model
- modelId



BACKBONE.JS

Backbone.js gives structure to web applications by providing **models** with key-value binding and custom events, **collections** with a rich API of enumerable functions, **views** with declarative event handling, and connects it all to your existing API over a RESTful JSON interface.

The project is [hosted on GitHub](#), and the [annotated source code](#) is available, as well as an online [test suite](#), an [example application](#), a [list of tutorials](#) and a [long list of real-world projects](#) that use Backbone. Backbone is available for use under the [MIT software license](#).

You can report bugs and discuss features on the [GitHub issues page](#), on Freenode IRC in the `#documentcloud` channel, post questions to the [Google Group](#), add pages to the [wiki](#) or send tweets to [@documentcloud](#).

Backbone is an open-source component of DocumentCloud.

Downloads & Dependencies (Right-click, and use "Save As")

[Development Version \(1.2.0\)](#)

69kb, Full source, tons of comments

[Production Version \(1.2.0\)](#)

*7.3kb, Packed and gzipped
(Source Map)*

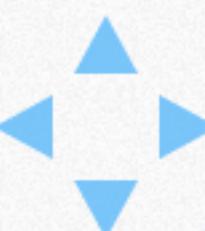
[Edge Version \(master\)](#)

*Unreleased, use at your own risk
[build](#) [passing](#)*

Backbone's only hard dependency is [Underscore.js](#) ($\geq 1.7.0$). For RESTful persistence and DOM manipulation with [Backbone.View](#), include [jQuery](#) ($\geq 1.11.0$), and [json2.js](#) for older Internet Explorer support. (*Mimics of the Underscore and jQuery APIs, such as [Lo-Dash](#) and [Zepto](#), will also tend to work, with varying degrees of compatibility.*)

Getting Started

When working on a web application that involves a lot of JavaScript, one of the first things you learn is to stop tying your data to the DOM. It's all too easy to create JavaScript applications that end up as tangled piles of jQuery selectors and callbacks, all trying frantically to keep data in sync between the HTML UI, your JavaScript logic, and the database on your server. For rich client-side applications, a more structured approach is often helpful.





A framework for creating ambitious web applications

```
npm install -g ember-cli  
ember new my-app
```

[More downloads](#)[Follow @emberjs](#)[Follow on g+](#)

MORE PRODUCTIVE OUT OF THE BOX.



Write dramatically less code with Ember's Handlebars integrated templates that update automatically when the underlying data changes.



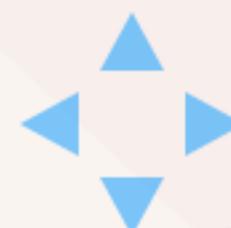
Don't waste time making trivial choices. Ember.js incorporates common idioms so you can focus on what makes your app special, not reinventing the wheel.



Ember.js is built for productivity. Designed with developer ergonomics in mind, its friendly APIs help you get your job done—fast.

AUTO-UPDATING HANDLEBARS TEMPLATES

Ember makes Handlebars templates even better, by ensuring your HTML stays up-to-date when the underlying model changes. To get started, you don't even need to write any JavaScript.



Knockout.

Simplify dynamic JavaScript UIs with the Model-View-View Model (MVVM) pattern

Download
v3.3.0 - 21kb min+gz



[release notes](#)

Key concepts



Declarative Bindings

Easily associate DOM elements with model data using a concise, readable syntax



Automatic UI Refresh

When your data model's state changes, your UI updates automatically



Dependency Tracking

Implicitly set up chains of relationships between model data, to transform and combine it



Templating

Quickly generate sophisticated, nested UIs as a function of your model data

More features

- ✓ Free, open source ([MIT license](#))
- ✓ Pure JavaScript — works with any web framework
- ✓ Small & lightweight — 54kb minified
... reduces to 20kb when using HTTP compression
- ✓ No dependencies
- ✓ Supports all mainstream browsers, even ancient ones
IE 6+, Firefox 3.5+, Chrome, Opera, Safari (desktop/mobile)
- ✓ Fully documented
API docs, live examples, and interactive tutorials included

Get started

[Interactive tutorials](#)

Learn the easy way with an in-browser code editor

[20-minute demo video](#)



New: Interactive tutorials

Get started with knockout.js quickly, learning to build *single-page applications, custom bindings* and more with [these interactive tutorials](#).

