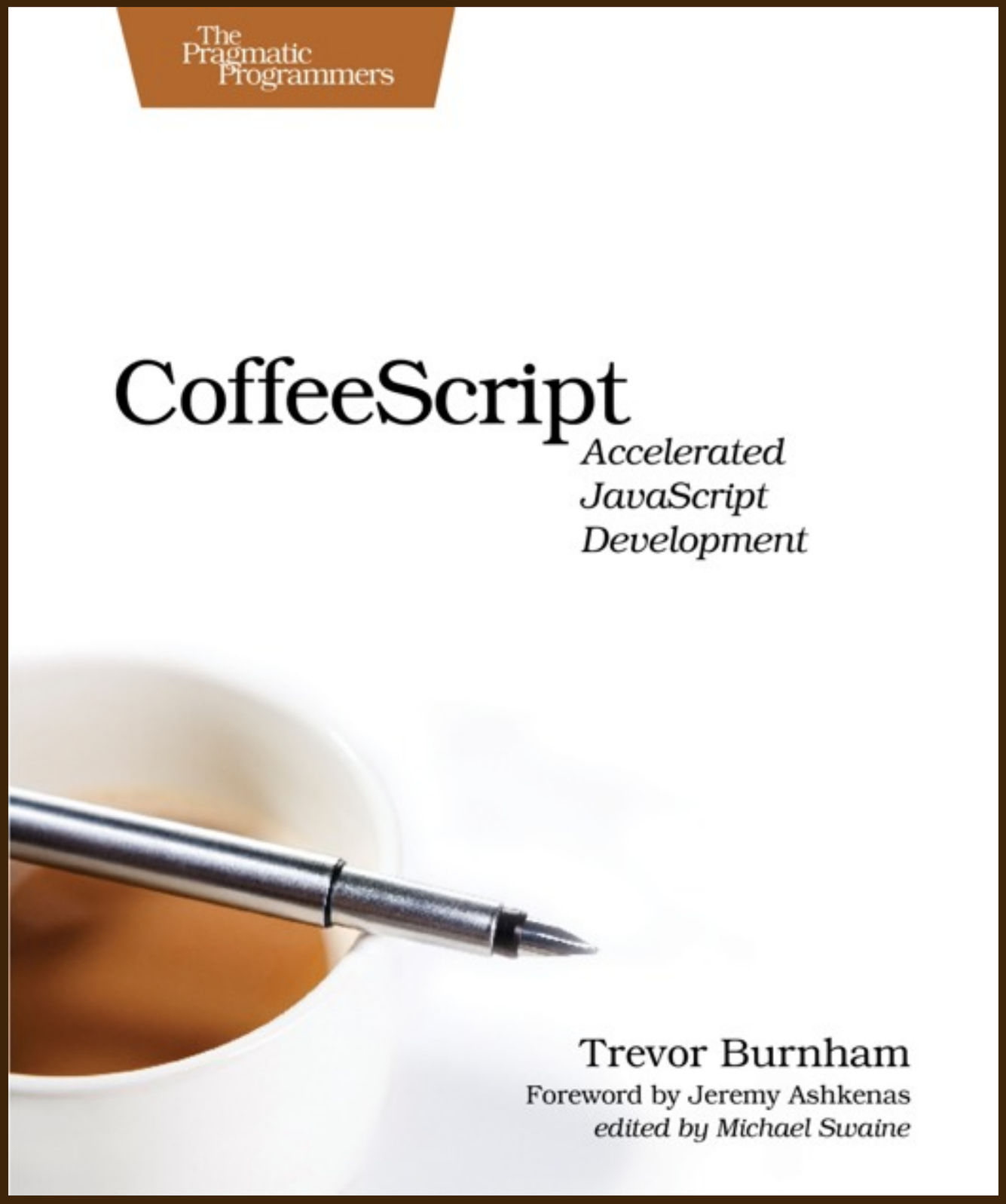
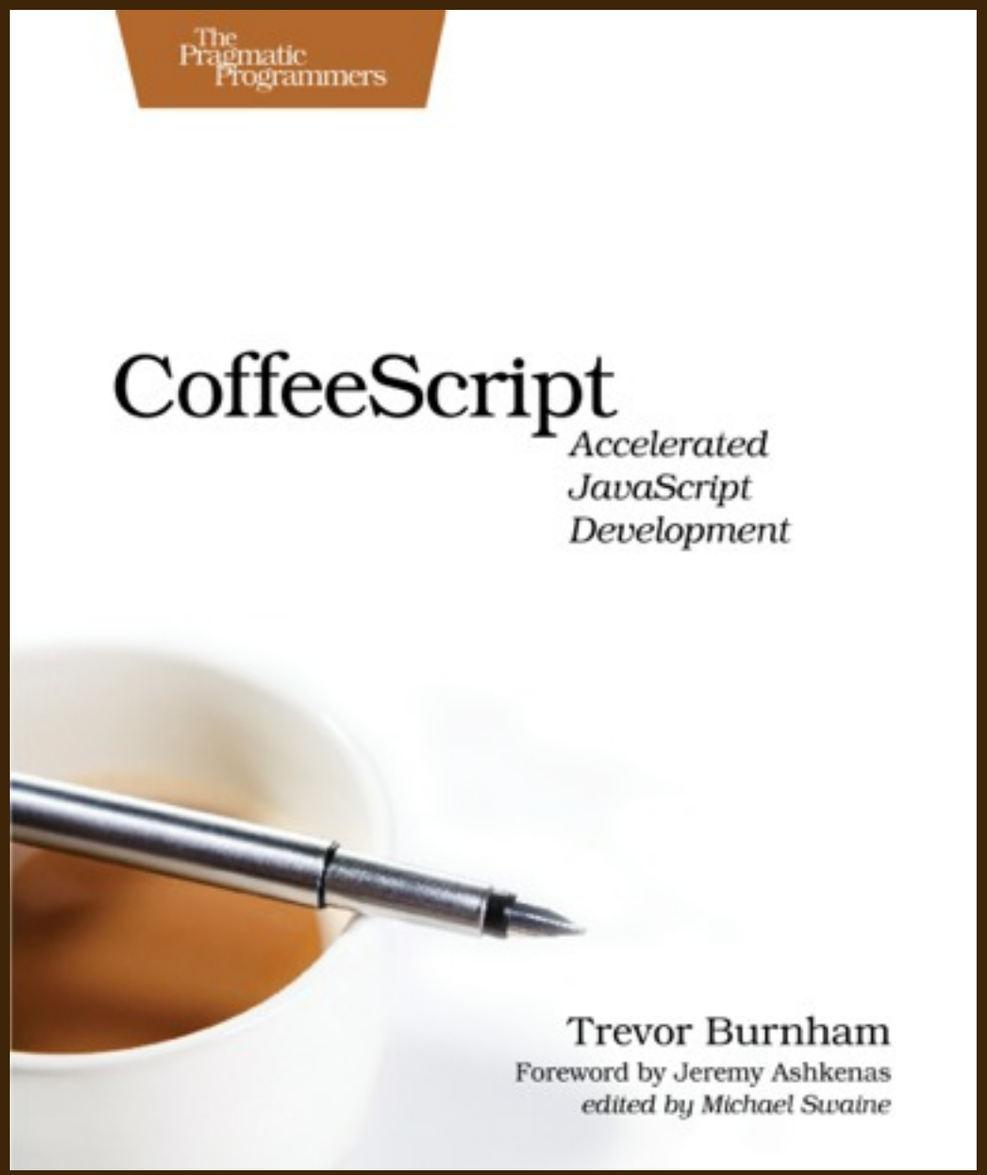
The CoffeeScript Edge

Presented by Trevor Burnham at Philly ETE 2012

Who Am I?



http://pragprog.com/book/tbcoffee/coffeescript



"This book helps readers become better JavaScripters in the process of learning CoffeeScript. What's more, this book is a blast to read."
—Brendan Eich

http://pragprog.com/book/tbcoffee/coffeescript

Preface

The Compleat History of JavaScript

1995 to 2003: Ancient JavaScript

Paul Graham on JavaScript:

