beta signup
www.famo.us

Web App Performance

the story of becoming **famo.us**

@stevenewcomb


wat

the story of becoming **famo.us**

about 2 years ago, we got funded to build
consumer identity app

it had a very **ambitious user interface**

the story of becoming **famo.us**

we decided to build it in **HTML5**

the story of becoming **famo.us**

it needed to work on **lots of devices** including

iPad1 + iPad2 + iPad3

phones, tablets, pcs and game consoles

the story of becoming **famo.us**

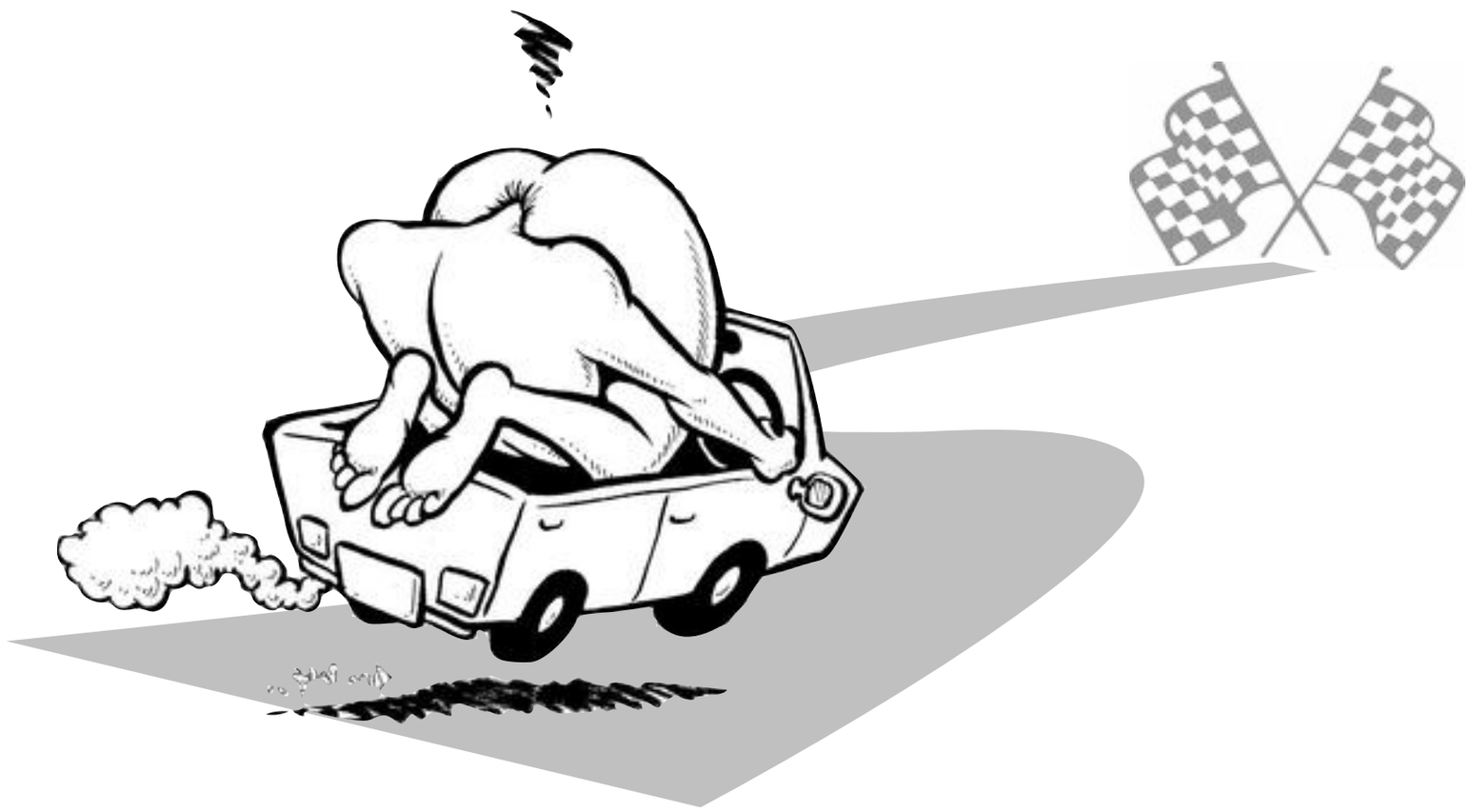
it needed to work on **iOS** and **Android** as a **web app** and inside a **native wrapper**

the story of becoming **famo.us**

it needed to **handle many inputs** including
touch, keyboard, mouse and gesturing systems

