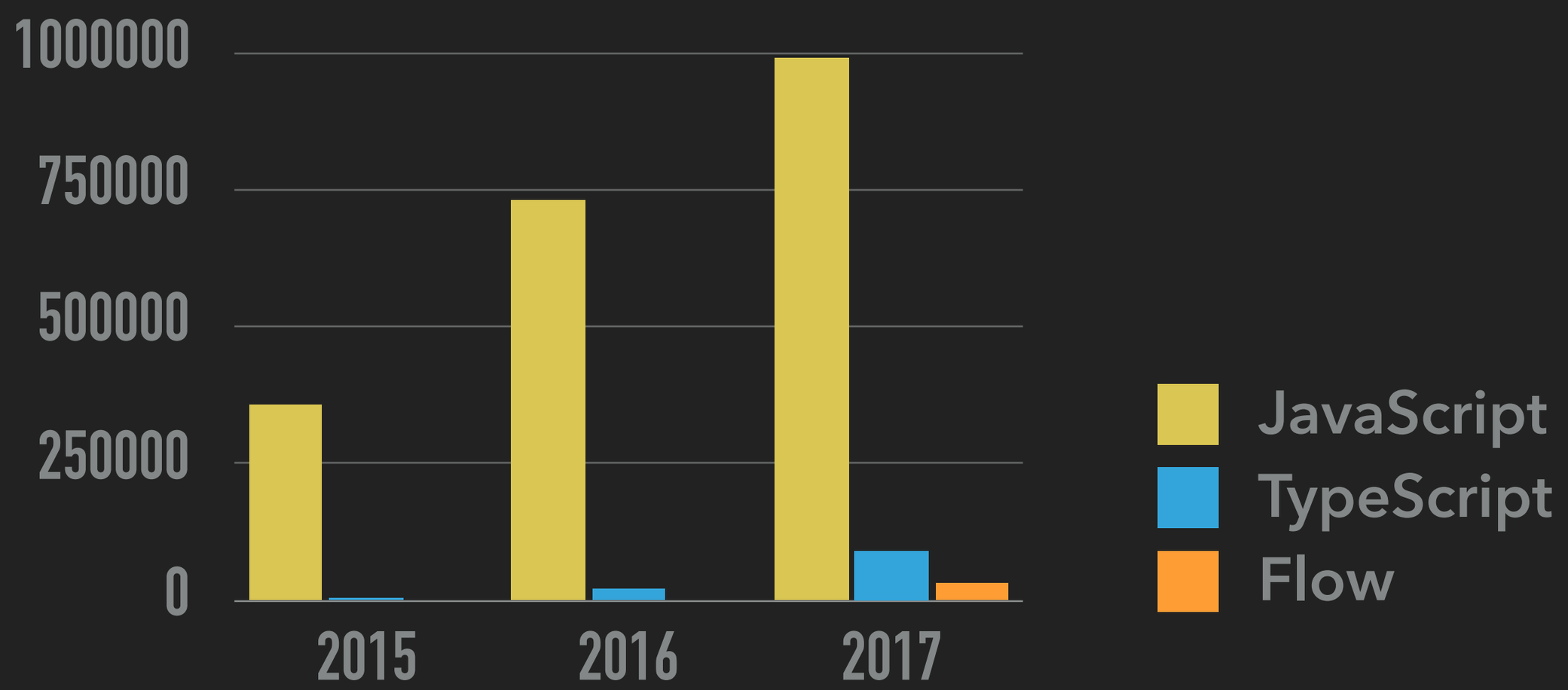
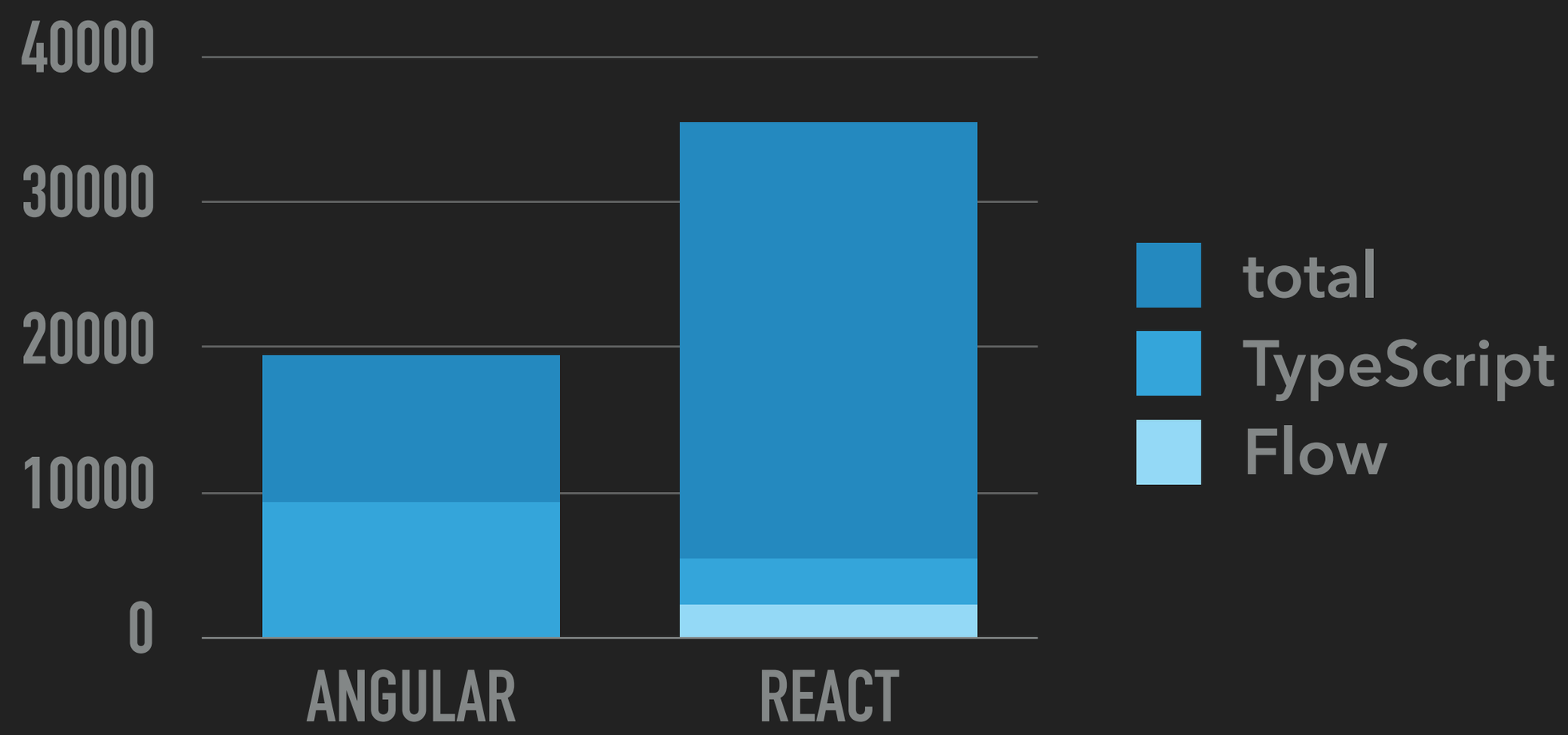
ODESSAJS

