ActionScript 2
&
The Future Of JavaScript That Never Was

@WebReflection

ActionScript 2
&
The Future Of JavaScript That Never Was

@WebReflection

but the topic is @cramforce fault!

Who am I

Who am I

. **year 2000,** WebMaster using Dreamweaver, ditching FrontPage, feeling cool because I knew how to drag and drop animated GIFs and Flash 4 dumb frames based swf

. **year 2003,** dragging GIFs wasn't exciting, I've opted for an Advanced Macromedia Flash MX course (ES3 AS1) and an Advanced Java Development at Sun Microsystem

Who am I

. **year 2000,** WebMaster using Dreamweaver, ditching FrontPage, feeling cool because I knew how to drag and drop animated GIFs and Flash 4 dumb frames based swf

. **year 2003,** dragging GIFs wasn't exciting, I've opted for an Advanced Macromedia Flash MX course (ES3 AS1) and an Advanced Java Development at Sun Microsystem

. **year 2004,** Certified Macromedia Flash MX 2004 Developer (ES3+4 via AS1+2)

. **year 2005,** Zend Certified Engineer (PHP4, 5, 5.1, 5.3, with SPL)

Who am I

. **year 2000,** WebMaster using Dreamweaver, ditching FrontPage, feeling cool because I knew how to drag and drop animated GIFs and Flash 4 dumb frames based swf

. **year 2003,** dragging GIFs wasn't exciting, I've opted for an Advanced Macromedia Flash MX course (ES3 AS1) and an Advanced Java Development at Sun Microsystem

. **year 2004,** Certified Macromedia Flash MX 2004 Developer (ES3+4 via AS1+2)

. **year 2005,** Zend Certified Engineer (PHP4, 5, 5.1, 5.3, with SPL)

. **last 5 years,** Mobile Web Development at NOKIA, Facebook, and currently at Twitter

Who am I

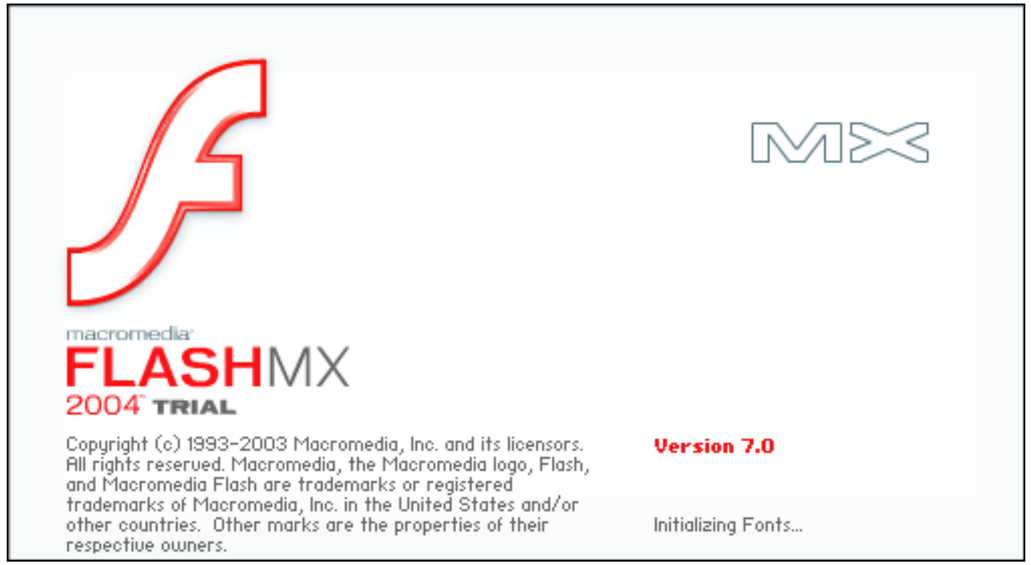
. **year 2000,** WebMaster using Dreamweaver, ditching FrontPage, feeling cool because I knew how to drag and drop animated GIFs and Flash 4 dumb frames based swf
. **year 2003,** dragging GIFs wasn't exciting, I've opted for an Advanced Macromedia Flash MX course (ES3 AS1) and an Advanced Java Development at Sun Microsystem
. **year 2004,** Certified Macromedia Flash MX 2004 Developer (ES3+4 via AS1+2)
. **year 2005,** Zend Certified Engineer (PHP4, 5, 5.1, 5.3, with SPL)

. **last 5 years,** Mobile Web Development at NOKIA, Facebook, and currently at Twitter

No Flash Here!

