A Book Nerd's Guide to JavaScript

@angustweets

an·gus
@angustweets

JavaScript is to Java as the Eiffel Tower is to the Montparnasse Tower.
#parisienJavaScriptHumour #not @dotjs

Practical, Useful, Mind Numbingly Boring

Nonsensical Work of Genius

What can Literature teach us about Programming?

Great literature is the creation of
open minds

As programmers our focus can be
very narrow

We’re encouraged to follow
only one style…

```
// bad
var name = "Bob Parr";
```

Because **this** is so easy to get wrong, limit its use to those places where it is required:

- in constructors
- in methods of objects
- to create bound functions (including in the creation of closures)

Strict equality checks (`===`) must be used in favor of abstract equality checks (`==`).

```
// bad
var items = new Array();
```

Avoid `eval` should never be used.

Furthermore, the use of `setInterval` should be avoided

```
// Bad
var foo = true;
var bar = false;
```

Actual excerpts from:

Google Style Guide

AirBnb Style Guide

JavaScript Garden

```
// bad
var name = "Bob Parr";
```

Strict equality checks (`===`) must be used in favor of abstract equality checks (`==`)

Because **this** is so easy to get wrong, limit its use to those places where it is required:

- in constructors
- in methods of objects (including in the creation of closures)

```
// bad
var items = new Array();
```

`eval` should never be used.

Furthermore, the use of `setInterval` should be avoided

```
// Bad
var foo = true;
var bar = false;
```