STRICT JAVASCRIPT

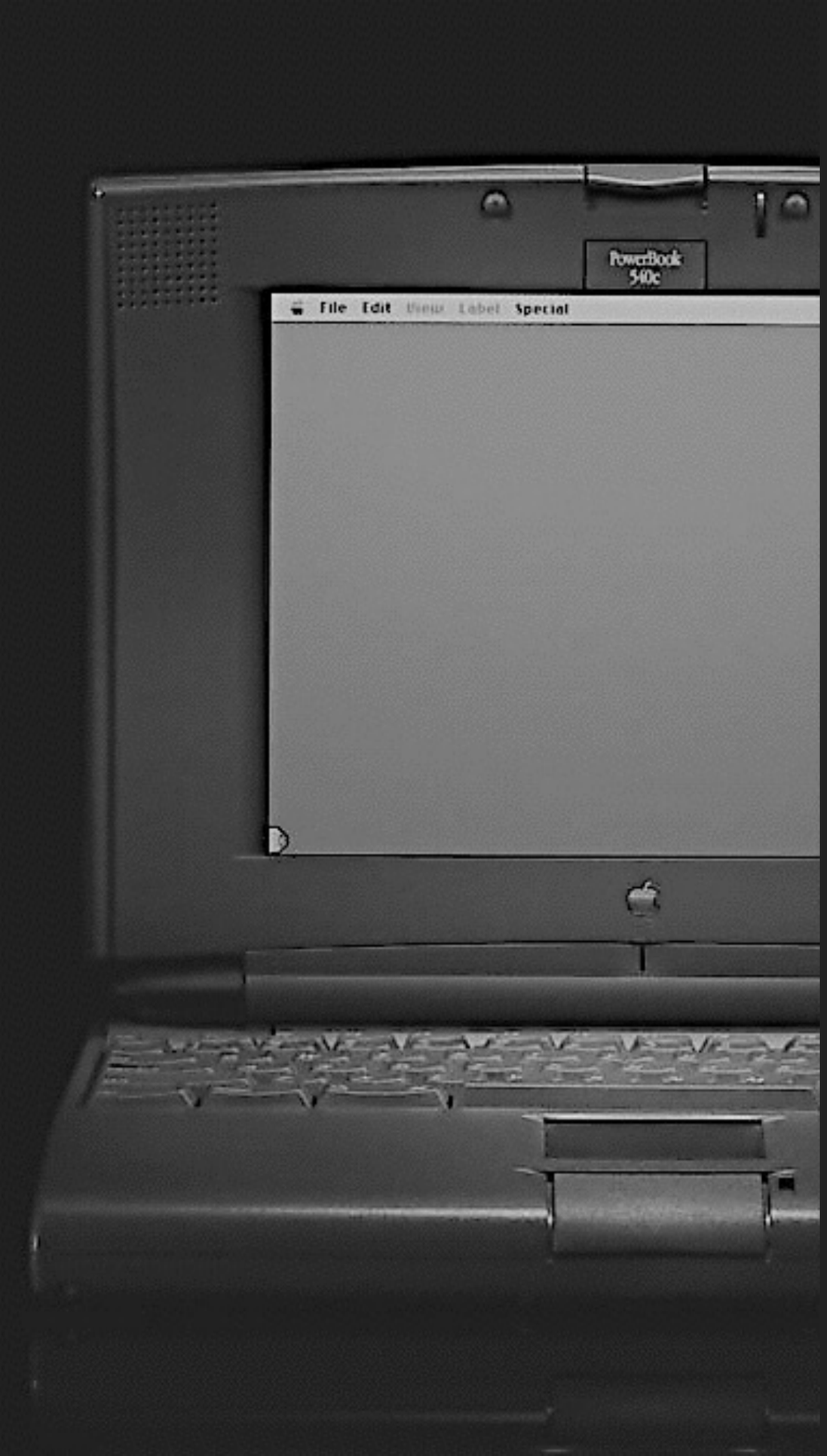
GITHUB LANGUAGES BY PULL REQUESTS



GITHUB FRAMEWORKS



https://bigquery.cloud.google.com/savedquery/563055368311:34dcf5adb2464fbb882349a307cb9067
https://bigquery.cloud.google.com/savedquery/563055368311:e3b4fc8a151f4ef08688a09242581948
https://bigquery.cloud.google.com/savedquery/563055368311:9172761d4f184b438da46b34bc46d39b
https://bigquery.cloud.google.com/savedquery/563055368311:8847e62ecef645b6ba0e3dccf49f532b