Who is ActionScript 2

Who is ActionScript 2



Who is ActionScript 2

Who is ActionScript 2

ECMAScript 4th Edition

Who is ActionScript 2

ECMAScript 4th Edition
AS3 - haxe

Who is ActionScript 2

. Classes, including Matrix, Point, Rectangle, color transformations, MOAR
. A module system (via import)
. Network Streams
. Binary Objects via AMF
. SharedObject storage between different SWFs + different domain + Sync option
. Locale and i18n features

Who is ActionScript 2

. Classes, including Matrix, Point, Rectangle, color transformations, MOAR
. A module system (via import)
. Network Streams
. Binary Objects via AMF
. SharedObject storage between different SWFs + different domain + Sync option
. Locale and i18n features

the **arguments** Object has bloody same Array methods

meanwhile, in 2014 JavaScript

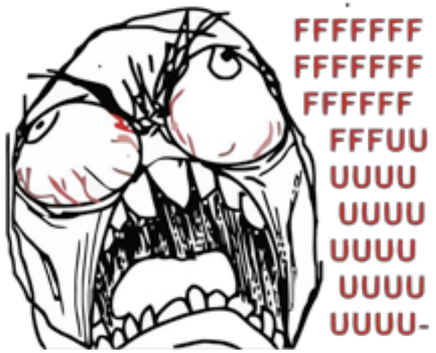
```
function seriously() {
  var args = Array.prototype.slice.call(arguments);
  // do some stuff ... then ...




}
```

meanwhile, in 2014 JavaScript

```
function seriously() {
  var args = Array.prototype.slice.call(arguments);
  // do some stuff ... then ...
  return
```




```
;
}
```

Who is ActionScript 2

. System.capabilities (UA never exposed anything useful)
. ExternalInterface (communicate between different languages)
. onEnterFrame (requestAnimationFrame with configurable FPS)

Who is ActionScript 2

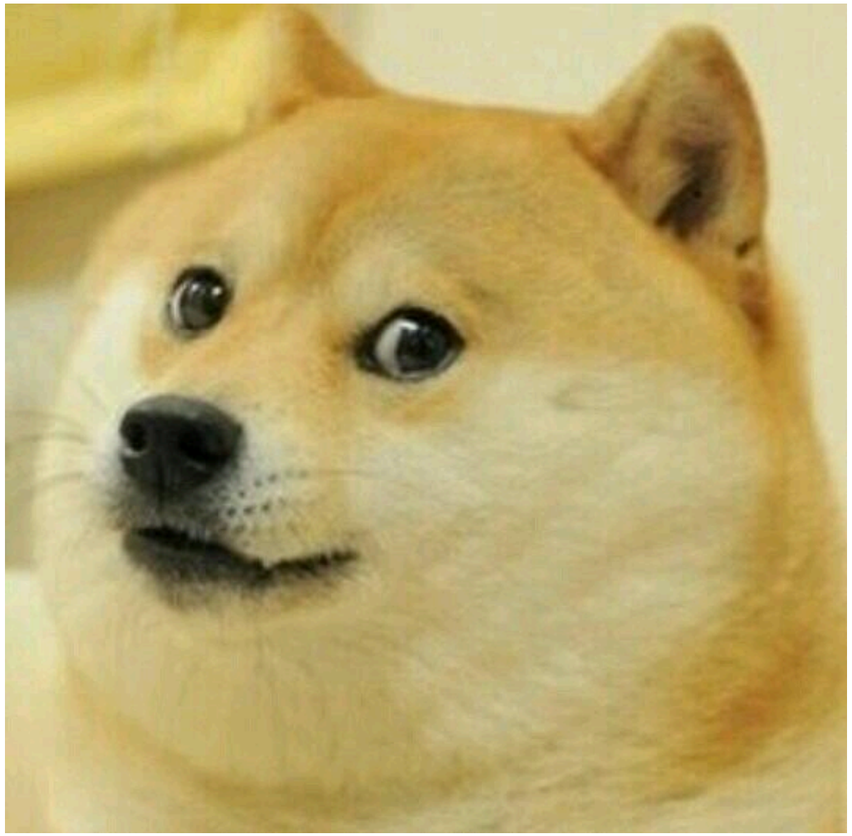
. System.capabilities (UA never exposed anything useful)
. ExternalInterface (communicate between different languages)
. onEnterFrame (requestAnimationFrame with configurable FPS)
. much more ... and this was **YEAR 2003**

