



Thomas Watson

@wa7son

github.com/watson



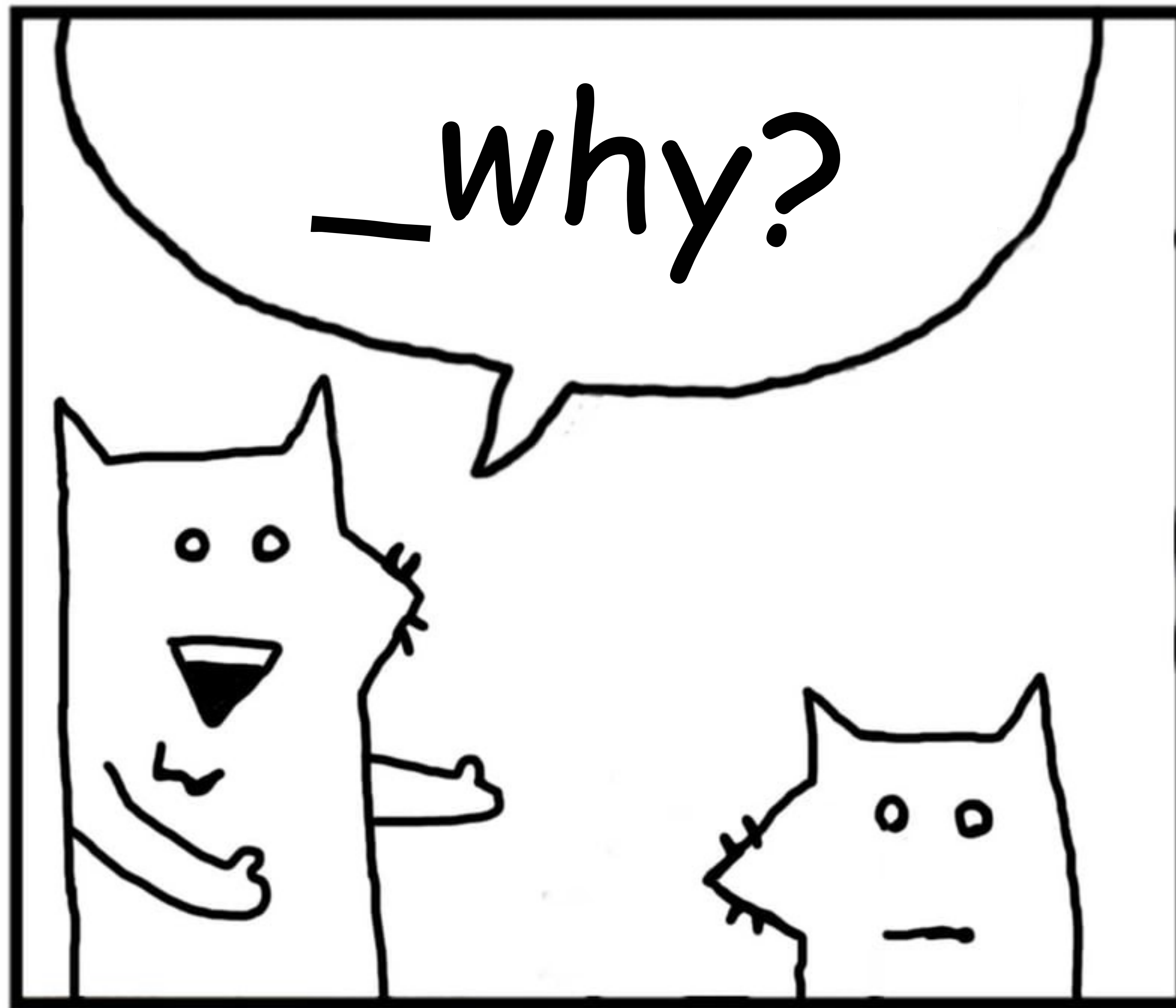
github.com/watson/talks



These go to eleven!



How to write code
that writes code



A black and white photograph of a person with long, dark, wavy hair, wearing a leopard-print top, singing passionately into a vintage-style microphone. The person's mouth is wide open, and their eyes are looking upwards. The background is dark and out of focus, with some light flares visible. The word "performance" is overlaid in a large, bold, pink sans-serif font across the center of the image.