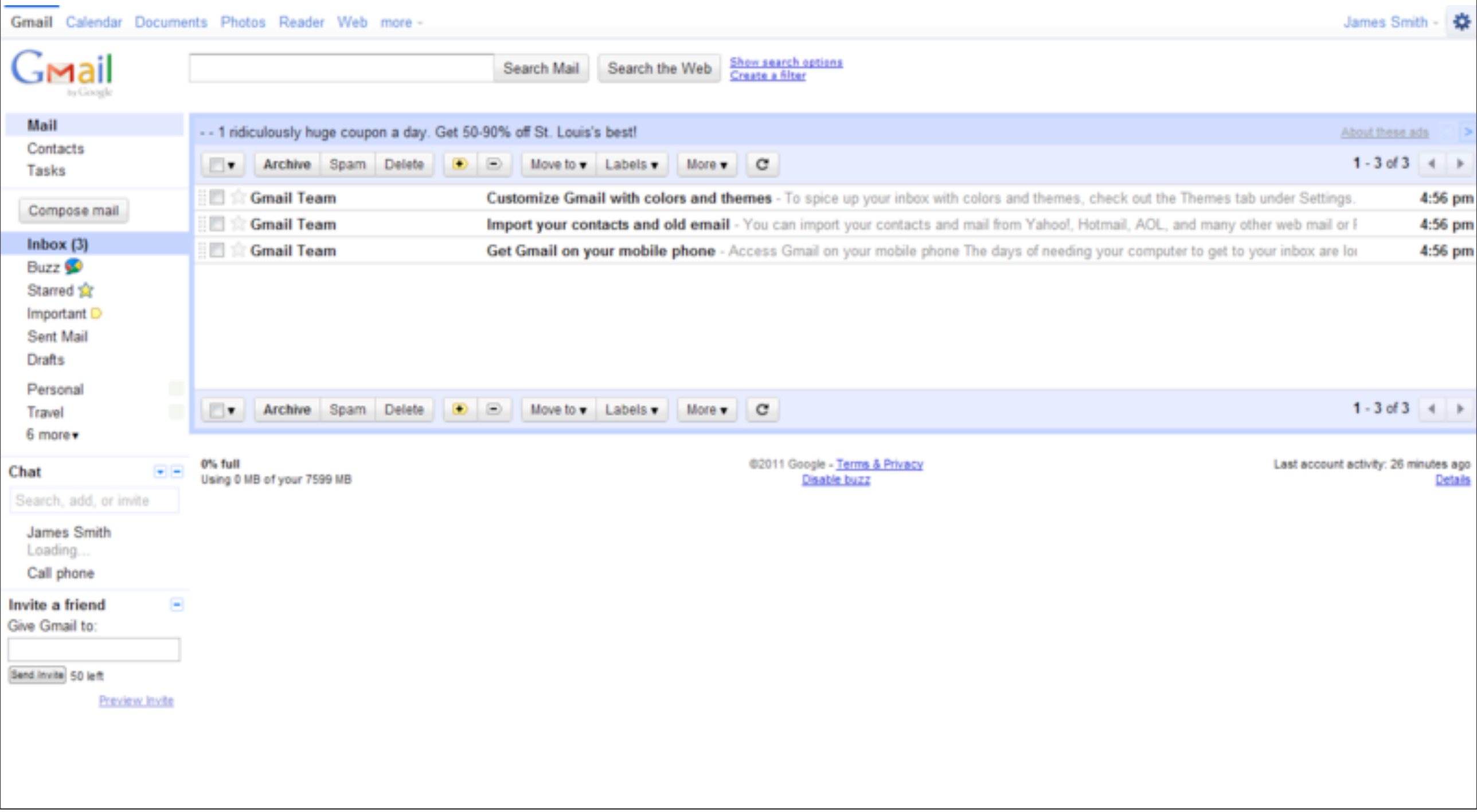
Paul Graham on JavaScript:

"I would not even use Javascript, if I were you... Most of the JavaScript I see on the Web isn't necessary, and much of it breaks."

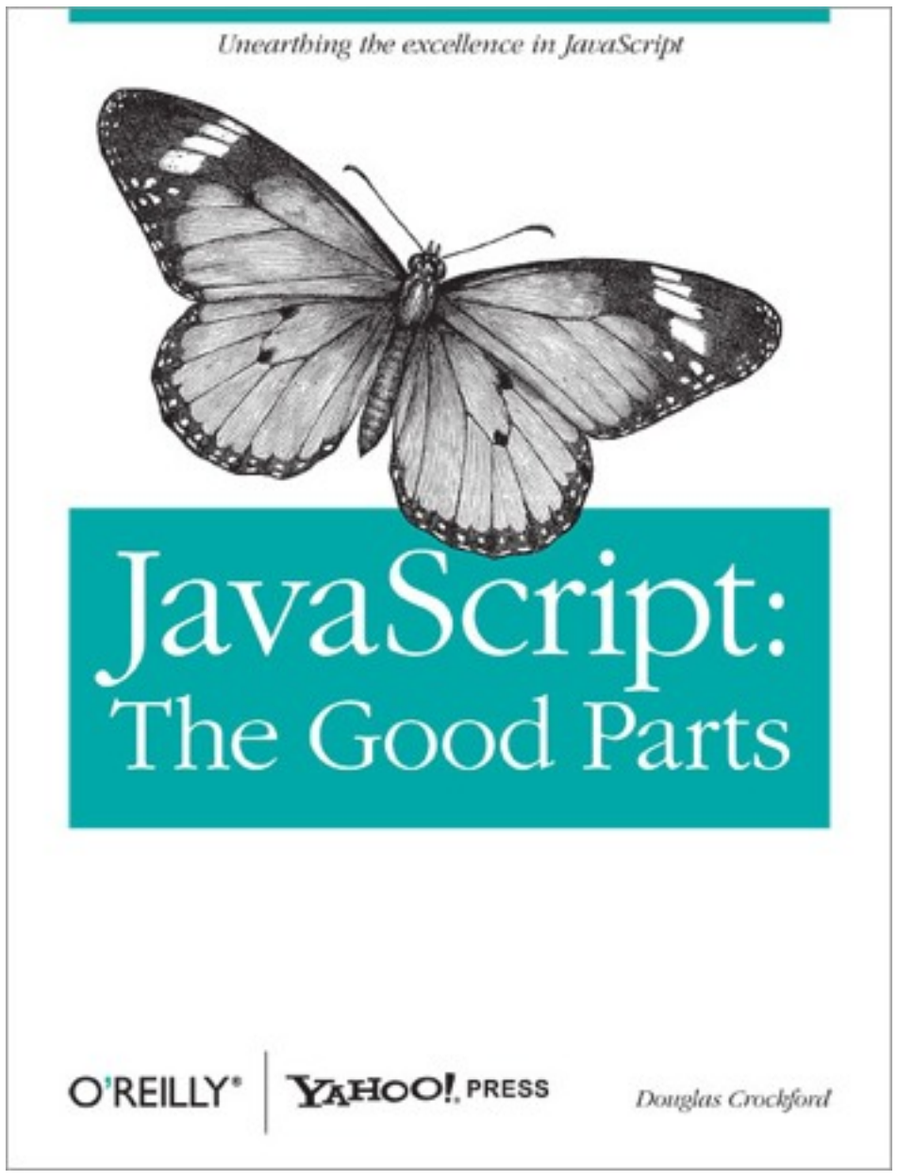
("The Other Road Ahead," 2001)

2004-2008: Medieval JavaScript

Ajax!


Gmail Calendar Documents Photos Reader Web more
James Smith
Search Mail
Search the Web
Show search options
Create a filter
Mail
Contacts
Tasks
Compose mail
Inbox (3)
Buzz
Starred
Important
Sent Mail
Drafts
Personal
Travel
6 more
Chat
Search, add, or invite
James Smith
Loading...
Call phone
Invite a friend
Give Gmail to:
Send Invite
50 left
Preview Invite
1 ridiculously huge coupon a day. Get 50-90% off St. Louis's best!
About these ads
Archive
Spam
Delete
Move to
Labels
More
1 - 3 of 3
Gmail Team
Customize Gmail with colors and themes - To spice up your inbox with colors and themes, check out the Themes tab under Settings.
4:56 pm
Gmail Team
Import your contacts and old email - You can import your contacts and mail from Yahoo!, Hotmail, AOL, and many other web mail or
4:56 pm
Gmail Team
Get Gmail on your mobile phone - Access Gmail on your mobile phone The days of needing your computer to get to your inbox are lo
4:56 pm
0% full
Using 0 MB of your 7599 MB
©2011 Google - Terms & Privacy
Disable buzz
Last account activity: 26 minutes ago
Details

Crockford!


Unearthing the excellence in JavaScript
JavaScript:
The Good Parts
O'REILLY
YAHOO! PRESS
Douglas Crockford


JSLint

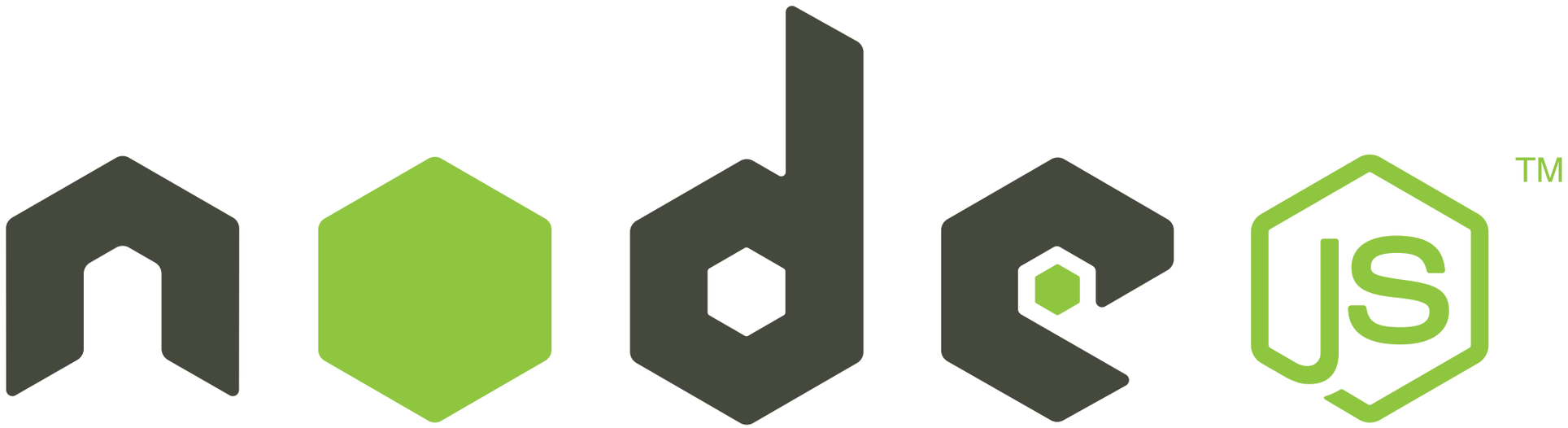
Solid libraries!