Who is ActionScript 2

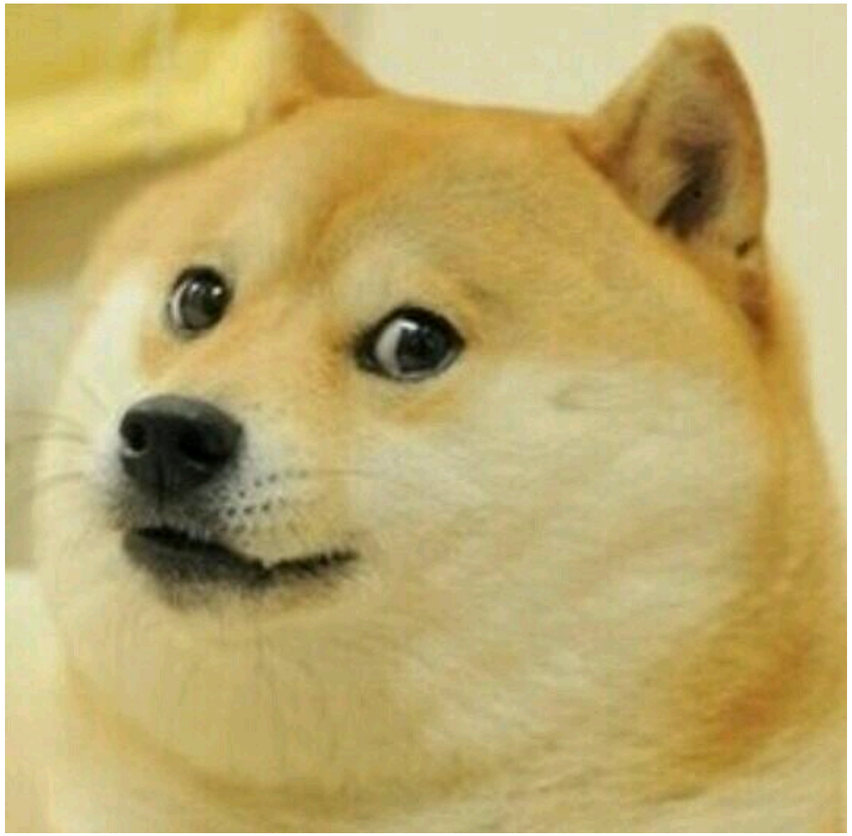
. System.capabilities (UA never exposed anything useful)
. ExternalInterface (communicate between different languages)
. onEnterFrame (requestAnimationFrame with configurable FPS)
. much more ... and this was **YEAR 2003**

... plus Optional Types !!!

Who is ActionScript 2



Who is ActionScript 2



:Wow

Who is ActionScript 2

```
class Greeter {
    var greeting:String;
    function Greeter(message:String) {
        this.greeting = message;
    }
    function greet(Void):String {
        return "Hello, " + this.greeting;
    }
}
var greeter:Greeter = new Greeter("world");
```

Who am I

Who am I

... at the very beginning ...

Who am I

Who am I

... “few” beers later ...

Who am I

BEHIND THE DESIGN
WEB 2.0 REFLECTION

Wednesday, May 23, 2007

JavaStrict Act II - AS2.0 like Strict Type with JavaScript

I've just updated my JavaStrict public, static, function.

It has been successfully tested with different cases and now it behaviour is like ActionScript 2.0

With this function, and dedicated Strict shortcut one, You can:

link

Who am I

```
charToCode = JavaStrict.returnValue(
 Number,
 JavaStrict.call(
  function(str){
   return str.charCodeAt(0);
  },
  String
 )
);
```

link

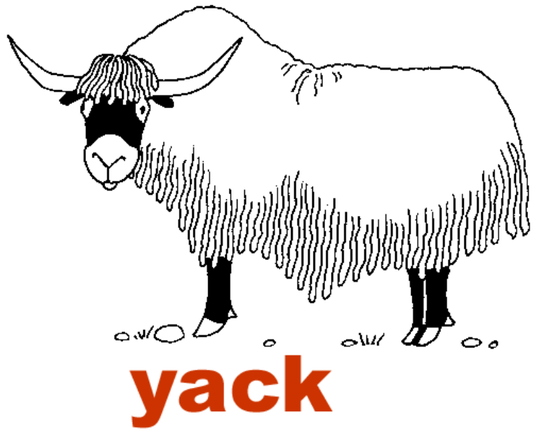
Who am I

```
charToCode = JavaStrict.returnValue(
 Number,
 JavaStrict.call(
  function(str){
   return str.charCodeAt(0);
  },
  String
 )
);
```

link

Who am I

```
charToCode = JavaStrict.returnValue(
 Number,
 JavaStrict.call(
  function(str){
   return str.charCodeAt(0);
  },
  String
 )
);
```



link

2009, dojo.lang.typed

2009, dojo.lang.typed

```
dojo.require("dojox.lang.typed");

TypedClass = dojox.lang.typed(
    dojo.declare("TypedClass", null, {
        constructor: function(makeDefaults){
            if(makeDefaults){
                this.aString = "start";
            }
        },
        add: function(a, b){
            return a + b;
        }
    }));
TypedClass.properties = {
    aString:String, // this is the same aString:{type:"string"}
};
TypedClass.methods = {
    add: {
        parameters:[
            Number,
            Number
        ],
        returns: {type:"number", description: "The sum"}
    }
}
```

Who am I

BEHIND THE DESIGN
WEB 2.0 REFLECTION

Friday, August 20, 2010

Object.defineProperty ... but Strict!

In my precedent post entitled A Pascal record via JavaScript I have showed a basic function able to emulate type hints behavior via JavaScript.
Even if that was a proof of concept, I consider other languages simulation of unsupported features an error, first of all because the behavior will rarely be exactly the expected one, secondly because our curent programming language may already have something similar to better learn and use.

A new ES5(direction)

As soon as I have written the Pascal example, I have realized that the good "old" Object.defineProperty, implemented in all major browsers (IE < 9 sucks, you know that ...), has basically the same meaning: define object accessors.

