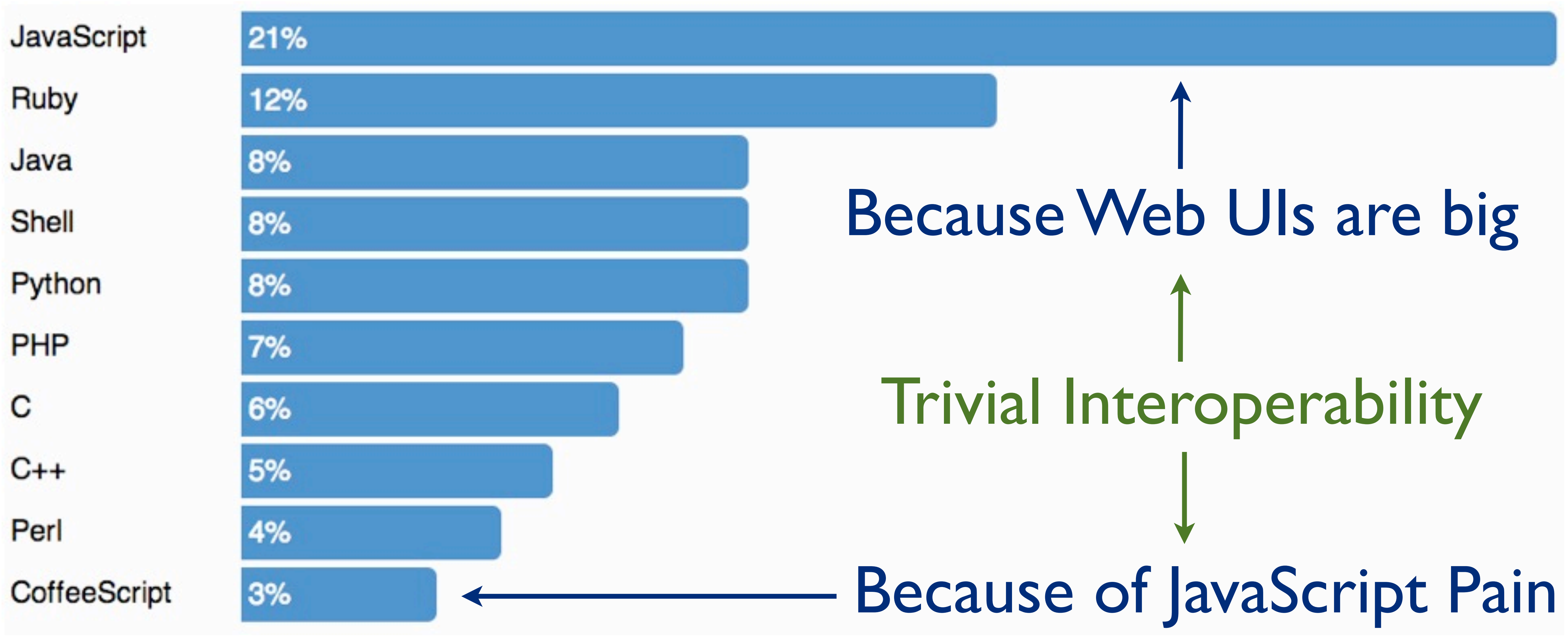
Functional CoffeeScript for Web UIs

@rtfeldman

Most Popular Languages on GitHub



Warts Removed

Niceties Added

"It's Just JavaScript"