Null Considered Harmful

If considered harmful

Closures considered harmful

"New" Statement Considered Harmful

Arrays considered somewhat harmful

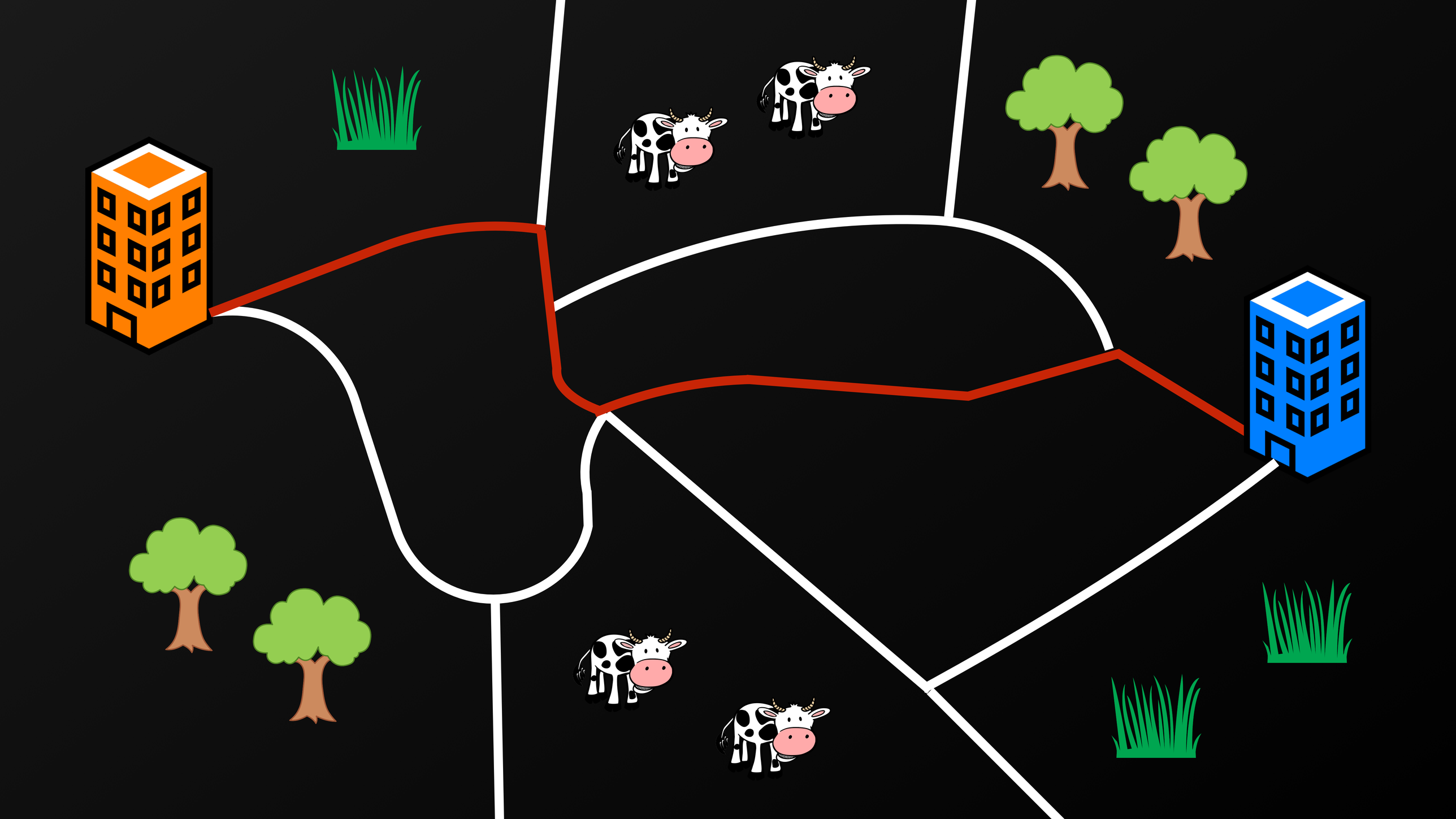
toString Considered Harmful,

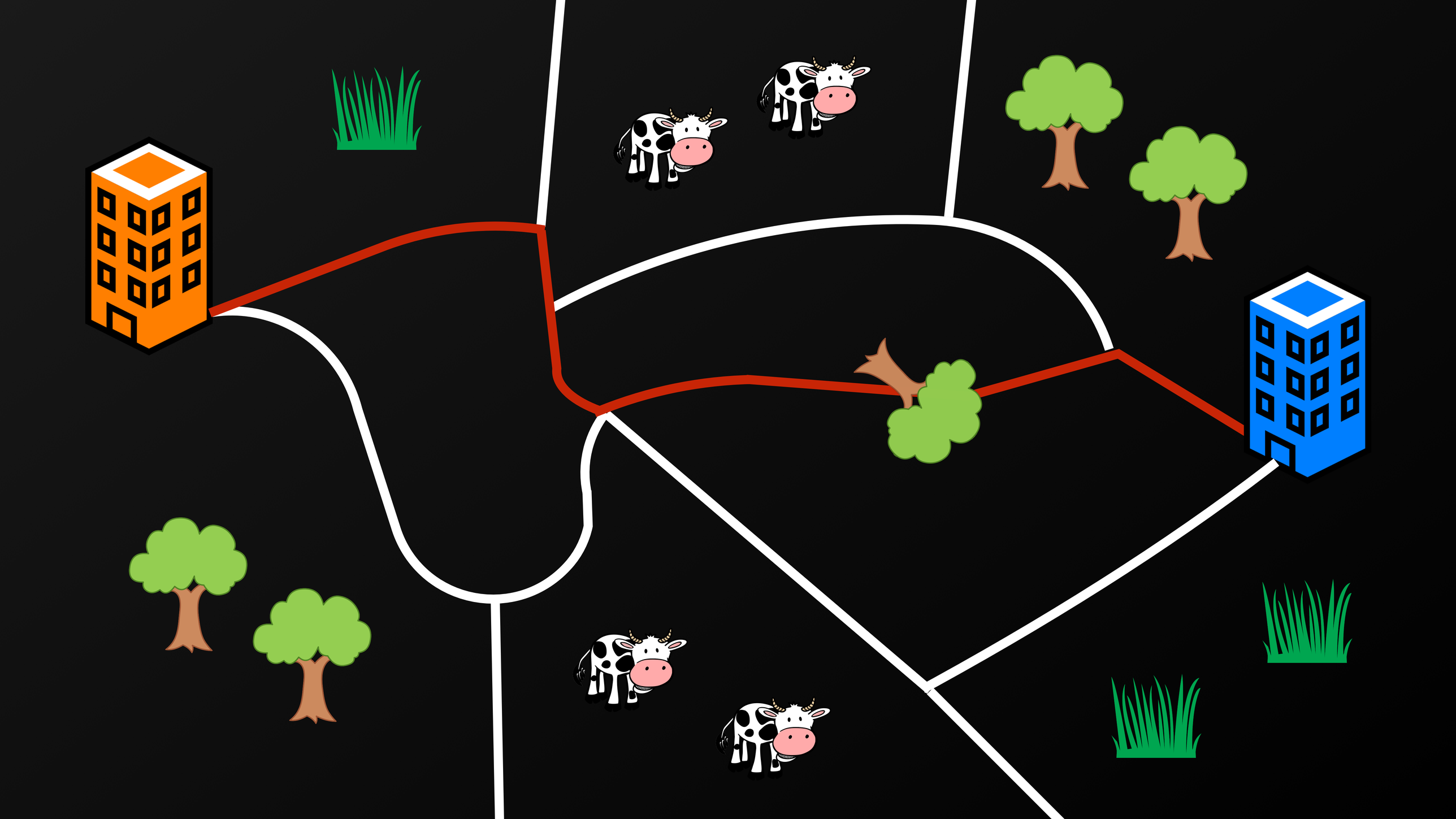
anonymous functions considered harmful

Global Variables Considered Harmful

Constructors Considered Harmful

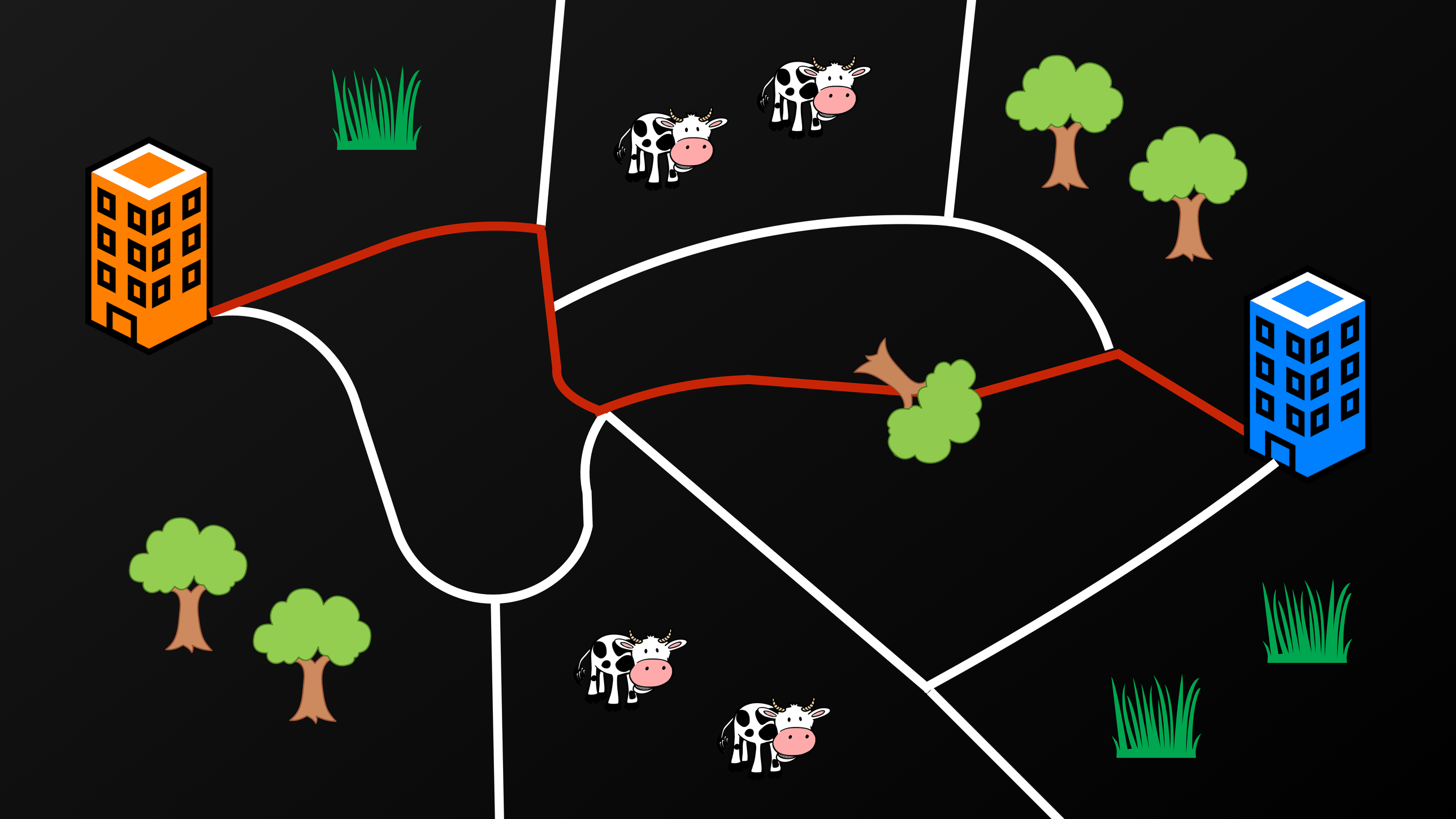
We’re left with only one way to get from A to B :(