Object.defineStrictProperties()

```
propertyName: {
    // ES5
    writable: true/false
    enumerable: true/false
    configurable: true/false
    get: Function
    set: Function
    value: any

    // my complementary, non obtrusive, info
    type: string|Function(as instanceof)|Object(as isPrototypeOf)
    returns: type|[type1, type2, typeN]
    arguments: [type]|[[type], [type]]
}
// overloads included + backward compatible
```

link

Object.defineStrictProperties()

```
get: {
    type: "function",
    returns: "string",
    arguments: [["string"], ["string", "string", "string"]],
    value: function (uri, user, pass) {
        this.open("get", uri, false, pass && user, pass);
        this.send(null);
        return this.responseText;
    }
}
```

link

Object.defineStrictProperties()

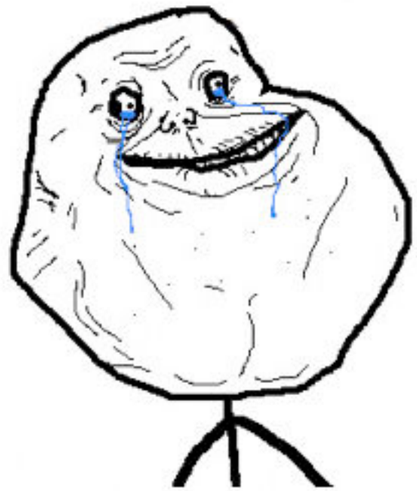
```
propertyName: {
    // ES5
    writable: true/false
    enumerable: true/false
    configurable: true/false
    get: Function
    set: Function
    value: any

    // my complementary, non obtrusive, info
    type: string|Function(as instanceof)|Object(as isPrototypeOf)
    returns: type|[type1, type2, typeN]
    arguments: [type]|[[type], [type]]
}
// overloads included + backward compatible
// it won't compromise performance in production
// Object.defineStrictProperties = Object.defineProperties; // DONE!
```

link

Object.defineStrictProperties()

Object.defineStrictProperties()



October 2012

October 2012

Home > News > Microsoft > Developing Projects

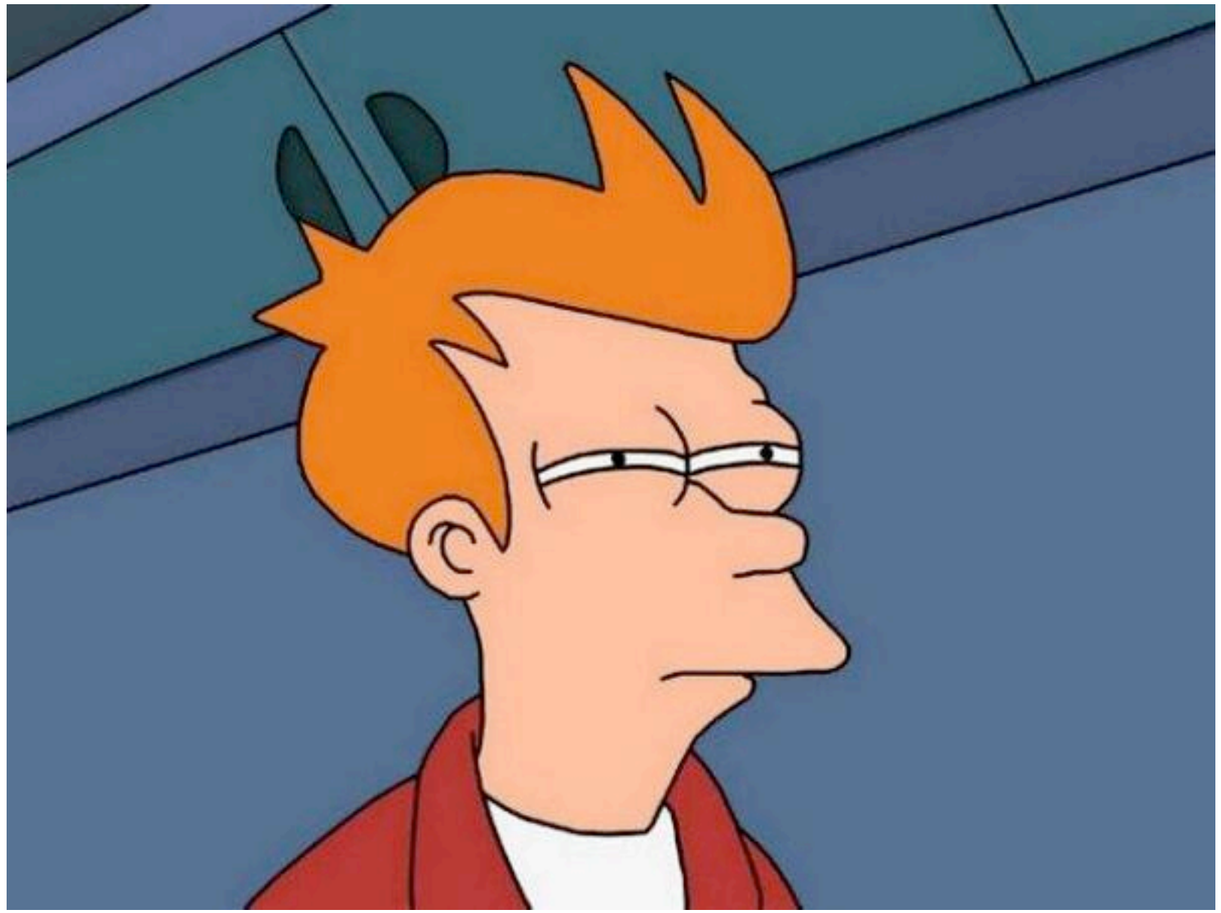
October 2nd, 2012, 07:43 GMT · By Bogdan Popa

