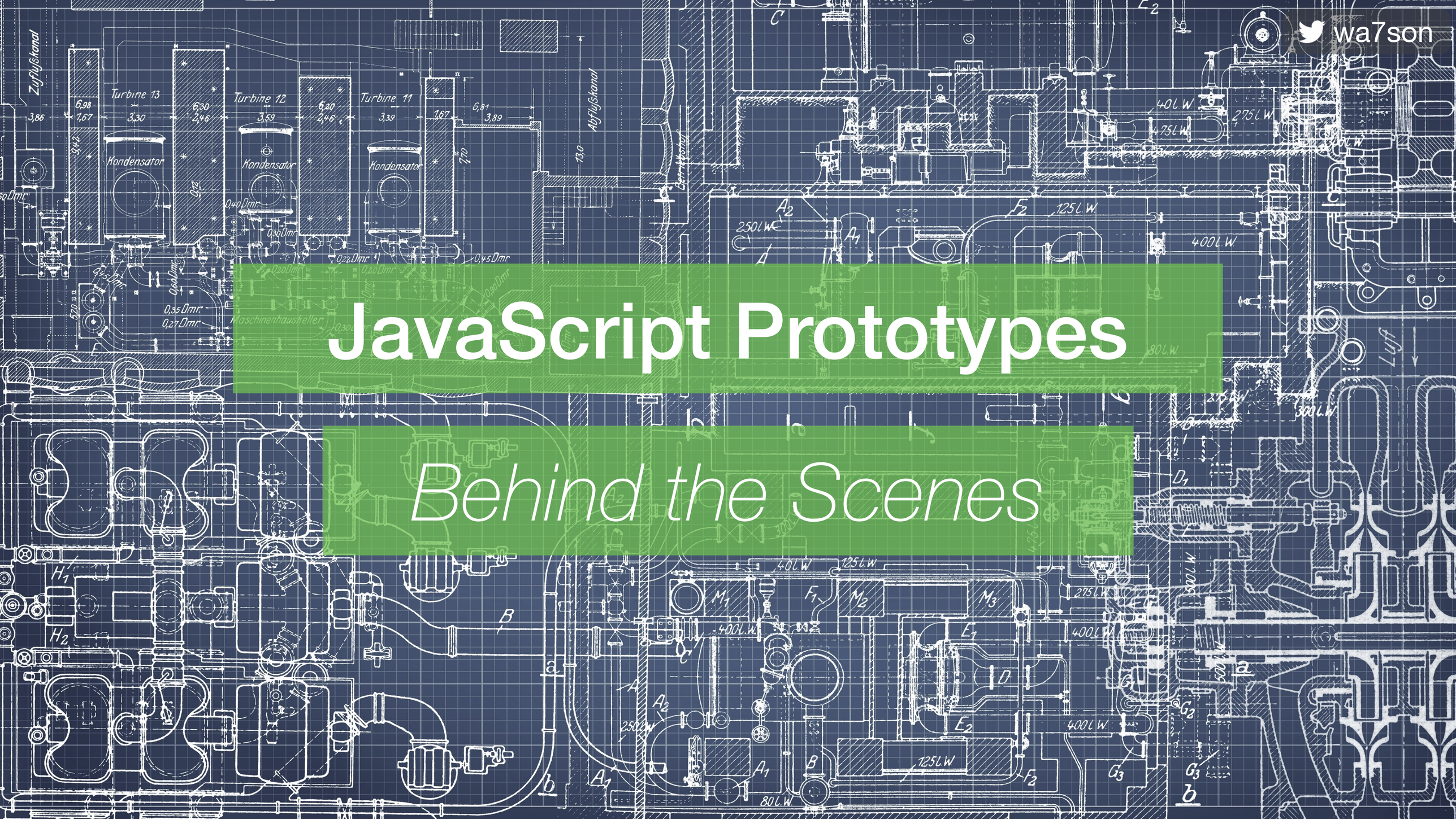

Thomas Watson
wa7son
github.com/watson

Who am I?