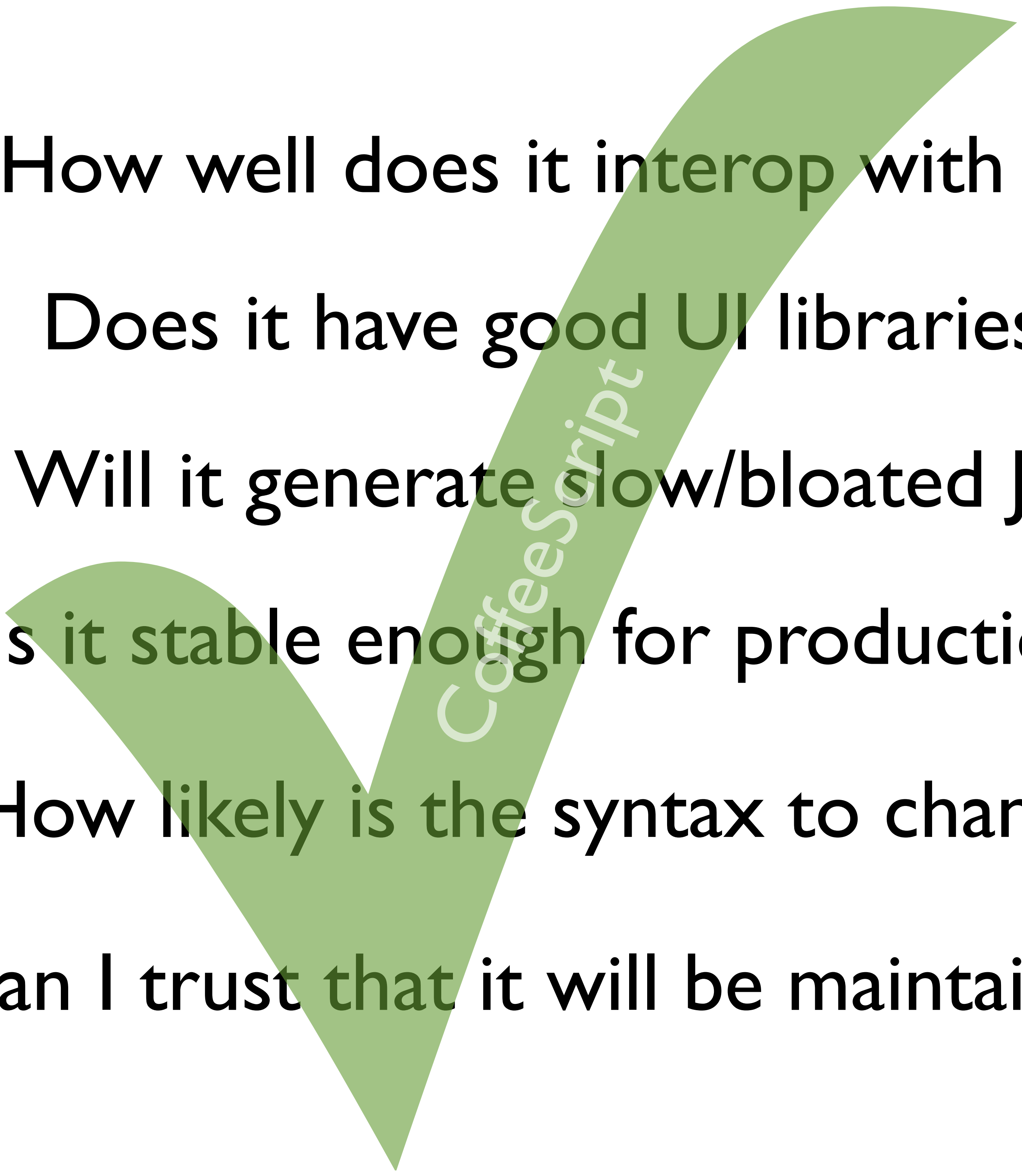
github.com/languages

Functional-Friendliness

x86 Assembly	JavaScript	CoffeeScript	Erlang
Not Machine Code	Expressions	Expressions Everywhere	Constants Everywhere
	First-Class Functions	Implicit Return	Immutability

altjs.org

ClojureScript
Elm
Roy
GHCjs
Haste
Pit
Ocamljs
Shen
Sibilant
LispyScript
...



How well does it interop with JS?

Does it have good UI libraries?

Will it generate slow/bloated JS?

Is it stable enough for production?

How likely is the syntax to change?