Solid libraries!

...And probably others!

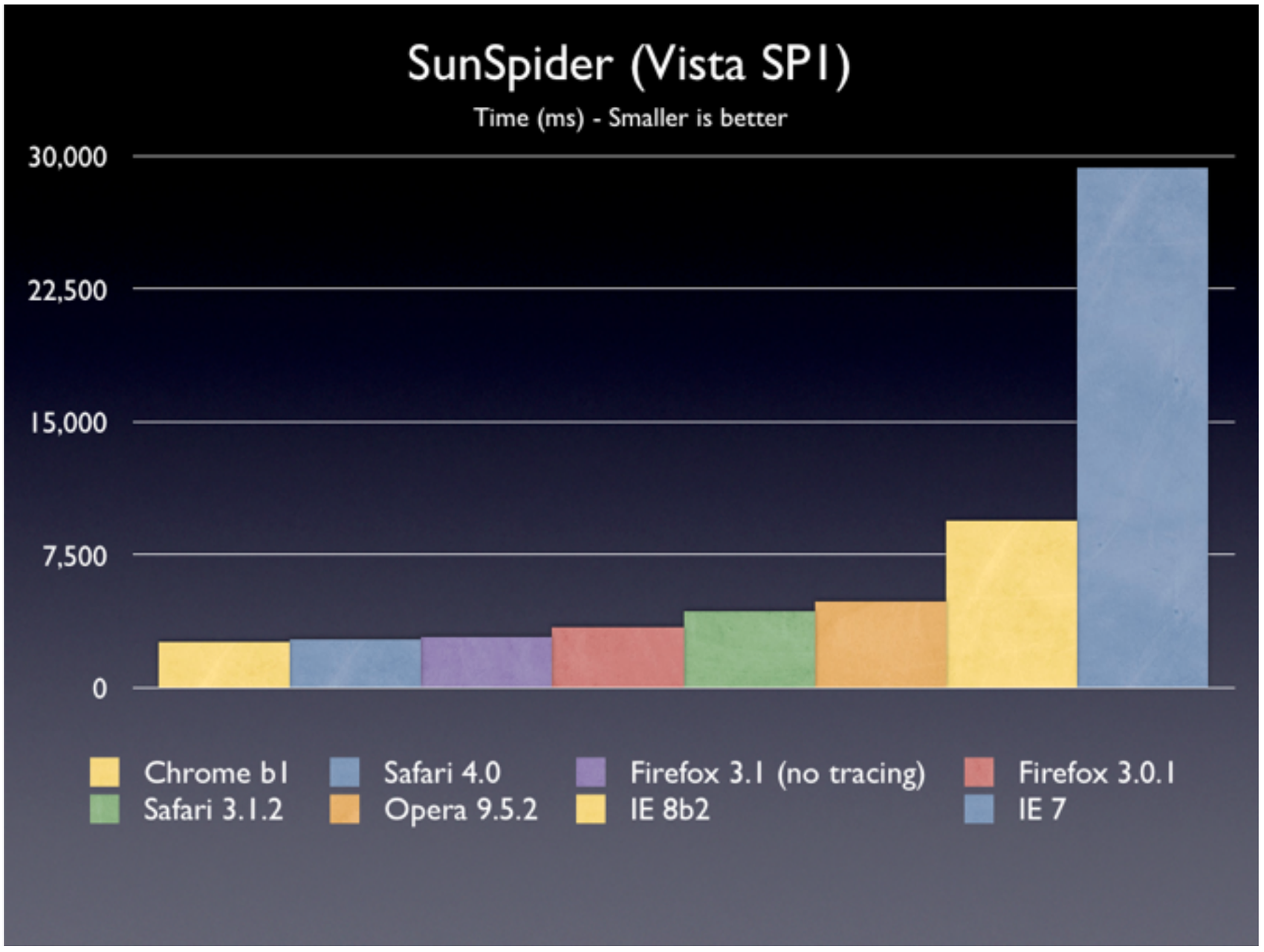
2009-: Modern JavaScript

JavaScript on the Server

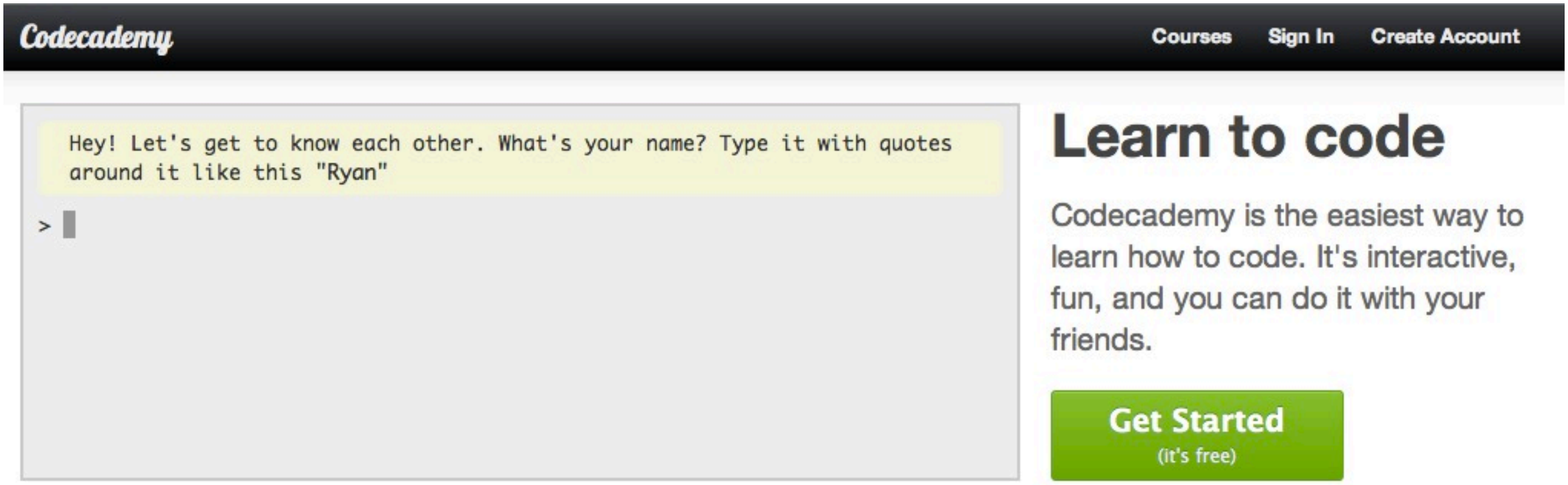


JavaScript as an Embedded Scripting Language

JavaScript is Now FAST!



The New First Language?



http://www.codecademy.com/

Everyone Who Knows JavaScript Feels Like Superman!


CoffeeScript

"A Little Language"

Timeline

Timeline

Christmas 2009: First announced on HN

Timeline

Christmas 2009: First announced on HN

Christmas 2010: 1.0 Released

Timeline

Christmas 2009: First announced on HN

Christmas 2010: 1.0 Released

April 13, 2011: Added to Ruby on Rails

sryche commented on 9f09aeb — April 14, 2011

sryche commented on 9f09aeb — April 14, 2011

I liked coffeescript before rails started using it.
I outhipstered you guys. Deal with it.

serradura commented on 9f09aeb — April 14, 2011

lucashungaro commented on 9f09aeb — April 14, 2011

.

sryche commented on 9f09aeb — April 14, 2011

reu commented on 9f09aeb — April 14, 2011

pbalduino commented on 9f09aeb — April 15, 2011

https://github.com/rails/rails/commit/9f09aeb8273177fc2d09ebdafcc76ee8eb56fe33

David Heinemeier Hansson

@dhh
DHH

Been writing a bunch more CoffeeScript lately. It's only getting better and nicer. What a leap forward for JavaScript.

4 Nov via Tweetie for Mac ☆ Favorite ↻ Retweet ↩ Reply

Ward Cunningham

Ward Cunningham

"CoffeeScript and the environment will all the powerful browsers is the closest I felt to the power I had twenty years ago in Smalltalk."

—Interview with InfoQ, November 2011

Brendan Eich (!)

Brendan Eich (!)

⟵ CoffeeScript user

Eich + Ashkenas at JsConf 2011

Who uses it?

Trello
Organize anything, together.

37signals

posterous™

Hotel Tonight

airbnb™

CloudApp

https://github.com/jashkenas/coffee-script/wiki/In-The-Wild

Ponder This...