- Thomas Watson
- Principal Software Engineer at Elastic
- Open Source developer at github.com/watson
- Node.js Core Member
- Tweets as @wa7son
- Slides: github.com/watson/talks

elastic

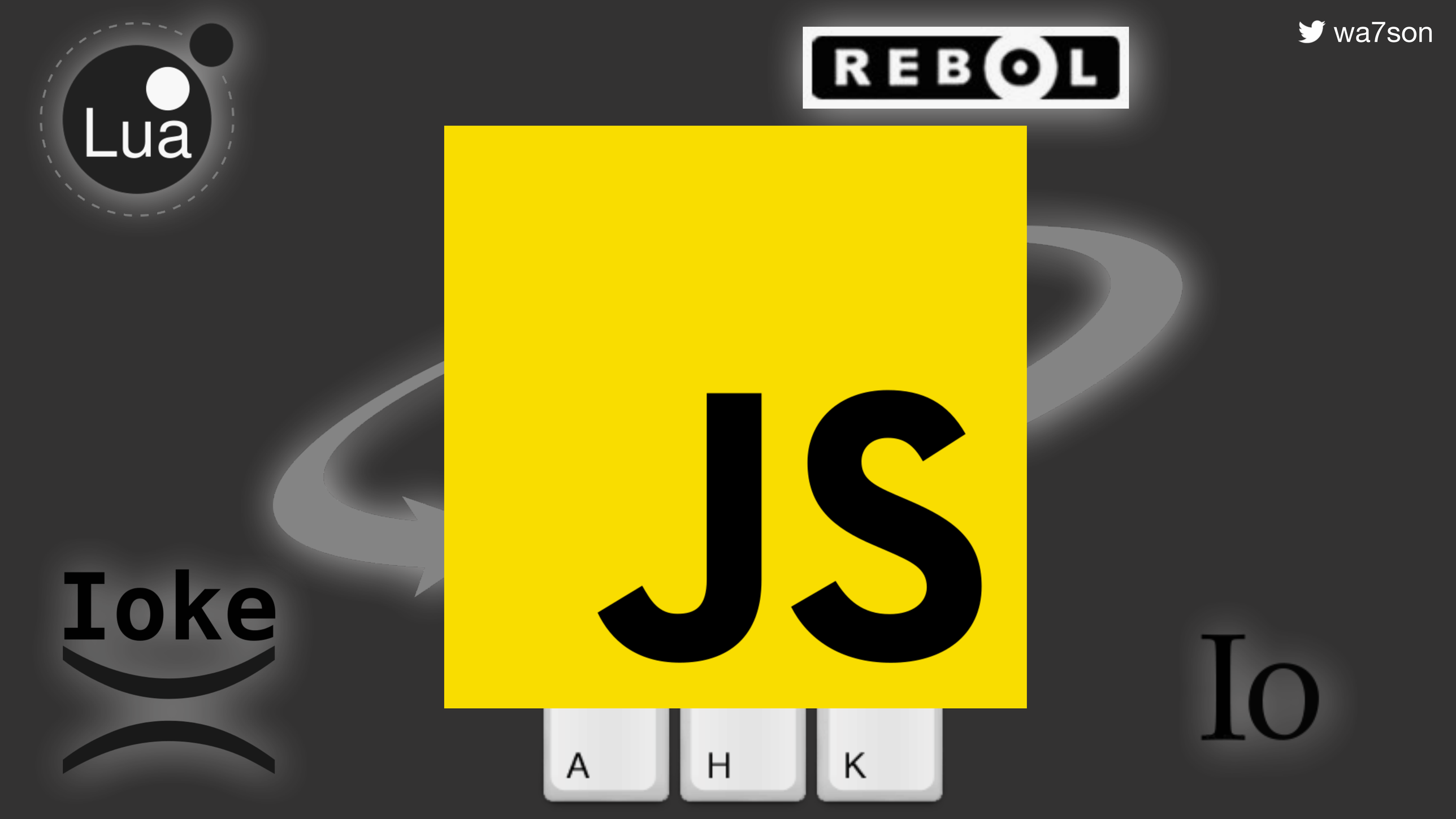
node


wa7son
JavaScript Prototypes
Behind the Scenes

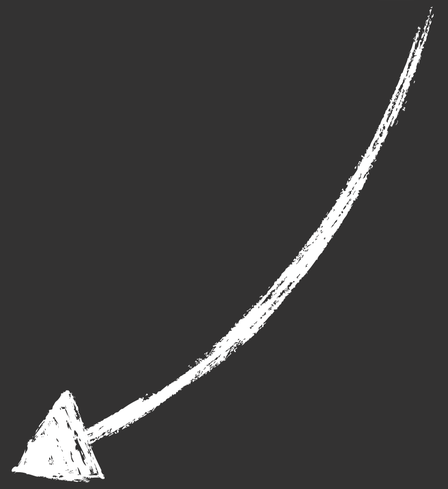
In the beginning...