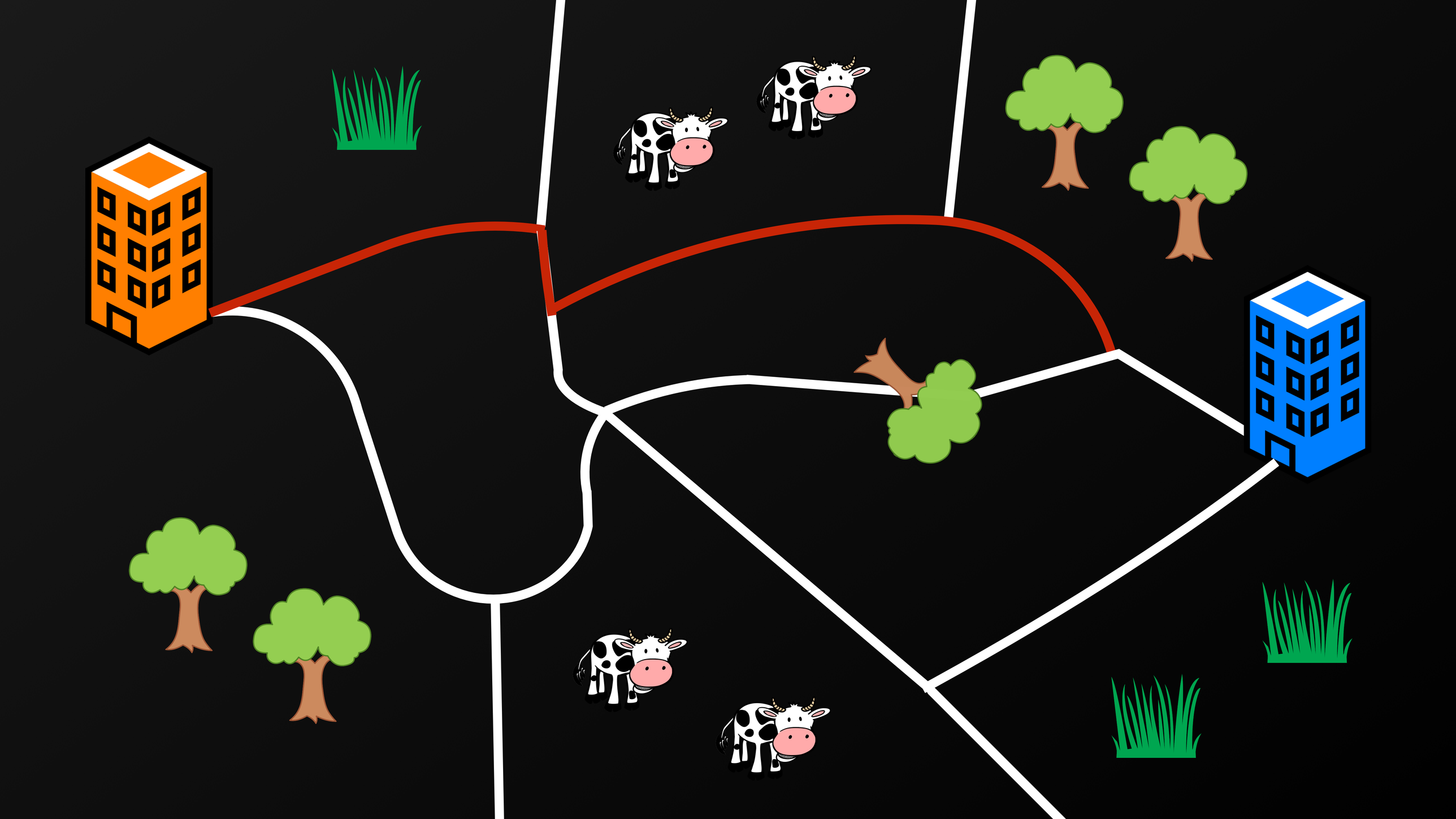




Literature is about
embracing the entire language

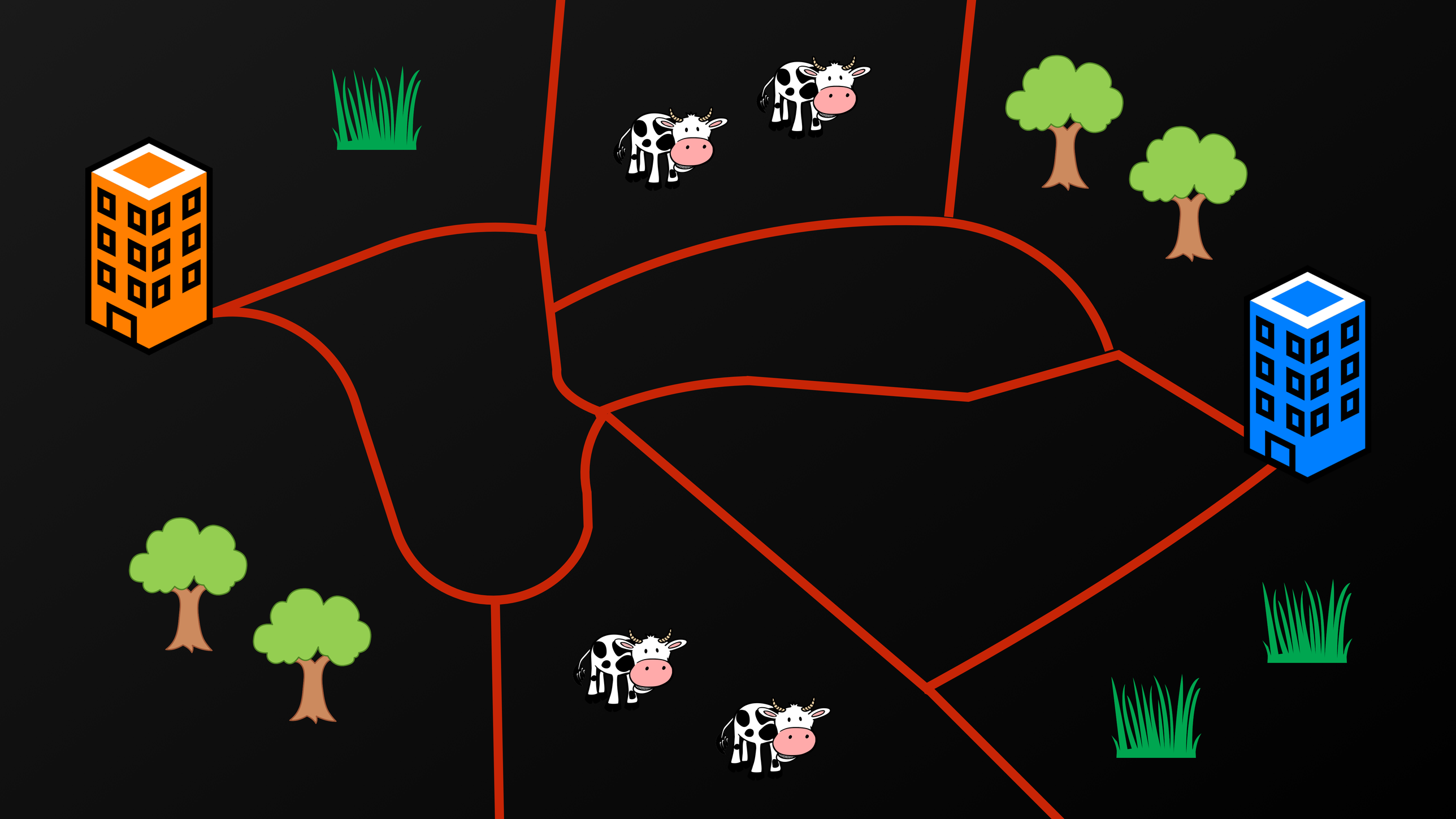
When programmers embrace the entire language we get choices…

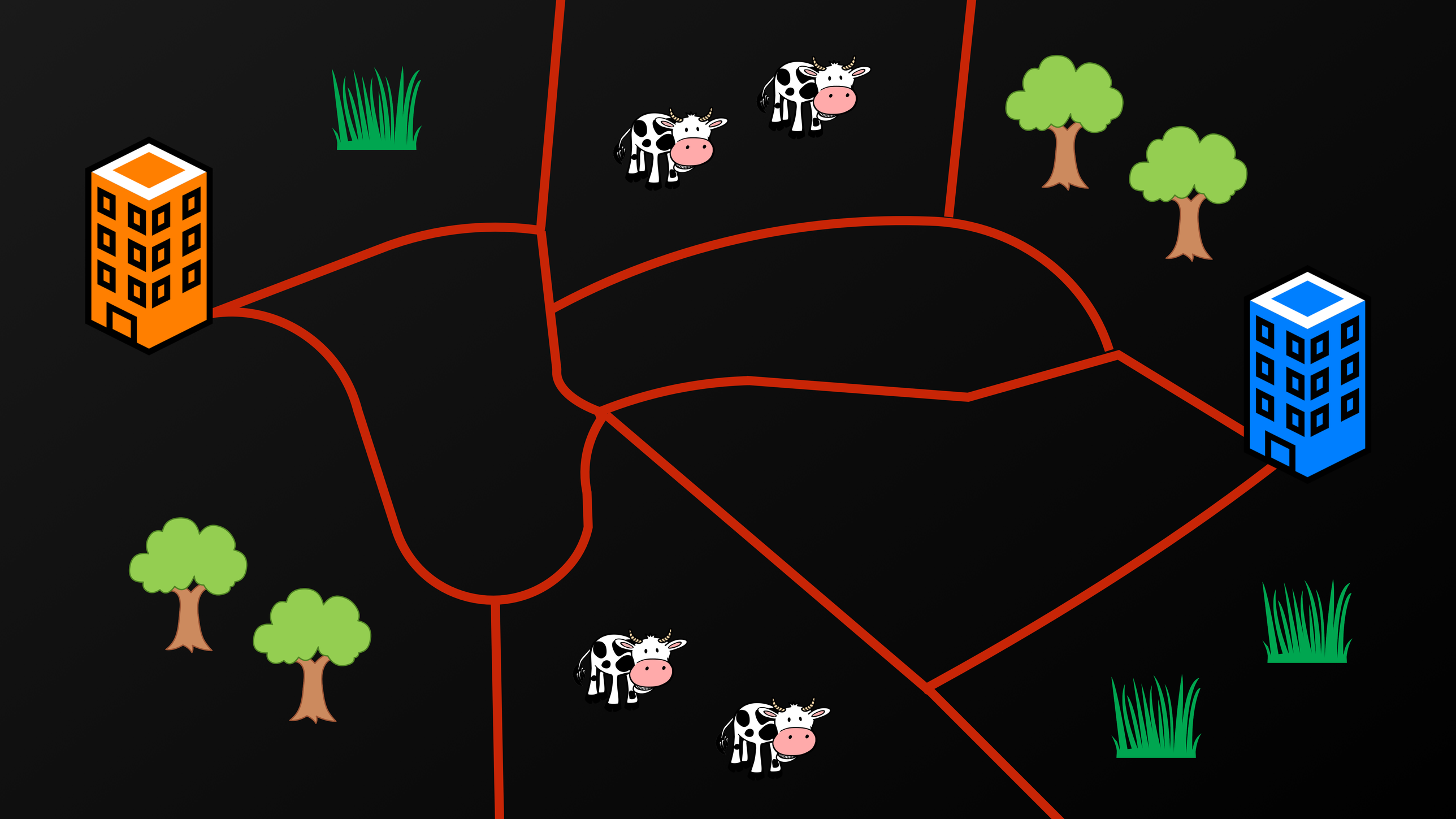




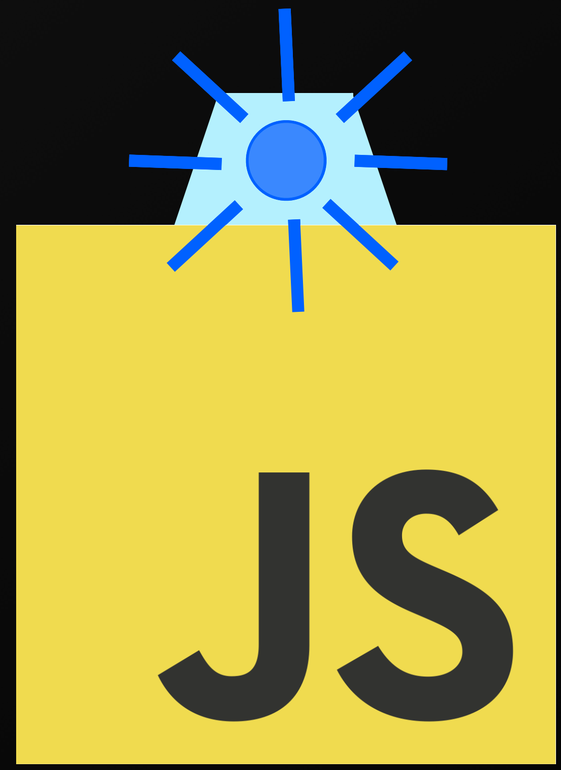
"There are indeed many ways to do the same thing in JS. That's one of the beauties of the language"