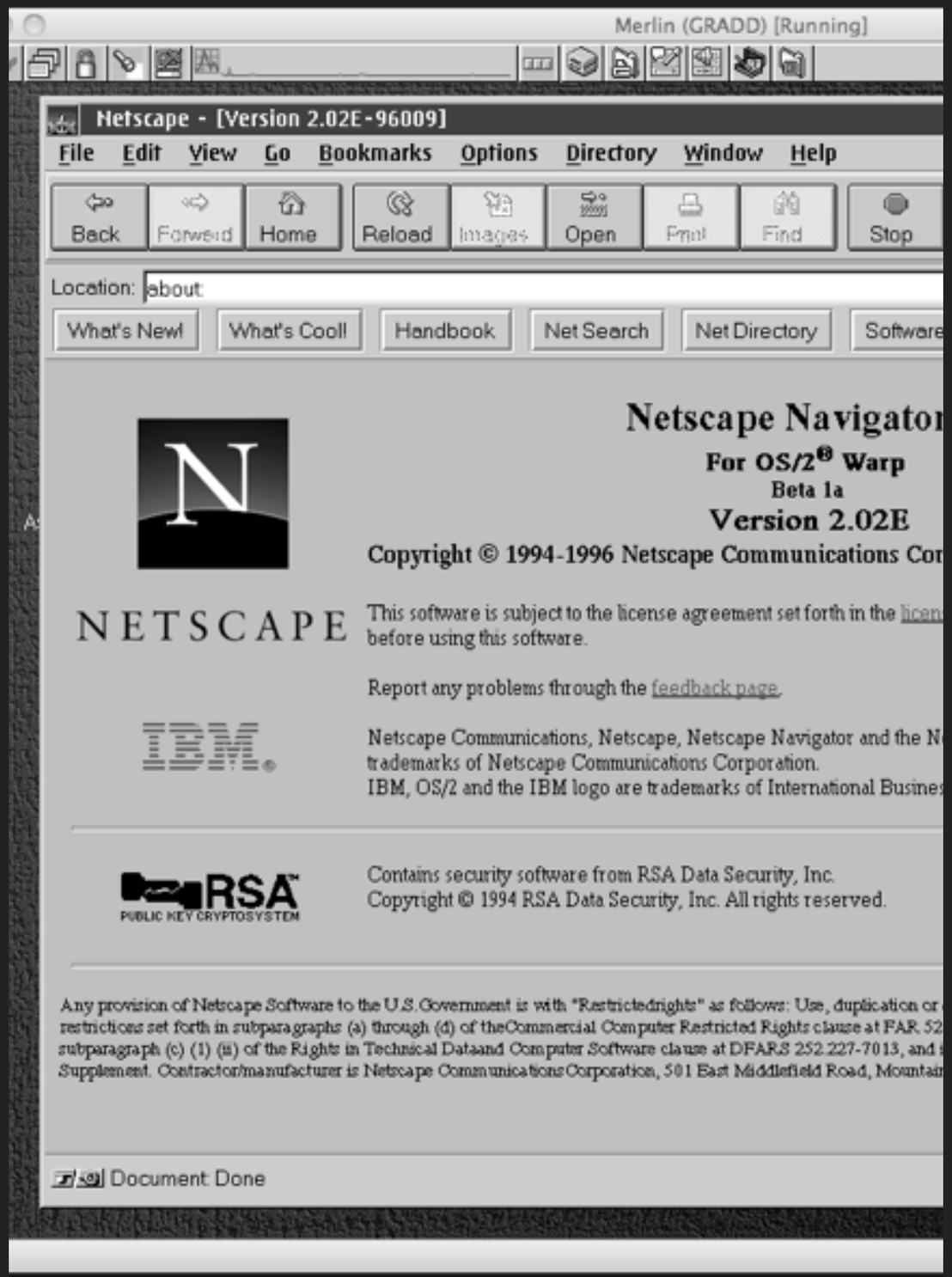


JS TYPE SAFETY

BRIEF HISTORY

1995

BIRTH


Merlin (GRADD) [Running]
Netscape - [Version 2.02E-96009]
File Edit View Go Bookmarks Options Directory Window Help
Back Forward Home Reload Images Open Print Find Stop
Location: about
What's New! What's Cool! Handbook Net Search Net Directory Software
Netscape Navigator
For OS/2® Warp
Beta 1a
Version 2.02E
Copyright © 1994-1996 Netscape Communications Cor
NETSCAPE
This software is subject to the license agreement set forth in the licen
before using this software.
Report any problems through the feedback page.
IBM
Netscape Communications, Netscape, Netscape Navigator and the N
trademarks of Netscape Communications Corporation.
IBM, OS/2 and the IBM logo are trademarks of International Busines
RSA
PUBLIC KEY CRYPTOSYSTEM
Contains security software from RSA Data Security, Inc.
Copyright © 1994 RSA Data Security, Inc. All rights reserved.
Any provision of Netscape Software to the U.S. Government is with "Restrictedrights" as follows: Use, duplication or
restrictions set forth in subparagraphs (a) through (d) of theCommercial Computer Restricted Rights clause at FAR 52
subparagraph (c) (1) (ii) of the Rights in Technical Dataand Computer Software clause at DFARS 252.227-7013, and
Supplement. Contractor/manufacturer is Netscape Communications Corporation, 501 East Middlefield Road, Mountai
Document: Done

I WAS RECRUITED TO NETSCAPE WITH THE PROMISE OF "DOING SCHEME" IN THE BROWSER.

Brendan Eich

LANGUAGES THAT INFLUENCED JS

Scheme	Strong	Dynamic
Java	Strong	Static
Self	Strong	Dynamic
JavaScript	Weak	Dynamic

HTML NEEDED A "SCRIPTING LANGUAGE", A PROGRAMMING LANGUAGE THAT WAS EASY TO USE BY AMATEURS AND NOVICES, WHERE THE CODE COULD BE WRITTEN DIRECTLY IN SOURCE FORM AS PART OF THE WEB PAGE MARKUP.