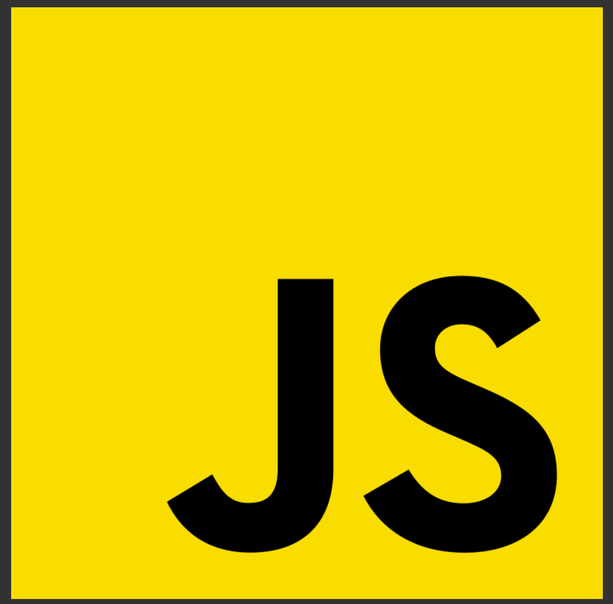
wa7son

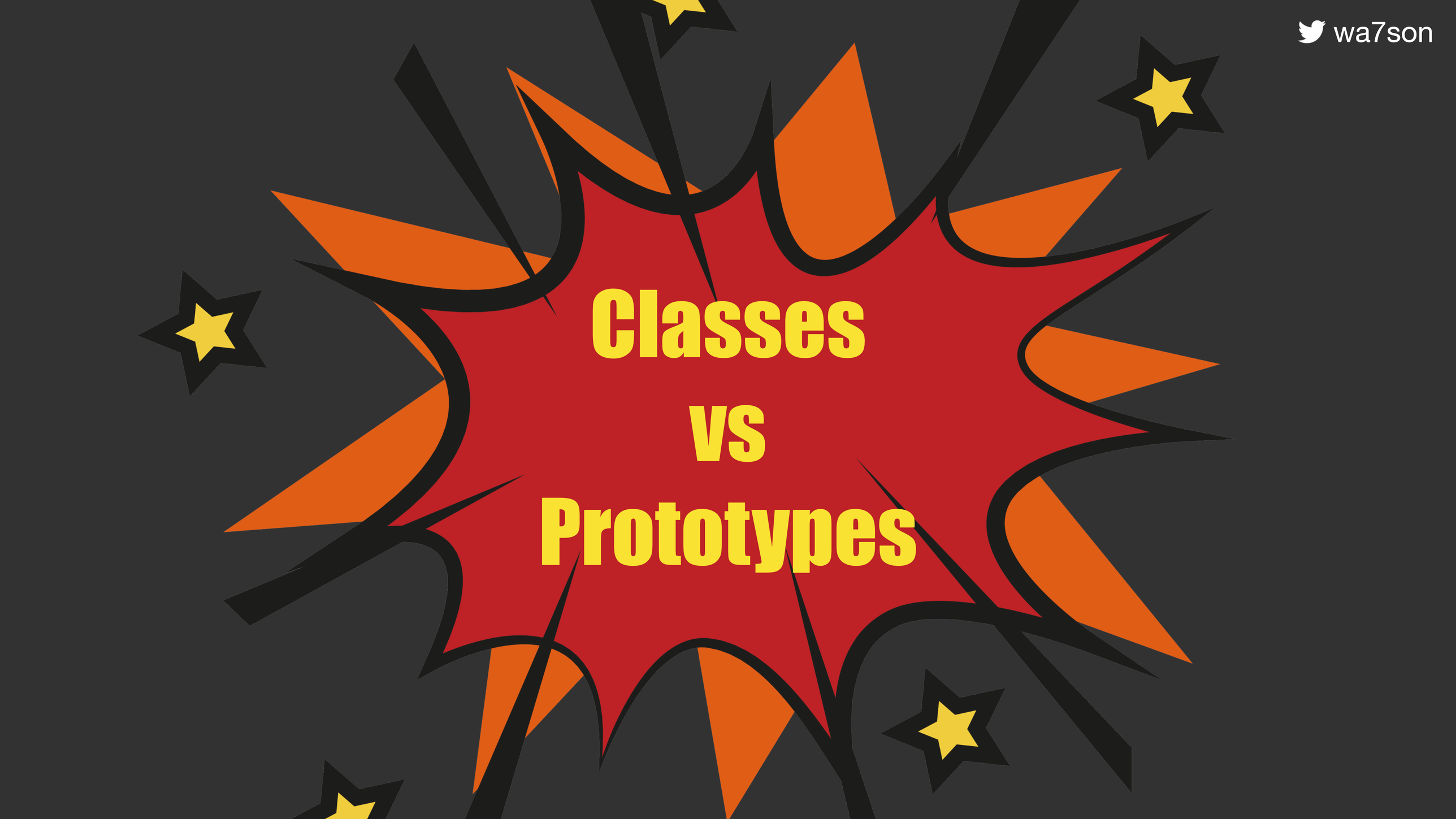