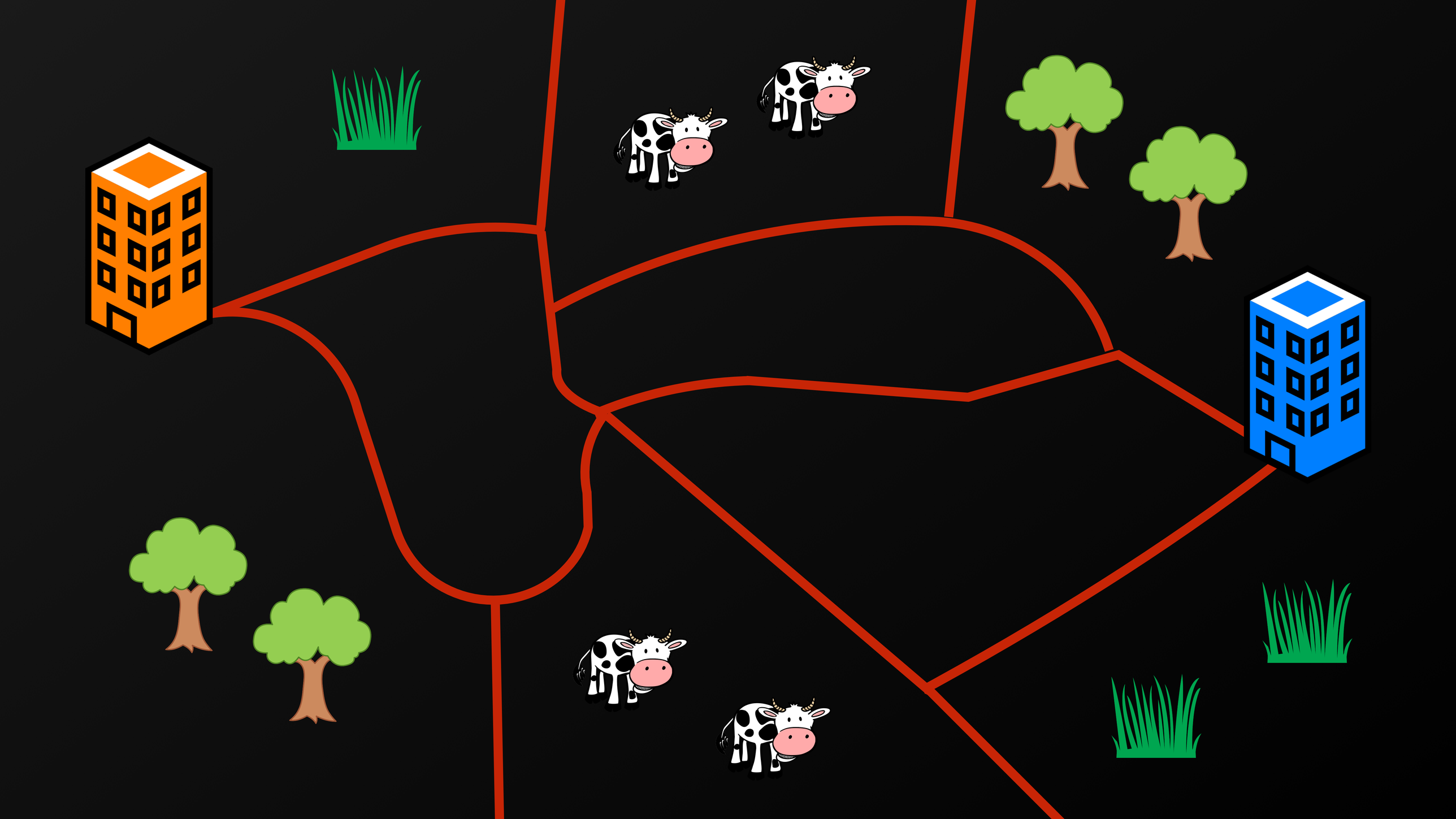
- Peter van der Zee (@kuvos)

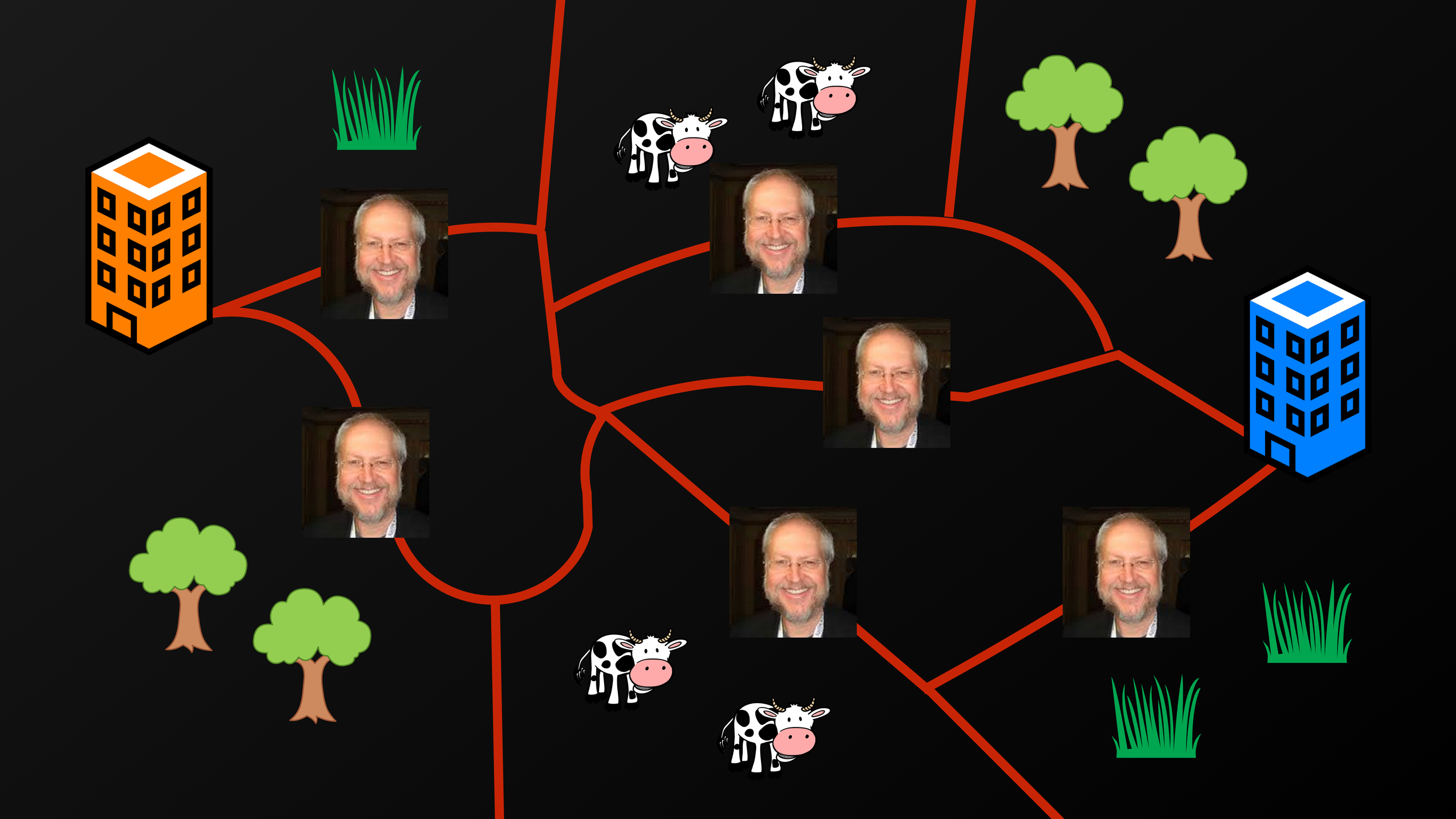




(at least until the
JavaScript Police
get here)

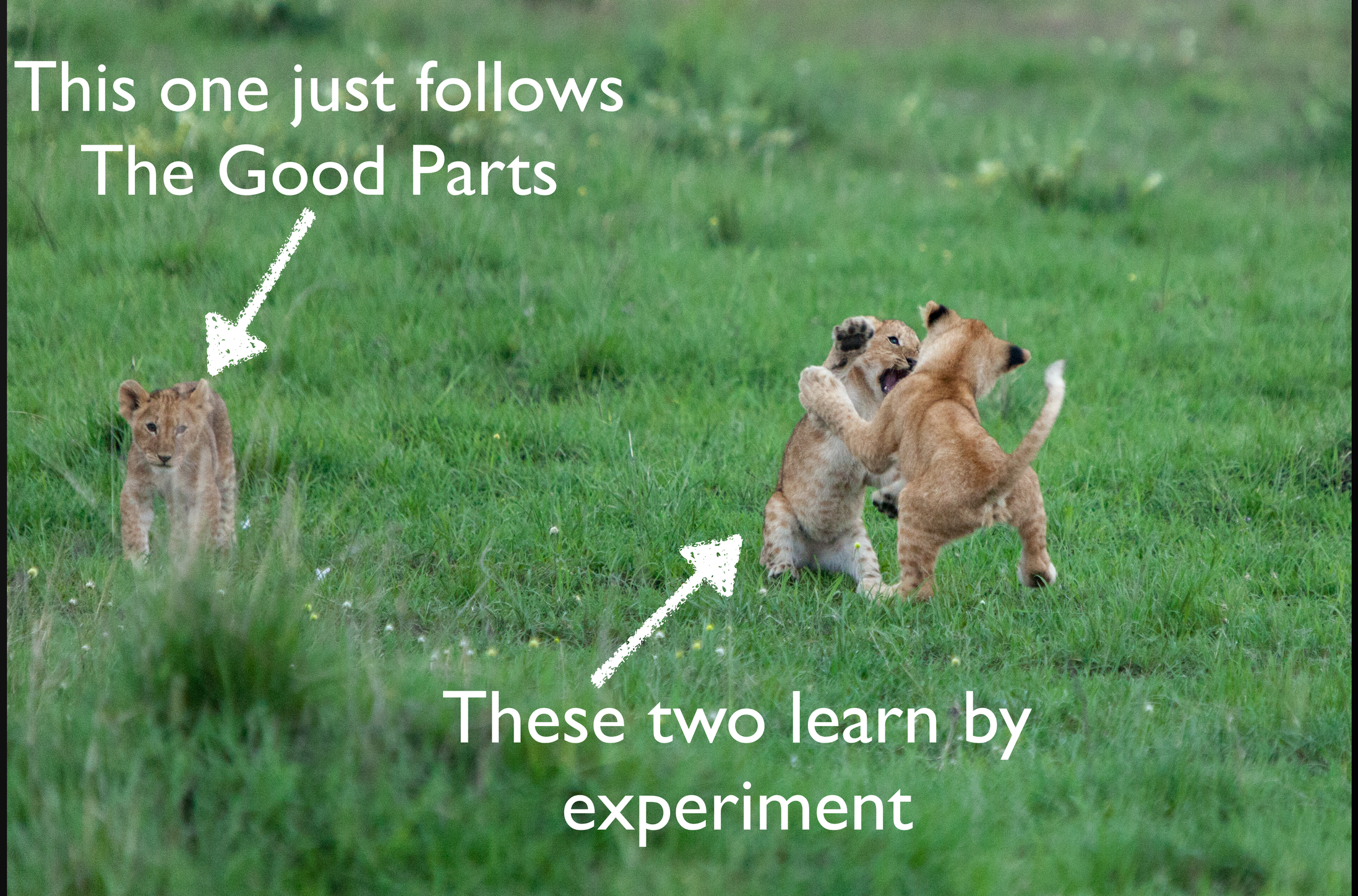






Perhaps most importantly…

…Open Minds
keep JavaScript fun.