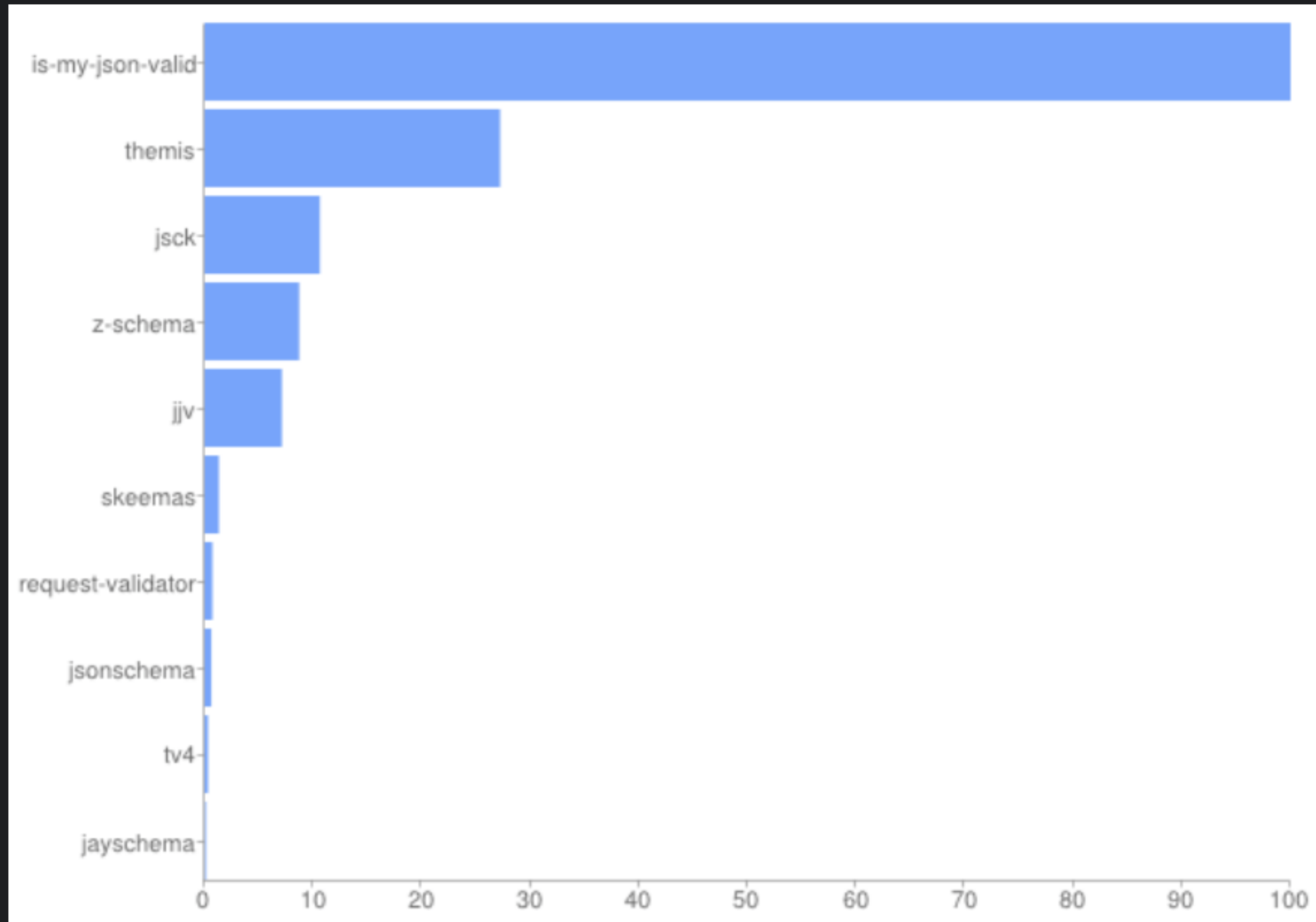
performance

github.com /
mafintosh /
is-my-json-valid

```
var validator = require('is-my-json-valid')

var validate = validator({
  required: true,
  type: 'object',
  properties: {
    hello: {
      required: true,
      type: 'string'
    }
  }
})

console.log('should be valid', validate({hello: 'world'}))
console.log('should not be valid', validate({}))
```

$f(\text{seed}, \text{data})$

$$f_2 = f_1(\textit{seed})$$
$$f_2(\textit{data})$$

An oversimplified example

```
function doMath (a, b) {  
  // ...do stuff with a...  
  return a + b  
}
```