CoffeeScript is the #12 most popular language on GitHub

https://github.com/languages/CoffeeScript

CoffeeScript: A Bird's-Eye View

Things CoffeeScript Isn't:

Things CoffeeScript Isn't:

Ruby/Python

Things CoffeeScript Isn't:

Ruby/Python

jQuery

Things CoffeeScript Isn't:

Ruby/Python

jQuery

GWT

Things CoffeeScript Isn't:

Ruby/Python

jQuery

GWT

Dart

Things CoffeeScript Isn't:

Ruby/Python

jQuery

GWT

Dart

```
alert 'Hello World!'  # 1 line!!!
```

"It's Just JavaScript"

"It's Just JavaScript"

```
a = b          # var a = b;
```

"It's Just JavaScript"

```
a = b       # var a = b;
x is y      # x === y;
```

"It's Just JavaScript"

```
a = b       # var a = b;
x is y      # x === y;
f arg       # f(arg);
```

"It's Just JavaScript"

```
a = b          # var a = b;
x is y         # x === y;
f arg          # f(arg);

f = -> x
# var f = function() {return x;};
```

"It's Just JavaScript"

```
a = b       # var a = b;
x is y      # x === y;
f arg       # f(arg);

f = -> x
# var f = function() {return x;};
```

No standard library. No additional types.
Nothing but sweet, sweet syntax!

Ruby-isms

Ruby-isms

```
"That costs $#{price}"
# "That costs $" + price
```

Wael Al-Sallami
@wa3l

Follow

CoffeeScript has Ruby-style string interpolation?!
my God this thing is so sweet I almost can taste it!

8:16 AM - 27 Jun 11 via Twitter for Mac · Embed this Tweet

Reply Retweet Favorite

Streamlined Literals (I)

Streamlined Literals (I)

```
[
   "Commas?"
   "We"
   "don't"
   "need"
   "no"
   "stinking"
   "commmas!"
 ]
```

Streamlined Literals (II)

Streamlined Literals (II)

```
outer:
  inner:
    "You got YAML in my JSON!"
```

```
{outer:
  {inner:
    "You got YAML in my JSON!"
  }
}
```

{key: key} Shorthand and Destructuring Assignment

{key: key} Shorthand and Destructuring Assignment

```
obj = {x}  # obj = {x: x};
```

{key: key} Shorthand and Destructuring Assignment

```
obj = {x}   # obj = {x: x};

{x} = obj   # x = obj.x;
```

{key: key} Shorthand and Destructuring Assignment

```
obj = {x}   # obj = {x: x};

{x} = obj   # x = obj.x;

func = ({x}) ->
func = function(o) {
  var x = o.x;
}
```

More Syntactic Sugar

More Syntactic Sugar

```
f() if z   # if (z) { f(); }
```

More Syntactic Sugar

```
f() if z   # if (z) { f(); }
f?()       # if (...) { f(); }
```

More Syntactic Sugar

```
f() if z   # if (z) { f(); }
f?()       # if (...) { f(); }
obj?       # obj != null;
```

More Syntactic Sugar

```
f() if z   # if (z) { f(); }
f?()       # if (...) { f(); }
obj?       # obj != null;
x ? y      # x != null ? x : y;
```

More Syntactic Sugar

```
f() if z   # if (z) { f(); }
f?()       # if (...) { f(); }
obj?       # obj != null;
x ? y      # x != null ? x : y;
```

Plus, significant whitespace...

The Beauty of Indentation

source: https://github.com/TrevorBurnham/connect-assets

The Beauty of Indentation

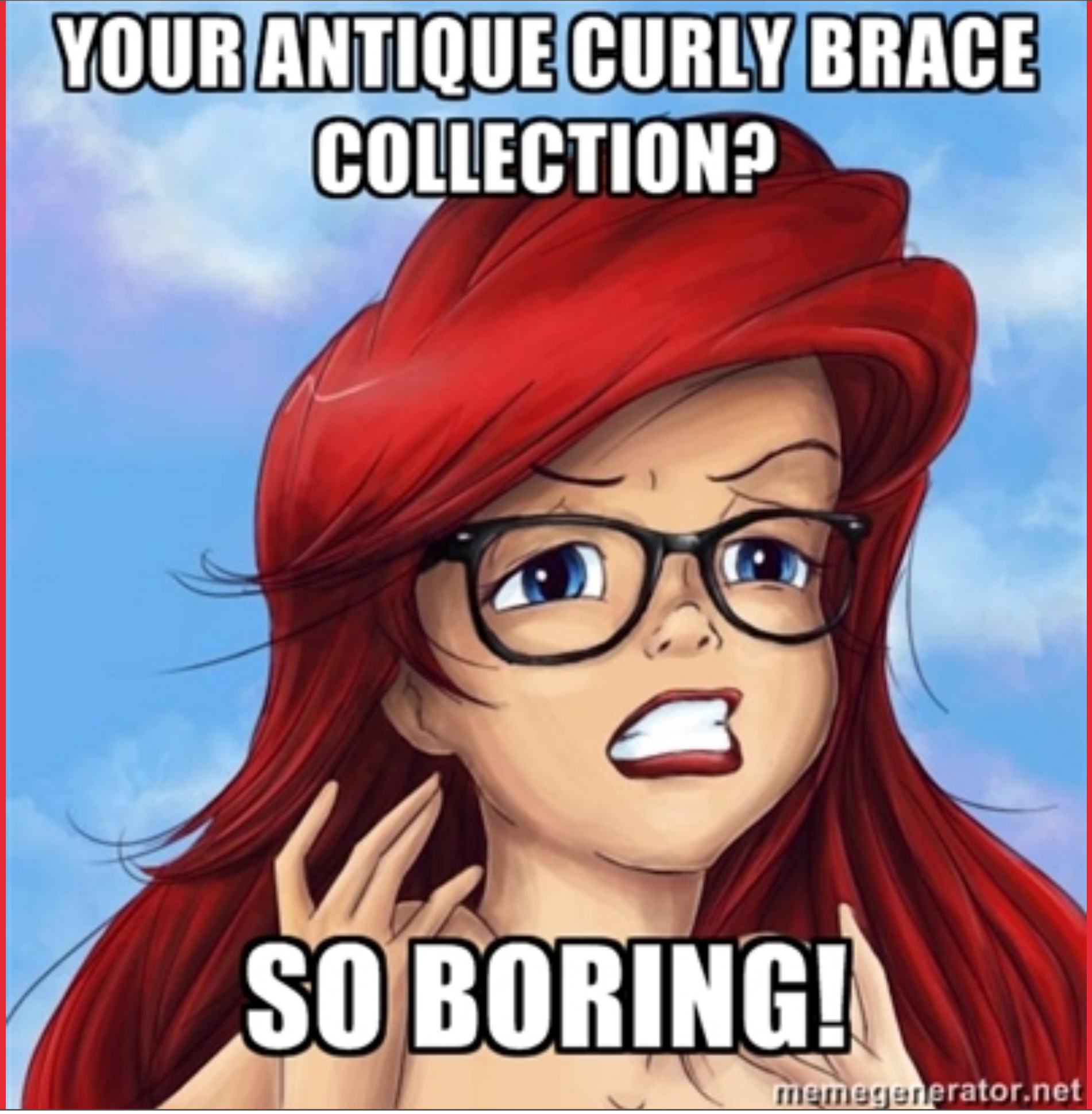
source: https://github.com/TrevorBurnham/connect-assets

```
for ext in exts
  sourcePath = stripExt(route) + ".#{ext}"
  try
    stats = fs.statSync @absPath(sourcePath)
  if ext is 'css'
    {mtime} = stats
    if timeEq mtime, @cache.map[route]?.mtime
      css = @cache.map[route].data
    else
      css = fs.readFileSync @absPath(sourcePath)
```

The Curly-Braced Equivalent

The Curly-Braced Equivalent

```
var css, ext, mtime, sourcePath, stats, _i, _len, _ref;
for (_i = 0, _len = exts.length; _i < _len; _i++) {
  ext = exts[_i];
  sourcePath = stripExt(route) + ("." + ext);
  try {
    stats = fs.statSync(this.absPath(sourcePath));
    if (ext === 'css') {
      mtime = stats.mtime;
      if (timeEq(mtime, (_ref = this.cache.map[route]) !=
null ? _ref.mtime : void 0)) {
        css = this.cache.map[route].data;
      } else {
        css = fs.readFileSync(this.absPath(sourcePath));
      }
    }
  } catch (_e) {}
}
```


YOUR ANTIQUE CURLY BRACE COLLECTION?
SO BORING!
memegenerator.net

II

Language Features

The Wrapper

The Wrapper

CoffeeScript in:

```
console.log i for i in arr
```

JavaScript out:

```
(function() {
  var i, _i, _len;
  for (_i = 0, _len = arr.length; _i < _len; _i++) {
    i = arr[_i];
    console.log(i);
  }
}).call(this);
```

Why Use The Wrapper?

source: http://stackoverflow.com/questions/5211638/

Pattern for CoffeeScript modules

4

8

While reviewing the source code for CoffeeScript on Github, I noticed that most, if not all, of the modules are defined as follows:

```
(function() {
    ...
}).call(this);
```

This pattern looks like it wraps the entire module in an anonymous function and calls itself.

What are the pros (and cons) of this approach? Are there other ways to accomplish the same goals?

javascript | node.js | coffeescript

link | edit | close | flag

asked **Mar 6 at 16:10**

James Sun

469 ●4 ●19

83% accept rate

YOU

GOTTA BELIEVE!