the story of becoming **famo.us**

we started knowing **nothing**



the story of becoming **famo.us**

we hit every **performance issue**
you can imagine

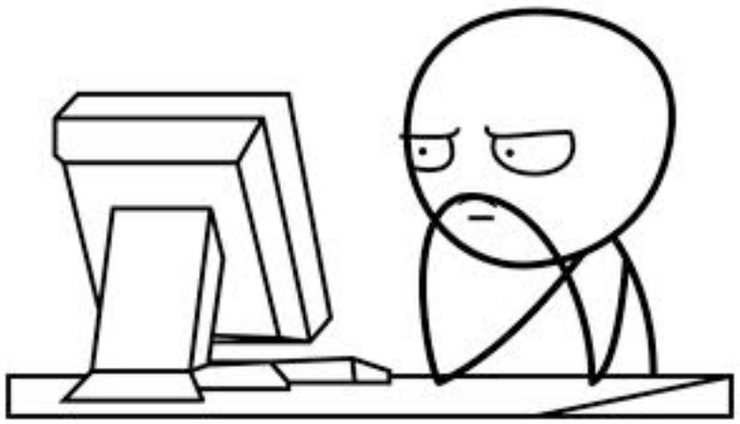
the story of becoming **famo.us**

we battled through many
false promisses

the story of becoming **famo.us**

we went through many
frustrating experiences

the story of becoming **famo.us**



we **learned** a lot of things

the story of becoming **famo.us**

we’d like to **share our journey** with you

our **first** big realization

CSS3 can completely **ruin** your performance

examples

- -webkit-box-shadow: **60px** black;
- -webkit-transform-style: **preserve-3D**;
- **-webkit-text-stroke**: 1px transparent;
- -webkit-transform: **translateZ(0)**;
- **-webkit-transition**: all 1s ease-in-out;
- **@keyframes** mymove

and 50 more things...

our **first** big realization

there are basically **no debug tools** to find these things

our **first** big realization

this causes **discomfort** between
designers and **engineers**

the **impact**

The realization had a dramatic impacts

confusing — we catalogued all of these quirks

sucked — we **refactored** all of our CSS

unfortunate — one designer was killed, one maimed

sucked — we had to **audit** any 3rd party library that touched CSS

reading material

Y U NO LINKS

our **second** big realization

Learning to code for computers first has **masked bad coding habits**

our **second** big realization

We had to **relearn** many things to optimize our code for mobile

DOM Manipulation

Event Handling

Data Handling

Image Handling

our **second** big realization

The realization had a dramatic impacts

interesting

- we learned the **performance optimized methods**

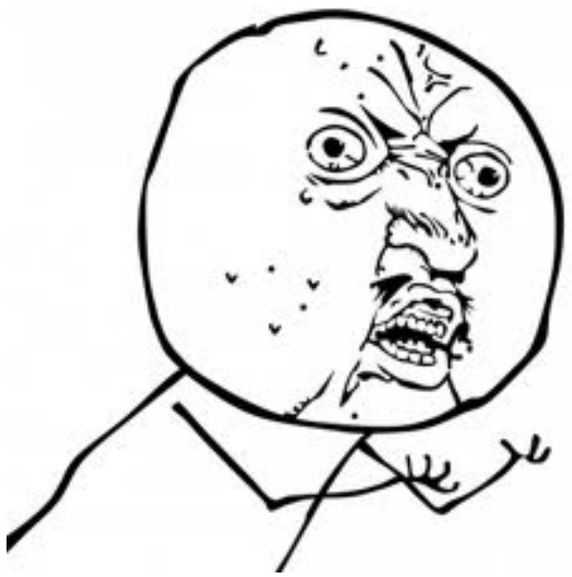
meh

- we **refactored** all of our app specific javascript libraries

sucked

- we had to **audit** any 3rd party library that touched DOM, events, data or images

reading material



Y U NO LINKS

our **fourth** big realization

Safari and Chrome

are not created equally

some **examples**

- webkit implementation
  - bezier curves and preserve-3D
- version differences
- uncanny valleys
- JS engines
- device differences
- retarded things we are still too angry to talk about

the real **work began**

The realization had a dramatic impacts

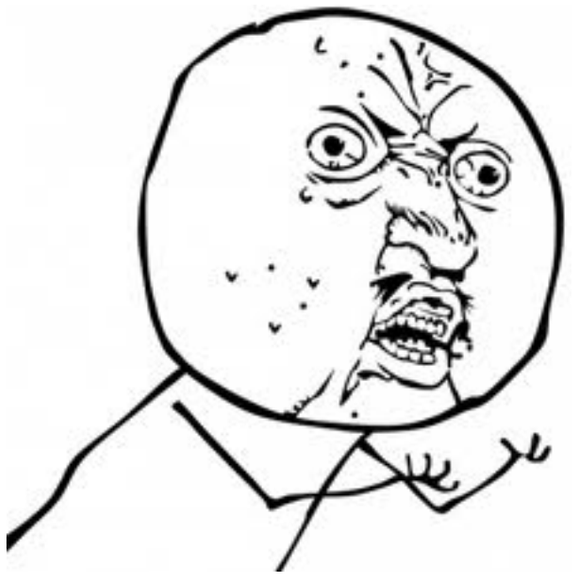
sucked
- we had to **catalogue** the differences

sucked
- we **refactored** all code to take these differences into account

unfortunate
- two devices were destroyed (Android)