Brendan Eich

2006 GOOGLE WEB TOOLKIT

gwt.js — ~/Documents/staticjs

Button.java

```
class Button {
  public void setText(String text) {
    DOM.setInnerText(getElement(), text);
  }
}
```

gwt.js

```
//Compiled Javascript (Safari):
$setInnerText(b.element, 'Click Me!');
function $setInnerText(elem, text) {
  while (elem.firstChild) {
    elem.removeChild(elem.firstChild);
  }
  if (text != null) {
    elem.appendChild($doc.createTextNode(text));
  }
}
// Compiled Javascript (IE):
b.element.innerText = 'Click Me!';
```

DOMImpl.java

```
class DOMImpl {
  public native void setInnerText(Element elem, String text){
    // Remove all children first.
    while (elem.firstChild) {
      elem.removeChild(elem.firstChild);
    }
    // Add a new text node.
    if (text != null) {
      elem.appendChild($doc.createTextNode(text));
    }
  };
}
```

DOMImplIE6.java

```
class DOMImplIE6 {
  public native void setInnerText(Element elem,
  String text){
    elem.innerText = text;
  };
}
```

gwt.js 12:24 LF UTF-8 JavaScript master +13 1 update

ITS GOAL IS TO ENABLE PRODUCTIVE DEVELOPMENT OF HIGH-PERFORMANCE WEB APPLICATIONS WITHOUT THE DEVELOPER HAVING TO BE AN EXPERT IN BROWSER QUIRKS, XMLHTTPREQUEST, AND JAVASCRIPT.

http://www.gwtproject.org/overview.html

2011 DART

main.dart.js — ~/Documents/staticjs

main.dart

```dart
// Copyright (c) 2017, webcamp. All rights reserved. Use of this source code
// is governed by a BSD-style license that can be found in the LICENSE file.

import 'dart:html';

void main() {
  querySelector('#output').text = 'Your Dart app is running.';
}
```

main.dart.js

```
GRectElement|SVGSVGElement|SVGScriptElement|SVGSetElement|SVGStopElement|SVGStyleElement|SVGSwitchElement|SVGSymbolElement|SVGTSpanElement|SVGTextContentElement|SVGTextElement|SVGTextPathElement|SVGTextPositioningElement|SVGTitleElement|SVGUseElement|SVGViewElement;Element"},aj:{"^":"f;","%":"DOMWindow|Window;EventTarget"},b6:{"^":"r;k:length=","%":"HTMLFormElement"},as:{"^":"aj;",
h:function(a){var z=a.nodeValue
return z==null?this.v(a):z},
"%":"Document|HTMLDocument;Node"},ba:{"^":"r;k:length=","%":"HTMLSelectElement"}}],["","",,P,{"^":""}],["","",,P,{"^":""}],["","",,F,{"^":"",
a4:function(){document.querySelector("#output").textContent="Your Dart app is running."}},1]]
setupProgram(dart,0)
J.i=function(a){if(typeof a=="number"){if(Math.floor(a)==a)return J.ap.prototype
return J.ao.prototype}if(typeof a=="string")return J.t.prototype
if(a==null)return J.aq.prototype
if(typeof a=="boolean")return J.an.prototype
if(a.constructor==Array)return J.A.prototype
if(typeof a!="object"){if(typeof a=="function")return J.D.prototype
return a}if(a instanceof P.b)return a
return J.a1(a)}
J.aI=function(a){if(typeof a=="string")return J.t.prototype
if(a==null)return a
if(a.constructor==Array)return J.A.prototype
if(typeof a!="object"){if(typeof a=="function")return J.D.prototype
return a}if(a instanceof P.b)return a
return J.a1(a)}
```

Dart/build/web/main.dart.js 742:1 (1, 94) LF UTF-8 JavaScript master 1 update

DART IS A DYNAMICALLY TYPED LANGUAGE, AND PROUD OF IT

https://www.dartlang.org/articles/design-decisions/why-dart-types

WITH STRONG MODE ENABLED, DART IS A TYPE SAFE LANGUAGE. "CLASSIC DART" REFERS TO DART BEFORE SOUNDNESS WAS ADDED TO THE LANGUAGE.

https://www.dartlang.org/guides/language/sound-dart

2011 STJS

HelloWorld.java — ~/Documents/staticjs

HelloWorld.java

```java
public class HelloWorld {
  public static void main(String[] args) {
    window.onload = new Callback1<DOMEvent>() {
      @Override
      public void $invoke(DOMEvent ev) {
        Form form = window.document.forms.$get(0);
        Element button = form.elements.$get("say");
        final Input text = form.elements.$get("to");
        button.onclick = new Function1<DOMEvent,
        Boolean>() {
          @Override
          public Boolean $invoke(DOMEvent ev) {
            alert("Hello " + text.value);
            return true;
          }
        };
      }
    };
  }
}
```