This one just follows
The Good Parts
These two learn by
experiment

As programmers we’re not encouraged to experiment

Yet experiment is what keeps JavaScript alive

“You need a lot of bad code to write good code” - Substack (dotJS 2014)

- if(!!x)
- fn && fn()
- immediately invoked functions
- module pattern
- promises

ES 5

Function.prototype.bind

ES 5

String.prototype.trim

I decided to take *experiment* to the next level…

IF HEMINGWAY WROTE JAVASCRIPT

Angus Croll

Wait What?

Twenty Five Authors solving Five JavaScript Problems

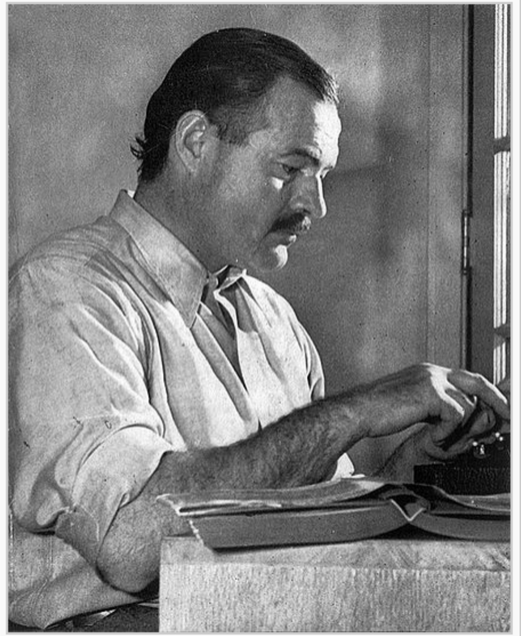
Hemingway developed the **Iceberg Theory**

Iceberg Theory

From Wikipedia, the free encyclopedia

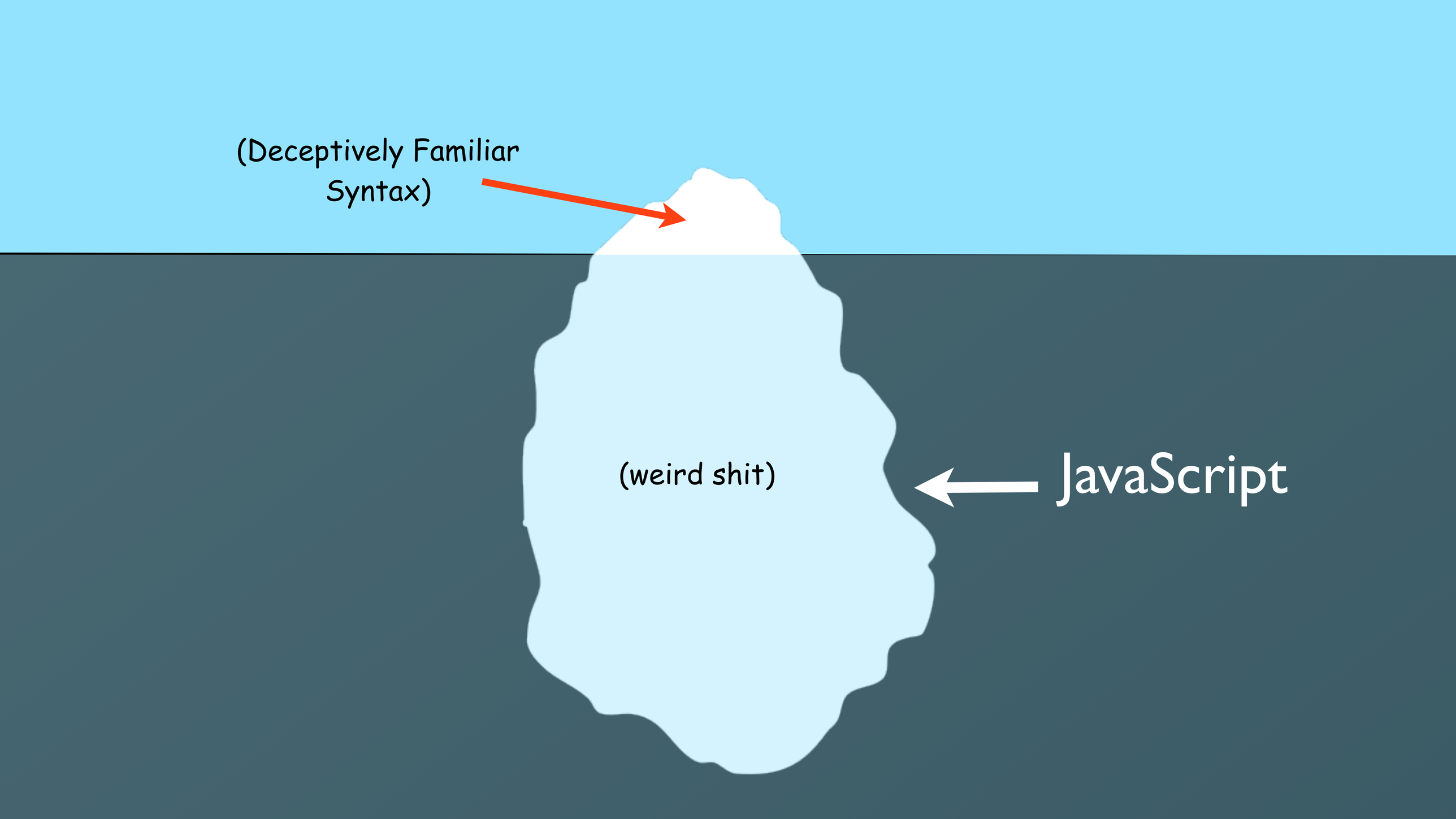
The **Iceberg Theory** (also known as the "theory of omission") is the writing style of American writer Ernest Hemingway. As a young journalist, Hemingway had to focus his newspaper reports on immediate events, with very little context or interpretation. When he became a writer of short stories, he retained this minimalistic style, focusing on surface elements without explicitly discussing underlying themes. Hemingway believed the deeper meaning of a story should not be evident on the surface, but should shine through implicitly. Critics such as Jackson Benson claim that the iceberg theory, along with his distinctive clarity of style, functioned to distance himself from the characters he created.