Microsoft Launches TypeScript Programming Language Based on JavaScript

Microsoft has just released a new programming language called TypeScript and designed to ease the process of creating client-side JavaScript for Windows, Internet Explorer and other browsers or operating systems.

October 2012

October 2012



2012 TypeScript

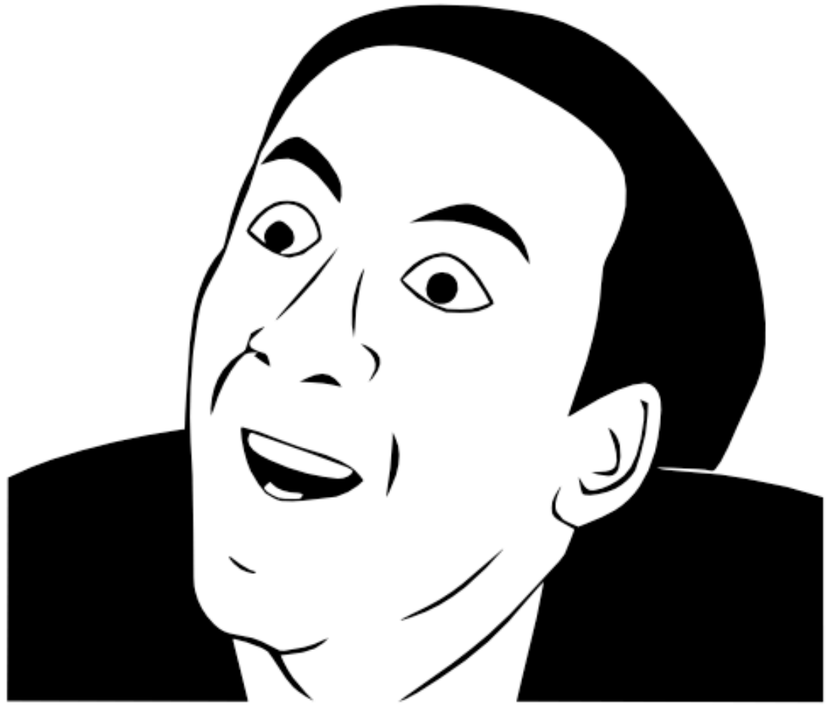
```
class Greeter {
    greeting:String;
    constructor(message:String) {
        this.greeting = message;
    }
    greet(Void):String {
        return "Hello, " + this.greeting;
    }
}
var greeter:Greeter = new Greeter("world");
```