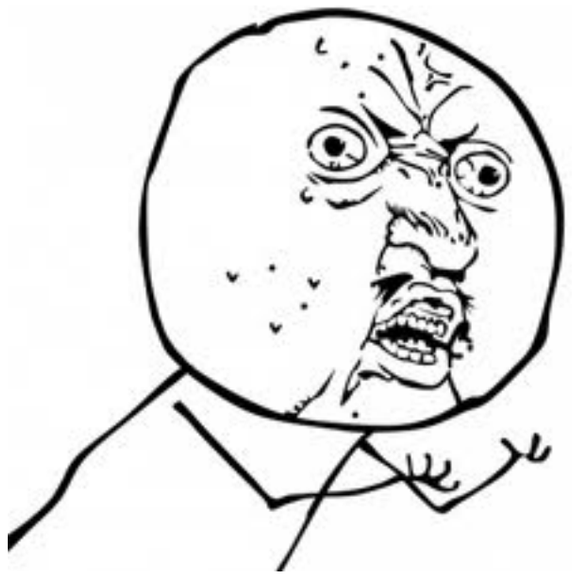
sucked
- we had to **audit** any 3rd party library to see how they handled the differences

reading material



Y U NO LINKS

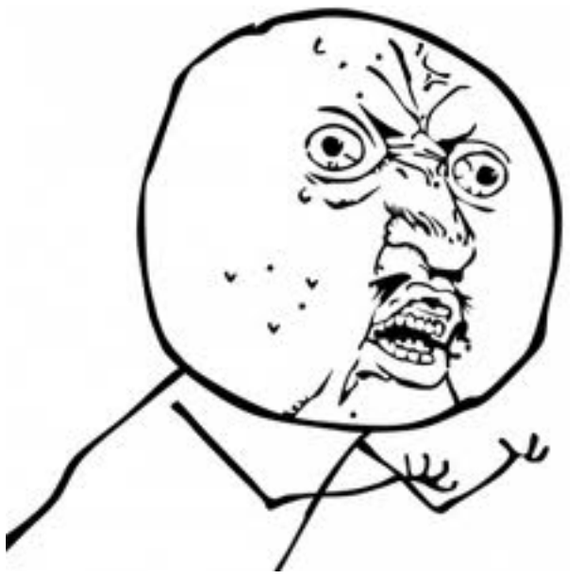
our **fifth** big realization



Y U NO WORK

a lot of HTML5 components **don’t work**
or don’t work like you’d expect

some **examples**



Y U NO WORK

- cache manifest
- android touch events
- HTML5 video

the real work **work began**

The realization had a dramatic impacts

frustrating

- we catalogued the **which ones didn't work**

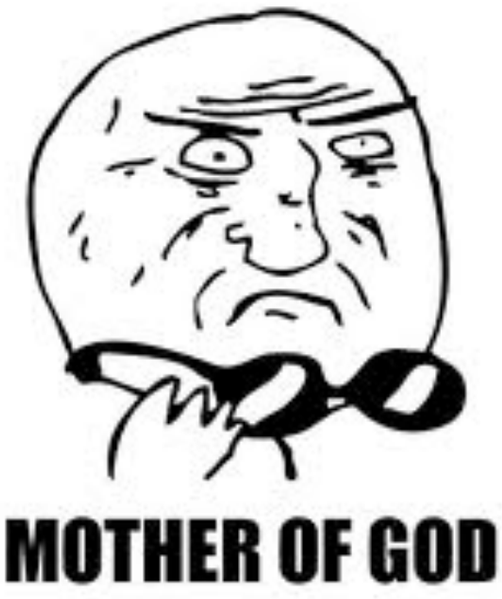
easy

- we **removed** all usage of these things

sucked

- we had to **audit** any 3rd party library to see how they handled the differences

our **sixth** big realization



a lot of our **favorite libraries** were **useless**

some **examples**

- jQuery Animations
- jQuery Plugins
- jQuery
- Isotope
- Any thing that was built originally for computers and touched the DOM or Events

the **impact**

The realization had a dramatic impacts

frustrating

- we catalogued the **which ones didn't work**

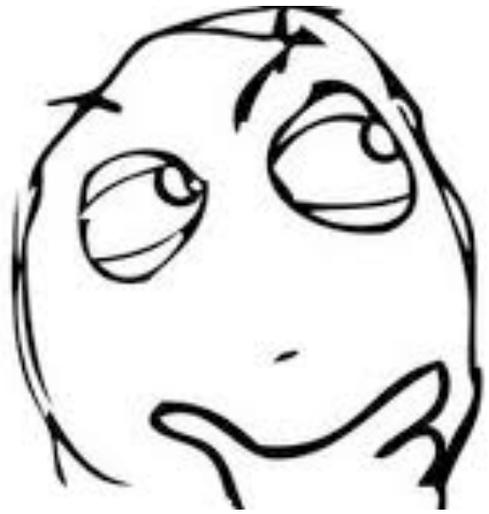
sucked

- we **removed** all usage of these things

sucked

- we re-wrote all the things we had to remove

DOM Manipulation Tricks



let's try some **exotic** DOM manipulation techniques

some **examples**

advanced DOM manipulation techniques

- DOM object re-use
- 3 panels method
- the event horizon method

GPU Tricks

let’s try some **exotic** frame rate techniques

some **examples**

Example of advanced DOM manipulation techniques

- Request animation frame
- Frame Rate throttling
- Fake multi-thread javascript

impacts on **effort**

In many ways dealing with the GPU is like working with a magical black box that you have little to no visibility into