Things That Don't Create Scope in JavaScript

Things That Don't Create Scope in JavaScript

Conditionals

Things That Don't Create Scope in JavaScript

Conditionals

Parentheses

Things That Don't Create Scope in JavaScript

Conditionals

Parentheses

Objects

Things That Don't Create Scope in JavaScript

Conditionals

Parentheses

Objects

Loops

Things That Don't Create Scope in JavaScript

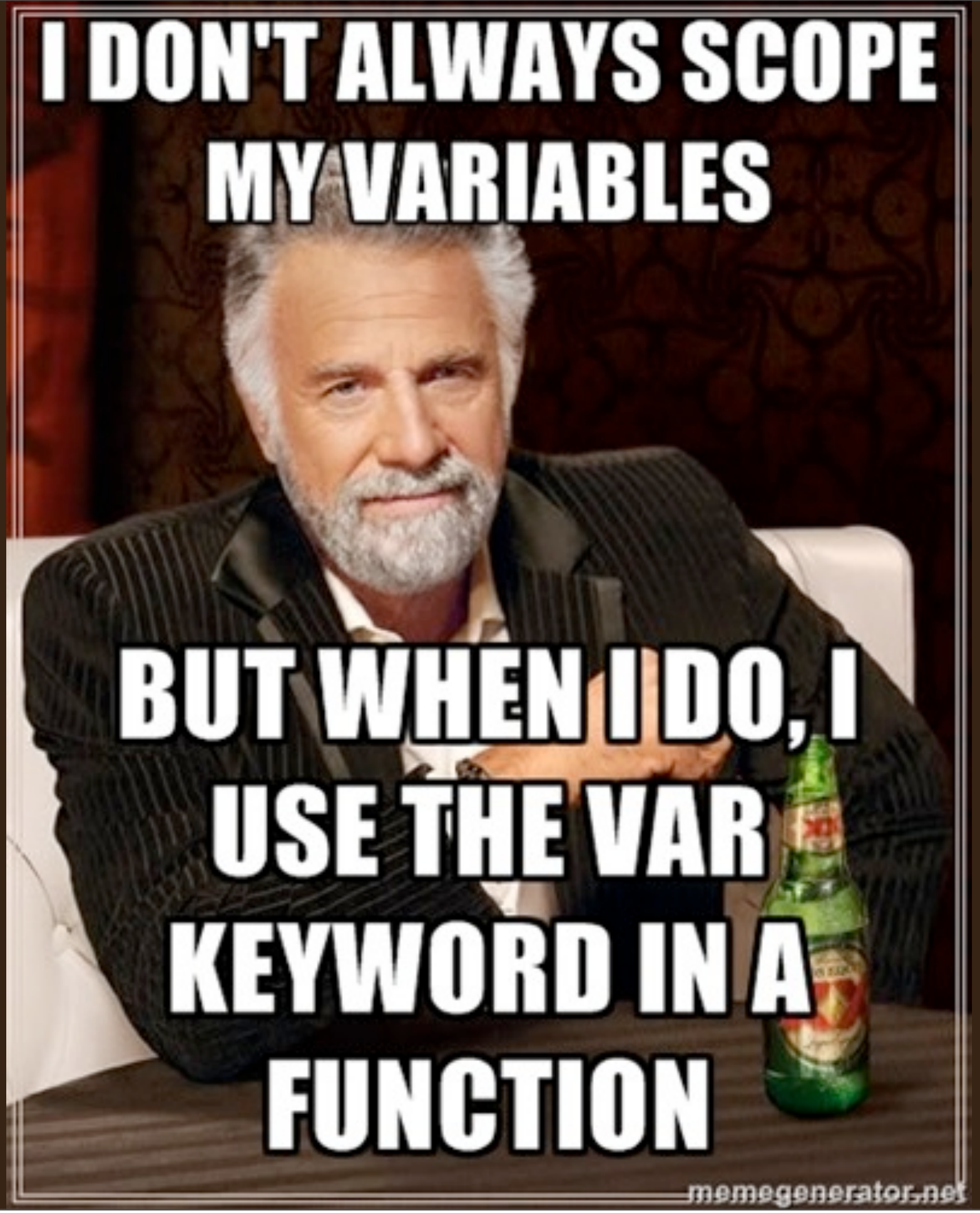
Conditionals

Parentheses

Objects

Loops

Files (!)


I DON'T ALWAYS SCOPE MY VARIABLES
BUT WHEN I DO, I USE THE VAR KEYWORD IN A FUNCTION
memegenerator.net

Don't Let This Happen to You!

31
10 / 2011

How One Missing `var` Ruined our Launch

Well, that was a veritable shitstorm (sorry for the language). Long story short, MelonCard was featured today on TechCrunch (along with other 500Startups companies, also on VentureBeat, Forbes, ...) and everything broke all at once. Every. little. thing. We had rolled out a huge change to MelonCard over the last few days to make our site a seamless "everything just updates" look-good / feel-good product using NodeJS long-polling with a slick KnockoutJS dynamic jQuery Templates front end. We did our due diligence of manual and unit testing, mixed with a full suite of Vows for Node. All systems check, full steam ahead, right? Not so fast.

Our system in NodeJS takes input from a user describing his state, say "I am waiting on these two records to be updated," and the server (based on a timestamp check) returns back either "Your records are up to date" or "Record xxx has changed to yyy." (It's actually a bit more complicated with Redis shared variables, sessions and other security checks to interface between Rails, mySQL, Redis, and Node). It's beautiful simple, but even simple code in NodeJS can be hell when things don't go quite as planned. That happened today.

After the articles dropped today, we faced a moderate load of excited users (let's say 50-100 new users over an hour period). All of a sudden, everything went haywire. No pages worked any more; our inboxes started to fill up with "Your product just hangs." I grabbed a coffee and readied for battle.

http://blog.meloncard.com/post/12175941935

CoffeeScripters Need No **`var`**!

CoffeeScripters Need No **var**!

```
console.log i for i in arr
```

```
(function() {
  var i, _i, _len;
  for (_i = 0, _len = arr.length; _i < _len; _i++) {
    i = arr[_i];
    console.log(i);
  }
}).call(this);
```

JS Best Practices, CoffeeScript Defaults!

JS Best Practices, CoffeeScript Defaults!

Parappa the Wrapper

JS Best Practices, CoffeeScript Defaults!

Parappa the Wrapper

Proper Indentation

JS Best Practices, CoffeeScript Defaults!

Parappa the Wrapper

Proper Indentation

Avoiding ==

JS Best Practices, CoffeeScript Defaults!

Parappa the Wrapper

Proper Indentation

Avoiding ==

Packaging extensible objects as “classes”

Plain, Ordinary JavaScript Objects

Plain, Ordinary JavaScript Objects

```
lifeEtAl =
  answer: 42
  showAnswer: ->
    console.log @answer # @ == this
```

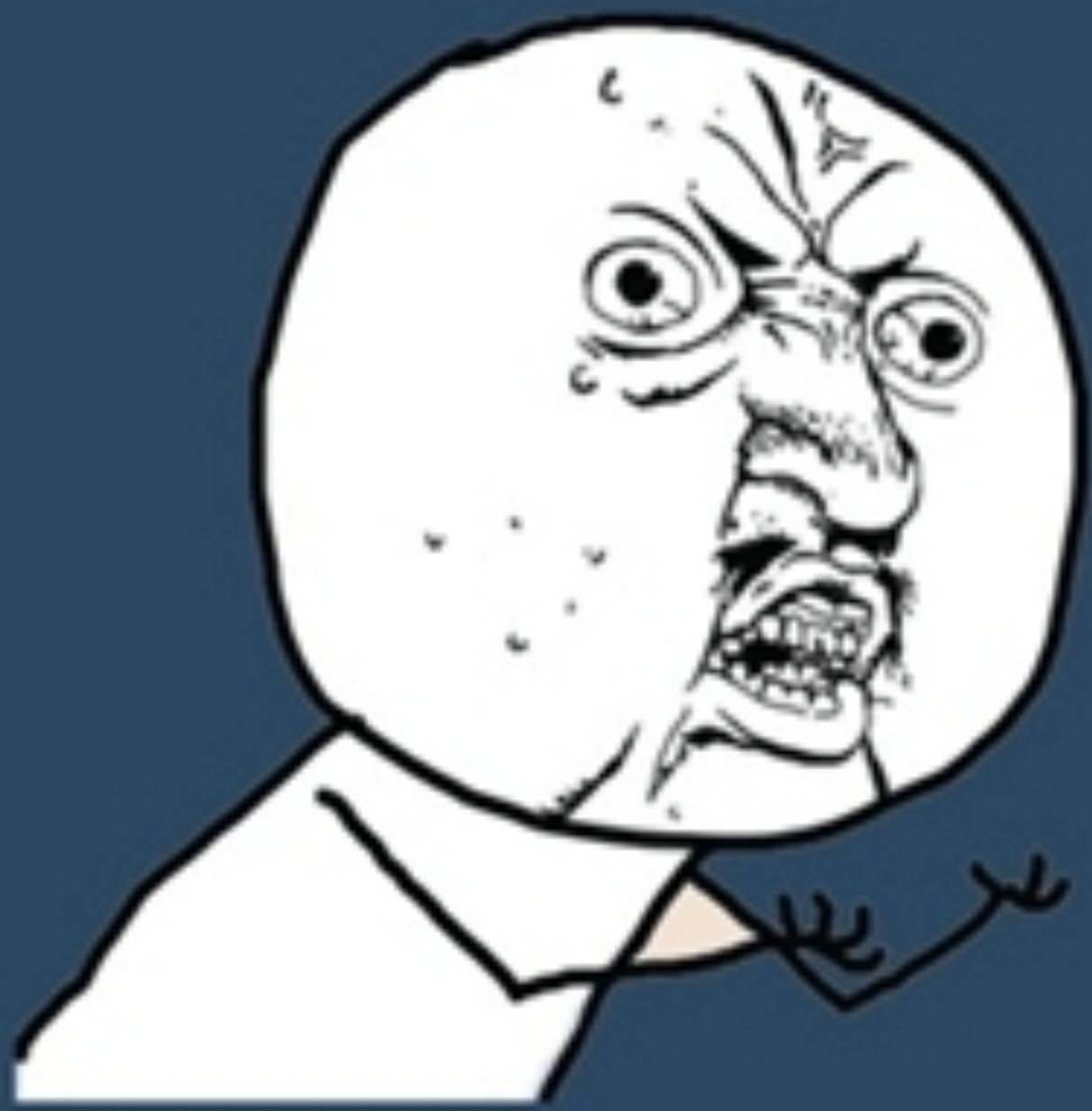
Plain, Ordinary JavaScript Objects

```
lifeEtAl =
  answer: 42
  showAnswer: ->
    console.log @answer # @ == this

lifeEtAl.showAnswer()
```