Io

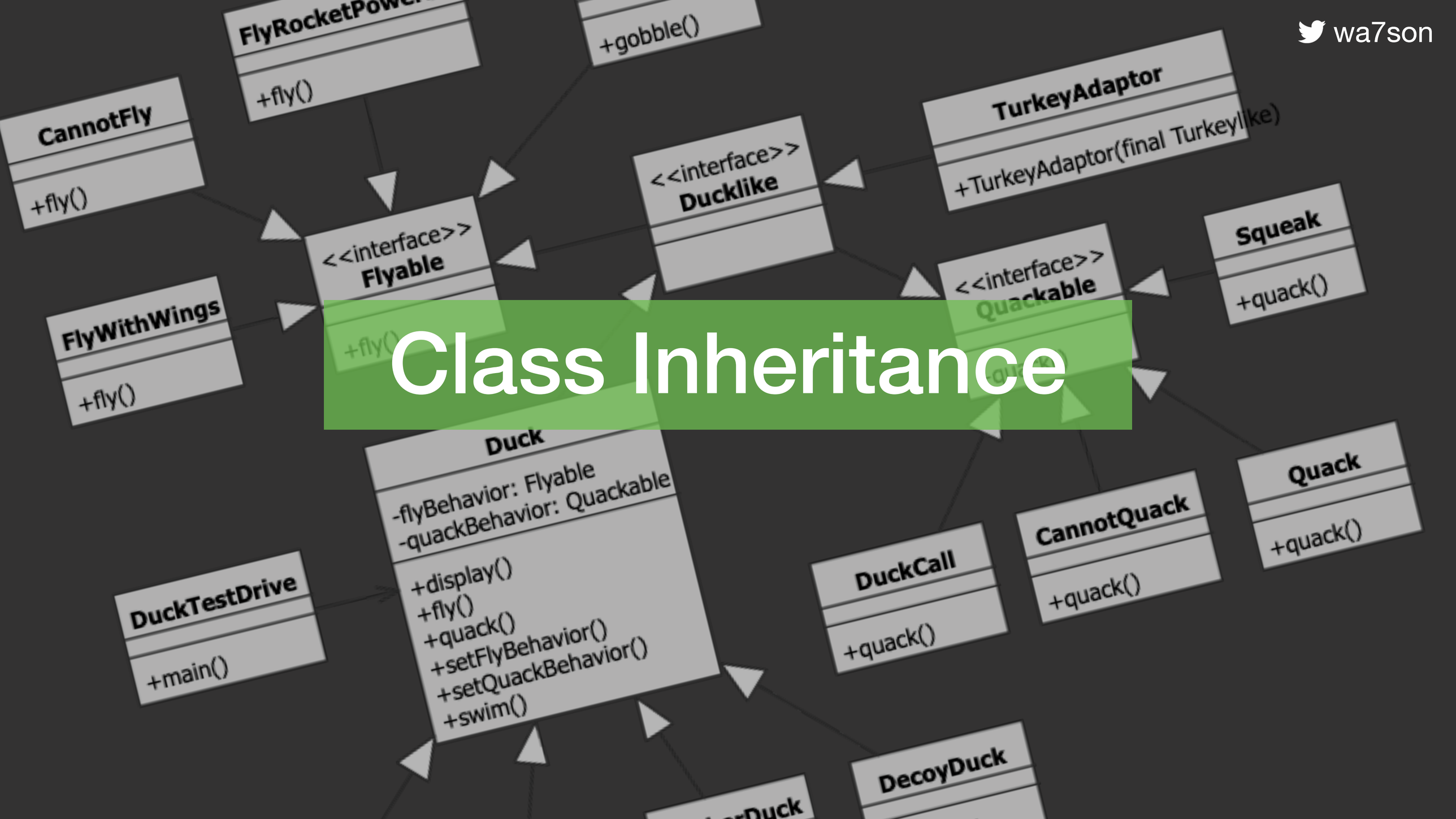

wa7son
Lua
REBOL
JS
Ioke
Io
A
H
K