helloworld.js

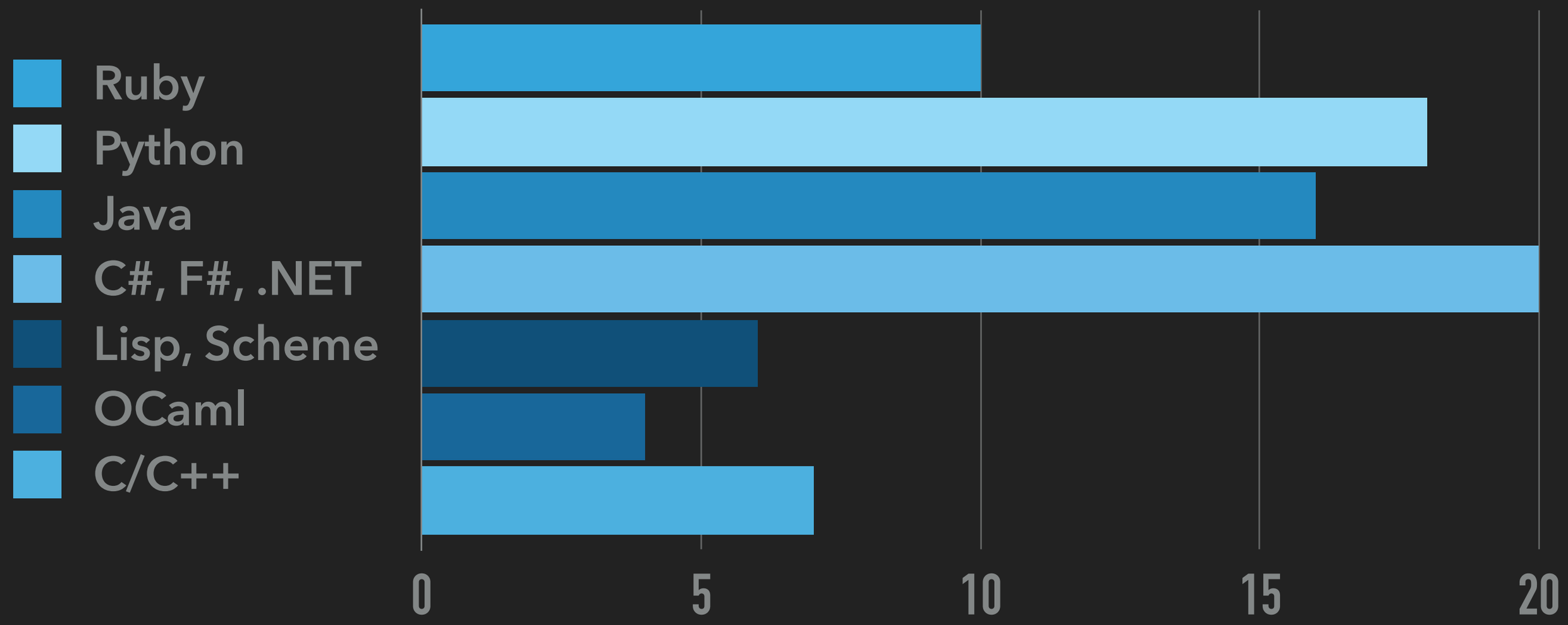
```js
var HelloWorld = function(){};
stjs.extend(HelloWorld, null, [], function(constructor, prototype){
    constructor.main = function(args) {
        window.onload = function(ev) {
            var form = window.document.forms[0];
            var button = form.elements["say"];
            var text = form.elements["to"];
            button.onclick = function(ev) {
                alert("Hello " + text.value);
                return true;
            };
        };
    };
}, {});

if (!stjs.mainCallDisabled) HelloWorld.main();
```

STJS/HelloWorld.java 1:1 LF UTF-8 Java master +20 1 update

ST-JS HELPS YOU MANAGE THE COMPLEXITY OF LARGE DYNAMIC SINGLE-PAGE WEB APPLICATIONS BY COMBINING THE STRONG, STATIC TYPING OF JAVA, WITH THE WEALTH OF EXISTING LIBRARIES OF JAVASCRIPT.

http://st-js.github.io/index.html

JS TYPE SAFETY | BRIEF HISTORY



https://github.com/jashkenas/coffeescript/wiki/list-of-languages-that-compile-to-js

2012 TYPESCRIPT

greeter.ts — ~/Documents/staticjs

greeter.ts

```typescript
class Student {
    fullName: string;
    constructor(public firstName, public middleInitial, public lastName) {
        this.fullName = firstName + " " + middleInitial + " " + lastName;
    }
}

interface Person {
    firstName: string;
    lastName: string;
}

function greeter(person : Person) {
    return "Hello, " + person.firstName + " " + person.lastName;
}

var user = new Student("Jane", "M.", "User");

document.body.innerHTML = greeter(user);

```

greeter.js

```javascript
var Student = (function () {
    function Student(firstName, middleInitial, lastName) {
        this.firstName = firstName;
        this.middleInitial = middleInitial;
        this.lastName = lastName;
        this.fullName = firstName + " " + middleInitial + " " + lastName;
    }
    return Student;
}());
function greeter(person) {
    return "Hello, " + person.firstName + " " + person.lastName;
}
var user = new Student("Jane", "M.", "User");
document.body.innerHTML = greeter(user);

```

TypeScript/greeter.ts 19:41 LF UTF-8 Plain Text master +20 1 update

TYPES ENABLE JAVASCRIPT DEVELOPERS TO USE HIGHLY-PRODUCTIVE DEVELOPMENT TOOLS AND PRACTICES LIKE STATIC CHECKING AND CODE REFACTORING WHEN DEVELOPING JAVASCRIPT APPLICATIONS.

https://www.typescriptlang.org/index.html

2014 FLOW

index.js — src — ~/Documents/staticjs

index.js

```
// @flow

function foo(x: ?number): number {
  if (x) {
    return x;
  }
  return 0;
}
```