An oversimplified example

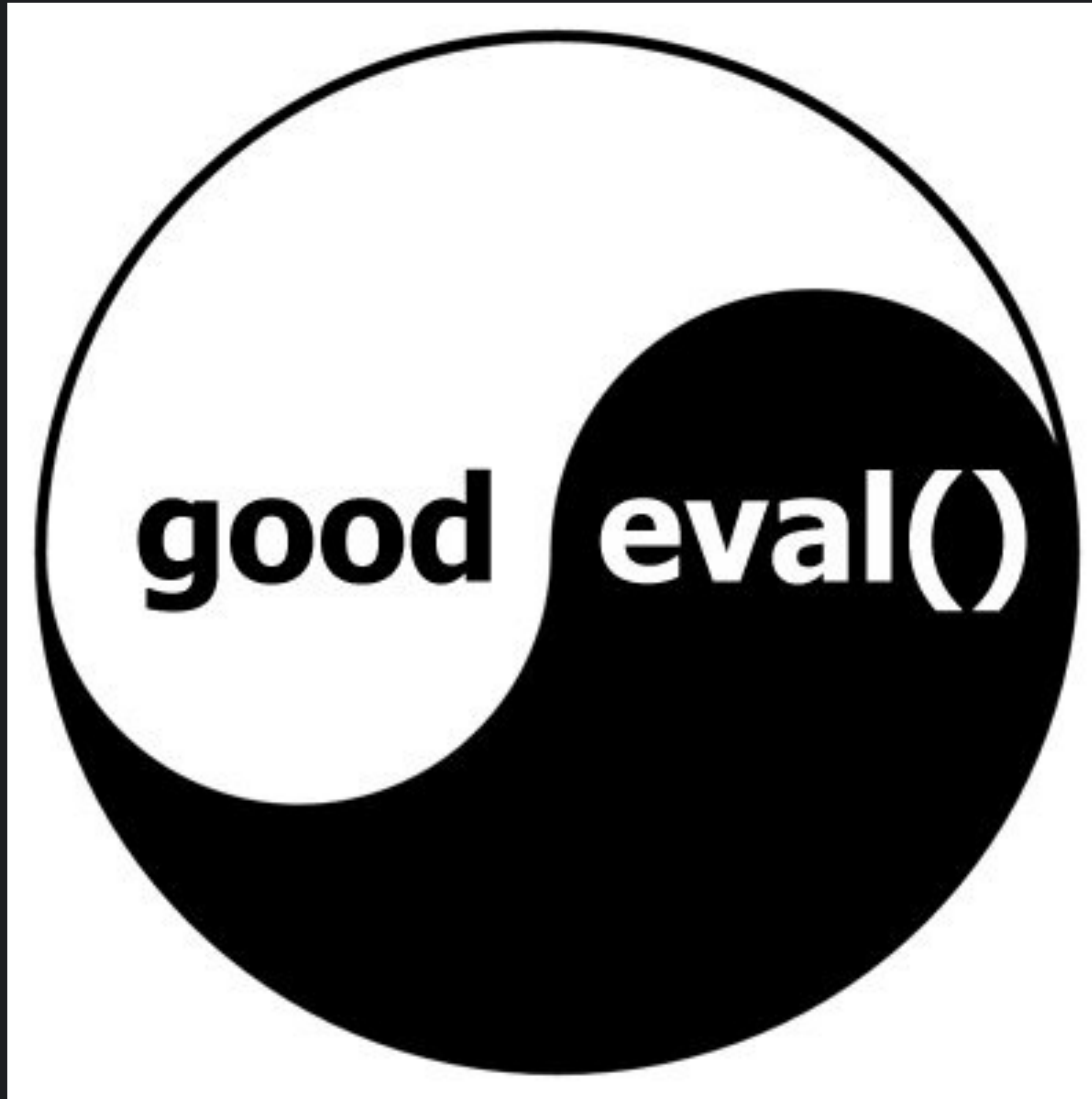
```
function doMath (a, b) {  
  // ...do stuff with a...  
  return a + b  
}  
  
function prepareMath (a) {  
  // ...do stuff with a...  
  return function doMath (b) {  
    return a + b  
  }  
}
```

An oversimplified example

```
function doMath (a, b) {  
  // ...do stuff with a...  
  return a + b  
}
```

```
function prepareMath (a) {  
  // ...do stuff with a...  
  return function doMath (b) {  
    return a + b  
  }  
}
```

```
function prepareMath (a) {  
  // ...do stuff with a...  
  var src = 'return ' + a + ' + b'  
  return new Function ('b', src)  
}
```


```
var admin = false
```

```
eval('var admin = true')
```

```
if (admin) console.log('user is admin')  
else console.log('user is NOT admin')
```

```
var admin = false
```

```
eval('var admin = true')
```

```
if (admin) console.log('user is admin')  
else console.log('user is NOT admin')
```

```
$ node example.js  
user is admin
```



```
var admin = false
```

```
var fn = new Function('var admin = true')  
fn()
```

```
if (admin) console.log('user is admin')  
else console.log('user is NOT admin')
```

```
var admin = false
```

```
var fn = new Function('var admin = true')  
fn()
```

```
if (admin) console.log('user is admin')  
else console.log('user is NOT admin')
```

```
$ node example.js  
user is NOT admin
```



```
var fn = new Function('var admin = true')  
  
console.log(fn.toString())
```

```
var fn = new Function('var admin = true')  
  
console.log(fn.toString())
```

```
$ node example.js  
function anonymous() {  
var admin = true  
}
```

```
var admin = false
```

```
var fn = new Function('admin = true')  
fn()
```

```
if (admin) console.log('user is admin')  
else console.log('user is NOT admin')
```



```
var admin = false

var fn = new Function('admin = true')
fn()

if (admin) console.log('user is admin')
else console.log('user is NOT admin')
```

```
$ node example.js
user is NOT admin
```

```
global.admin = false
```

```
var fn = new Function('admin = true')  
fn()
```

```
if (global.admin) console.log('user is admin')  
else console.log('user is NOT admin')
```

```
global.admin = false
```

```
var fn = new Function('admin = true')  
fn()
```

```
if (global.admin) console.log('user is admin')  
else console.log('user is NOT admin')
```

```
$ node example.js  
user is admin
```