- buffer size
- resource limits
- object count

Good luck figure out when your app is about to blow the GPU up

reading material

Y U NO LINKS

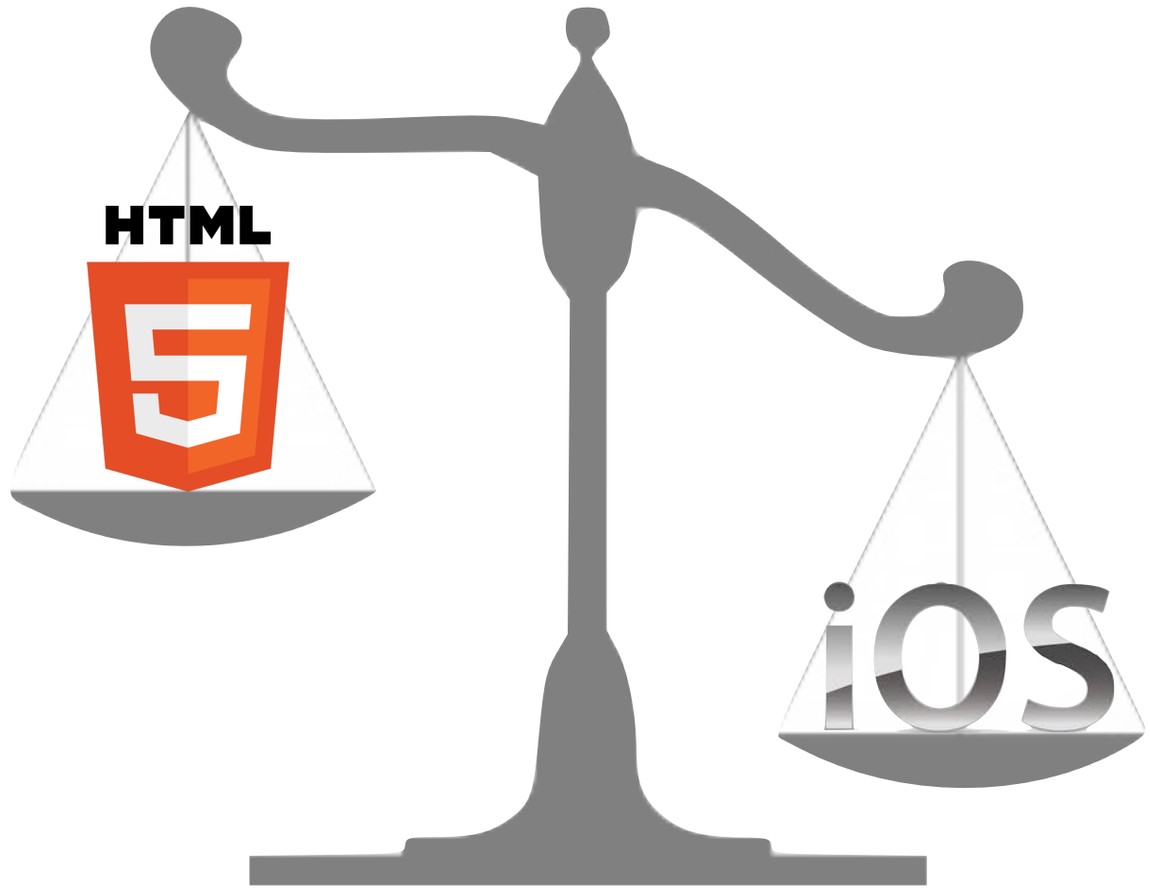
success (kind of)

1 year later (March 2011)


JANK

a **fundamental question**

we were able to make it **performant**, **but at what price?**



the **reality check**

Coding for mobile apps is much less forgiving that coding for websites

- performance problems are often a factor of many things interacting with each other

- as your codebase grows, the number of either library specific or app specific interactions that affect performance constantly grows

- every time a new performance problem arises, the complexity of solving it grows

what was the **real solution?**

why not just build everything in **webGL** or **Canvas?**

what do you lose with **WebGL** or **Canvas?**

Event handling at the object level

Portability of exisiting DOM, including
- **Text** layout in 2D
- Most **CSS** effects
- **Buttons** and **form** elements
- Text **selection** / highlighting