wa7son






wa7son
Classes
vs
Prototypes



Class Inheritance

Image

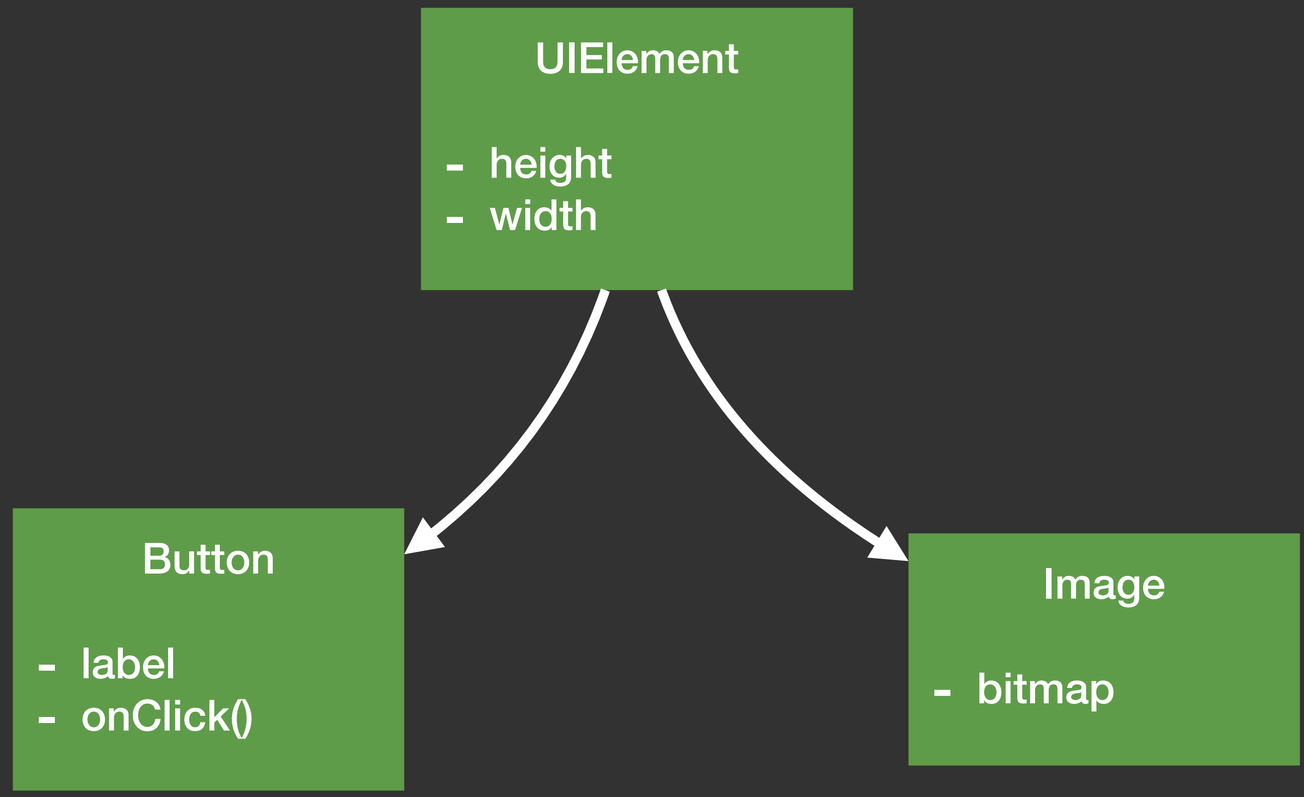