Plain, Ordinary JavaScript Objects

```
lifeEtAl =
  answer: 42
  showAnswer: ->
    console.log @answer # @ == this

lifeEtAl.showAnswer()

setTimeout lifeEtAl.showAnswer, 10
```

JAVASCRIPT



Y U NO BIND FUNCTIONS???

Classes to the Rescue!

Classes to the Rescue!

```
class LifeEtAl
  answer: 42
  showAnswer: =>   # fat ->
    console.log @answer
```

Classes to the Rescue!

```
class LifeEtAl
  answer: 42
  showAnswer: =>   # fat ->
    console.log @answer

myLife = new LifeEtAl
```

Classes to the Rescue!

```
class LifeEtAl
  answer: 42
  showAnswer: =>   # fat ->
    console.log @answer

myLife = new LifeEtAl

setTimeout myLife.showAnswer, 10
```

How About a Little Privacy?

How About a Little Privacy?

```
class LifeEtAl
  answer = 42
  showAnswer: ->
    console.log answer
```

How About a Little Privacy?

```
class LifeEtAl
  answer = 42
  showAnswer: ->
    console.log answer

myLife = new LifeEtAl
```

How About a Little Privacy?

```
class LifeEtAl
  answer = 42
  showAnswer: ->
    console.log answer

myLife = new LifeEtAl

setTimeout myLife.showAnswer, 10
```

Using a Constructor

Using a Constructor

```
class Circle
  twoPi = Math.PI * 2
  constructor: (@radius) ->
    @radiusSqr = Math.pow @radius, 2
  diameter: => twoPi * @radius
  area: => Math.PI * @radiusSqr
```

Using a Constructor

```
class Circle
  twoPi = Math.PI * 2
  constructor: (@radius) ->
    @radiusSqr = Math.pow @radius, 2
  diameter: => twoPi * @radius
  area: => Math.PI * @radiusSqr

c = new Circle(5)
console.log c.diameter() # 31.4159
console.log c.area()     # 78.5398
```

Let's Try a Little Inheritance

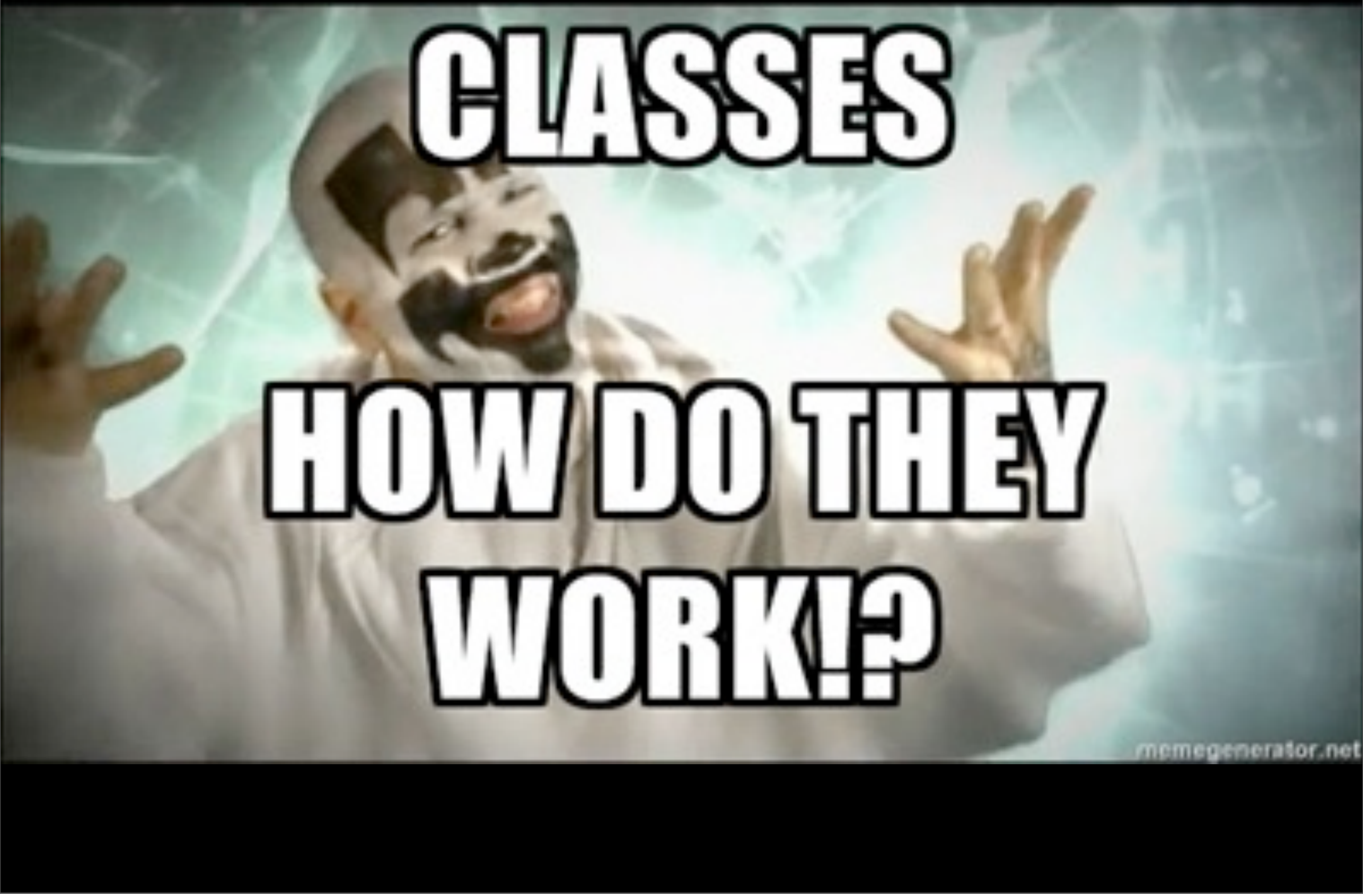
Let's Try a Little Inheritance

```
class Document extends Backbone.Model
  defaults:
    title: 'Untitled'
```

Doing Inheritance via a JavaScript Library

Doing Inheritance via a JavaScript Library

```
var Document = Backbone.Model.extend({
  defaults: {
    title: 'Untitled'
  }
});
```


CLASSES
HOW DO THEY WORK!?
memegenerator.net

I. The Prototype Chain

I. The Prototype Chain

```
class Document extends Backbone.Model
  defaults:
    title: 'Untitled'
```

I. The Prototype Chain

```
class Document extends Backbone.Model
  defaults:
    title: 'Untitled'

doc = new Document
```

I. The Prototype Chain

```
class Document extends Backbone.Model
  defaults:
    title: 'Untitled'

doc = new Document

Document:: is Document.prototype
```

I. The Prototype Chain

```
class Document extends Backbone.Model
  defaults:
    title: 'Untitled'

doc = new Document

Document:: is Document.prototype

doc.defaults is Document::defaults
```

I. The Prototype Chain

```coffeescript
class Document extends Backbone.Model
  defaults:
    title: 'Untitled'

doc = new Document

Document:: is Document.prototype

doc.defaults is Document::defaults

doc.validate is Backbone.Model::validate
```

I. The Prototype Chain

```
class Document extends Backbone.Model
  defaults:
    title: 'Untitled'

doc = new Document

Document:: is Document.prototype

doc.defaults is Document::defaults

doc.validate is Backbone.Model::validate

doc.hasOwnProperty is Object::hasOwnProperty
```