Can I trust that it will be maintained?

```
var names = ["sam", "taylor", "morgan"];
var capitalizedNames = [];

for (var i=0; i < names.length; i++) {
    var name = names[i];
    var capitalized = name[0].toUpperCase()
        + name.slice(1);

    capitalizedNames.push(capitalized);
}

console.log(capitalizedNames);
```

```
names = ["sam", "taylor", "morgan"]
capitalizedNames = []

for name in names
  capitalized = name[0].toUpperCase() +
    name.slice(1)

  capitalizedNames.push capitalized

console.log capitalizedNames
```

underscorejs.org

```
_.map

_.reduce

_.filter
```

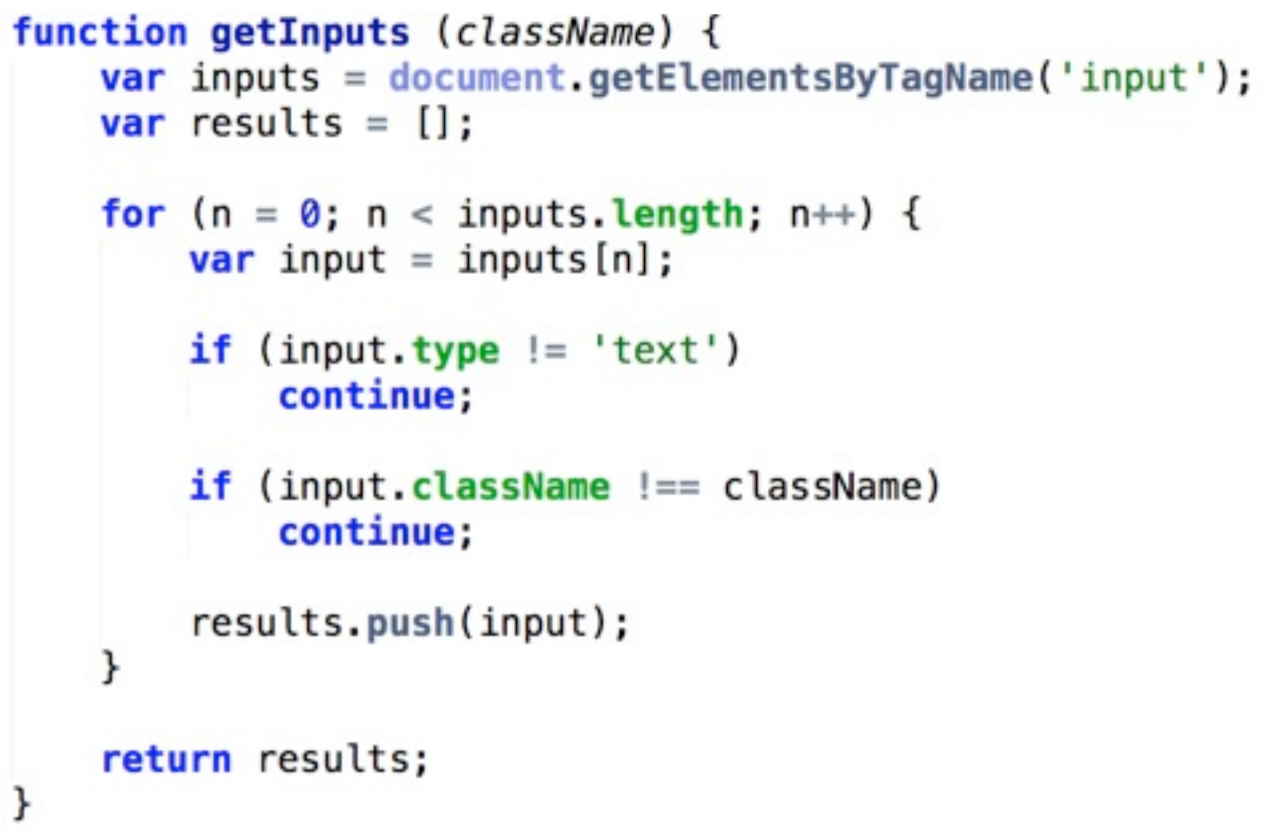
```
var capitalizedNames = _.map(
    names, function(name) {
        return name[0].toUpperCase()
            + name.slice(1);
    });
```

```
capitalizedNames = _.map names, (name) ->
    name[0].toUpperCase() + name.slice(1)
```