- height
- width
- bitmap

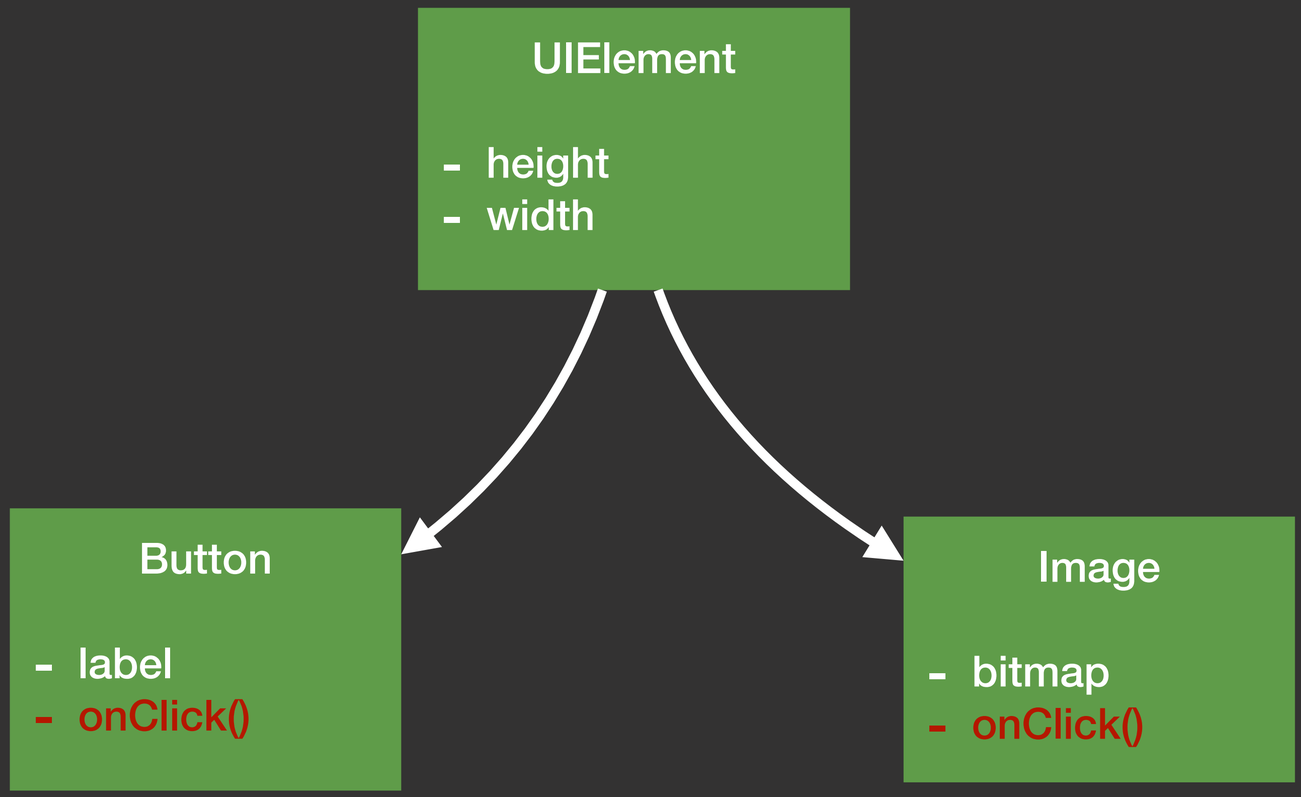