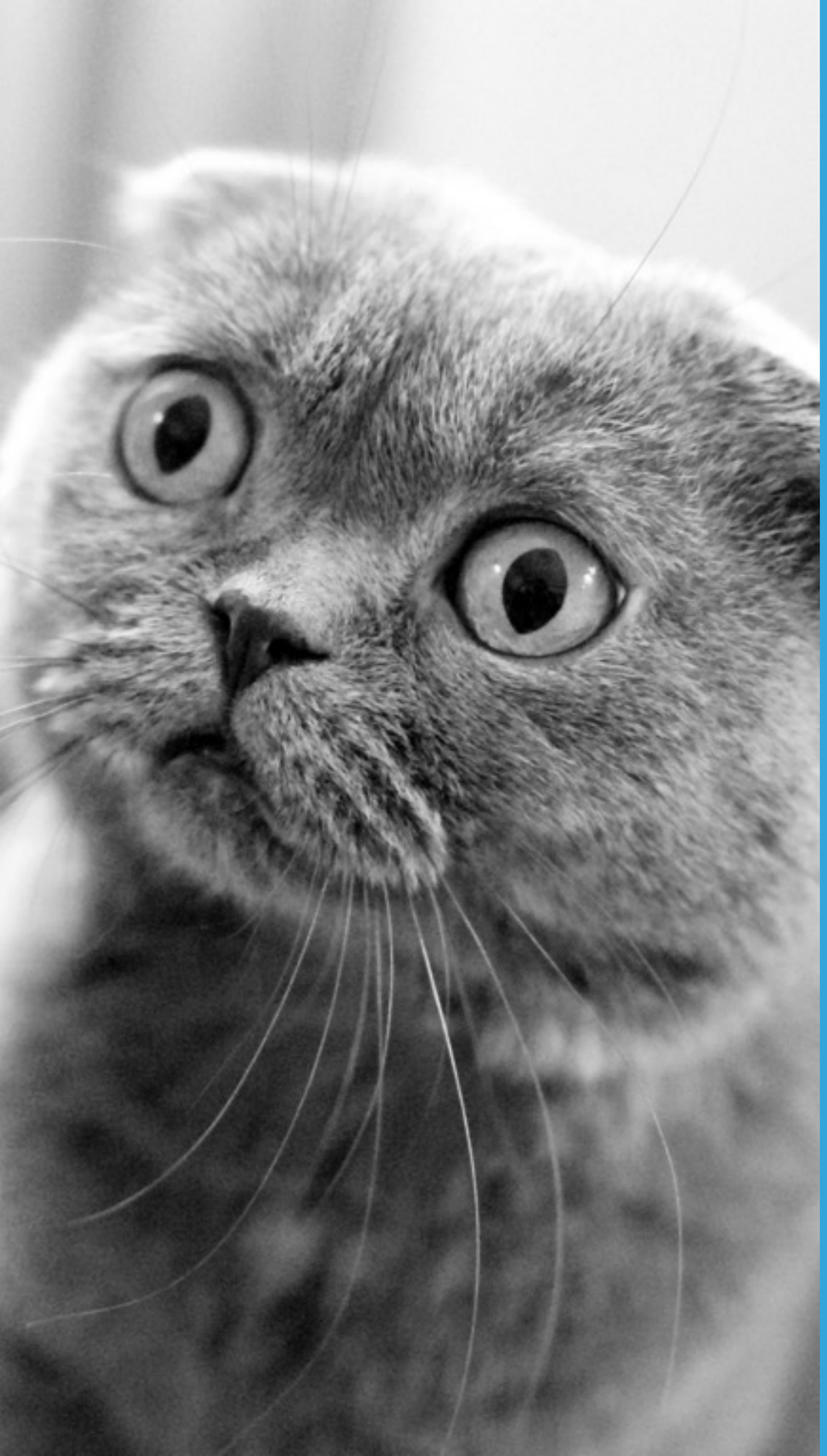
index.js

```
function foo(x) {
  if (x) {
    return x;
  }
  return 0;
}
```

Flow/src/index.js 7:11 LF UTF-8 JavaScript master +9 1 update

FLOW IS A STATIC TYPE CHECKER FOR YOUR JAVASCRIPT CODE. IT DOES A LOT OF WORK TO MAKE YOU MORE PRODUCTIVE. MAKING YOU CODE FASTER, SMARTER, MORE CONFIDENTLY, AND TO A BIGGER SCALE.

https://flow.org/en/docs/getting-started



WHY?

REVEALING TYPE-RELATED BUGS AT COMPILE TIME

Atom File Edit View Selection Find Packages Window Help Mon 1:06 AM webcamp

trick.js — js — ~/Documents/staticjs

trick.js

```
/*
* JavaScript coercion
*/
var objArr = {}+[];

var arrObj = []+{};

var arrNum = []+1;

var arrNum = []*1;

var arrNum1 = [2]*2;
```

trick.js

```
var arrNum2 = [2,1]+2;

var arrNum3 = [2,1]-2;

var arrs = [1]+[2];
```

>

js/trick.js 0 0 0 13:1 LF UTF-8 JavaScript master +19 1 update

REVEALING TYPE-RELATED BUGS AT COMPILE TIME

coersion.ts — ~/Documents/staticjs

trick.js | coersion.ts

```
1  /*
2  * JavaScript coercion
3  */
4  var objArr = {}+[];
5
```

Operator '+' cannot be applied to types '{}' and 'undefined[]'.

Severity	Provider	Description	Line
Error	Typescript	Operator '+' cannot be applied to types '{}' and 'undefined[]'.	4:14

js/coersion.ts 1 0 0 1:1 LF UTF-8 TypeScript 2.3.0-dev.20170319 No project master +5 1 update

REVEALING TYPE

MORE READABLE CODE

Atom File Edit View Selection Find Packages Window Help

readable.js — ~/Documents/staticjs

readable.js

```
1  function Product(){
2  }
3
4  Product.prototype.setCategory = function (category) {
5    this.category = category;
6  };
7
```

js/readable.js 0 0 0 7:1 LF UTF-8 JavaScript master +7 1 update

MORE READABLE CODE

readable.ts — ~/Documents/staticjs

readable.js | readable.ts

```
1  interface Category{
2    name: string;
3    id: string;
4  }
5
6  class Product{
7    private category:Category;
8    setCategory(category:Category){
9      this.category = category;
10   }
11 }
12
```

js/readable.ts 0 0 0 12:1 LF UTF-8 TypeScript 2.3.0-dev.20170319 No project master +12 1 update

MORE MAINTAINABLE CODE

maintainable.js

```
function Product(){
}

Product.prototype.setCategory = function (category) {
  this.category = category;
};
```

maintainable2.js

```
function searchProductsByCategory(products,
category) {
  return products.filter(product=>{
    return product.category.id == category.id;
  })
}
```