Live example

Run it:

Choose a ticket class:

Source code:

```
Choose a ticket class:  
<select data-bind="options: tickets,  
           optionsCaption: 'Choose...',  
           optionsText: 'name',  
           value: chosenTicket"></select>  
  
<button data-bind="enable: chosenTicket,  
           click: resetTicket">Clear</button>  
  
<p data-bind="with: chosenTicket">  
  You have chosen <b data-bind="text: name"></b>  
</p>
```

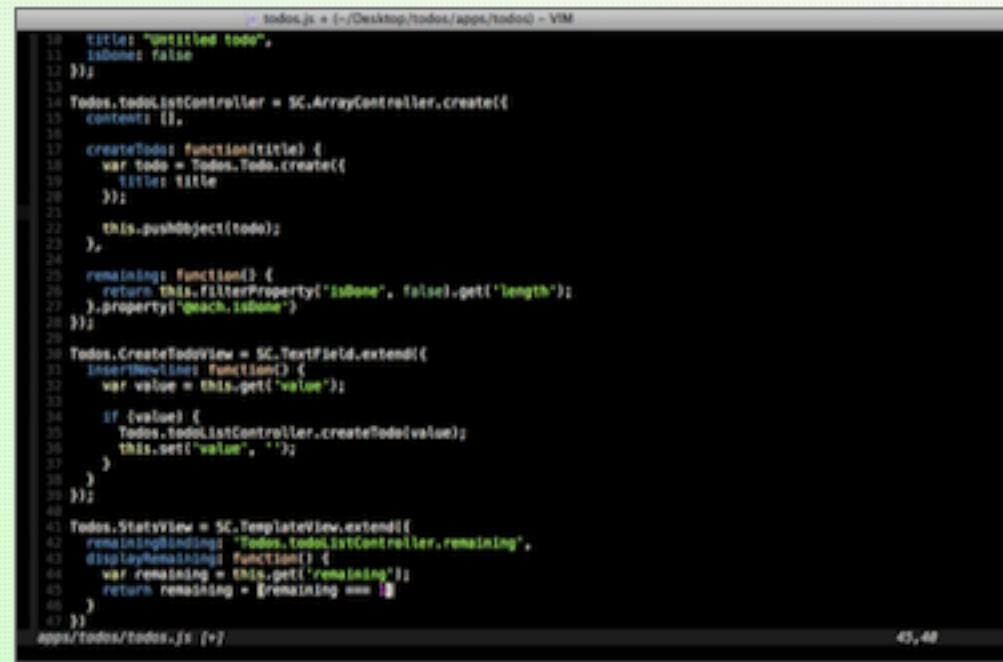
Binding attributes
declaratively link
DOM elements
with model
properties



SproutCore is an open-source framework for building blazingly fast, innovative user experiences on the web.

 [INSTALL SPROUTCORE](#) [VIEW ON GITHUB](#)

New Tutorial to Help You Get Up and Running!



```
git:(master) ✘ todos.js + (-/Desktop/todos/app/todos) - VIM
1 // ...
2 // ...
3 Todos.todoListController = SC.ArrayController.create{
4   contents: []
5 };
6 Todos.createTodo = function(title) {
7   var todo = Todos.Todo.create({
8     title: title
9   });
10   this.pushObject(todo);
11 },
12 Todos.remaining = function() {
13   return this.filterProperty('isDone', false).get('length');
14 },
15 Todos.CreateTodoView = SC.TextField.extend{
16   insertNewline: function() {
17     var value = this.get("value");
18     if (value) {
19       Todos.todoListController.createTodo(value);
20       this.set("value", '');
21     }
22   }
23 };
24 Todos.StatView = SC.TemplateView.extend{
25   remainingBinding: "Todos.todoListController.remaining",
26   displayRemaining: function() {
27     var remaining = this.get("remaining");
28     var remainingString = "Remaining: " + remaining;
29   }
30 };
31 app/todos/todos.js [v]
```

You've heard all about SproutCore and now it's time to see for yourself. It's easier than ever to get started with our new tutorial. In it you'll learn:

- How to install SproutCore on your machine
- How to build your first application
- How to add new interactive elements
- ...and much more!