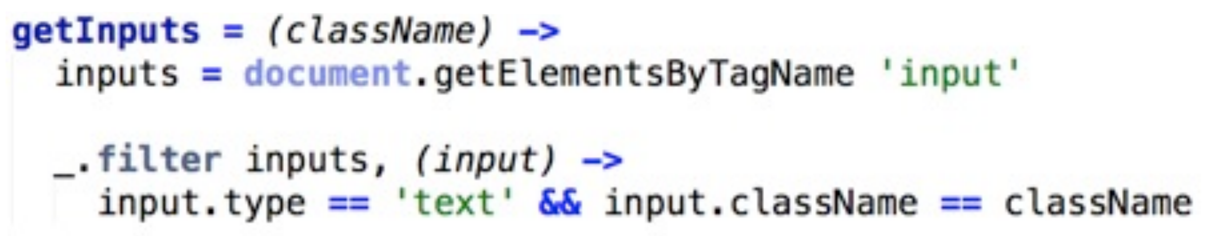
```
names = ["sam", "taylor", "morgan"]

capitalize = (str) ->
    str[0].toUpperCase() + str.slice(1)

console.log _.map names, capitalize
```

```javascript
/* Return all the INPUT elements
 * which have the given class name. */
function getInputs (className) {
    var inputs = document.getElementsByTagName('input');
    var results = [];

    for (n = 0; n < inputs.length; n++) {
        var input = inputs[n];

        if (input.type != 'text')
            continue;

        if (input.className !== className)
            continue;

        results.push(input);
    }

    return results;
}
```

```coffeescript
getInputs = (className) ->
  inputs = document.getElementsByTagName 'input'

  _.filter inputs, (input) ->
    input.type == 'text' && input.className == className
```



```javascript
function getInputs (className) {
    var inputs = document.getElementsByTagName('input');
    var results = [];

    for (n = 0; n < inputs.length; n++) {
        var input = inputs[n];

        if (input.type != 'text')
            continue;

        if (input.className !== className)
            continue;

        results.push(input);
    }

    return results;
}
```



```coffeescript
getInputs = (className) ->
  inputs = document.getElementsByTagName 'input'

  _.filter inputs, (input) ->
    input.type == 'text' && input.className == className
```

```javascript
document.querySelectorAll(
   "input[type=text]." + className)
```

js2coffee.org

JavaScript → CoffeeScript

CoffeeScript → JavaScript
(`coffee -c`)