Button

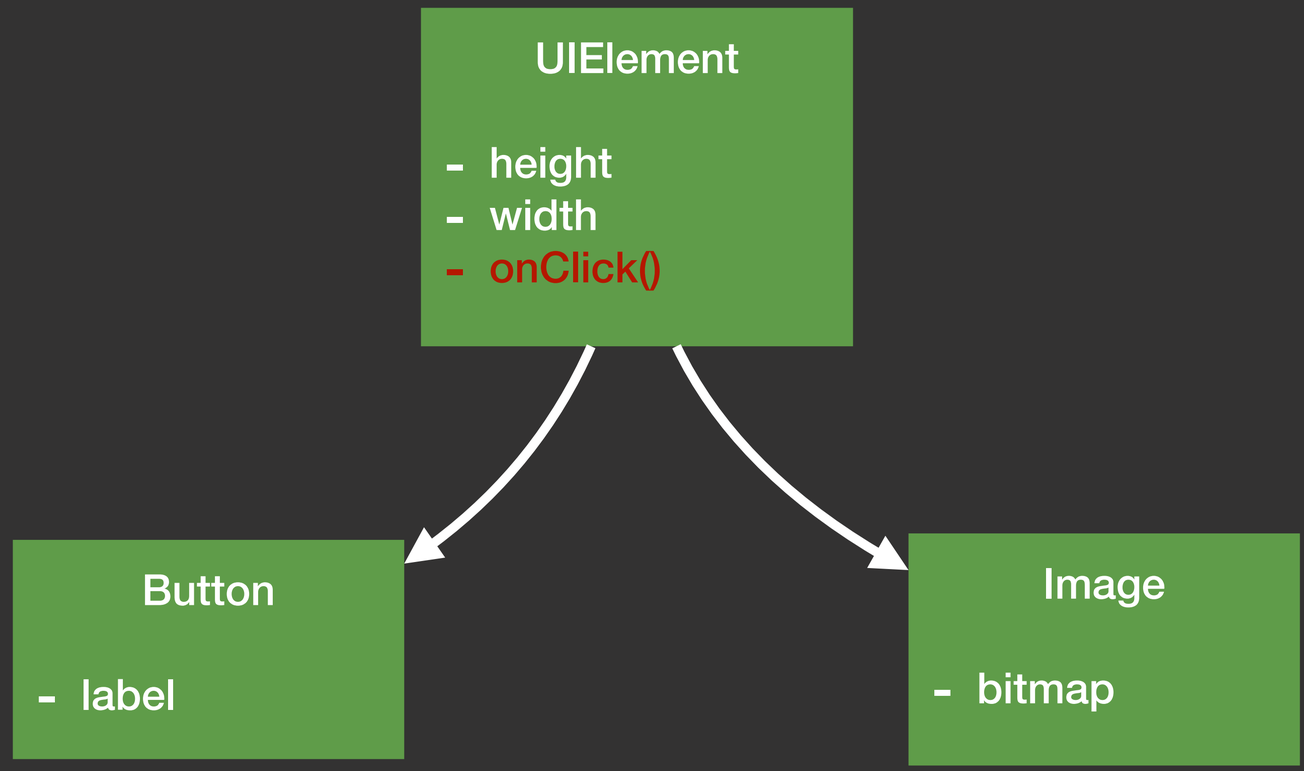
- height
- width
- label
- onClick()