Contextual information of objects within

WebGL is like a hologram while HTML + CSS transforms is the real thing

what was the **real solution?**

there wasn't anything specifically designed for **apps**

documents	apps	2D rendered	3D rendered
HTML	?	canvas	WebGL

what was the **real problem?**

browsers were built to **render documents** and **not apps**

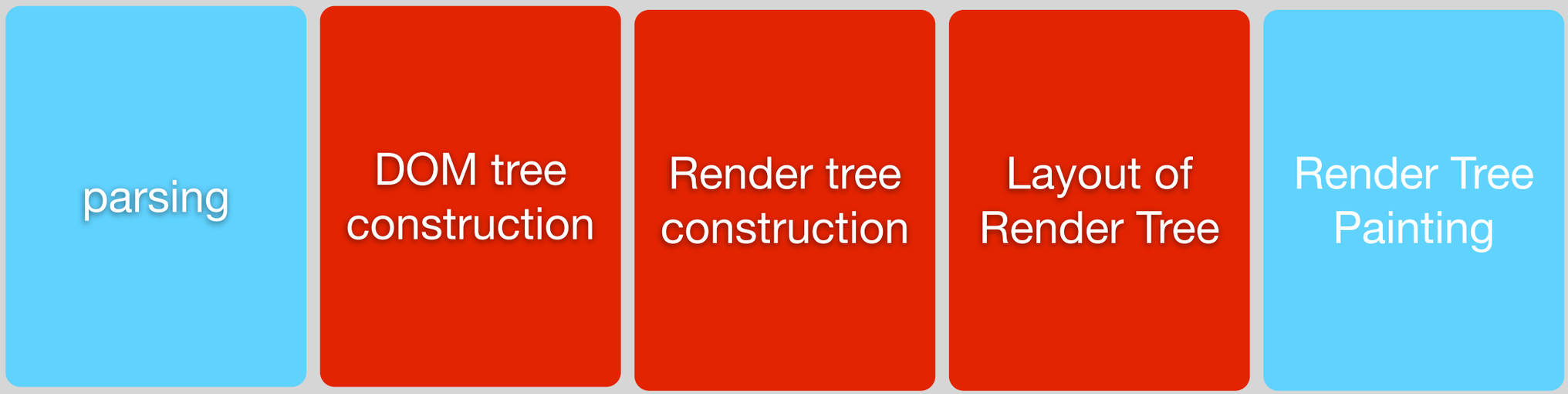
Simplified Render

parsing	DOM tree construction	Render tree construction	Layout of Render Tree	Render Tree Painting

WebCore

understanding **WebKit**

designed to render documents



WebCore

understanding **WebKit**

Physics Engine

Render Engine

Rules Framework

famo.us

understanding **WebKit**

modify the inefficient parts

parsing

DOM tree construction

Render tree construction

Layout of Render Tree

Render Tree Painting

Physics Engine

Render Engine

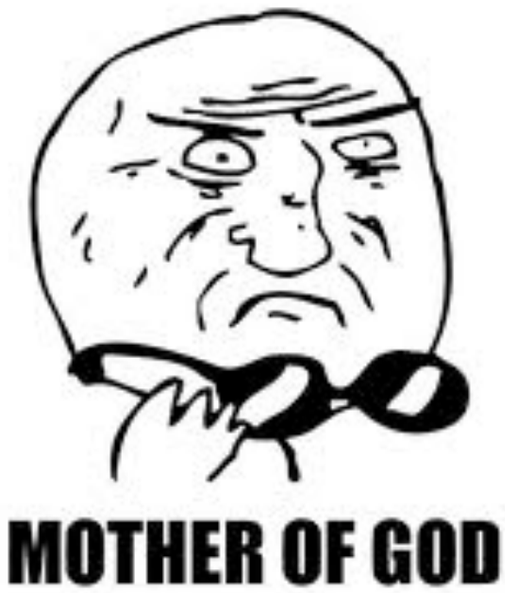
Rules Framework

famo.us

building the **render engine**

- **Energy Module** (defines and instantiates energy agents)
- **Camera Kinematics** Module (controls camera movement behavior & events)
- **Camera Module** (positions camera and adjusts perspective)
- **TransitionHelper Module** (handles animation of multiple surfaces)
- **Surface Kinematics Module** (positions surfaces and applies kinematics)
- **Matrix Module** (performs efficient matrix math)