2003 ActionScript 2

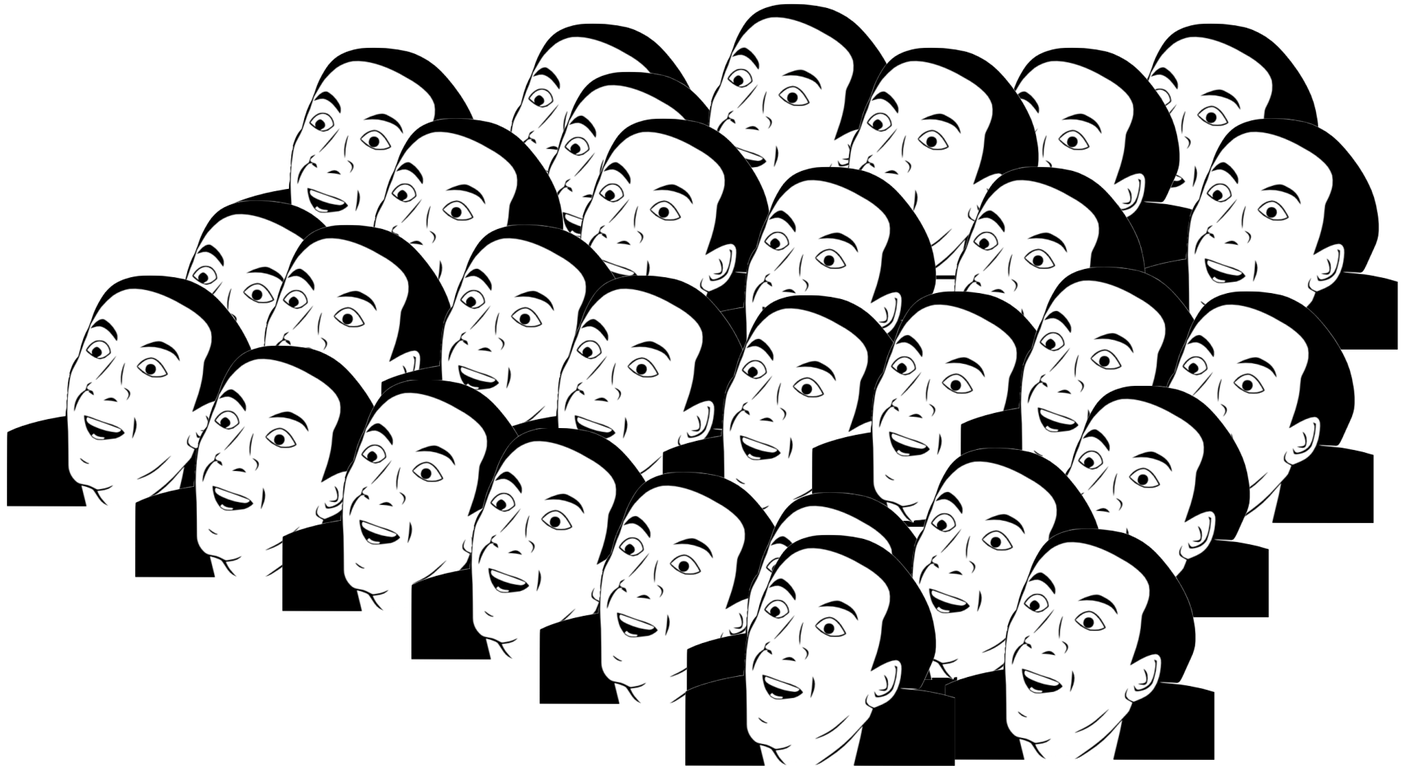
```
class Greeter {
    var greeting:String;
    function Greeter(message:String) {
        this.greeting = message;
    }
    function greet(Void):String {
        return "Hello, " + this.greeting;
    }
}
var greeter:Greeter = new Greeter("world");
```