 [GET STARTED NOW!](#)

All The Tools For The Next Generation



The Power of HTML5
Leverage the latest in web technologies and specifications.



Accessible Everywhere
SproutCore apps give you a native experience—on the web.



Clean MVC Architecture
Keep your code sensible and organized for easy maintenance.



Scale, Scale, Scale
Your app will grow with you, no matter how big you get.



Incredible Speed
Client-side logic means no more waiting on your servers.



Top Notch Theming
Built-in tools to help you and your app look good. Seriously.



Spine

Build Awesome JavaScript
MVC Applications

[Download \(version 1.5.0\)](#)

MVC

The Model View Controller pattern is at the heart of Spine, and absolutely integral to modern JavaScript applications.

Simplicity

Spine is a simple and lightweight framework, and doesn't consist of a vast amount of complex widgets to configure and theme.

Documentation

In addition to very approachable source code Spine strives to have great documentation.

[More information...](#)

[Home](#) [Documentation](#) [GitHub](#) [Spine Mobile](#)

[About](#)



Build better apps, faster

DOWNLOAD 2.2.6
(One file. Everything you need.)

CUSTOMIZE ▾

Works with: jQuery, Zepto, Dojo, Mootools, YUI

**FLEXIBLE**

Works with jQuery, Dojo, Mootools, YUI, and Zepto.
Reuse your existing templates.

**POWERFUL**

2-way binding, restful models,
custom tags, observables,
memory safety, and more.

**FAST**

Fast templates, responsive
widgets, and you can learn it in
a day.

CanJS is a JavaScript library that makes developing complex applications simple and fast. Easy-to-learn, small, and unassuming of your application structure, but with modern features like custom tags and 2-way binding. Creating apps is easy and maintainable.

Follow @canjs

1,184 followers

Star

Fork

26 articles

27 apps submitted

8 plugins submitted

1140 forum posts

5566 commits

Simple To Use

OBSERVABLES AND LIVE BINDING DO THE WORK FOR YOU.

EDIT

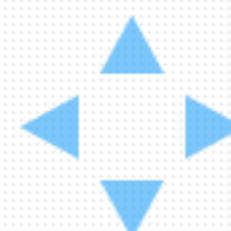
```
var Todo = can.Model.extend({
  findAll: 'GET /todos',
  findOne: 'GET /todos/{id}',
  update: 'PUT /todos/{id}',
  destroy: 'DELETE /todos/{id}'
}, {});

can.Component.extend({
  tag: 'todos-app',
  scope: {
    selectedTodo: null,
    todos: new Todo.List(),
    select: function(todo) {
      this.attr('selectedTodo', todo);
    },
    save: function(todo) {
      todo.save();
      this.removeAttr('selectedTodo');
    }
  }
});
```

```
<todos-app>
  <h2>Todays to-dos</h2>
  {{#selectedTodo}}
    <input type="text"
      can-value="description"
      can-change="save">
  {{/selectedTodo}}
  <ul>
    {{#each todos}}
      <li>
        <input type="checkbox"
          can-value="complete">
        <span class="{{#if complete}}done{{/if}}">
          can-click="select">
          {{description}}
        </span>
        <button can-click="destroy"></button>
      </li>
    {{/each}}
  </ul>
</todos-app>
```

Todays to-dos

- Download CanJS X
- Read the guides X
- Build your app X
- Become immortal X
- Haircut @ 2pm X





Flight

.....

An event-driven web framework, from Twitter

[VIEW ON GITHUB](#)

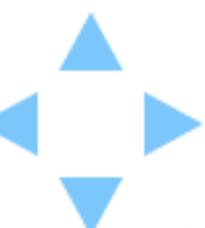
Overview

Flight is a lightweight, component-based JavaScript framework that maps behavior to DOM nodes. Twitter uses it for their web applications. By way of example, we've included a [simple email client demo](#) (browse the [source](#)) built over the Flight framework. There's also a flight implementation over on the [todoMVC](#) site ([source](#)), courtesy of [@mkuklis](#)

Flight uses [jQuery](#) and requires a module loader with support for AMD, like [WebPack](#) or [Require.js](#). Please read the [Flight documentation](#) for installation instructions.