Thomas, at least *mention* prototype pollution,
so people do not think you have lost your marbles

```
global.admin = false
```

```
var fn = new Function('admin = true')  
fn()
```

```
if (global.admin) console.log('user is admin')  
else console.log('user is NOT admin')
```

```
$ node example.js  
user is admin
```

```
var src = '\n"Hello World"\n'  
  
var fn = new Function('return (' + src + ')')  
  
console.log(fn())
```

```
var src = '\n"Hello World"\n'  
  
var fn = new Function('return (' + src + ')')  
  
console.log(fn())
```

```
$ node example.js  
Hello World
```



```
var src = 'n + 1'
```

```
var fn = new Function('n', 'return (' + src + ')')
```

```
console.log(fn(1))
```

```
var src = 'n + 1'
```

```
var fn = new Function('n', 'return (' + src + ')')
```

```
console.log(fn(1))
```

```
$ node example.js  
2
```

```
var n = 2  
var src = '2 * ' + n  
  
var fn = new Function('return (' + src + ')')  
  
console.log(fn())
```



```
var n = 2  
var src = '2 * ' + n  
  
var fn = new Function('return (' + src + ')')  
  
console.log(fn())
```

```
$ node example.js  
4
```

```
var n = 'console.log("owned") '  
var src = '2 * ' + n  
  
var fn = new Function('return (' + src + ')')  
  
console.log(fn())
```

```
var n = 'console.log("owned") '  
var src = '2 * ' + n  
  
var fn = new Function('return (' + src + ')')  
  
console.log(fn())
```

```
$ node example.js  
owned  
NaN
```

```
var n = 'console.log("owned") '  
var src = util.format('2 * %j ', n)  
  
var fn = new Function('return (' + src + ')')  
  
console.log(fn())
```



```
var n = 'console.log("owned") '  
var src = util.format('2 * %j ', n)  
  
var fn = new Function('return (' + src + ')')  
  
console.log(fn())
```

```
$ node example.js  
NaN
```




use cases

Regular Expressions

```
var r = new RegExp( 'foo(bar)?' )
```

```
r.test( 'foo' )
```

Templates

```
<div class="entry">  
  <h1>{{title}}</h1>  
  <div class="body">  
    {{body}}  
  </div>  
</div>
```


JSON Schema

```
{  
  type: 'object',  
  properties: {  
    aProperty: {  
      type: 'string'  
    }  
  }  
}
```

MongoDB query language

```
{  
  foo: {  
    $gt: 42,  
    $lt: 100  
  }  
}
```

Let's write a query
language interpreter

Our language

```
{  
  $eq: value,  
  $gt: value,  
  $lt: value,  
  $not: {...} // negate this expression  
}
```



coding
time

github.com /
mafintosh /
generate-object-property

```
var gen = require('generate-object-property')  
console.log(gen('a', 'b')) // => a.b  
console.log(gen('a', 'foo-bar')) // => a["foo-bar"]
```

github.com /
mafintosh /
generate-function

```
var genfun = require('generate-function')

var addNumber = function(val) {
  var fn = genfun()
    ('function add(n) {'
    ('return n + %d', val) // supports format strings
    ('}')

  return fn.toFunction() // will compile the function
}

var add2 = addNumber(2)
console.log('1+2=', add2(1))
console.log(add2.toString()) // logs generated function
```


A scenic view of a harbor at night, likely in Copenhagen, Denmark. The water is calm, reflecting the warm lights from the buildings and the cool blue tones of the twilight sky. Several boats are docked along the quay, including a large dark wooden sailing ship in the foreground. The buildings are multi-story, with many windows glowing with light. Outdoor seating areas with umbrellas are visible along the waterfront.

Thank you

@wa7son

github.com/watson