II. Superclass Methods Can Be Invoked With `super`

II. Superclass Methods Can Be Invoked With `super`

```
class AppleDevice
  constructor: (cost) ->
    bankAccount.deduct cost
    bankAccount.deduct cost / 4 # AppleCare
```

II. Superclass Methods Can Be Invoked With `super`

```
class AppleDevice
  constructor: (cost) ->
    bankAccount.deduct cost
    bankAccount.deduct cost / 4 # AppleCare

class iPhone extends AppleDevice
  constructor: (cost) ->
    super  # equivalent to super(cost)
    setInterval (->
      bankAccount.deduct cost / 4
    ), ONE_MONTH
```

III. Superclass Properties are Copied

III. Superclass Properties are Copied

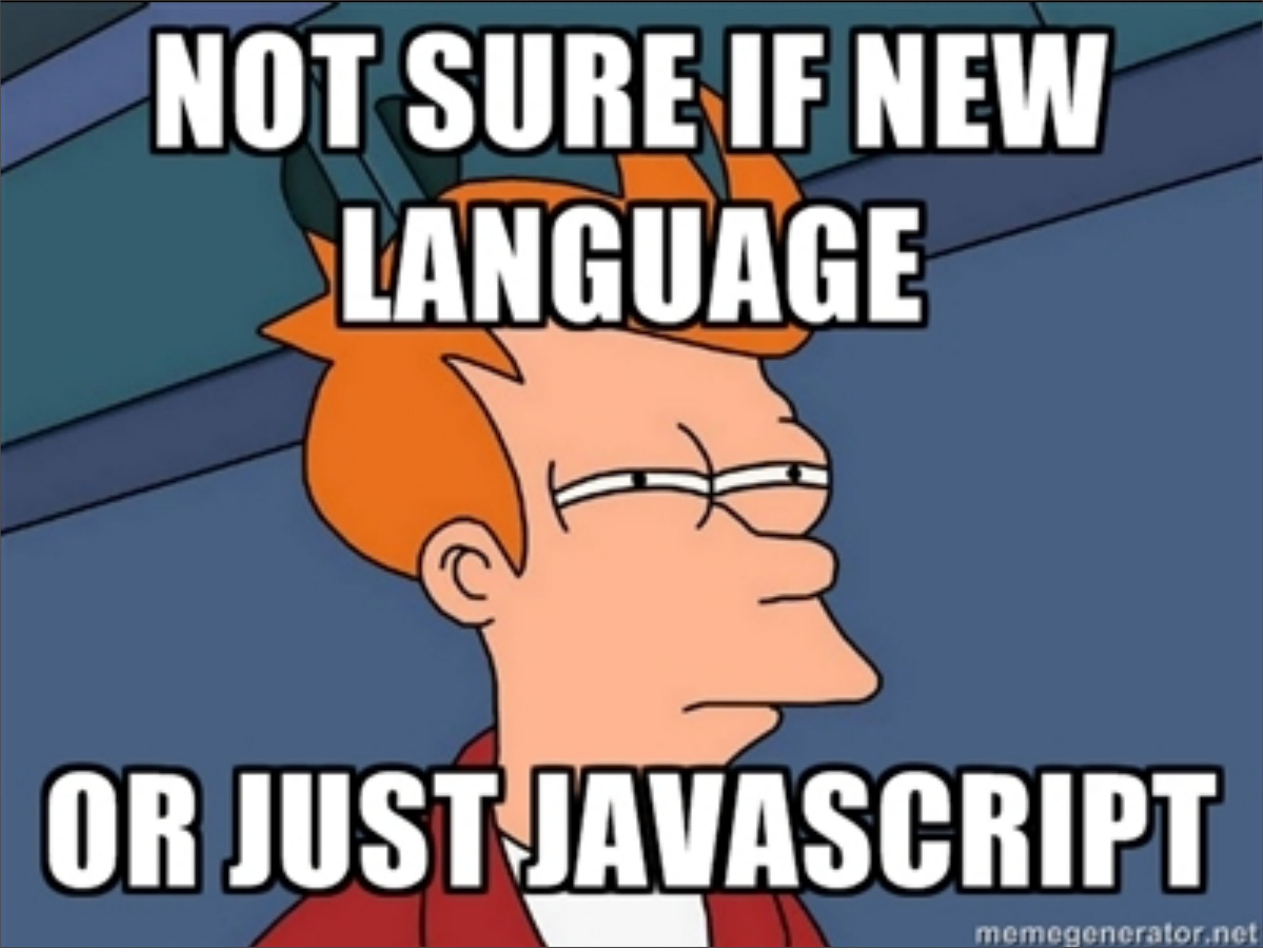
```
class Primate
  @thumbs = 'opposable'
```

III. Superclass Properties are Copied

```
class Primate
  @thumbs = 'opposable'

class Human extends Primate
```

III. Superclass Properties are Copied

```
class Primate
  @thumbs = 'opposable'

class Human extends Primate

Human.thumbs is Primate.thumbs is 'opposable'
```


NOT SURE IF NEW
LANGUAGE
OR JUST JAVASCRIPT
memegenerator.net

III

Working with CoffeeScript

Installing `coffee`

Installing `coffee`

Step 1: Install Node

Installing `coffee`

Step 1: Install Node

Step 2: `npm install -g coffee-script`

Installing `coffee`

Step 1: Install Node

Step 2: `npm install -g coffee-script`

Want a different version?

```
npm install -g coffee-script@1.2.0
```

Installing `coffee`

Step 1: Install Node

Step 2: `npm install -g coffee-script`

Want a different version?

```
npm install -g coffee-script@1.2.0
```

Compiling

Compiling

```
coffee -w
```

Compiling

```
coffee -w
```

Rails / other web frameworks

Compiling

`coffee -w`

Rails / other web frameworks

Middleman / other *static* web frameworks

Compiling

`coffee -w`

Rails / other web frameworks

Middleman / other *static* web frameworks

LiveReload (Mac)

Compiling

`coffee -w`

Rails / other web frameworks

Middleman / other *static* web frameworks

LiveReload (Mac)

Write a Cakefile

A Piece of Cake

source: https://github.com/sstephenson/node-coffee-project/

A Piece of Cake

source: https://github.com/sstephenson/node-coffee-project/

```
build = (watch, callback) ->
  if typeof watch is 'function'
    callback = watch
    watch = false
  options = ['-c', '-o', 'lib', 'src']
  options.unshift '-w' if watch
  coffee = spawn 'coffee', options
  coffee.stdout.on 'data', (data) -> print data.toString()
  coffee.stderr.on 'data', (data) -> print data.toString()
  coffee.on 'exit', (status) -> callback?() if status is 0

task 'build', 'Compile CoffeeScript source files', ->
  build()

task 'test', 'Run the test suite', ->
  build ->
    require.paths.unshift __dirname + "/lib"
    {reporters} = require 'nodeunit'
    process.chdir __dirname
    reporters.default.run ['test']
```


YO DAWG, I HEARD YOU
LIKE COFFEESCRIPT
SO I MADE YOU A CAKEFILE SO YOU CAN
DO YOUR COFFEESCRIPT PROJECT BUILD
SCRIPTS IN COFFEESCRIPT
memegenerator.net

Editing

Text editor plugins

New Page | Edit Page | Page History

- Brewer: **rbrcurtis**'s Brewer project browser based editor uses the ace editor (which cloud9 also uses) to show the compiled JS alongside your coffeescript, much like the coffeescript homepage, and runs a simple nodejs webserver on your machine to provide file manipulation functionality.
- Cloud9IDE: **tanepiper**'s Cloud9IDE CoffeeScript Live extension - provides a live debug environment for .coffee files in Cloud9. Of course, written in CoffeeScript too!
- Coda/SubEthaEdit:
  - **nfiniteset**'s CoffeeScript Coda/SubEthaEdit mode — A syntax mode that provides syntax coloring.
  - **parkerl**'s CoffeeScript.codaplugin - Rough support for compiling and running in Coda's preview
- CodeMirror: coffeescript-codemirror-mode
- Emacs:
  - **defunkt**'s CoffeeScript Major Mode — provides syntax highlighting, indentation support, and some bonus commands.
  - coffeelintnode - provides on-the-fly coffeelint-ing of a coffee-script buffer using node.js, coffeelint and flymake-mode.
- gedit: **wavded**'s gedit-coffeescript — provides syntax highlighting.
- IntelliJ IDEA/RubyMine/PyCharm/PHPStorm/WebStorm:
  - **yeungda**'s coffeescript-idea — provides syntax highlighting. development halted: see next item.
  - **netzpirat**'s CoffeeBrew — an extensive refactoring and improvement upon coffeescript-idea. development halted: see next item.
  - Official Support is *planned* for RubyMine 4, using CoffeeBrew as a base. Whether or not this support will be more widely available is currently up in the air. Please add your comments to the bug tracker if you would like to see CoffeeScript support in other jetbrains products than just RubyMine, with all of the features one would expect from a jetbrains product (completion, inspections/intentions, auto-fix, go-to-definition, etc).