Why Flight?

Flight is distinct from existing frameworks in that it doesn't prescribe or provide any particular approach to rendering or providing data to a web application. It's agnostic to how requests are routed, which templating language you use or even if you render your HTML on the client or the server. While some web frameworks encourage developers to arrange their code around a prescribed model layer, Flight is organized around the existing DOM model with functionality





TUTORIAL DOCS SUBPROJECTS PACKAGES THE METEOR PROJECT BLOG

SIGN IN

The JavaScript App Platform

Discover the easiest way to build amazing web and mobile apps in JavaScript

INSTALL METEOR 1.1.0.2

START TUTORIAL



Star

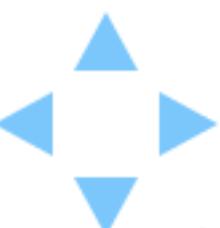
25440



Follow @meteorjs

40.8K followers

Get on the mailing list - the latest Meteor updates, and nothing else.

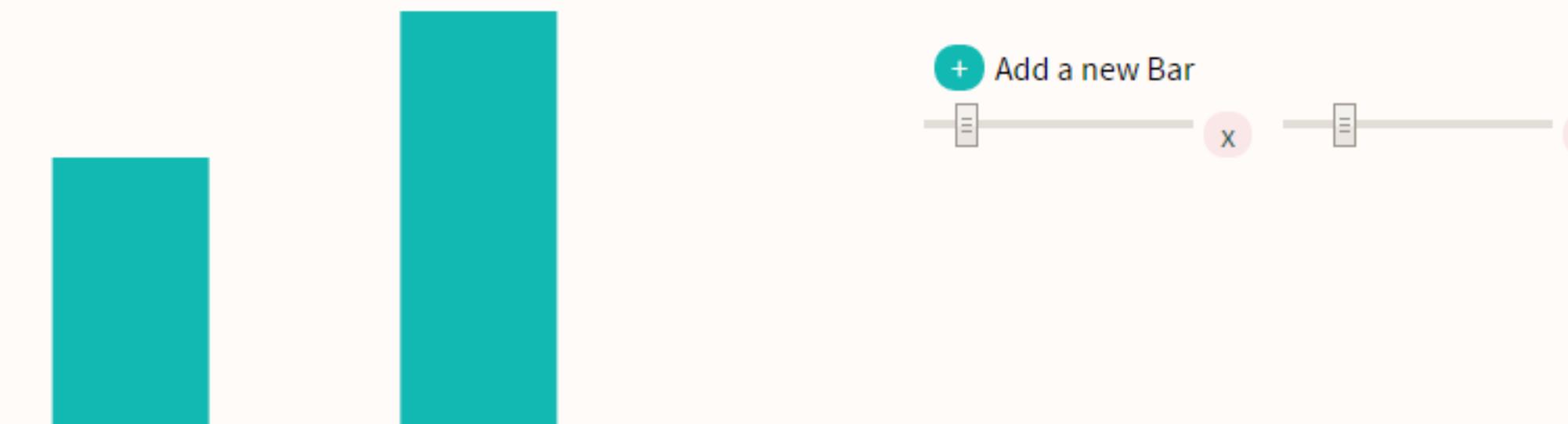


Expect more from your MVC.

DerbyJS is a full-stack framework for writing modern web applications.

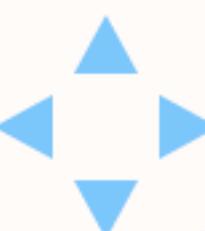
[Get started](#)

Realtime collaboration



Effortlessly sync data across clients and servers with automatic conflict resolution powered by [ShareJS's](#) operational transformation of JSON and text.

Server Rendering



DURANDAL

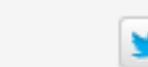
Single Page Apps Done Right

[GitHub Project](#) Version 2.1.0

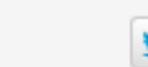
Star



Fork



Follow @durandaljs



Tweet

607

Google groups

Meet Durandal

Your search for a SPA framework ends here.