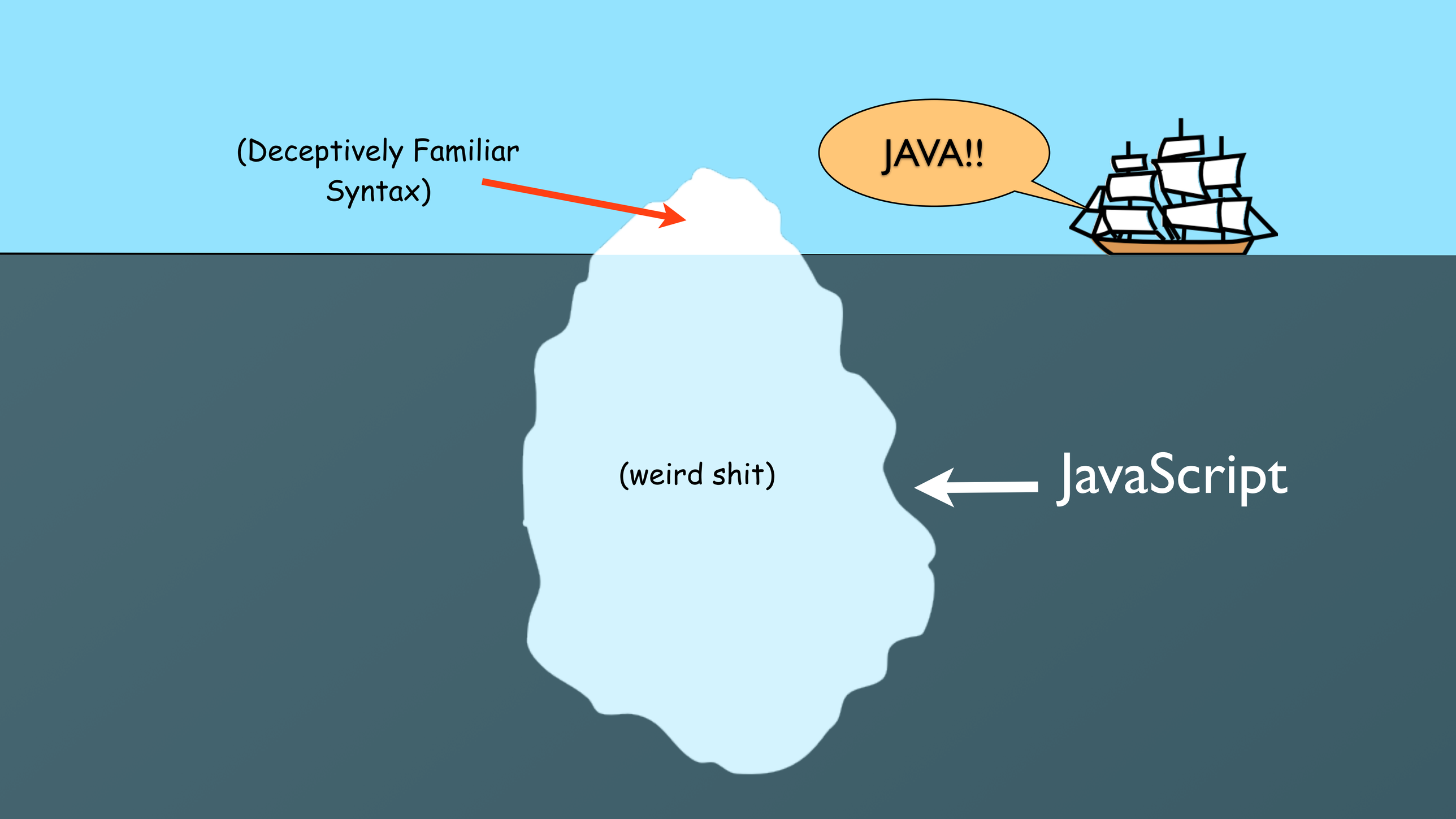
Contents [hide]

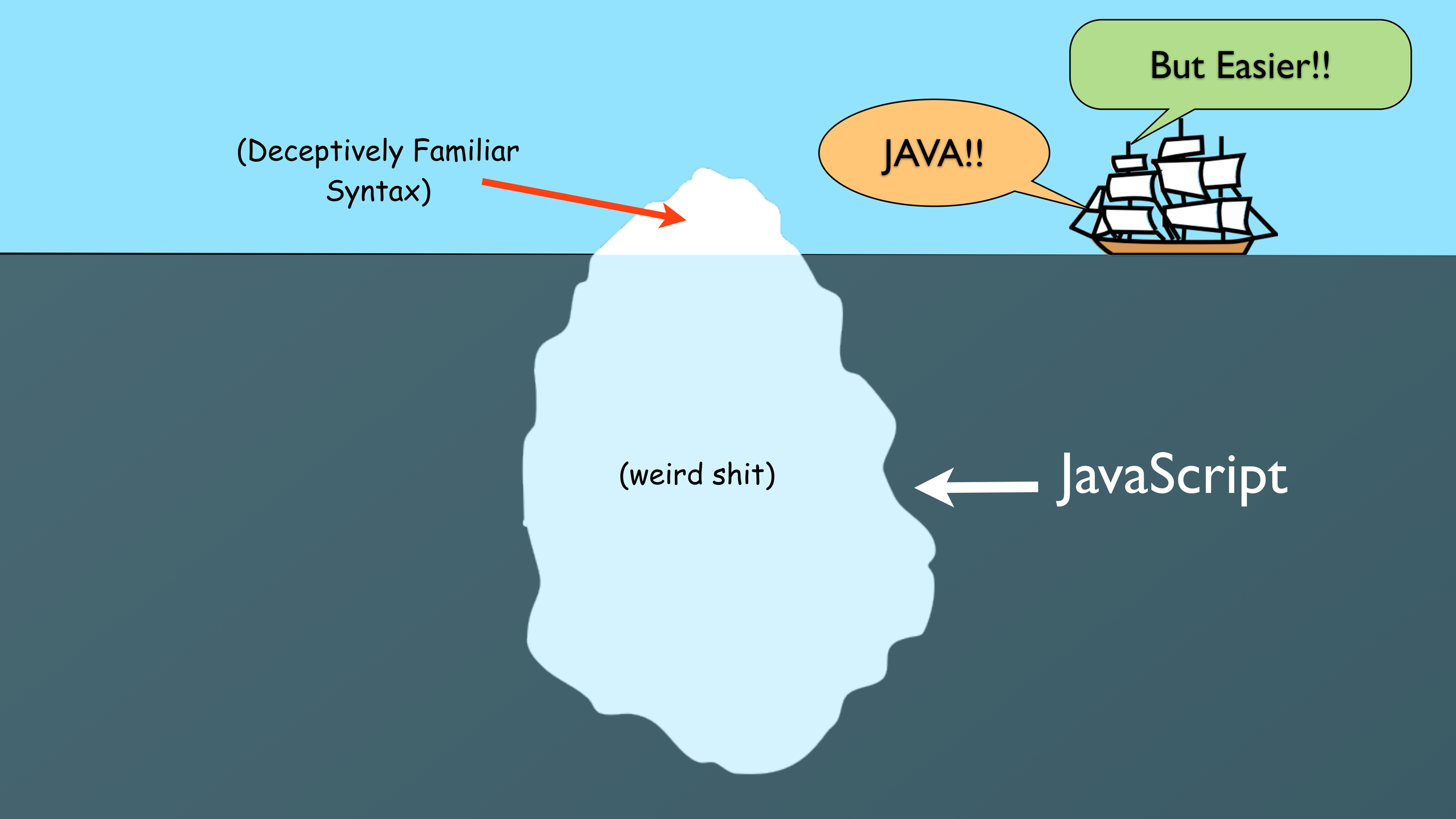


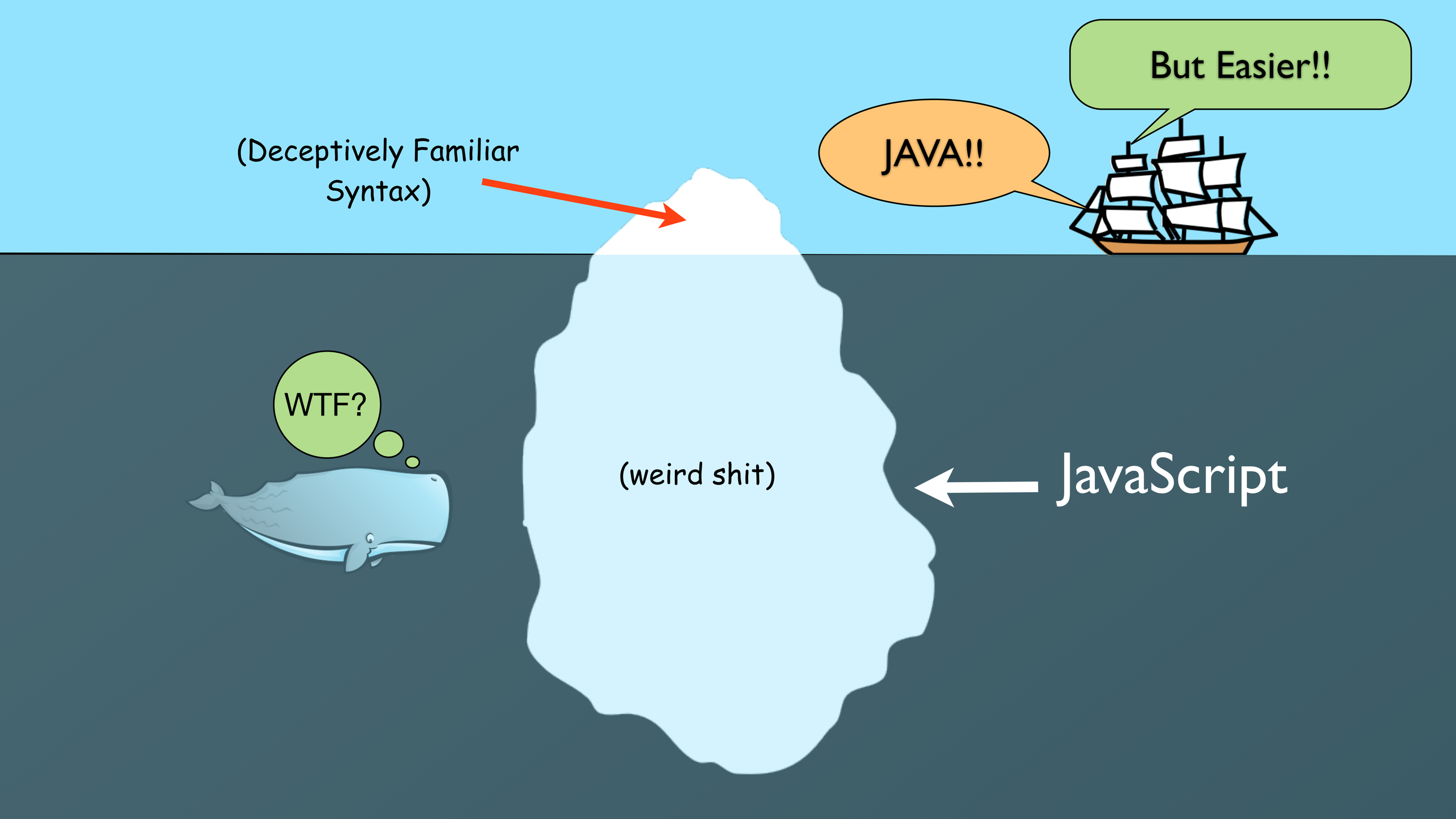
Ernest Hemingway as photographed for 1940 edition of *For Whom the Bell Tolls*