how does it **work?**



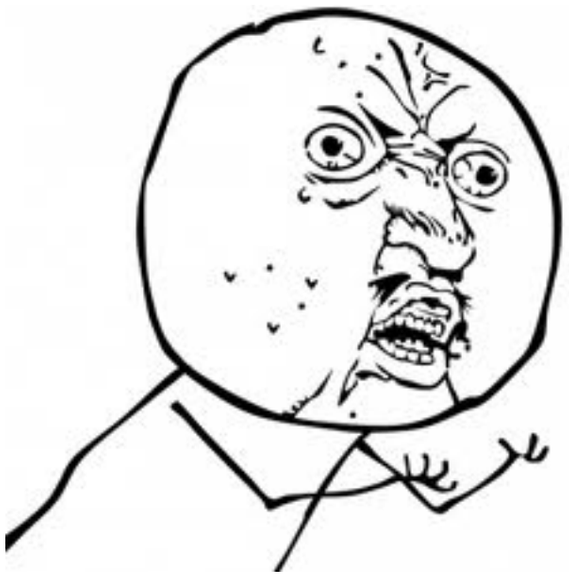
ignore the CSS3 transition primitives

compute the transforms in our render engine

inject our computed transforms into
`-webkit-transform: matrix3D`

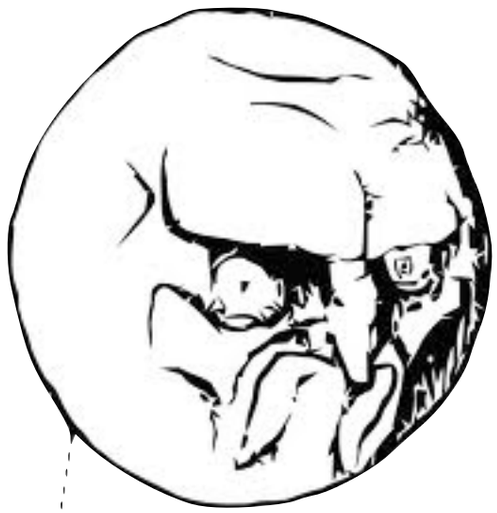
effectively **skipping** the browsers inefficient rendering process meant for documents

DEMO PLEASE



Y U NO SHOW ME

Physics Engine



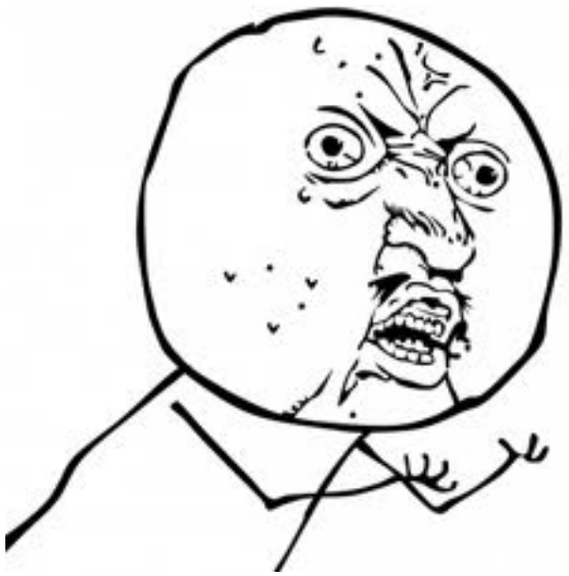
use **kinematic actors** to effect motion and user interaction in the system

enable **designers** to tune without touching CSS

enable **engineers** to build **custom components**

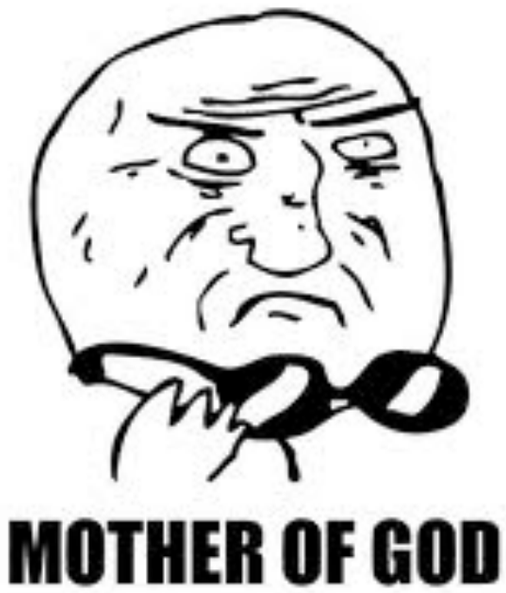
enable enginers to **build new kenematic actors** or change the engine itself