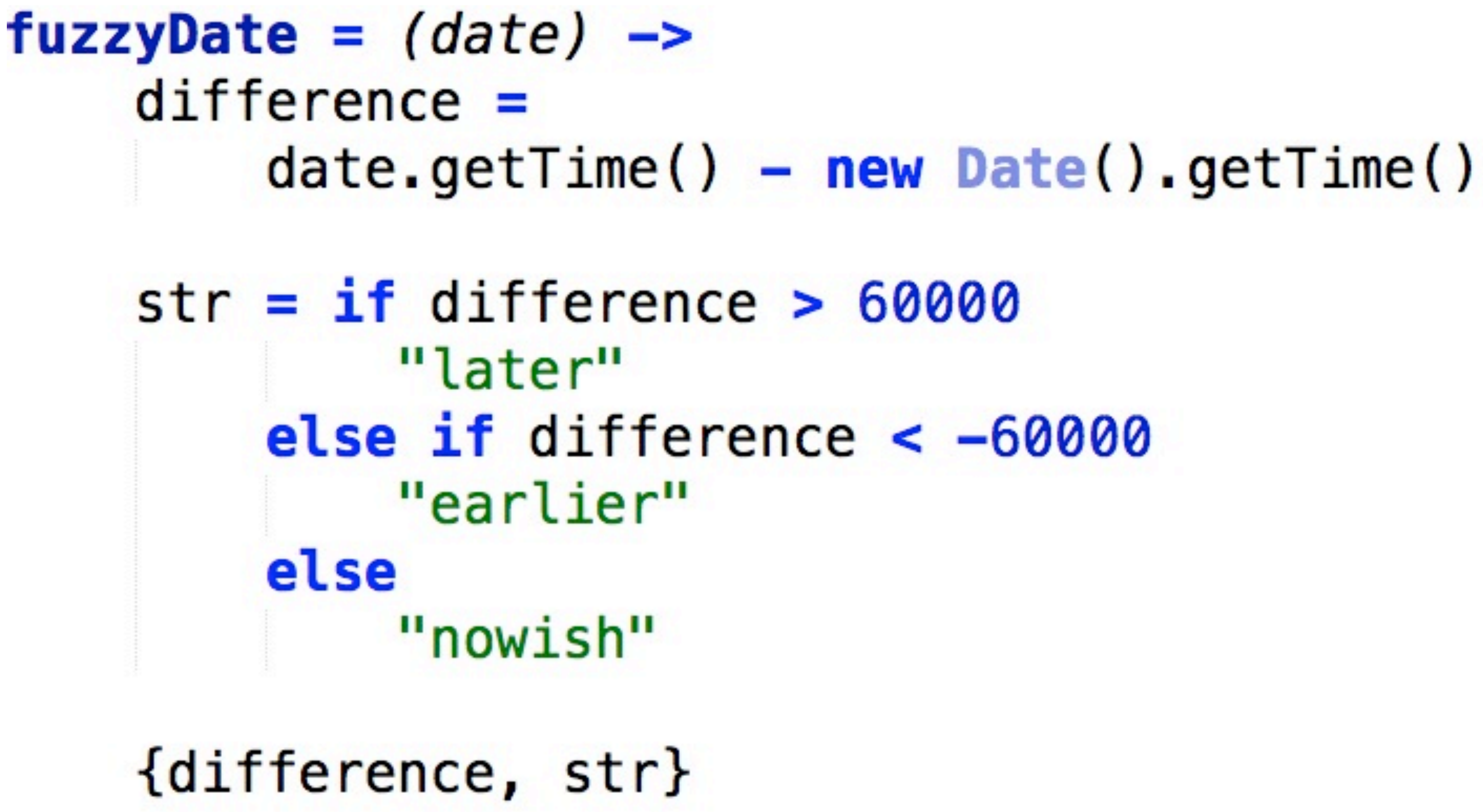
```
fuzzyDate = (date) ->
    difference =
        date.getTime() - new Date().getTime()

    str = if difference > 60000
            "later"
        else if difference < -60000
            "earlier"
        else
            "nowish"

    {difference, str}
```

```
function fuzzyDate (date) {
    var difference =
        date.getTime() - new Date().getTime();

    var str;

    if (difference > 60000) {
        str = "later";
    } else if (difference < -60000) {
        str = "earlier";
    } else {
        str = "nowish";
    }

    return {difference: difference, str: str}
}
```

```
const weekends = ["Sat", "Sun"];
```

```
`const weekends = ["Sat", "Sun"];`
```

```
`const difference =
    date.getTime() - new Date().getTime()`
```



```
fuzzyDate = (date) ->
    difference =
        date.getTime() - new Date().getTime()

    str = if difference > 60000
            "later"
        else if difference < -60000
            "earlier"
        else
            "nowish"

    {difference, str}
```

~~`const str = if difference > 60000`~~

~~`const str; if (difference > 60000)`~~

DISCIPLINE

IT’S WHAT’S FOR BREAKFAST

Closure Compiler

developers.google.com/closure/compiler

JavaScript → Optimized JavaScript

CoffeeScript → JavaScript → Optimized JavaScript

```
###* @const ###
str = if difference > 60000
```

```
/** @const */
var str;
if (difference > 60000)
```