Here is Hemingway’s Iceberg Theory applied to **JavaScript**…


(Deceptively Familiar Syntax)
(weird shit)
JavaScript


(Deceptively Familiar Syntax)
JAVA!!
(weird shit)
JavaScript


But Easier!!
JAVA!!
(Deceptively Familiar Syntax)
(weird shit)
JavaScript

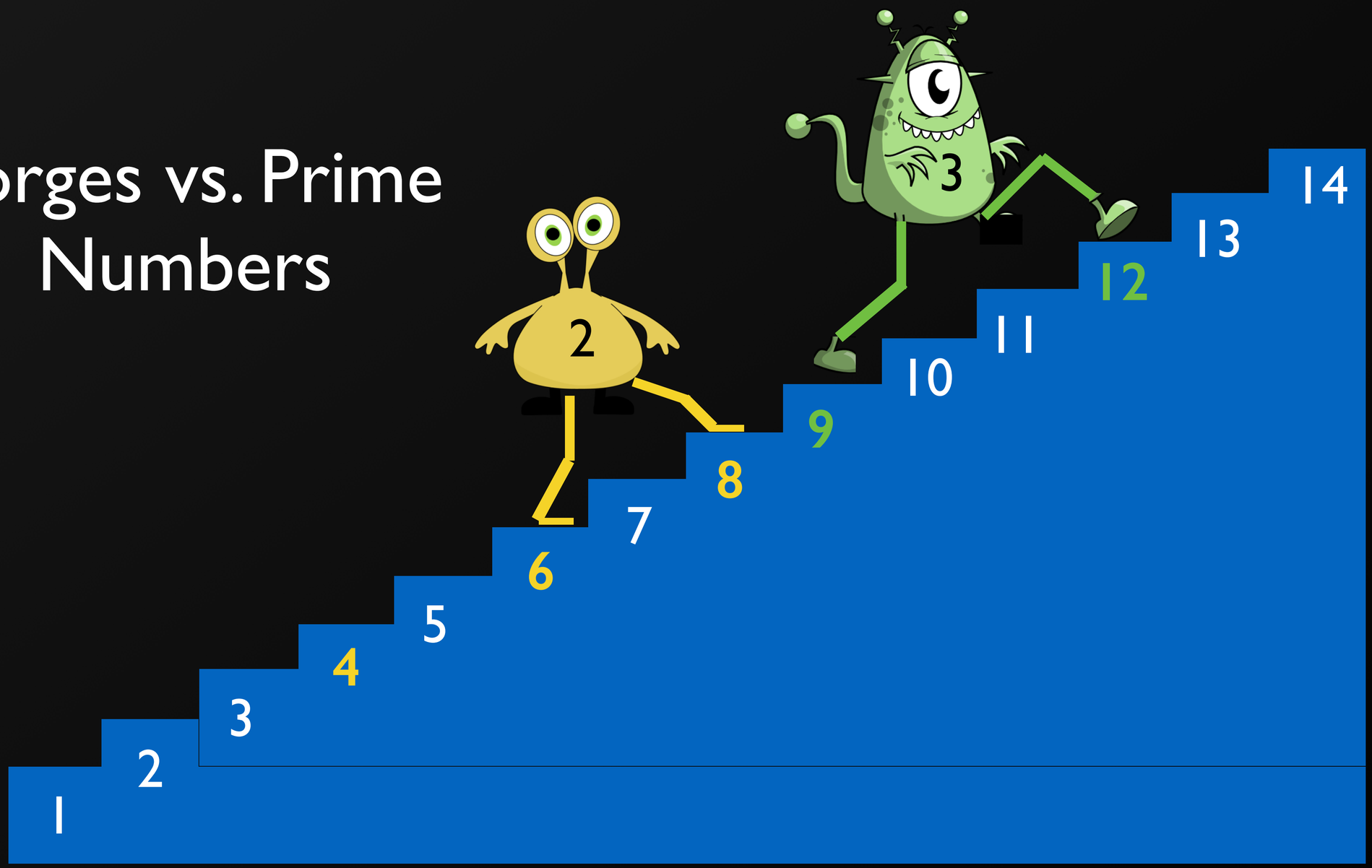

But Easier!!
JAVA!!
(Deceptively Familiar Syntax)
WTF?
(weird shit)
JavaScript

Generate Prime Numbers

JORGE LUIS BORGES

```
// A succession of creatures mount the stairs;
// each creature's stride exceeds that of its predecessor
var monstersAscendingAStaircase = function(numberOfSteps) {
  var stairs = []; stepsUntrodden = [];
  var largestGait = Math.sqrt(numberOfSteps);
  for (var i = 2; i <= largestGait; i++) {
    if (!stairs[i]) {
      for (var j = i * i; j <= numberOfSteps; j += i) {
        stairs[j] = 'stomp';
      }
    }
  }

  // Long-limbed monsters won't tread on prime numbered stairs.
  for (var i = 2; i <= numberOfSteps; i++) {
    if(!stairs[i]) {
      stepsUntrodden.push(i);
    }
  }

  // Here, then, is our answer.
  return stepsUntrodden;
};
```

Borges vs. Prime Numbers



LEWIS CARROLL

```
function downTheRabbitHole(growThisBig) {
  var theFullDeck = Array(growThisBig);
  var theHatter = Function('return this/4').call(2*2);
  var theMarchHare = Boolean("The frumious Bandersnatch!");

  var theVerdict = "the white rabbit".split(/the march
hare/).slice(theHatter);

  //into the pool of tears...
  eval(theFullDeck.join("if (!theFullDeck[++theHatter]) {\
      theMarchHare = 1;\
      theVerdict.push(theHatter);\
      " + theFullDeck.join("theFullDeck[++theMarchHare *
theHatter]=true;") + "}")
  );

  return theVerdict;
}
```

```
function downTheRabbitHole(growThisBig) {
  var theFullDeck = Array(growThisBig);
  var theHatter = Function('return this/4').call(2*2);
  var theMarchHare = Boolean("The frumious Bandersnatch!");

  var theVerdict = "the white rabbit".split(/the march
hare/).slice(theHatter);

  //into the pool of tears...
  eval(theFullDeck.join("if (!theFullDeck[++theHatter]) {\
      theMarchHare = 1;\
      theVerdict.push(theHatter);\
      " + theFullDeck.join("theFullDeck[++theMarchHare *
theHatter]=true;") + "}")
  );

  return theVerdict;
}
```

```
function downTheRabbitHole(growThisBig) {
  var theFullDeck = Array(growThisBig);
  var theHatter = Function('return this/4').call(2*2);
  var theMarchHare = Boolean("The frumious Bandersnatch!");

  var theVerdict = "the white rabbit".split(/the march
hare/).slice(theHatter);

  //into the pool of tears...
  eval(theFullDeck.join("if (!theFullDeck[++theHatter]) {\
      theMarchHare = 1;\
      theVerdict.push(theHatter);\
      " + theFullDeck.join("theFullDeck[++theMarchHare *
theHatter]=true;") + "}")
  );

  return theVerdict;
}
```

```
new Array(5).join('A' + new Array(5).join('a'));

>
```

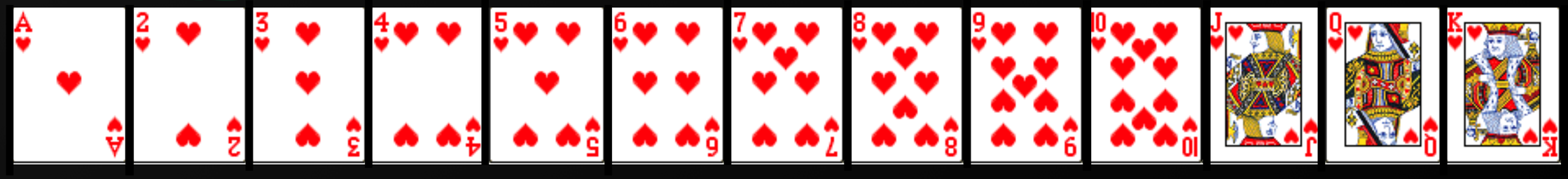
```
new Array(5).join("A" + new Array(5).join('a'));

> "AaaaaAaaaaAaaaaAaaaa"
```

```
function downTheRabbitHole(growThisBig) {
  var theFullDeck = Array(growThisBig);
  var theHatter = Function('return this/4').call(2*2);
  var theMarchHare = Boolean("The frumious Bandersnatch!");

  var theVerdict = "the white rabbit".split(/the march
hare/).slice(theHatter);

  //into the pool of tears...
  eval(theFullDeck.join("if (!theFullDeck[++theHatter]) {\
      theMarchHare = 1;\
      theVerdict.push(theHatter);\
      " + theFullDeck.join("theFullDeck[++theMarchHare *
theHatter]=true;") + "}")
  );

  return theVerdict;
}
```