9 Years In Between!

Who am I

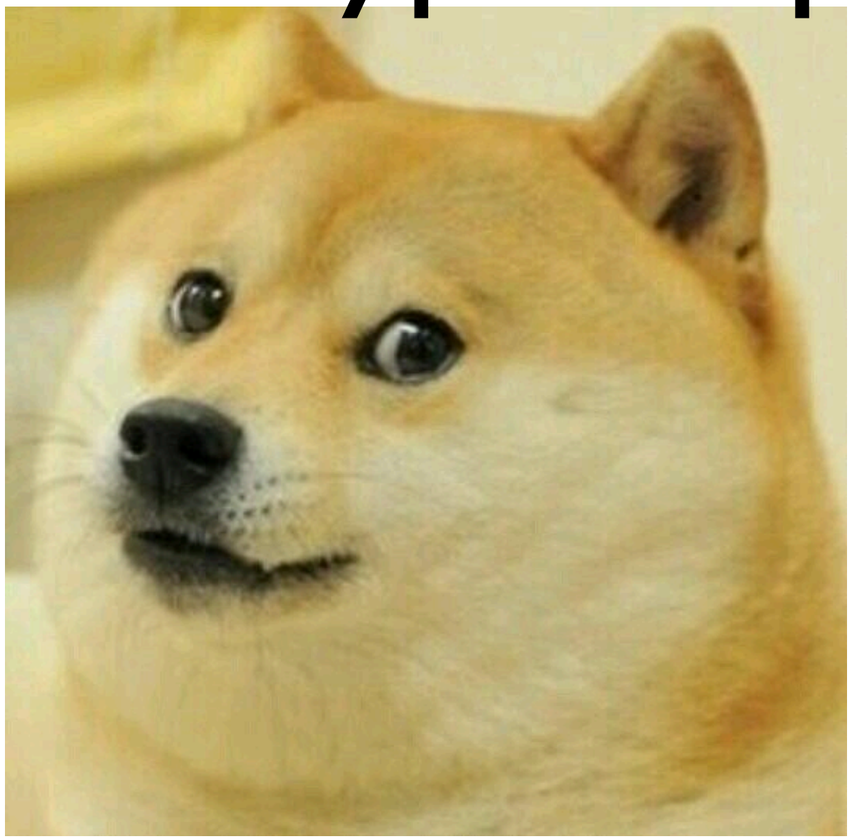


meanwhile, in Adobe



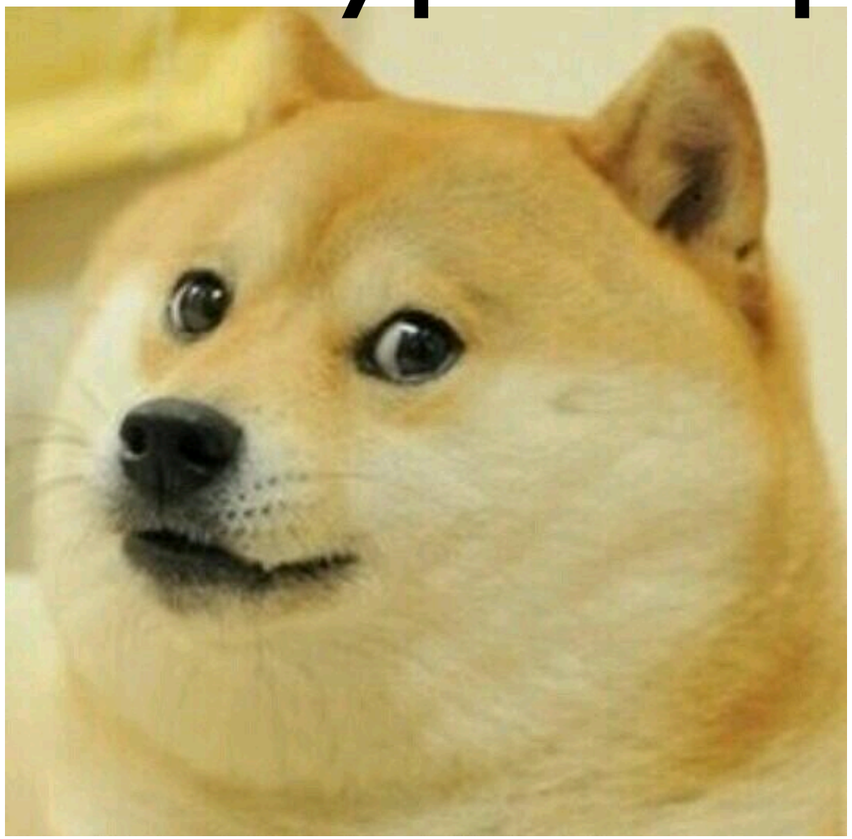
meanwhile, in Adobe

2012 TypeScript

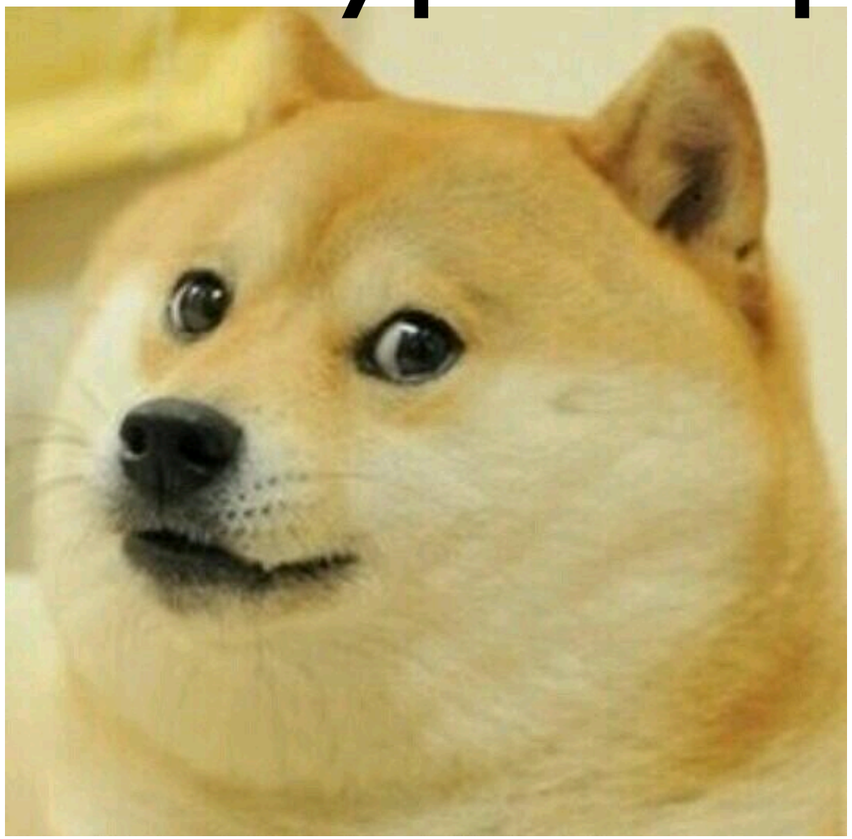


2012 TypeScript

Such Progress!