DEMO PLEASE



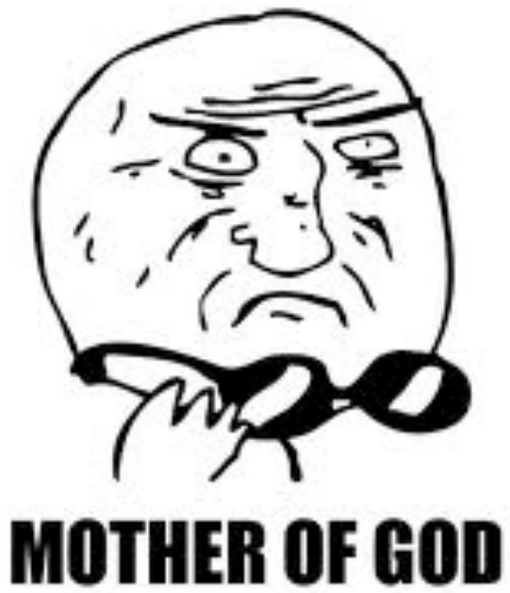
Y U NO SHOW ME

step **3**



let's build our own **framework**

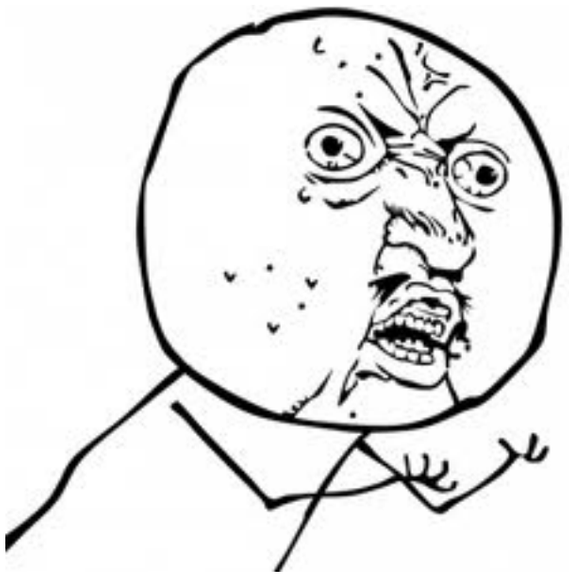
specifically



manage all of the **input mechanisms** like touch, mouse, keyboard and gestures

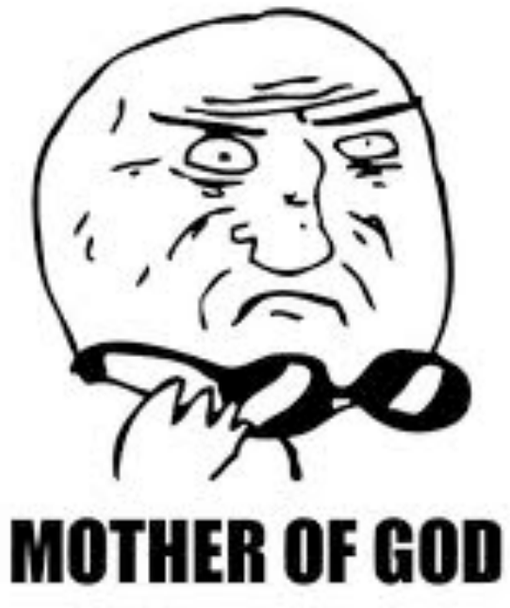
manage the **differences** amongst **browsers, versions and devices**

DEMO PLEASE



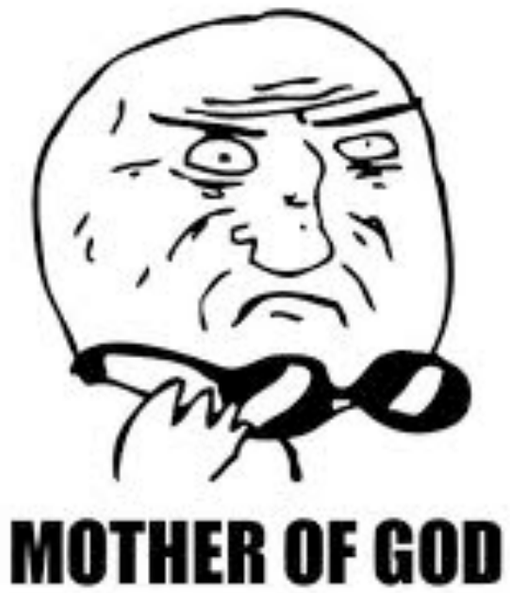
Y U NO SHOW ME

the **realization**



the **engine** and the **framework** were **more important** to us than the product

the **framework**



scaffolds, ui and ux components and all the things you need to build apps

let developers build and share their own app templates, scaffolds and components too

enable developers to get down to the **source code** if they need to

the **FAQs**

Are you planning on being full stack or doing just one thing well and integrating with other MVCs?
designed to be just the view layer in an MVC

Do you plan on working with Backbone.js?
YES

Will there be a public GitHub repo with an open source license?
YES

Will there be a commercial license?
YES

how to get **involved**

beta

@befamous

jobs@famo.us

links

CSS 3D Transforms
https://www.webkit.org/blog/386/3d-transforms/
http://www.webkit.org/blog-files/3d-transforms/poster-circle.html
http://desandro.github.com/3dtransforms/
http://html5rubik.com/tutorial/
http://html5rubik.com/tutorial/step3/index.html
http://www.paulrhayes.com/experiments/sphere/
http://css3.bradshawenterprises.com
http://9elements.com/html5demos/matrix3d/
http://www.the-art-of-web.com/css/3d-transforms/
http://dev.opera.com/articles/view/understanding-the-css-transforms-matrix/
http://24ways.org/2010/intro-to-css-3d-transforms
http://www.webkit.org/blog-files/3d-transforms/transform-style.html
http://www.edankwan.com/lab/css3dEarth
http://www.netmagazine.com/features/20-stunning-examples-css-3d-transforms
http://cbateman.com/blog/head-coupled-3d-transforms/
http://coding.smashingmagazine.com/2012/04/17/beercamp-an-experiment-with-css-3d/
http://acko.net/blog/making-love-to-webkit/
https://news.ycombinator.com/item?id=3470736
http://www.satine.org/archives/2009/07/11/snow-stack-is-here/
http://coding.smashingmagazine.com/2012/01/06/adventures-in-the-third-dimension-css-3-d-transforms/