MORE MAINTAINABLE CODE

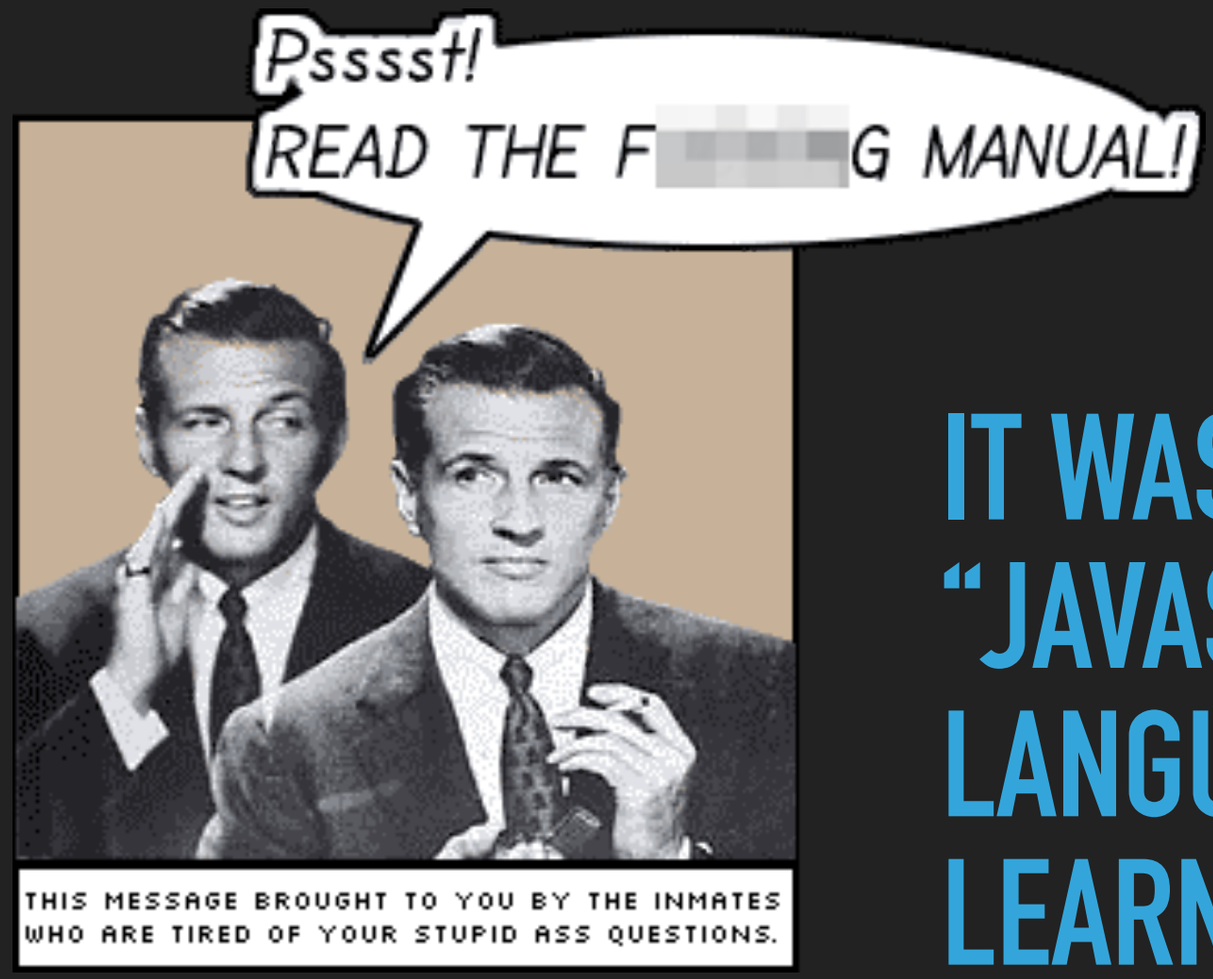
maintainable.ts

```
1   export interface Category{
2     name: string;
3     id: string;
4   }
5
6   export class Product{
7     category:Category;
8     setCategory(category:Category){
9       this.category = category;
10    }
11  }
12
```

maintainable2.ts

```
1   import {Product, Category} from './maintainable';
2
3   function
    searchProductsByCategory(products:Product[],
    category:Category){
4     return products.filter(product=>{
5       return product.category.id == category.id;
6     })
7   }
8
```

WHY NOT?



IT WAS ONCE SAID, "JAVASCRIPT IS THE ONLY LANGUAGE DEVELOPERS DON'T LEARN TO USE BEFORE USING IT."

David Walsh

Foreword for "You Don't Know JS: Types & Grammar" by Kyle Simpson

IN A DYNAMICALLY TYPED LANGUAGE, THERE'S NO NEED FOR TYPE CONSTRUCTORS. . . . INSTEAD, DEVELOPERS CAN USE DUCK TYPING, AND OPTIONALLY PERFORM RUNTIME TYPE CHECKS.

params.js — ~/Documents/staticjs

params.js

```javascript
const fn = (fn, {required = []}) => (params = {}) => {
  const missing = required.filter(param => !(param in params));

  if (missing.length) {
    throw new Error(`${ fn.name }() Missing required parameter(s):
    ${ missing.join(', ') }`);
  }

  return fn(params);
};

const createEmployee = fn(
  ({
    name = '',
    hireDate = Date.now(),
    title = 'Worker Drone'
  } = {}) => ({
    name, hireDate, title
  }),
  {
    required: ['name']
  }
);

console.log(createEmployee({ name: 'foo' })); // works
createEmployee(); // createEmployee() Missing required parameter(s): name
```

why_not/params.js 0 0 0 24:1 LF UTF-8 JavaScript master +27 3 updates

Eric Elliott
"You Might Not Need TypeScript (or Static Types)"

JS TYPE SAFETY | WHY NOT?

paramsTS.ts — ~/Documents/staticjs

paramsTS.ts

```
1  const createEmployee = (name:string,
   hireDate?:number, title?:string)=>{
2    return {name, hireDate, title};
3  };
4
5  let employee = createEmployee();
6
```

Supplied parameters do not match any signature of call target.