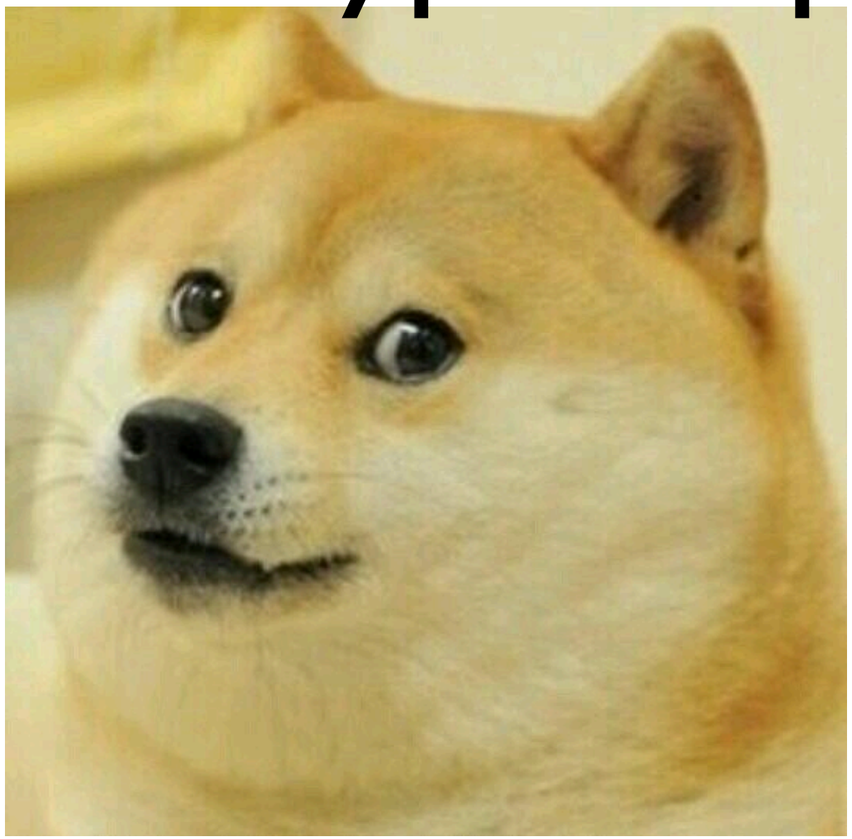


2012 TypeScript



Much Revolution!

2012 TypeScript



:Wow

meanwhile, in JS.next

. **transpile All The Things!** thanks to transpilers similar to TypeScript, Traceur, CoffeScript or others. Still nobody explained to me how are we supposed to serve the new version or the transpiled one to old and new browsers once ES6 will be supported natively by these one ... right, UA sniffing on the server does that

. **classes landed in ES6 specs already** + AFAIK TypeScript is already slightly different

. **generators, iterators, destructuring, arrow function,** and MOAR coming

. **A module system** that will be different from all others

. **typed objects / StructType will replace typed notation**

meanwhile, in JS.next

. **transpile All The Things!** thanks to transpilers similar to TypeScript, Traceur, CoffeScript or others. Still nobody explained to me how are we supposed to serve the new version or the transpiled one to old and new browsers once ES6 will be supported natively by these one ... right, UA sniffing on the server does that

. **classes landed in ES6 specs already** + AFAIK TypeScript is already slightly different

. **generators, iterators, destructuring, arrow function,** and MOAR coming

. **A module system** that will be different from all others

. **typed objects / StructType will replace typed notation**

. **arguments in Proxy behaves like Array ...** AFAIK no changes in regular arguments

Thank You!

all images have been randomly taken here and there and all credits go to people that put those images online including Brian face at mobile web nerd talk

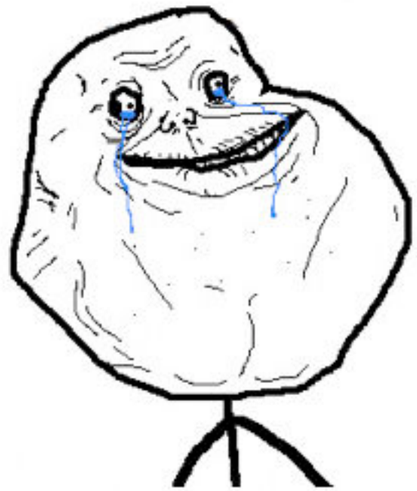
@WebReflection

Thank You!

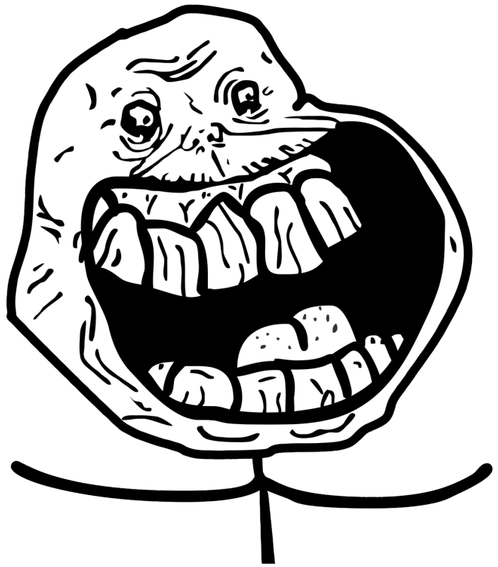
all images have been randomly taken here and there and all credits go to people that put those images online including Brian face at mobile web nerd talk

@WebReflection

Object.defineStrictProperties()



Object.create(proto, strictDescriptors)



Object.create(proto, strictDescriptors)

a concrete Object.create/defineProperty/ies drop-in/off script behind 60+ tests
with fully covered features and Open Source in github

https://github.com/WebReflection/define-strict-properties

@WebReflection

Thank You Again!

... and see you in 9 years from now ...

@WebReflection