3D Linear Algebra - Advanced
http://www.robertblum.com/articles/2005/02/14/decomposing-matrices
http://en.wikipedia.org/wiki/Householder_transformation
http://en.wikipedia.org/wiki/QR_decomposition
http://en.wikipedia.org/wiki/Transformation_matrix

links

Documentation
https://developer.mozilla.org/en-US/docs/CSS/transform?redirectlocale=en-US&redirectslug=CSS%2F-moz-transform
https://developer.apple.com/library/safari/#documentation/appleapplications/reference/SafariCSSRef/Articles/Functions.html
http://www.w3.org/TR/css3-2d-transforms/
http://www.w3.org/TR/css3-3d-transforms/
https://developer.apple.com/library/safari/#documentation/AudioVideo/Reference/WebKitCSSMatrixClassReference/WebKitCSSMatrix/WebKitCSSMatrix.html

Troubleshooting
http://stackoverflow.com/questions/6843367/css3-transforms-cause-screen-flickering-or-aliased-font

Animation, requestAnimationFrame and performance
http://paulirish.com/2011/requestanimationframe-for-smart-animating/
https://developer.mozilla.org/en-US/docs/CSS/@keyframes
http://www.w3.org/TR/css3-animations/
http://creativejs.com/resources/requestanimationframe/
https://developer.mozilla.org/en-US/docs/DOM/window.requestAnimationFrame?redirectlocale=en-US&redirectslug=DOM%2Fwindow.mozRequestAnimationFrame
http://www.html5rocks.com/en/tutorials/speed/animations/
http://www.html5rocks.com/en/tutorials/doodles/lem/
http://www.planetb.ca/2012/03/html5-game-programming-gems-requestanimationframe/
TranslateZ(0), preserve-3D and performance
http://stackoverflow.com/questions/10814178/css-performance-relative-to-translatez0
http://creativejs.com/2011/12/day-2-gpu-accelerate-your-dom-elements/
http://stickmanventures.com/labs/demo/spinning-gears-Chrome-preserve-3d/
http://stackoverflow.com/questions/7908493/considerations-for-css3-transition-performance
http://albertogasparin.it/articles/2011/06/ios-css-animations-performances/
http://jsperf.com/translate3d-vs-xy/29

links

Other CSS Issues
http://nerds.airbnb.com/box-shadows-are-expensive-to-paint

Interesting Discussions on HTML5 performance and feature support
http://lists.w3.org/Archives/Public/public-coremob/2012Sep/0021.html
http://news.ycombinator.com/item?id=4526593
https://plus.google.com/1063004076792571546 89/posts/PBxtaphMDGJ
https://plus.google.com/1063004076792571546 89/posts/NEAuwZ7v27B
http://updates.html5rocks.com/2012/07/How-to-measure-browser-graphics-performance

Information and videos on how browsers work
http://www.html5rocks.com/en/tutorials/internals/howbrowserswork/
http://vimeo.com/44182484
http://www.youtube.com/watch?v=xuMWhto62Eo

Touch Issues, Discussions and Solutions
http://www.html5rocks.com/en/mobile/touch/
http://smus.com/mouse-touch-pointer/
https://github.com/borismus/pointer.js
https://dl.dropbox.com/u/7479257/pointevents_strawman.txt
http://www.lukew.com/ff/entry.asp?1533

Information on browser support:
https://developers.facebook.com/html5/blog/post/2012/04/03/ringmark-is-now-open-source/

Information on CSS3 Transitions (nota bene: they aren't performant enough so we built our own animation curves for transitions in famo.us)
http://dev.w3.org/csswg/css3-transitions/

links

Appcache and storage
http://www.alistapart.com/articles/application-cache-is-a-douchebag/
https://speakerdeck.com/u/jaffathecake/p/application-cache-douchebag

General Javascript Performance Advice
http://blog.tojicode.com/2012/04/if-i-built-physics-engine.html
http://blog.tojicode.com/2012/03/javascript-memory-optimization-and.html
https://www.scirra.com/blog/76/how-to-write-low-garbage-real-time-javascript
http://net.tutsplus.com/tutorials/javascript-ajax/stop-nesting-functions-but-not-all-of-them/
http://www.html5rocks.com/en/tutorials/speed/v8/
http://news.ycombinator.com/item?id=4643611

Three panel method approach to Infinite scroll
http://cubiq.org/swipeview

Code review of Quake3 and Doom3, which offer lots of good insight into producing a performance 3D app that works over a network where latency is a problem
http://fabiensanglard.net/quake3/index.php
http://fabiensanglard.net/doom3/index.php