Very Imperative

Wrong Sugar

The Sweet Spot: Libraries

DOM

jQuery

Ext

D3

FRP

Arguments over Statements

```
# Assume these variables are already populated
req = new XMLHttpRequest()
req.onreadystatechange = ->
    if req.readyState == 4
        success()

req.open             "POST", url
req.setRequestHeader "content-type", "application/json"
req.send             JSON.stringify data
```

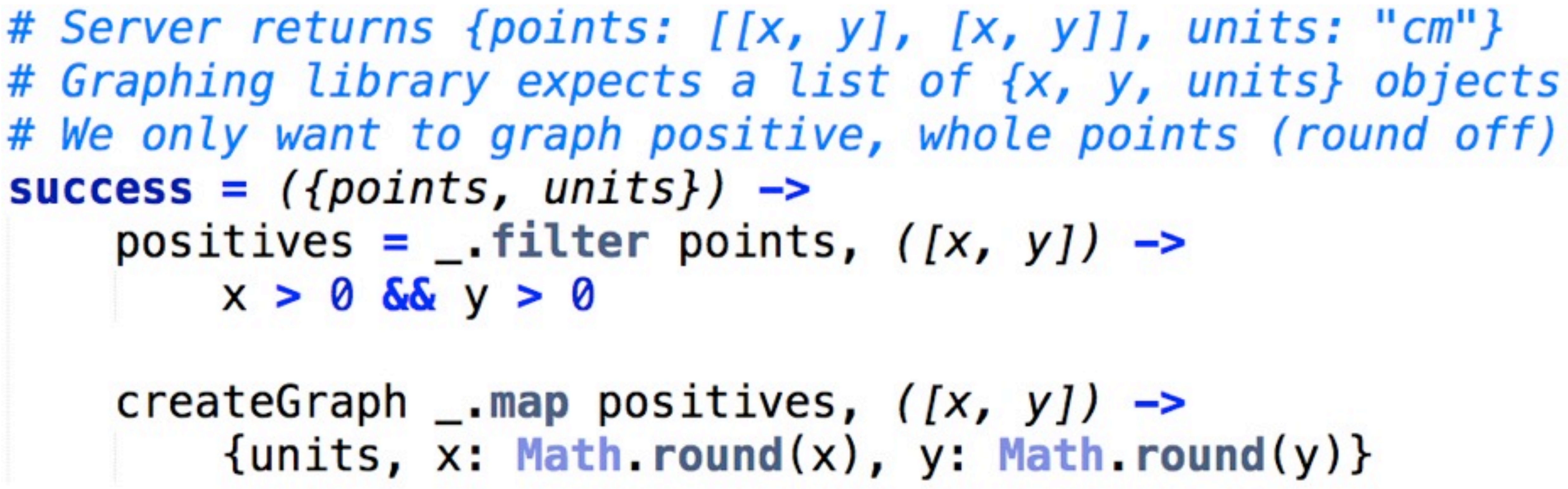
```
callback = (data, status) ->
    points = data.points
    units  = data.units

    # Logic goes here...
```

```
callback = ({points, units}, status) ->
    # Logic goes here...
```

Destructuring Assignment

```
_.map coordinates, (pair) ->
    x = pair[0]
    y = pair[1]

    # Logic goes here...
```

```
_.map coordinates, ([x, y]) ->
    # Logic goes here...
```



```coffeescript
# Server returns {points: [[x, y], [x, y]], units: "cm"}
# Graphing library expects a list of {x, y, units} objects
# We only want to graph positive, whole points (round off)
success = ({points, units}) ->
    positives = _.filter points, ([x, y]) ->
        x > 0 && y > 0

    createGraph _.map positives, ([x, y]) ->
        {units, x: Math.round(x), y: Math.round(y)}
```