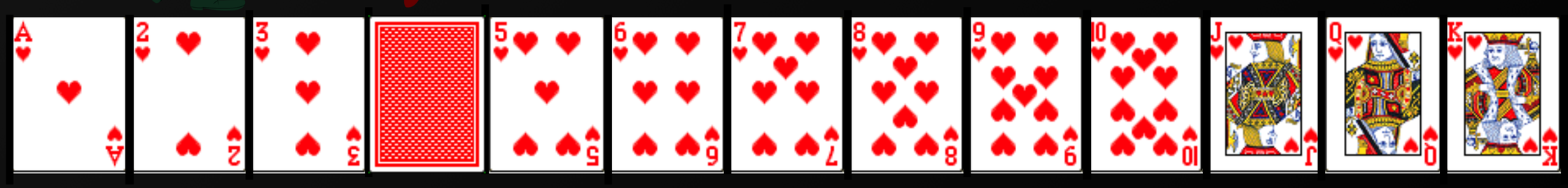
```
eval("if(!theFullDeck[++theHatter]){theMarchHare=1;theVerdict.push(theHatter);theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;}if(!theFullDeck[+
+theHatter]){theMarchHare=1;theVerdict.push(theHatter);theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;}if(!theFullDeck[+
+theHatter]){theMarchHare=1;theVerdict.push(theHatter);theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;}if(!theFullDeck[+
+theHatter]){theMarchHare=1;theVerdict.push(theHatter);theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;}if(!theFullDeck[+
+theHatter]){theMarchHare=1;theVerdict.push(theHatter);theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;}if(!theFullDeck[+
+theHatter]){theMarchHare=1;theVerdict.push(theHatter);theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;theFullDeck[+
+theMarchHare*theHatter]=true;theFullDeck[++theMarchHare*theHatter]=true;}");
```

```
eval(“if (!theFullDeck[++theHatter]) {
  theMarchHare = 1;
  theVerdict.push(theHatter);
  theFullDeck[++theMarchHare * theHatter] = true;
  theFullDeck[++theMarchHare * theHatter] = true;
  theFullDeck[++theMarchHare * theHatter] = true;
  theFullDeck[++theMarchHare * theHatter] = true;
  theFullDeck[++theMarchHare * theHatter] = true;
  theFullDeck[++theMarchHare * theHatter] = true;
}
if (!theFullDeck[++theHatter]) {
  theMarchHare = 1;
  theVerdict.push(theHatter);
  theFullDeck[++theMarchHare * theHatter] = true;
  theFullDeck[++theMarchHare * theHatter] = true;
  theFullDeck[++theMarchHare * theHatter] = true;
  theFullDeck[++theMarchHare * theHatter] = true;
  theFullDeck[++theMarchHare * theHatter] = true;
  theFullDeck[++theMarchHare * theHatter] = true;
}
if (!theFullDeck[++theHatter]) {
  theMarchHare = 1;
  theVerdict.push(theHatter);
```

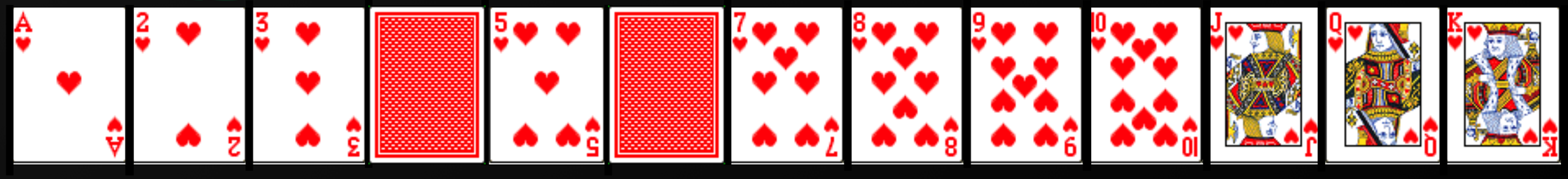
Alice in Prime Numberland



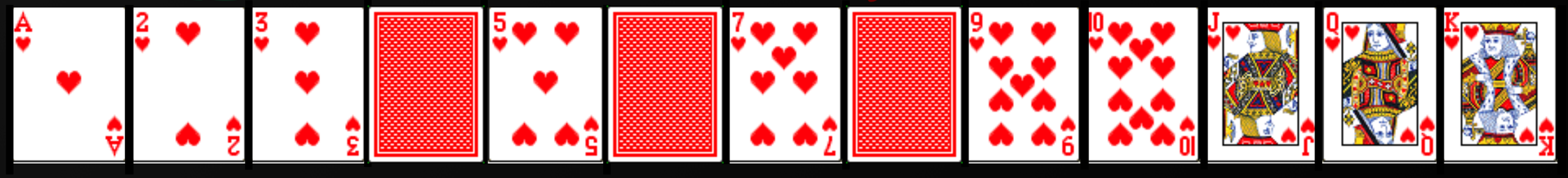
Alice in Prime Numberland



Alice in Prime Numberland

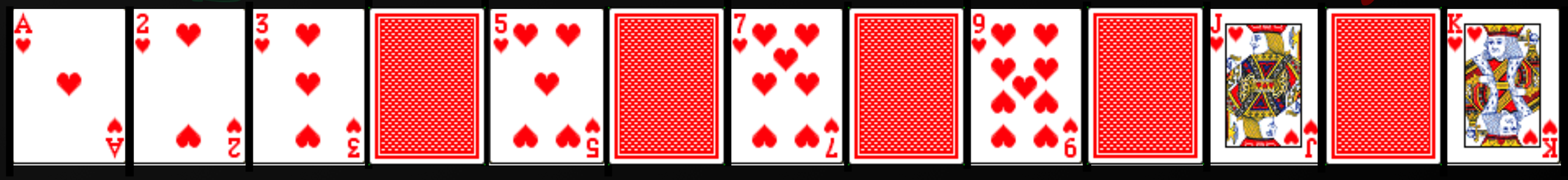


Alice in Prime Numberland

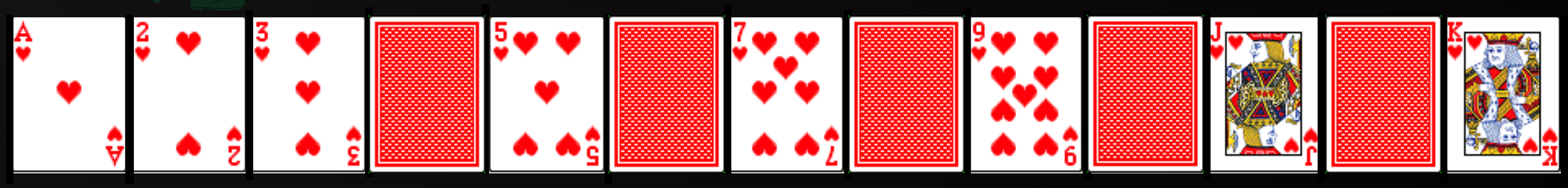


Alice in Prime Numberland

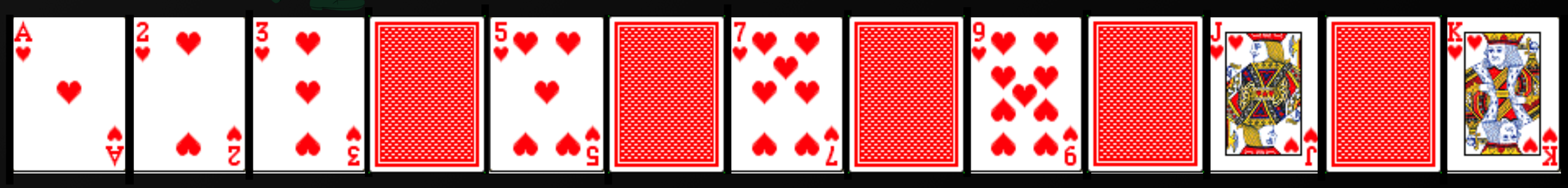
Alice in Prime Numberland



Alice in Prime Numberland



Alice in Prime Numberland



Alice in Prime Numberland

Alice in Prime Numberland

Alice in Prime Numberland

So what did we learn?


Playing is Learning

Self-taught programmers are often the best programmers because they get to make mistakes

Experiment, **Play**, **Have Fun** and Keep an **Open Mind**.

Fin, merci!

anguscroll.com/hemingway

@angustweets

Image credits

Books: http://pixgood.com/classic-book-spines.html/
Montparnasse Tower: http://allparisguides.com
Eiffel Tower:
Yellow Monster: http://content.mycutegraphics.com/
Green Monster: http://wondersofdisney2.yolasite.com/
White Rabbit: http://www.clker.com/
Mad Hatter, **March Hare**: http://wondersofdisney2.yolasite.com/
Cow, **tree, office buildings**: clipartpanda.com/
Grass: imgarcade.com/
Lion cubs: https://www.flickr.com/photos/mbiddulph/6947067048/
Cheetah cubs: http://www.bhmpics.com/view-cheetah_cubs_playing-1920x1080.html/