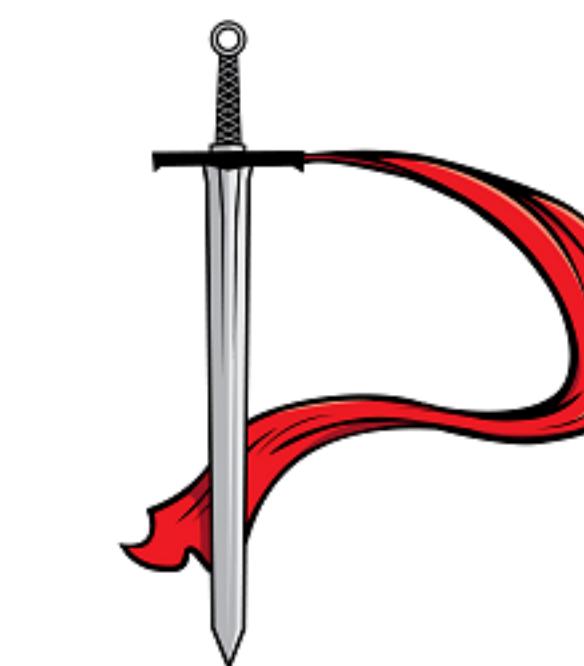
Comfortable

We didn't try to re-invent the wheel. Durandal is built on libs you know and love like [jQuery](#), [Knockout](#) and [RequireJS](#). There's little to learn and building apps feels comfortable and familiar. Dive in and enjoy.

Feature-rich

MVC? MVP? MVVM? Yes. Messaging, navigation, modals? Check. Durandal has the features you need to build whatever apps you can imagine; the apps of today and of tomorrow. Let your creativity soar.

Versatile



Features

- Clean MV* Architecture
- JS & HTML Modularity
- Simple App Lifecycle
- Eventing, Modals, Message Boxes, etc.
- Navigation & Screen State Management
- Consistent Async Programming w/ Promises
- App Bundling and Optimization
- Use any Backend Technology
- Built on top of [jQuery](#), [Knockout](#) & [RequireJS](#)
- Integrates with popular CSS libraries such as [Bootstrap](#) and [Foundation](#)
- Make Your Own Templatable and Bindable Widgets
- Fully Testable



React

A JAVASCRIPT LIBRARY FOR BUILDING USER INTERFACES

[Get Started](#)[Download React v0.13.3](#)

JUST THE UI

Lots of people use React as the V in MVC. Since React makes no assumptions about the rest of your technology stack, it's easy to try it out on a small feature in an existing project.

VIRTUAL DOM

React abstracts away the DOM from you, giving a simpler programming model and better performance. React can also render on the server using Node, and it can power native apps using [React Native](#).

DATA FLOW

React implements one-way reactive data flow which reduces boilerplate and is easier to reason about than traditional data binding.

A Simple Component

React components implement a `render()` method that takes input data and returns what to display. This example uses an XML-like syntax called JSX. Input data that is passed into the component can be accessed by `render()` via `this.props`.

JSX is optional and not required to use React. Try clicking on "Compiled JS" to see the raw JavaScript code produced by the JSX compiler.

[Live JSX Editor](#)[Compiled JS](#)

```
var HelloMessage = React.createClass({
  render: function() {
    return <div>Hello {this.props.name}</div>;
  }
});
```

Hello John



Flux

APPLICATION ARCHITECTURE FOR BUILDING USER INTERFACES

Flux is the application architecture that Facebook uses for building client-side web applications. It complements React's composable view components by utilizing a unidirectional data flow. It's more of a pattern rather than a formal framework, and you can start using Flux immediately without a lot of new code.



[Learn more about Flux](#)

© 2014-2015 Facebook Inc.



Sample applications

For each of the JavaScript **web frameworks** in the previous section, you will find a sample application at

<https://github.com/valera-rozuvan/web-frameworks-overview/blob/master/sample-apps.md>.



This is the end. Thank you =).



2015 @ Valera Rozuvan | <http://valera.rozuvan.net/>