https://github.com/jashkenas/coffee-script/wiki/Text-editor-plugins

Documenting

source: http://coffeescript.org/documentation/docs/grammar.html

Our handy DSL for Jison grammar generation, thanks to Tim Caswell. For every rule in the grammar, we pass the pattern-defining string, the action to run, and extra options, optionally. If no action is specified, we simply pass the value of the previous nonterminal.

```
o = (patternString, action, options) ->
  patternString = patternString.replace /\s{2,}/g, ' '
  return [patternString, '$$ = $1;', options] unless action
  action = if match = unwrap.exec action then match[1] else "(#{action}())"
  action = action.replace /\bnew /g, '$&yy.'
  action = action.replace /\b(?:Block\.wrap|extend)\b/g, 'yy.$&'
  [patternString, "$$ = #{action};", options]
```

Grammatical Rules

In all of the rules that follow, you'll see the name of the nonterminal as the key to a list of alternative matches. With each match's action, the dollar-sign variables are provided by Jison as references to the value of their numeric position, so in this rule:

```
"Expression UNLESS Expression"
```

`$1` would be the value of the first `Expression`, `$2` would be the token for the `UNLESS` terminal, and `$3` would be the value of the second `Expression`.

```
grammar =
```

The **Root** is the top-level node in the syntax tree. Since we parse bottom-up, all parsing must end here.

```
Root: [
  o '',                                       -> new Block
  o 'Body'
  o 'Block TERMINATOR'
]
```

Any list of statements and expressions, separated by line breaks or semicolons.

```
Body: [
  o 'Line',                                   -> Block.wrap [$1]
  o 'Body TERMINATOR Line',                   -> $1.push $3
  o 'Body TERMINATOR'
]
```

Testing

Testing

Mocha

Testing

Mocha

Mocha

Testing

Mocha

Mocha

Mocha

Testing

Mocha

Mocha

Mocha

http://visionmedia.github.com/mocha/

IU

The Future of Web Apps...?

A New Area of Confusion...

How to dynamically update a Jade element from client-side CoffeeScript?

1

I'm trying to dynamically update a table of values using SocketStream. I have a Jade template defining the table:

app.jade:

```
table
  thead
    tr
      th key
      th value
    tbody
      - var jadeItems = [{key:'Test',value:'3.1415'}, {key:'Test2',value:'2.1878'}
      - each item in jadeItems
        tr
          td= item.key
          td= item.value
```

This works for the static data, and now I need to make it dynamic. I have client-side CoffeeScript that receives a SocketStream message containing the new values for the table in JSON format:

app.coffee:

```
SS.events.on('message', (message) ->
  jadeItems = JSON.parse(message)
)
```

I am trying to figure out how to replace the JSON value of items in Jade with with the content of the message, but the 'jadeItems' variable is out of scope in the client-side CoffeeScript.

tagged

coffeescrip

publish-sut

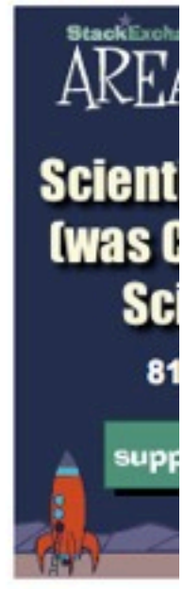
jade × 1

socketstrea

asked 7

viewed 2

active 7



http://stackoverflow.com/questions/7996883/

Where is Your Code?

Does anyone know the Jade syntax for getting the value of a Jade element? Or is there a way to assign to the items variable defined in Jade from within the client-side CoffeeScript? Are there any solid references for Jade syntax?

coffeescript | publish-subscribe | jade | socketstream

link | edit | close | flag

asked **Nov 3 at 14:47**

retrodrone

1,700 ●2 ●4 ●21

100% accept rate

To clarify: You're using Jade client-side? – Trevor Burnham Nov 3 at 14:59

Ooh good question. Maybe not? It's created by SocketStream in the Views directory by default. I assume it's served to the client as is but maybe I'm wrong. I'm new to all this stuff but excited by what I see. Maybe I can update it from server-side CoffeeScript. – retrodrone Nov 3 at 15:01

http://stackoverflow.com/questions/7996883/

Where is Your Code?

Does anyone know the Jade syntax for getting the value of a Jade element? Or is there a way to assign to the items variable defined in Jade from within the client-side CoffeeScript? Are there any solid references for Jade syntax?

coffeescript | publish-subscribe | jade | socketstream

link | edit | close | flag

asked **Nov 3 at 14:47**

retrodrone

1,700 ●2 ●4 ●21

100% accept rate

To clarify: You're using Jade client-side? – Trevor Burnham Nov 3 at 14:59

Ooh good question. Maybe not? It's created by SocketStream in the Views directory by default. I assume it's served to the client as is but maybe I'm wrong. I'm new to all this stuff but excited by what I see. Maybe I can update it from server-side CoffeeScript. – retrodrone Nov 3 at 15:01

http://stackoverflow.com/questions/7996883/

Breaking the Client-Server Boundaries

Breaking the Client-Server Boundaries

Stitch (37signals)

Breaking the Client-Server Boundaries

Stitch (37signals)

browserify

Breaking the Client-Server Boundaries

Stitch (37signals)

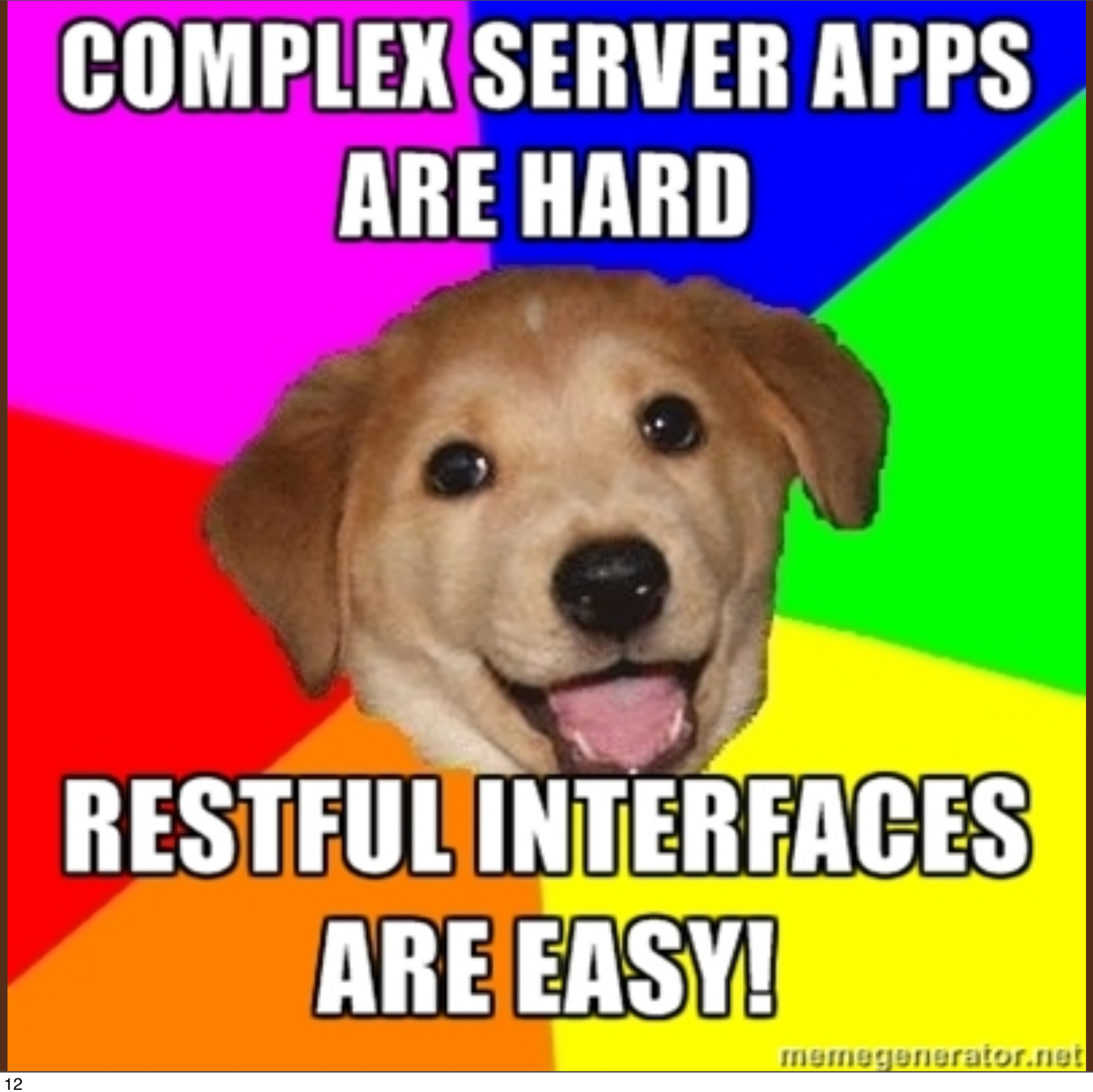
browserify

jsdom

IF I CAN WRITE CODE IN A GREAT LANGUAGE THAT CAN RUN IN BOTH BROWSERS AND SERVERS

WILL THAT CHANGE THE WAY I THINK ABOUT WEB APPS?

memegenerator.net


COMPLEX SERVER APPS
ARE HARD
RESTFUL INTERFACES
ARE EASY!
memegenerator.net

Thanks! Questions?

Follow me **@trevorburnham** and **@coffeescript**

Check out my new book, **Async JavaScript**
http://leanpub.com/asyncjs