paramsFlow.js

```
1  //@flow
2  const createEmployee = (name:string,
   hireDate:?number, title:?string)=>{
3    return {name, hireDate, title}
4  };
5
6  let employee = createEmployee();
7
```

params.js

```
1  const fn = (fn, {required = []}) => (params = {})
   => {
2    const missing = required.filter(param =>
     !(param in params));
3
4    if (missing.length) {
                                    ame }() Missing
       required parameter(s):
6      ${ missing.join(', ') }`);
7    }
8
9    return fn(params);
10 };
11
12 const createEmployee = fn(
13   ({
14     name = '',
15     hireDate = Date.now(),
16     title = 'Worker Drone'
17   } = {}) => ({
18     name, hireDate, title
19   }),
20   {
```

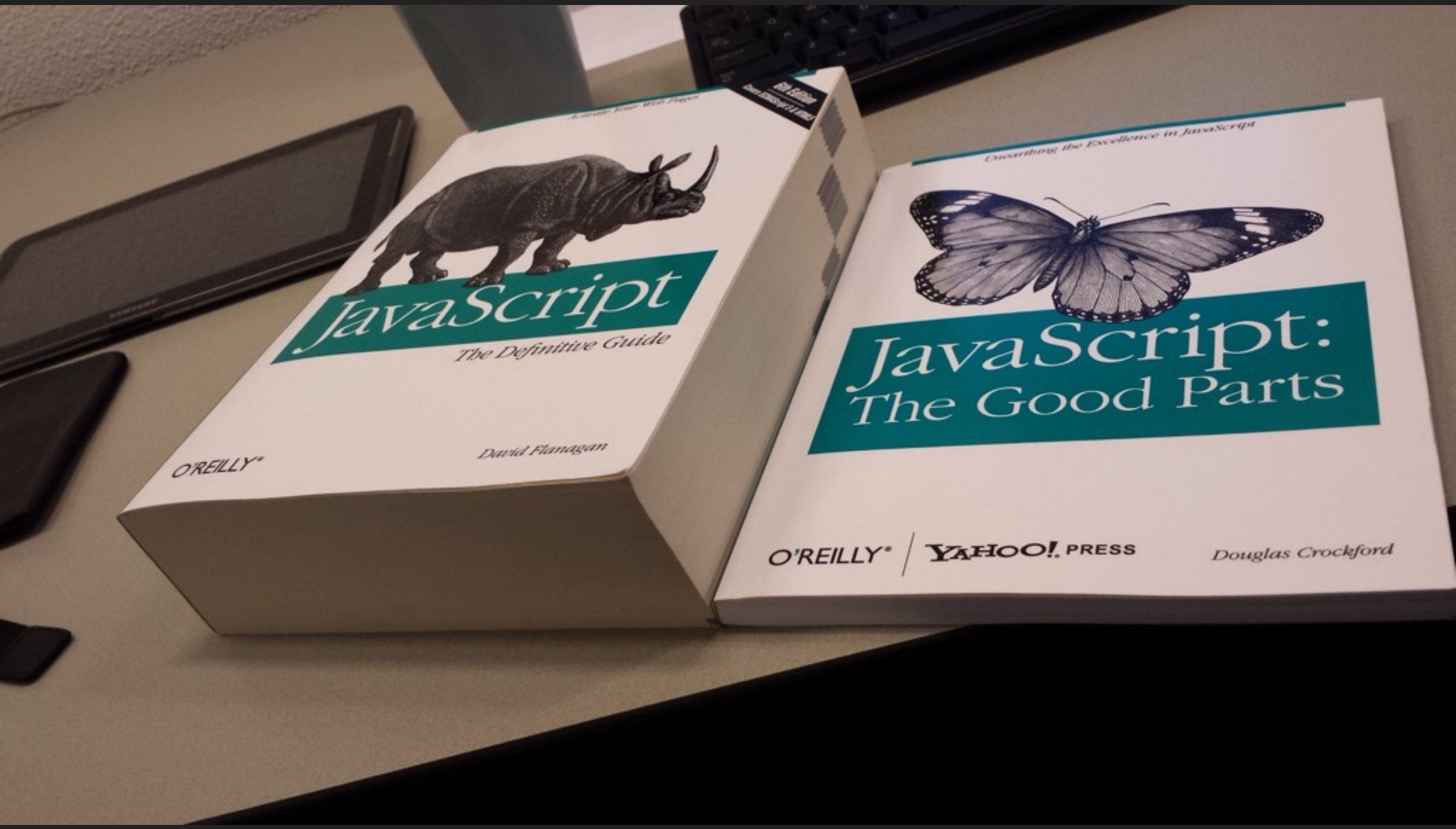
Severity	Provider	Description	Line
Error	Typescript	Supplied parameters do not match any signature of call target.	5:16

why_not/paramsTS.ts 5:19 LF UTF-8 TypeScript 2.3.0-dev.20170319 No project master +6 3 updates

I HAVE FOUND IN MY WORK THAT THE SORTS OF ERRORS THAT STRONG TYPE CHECKING FINDS ARE NOT THE ERRORS I WORRY ABOUT. ON THE OTHER HAND, I FIND LOOSE TYPING TO BE LIBERATING. I DON'T NEED TO FORM COMPLEX CLASS HIERARCHIES. AND I NEVER HAVE TO CAST OR WRESTLE WITH THE TYPE SYSTEM TO GET THE BEHAVIOR THAT I WANT.

Douglas Crockford
"JavaScript: The Good Parts"

JS TYPE SAFETY | WHY NOT?



JS TYPE SAFETY | WHY NOT?

cast.js — ~/Documents/staticjs

cast.js

```
1   var answer = 42.5 | 0;
```

1 0 why_not/cast.js* LF UTF-8 Babel

Cast.java — ~/Documents/staticjs

Cast.java

```
1   int answer = (int) 42.5;
2
```

1 0 why_not/Cast.java LF UTF-8 Java master 8930 files 10 updates

JS TYPE SAFETY | WHY NOT?

crockford.js — ~/Documents/staticjs

crockford.js

```
1   Number.method('integer', function (  ) {
2       return Math[this < 0 ? 'ceil' :
    •   'floor'](this);
3   });
4
```

why_not/crockford.js 0 0 0 3:4 LF UTF-8 Babel master 8931 files 10 updates

WHAT'S NEXT?