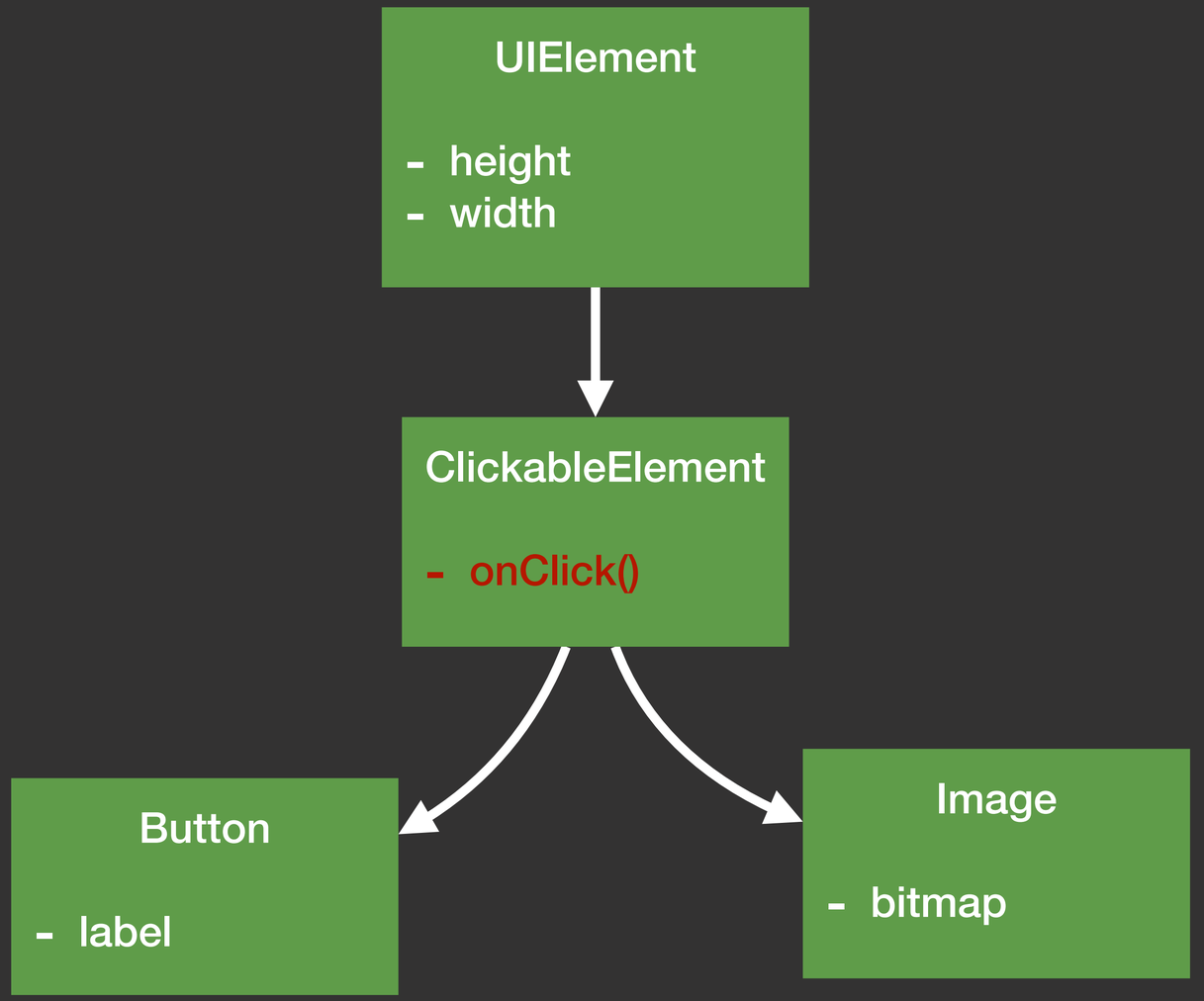
Image

- height
- width
- bitmap


UIElement
- height
- width
Button
- label
- onClick()
Image
- bitmap


UIElement
- height
- width
Button
- label
- onClick()
Image
- bitmap
- onClick()


UIElement
- height
- width
- onClick()
Button
- label
Image
- bitmap


UIElement
- height
- width
ClickableElement
- onClick()
Button
- label
Image
- bitmap


wa7son
Banana / Gorilla Problem
“You wanted a banana but what you got
was a gorilla holding the banana
and the entire jungle”
- Joe Armstrong, the creator of Erlang

Prototypes

Prototypes

Delegates

- Shared properties / methods
- Don’t find what you’re looking for on the object? Look at its prototype!
- Conserve memory

Java'isk Object Creation

```
function Cat (name) {
  this.name = name
}

Cat.prototype.talk = function () {
  return `Meow, I'm ${this.name}!`
}

const whiskers = new Cat('Mr. Whiskers')

whiskers.talk() // Meow, I'm Mr. Whiskers!
```

Cleaner Prototype Object Creation

```
const proto = {
  talk () {
    return `Meow, I'm ${this.name}!`
  }
}

const whiskers = Object.create(proto)
whiskers.name = 'Mr. Whiskers'

whiskers.talk() // Meow, I'm Mr. Whiskers!
```

Prototypes

Delegates

- Shared properties / methods
- Don’t find what you’re looking for on the object? Look at its prototype!
- Conserve memory

Cloning / Concatenation

- Default state / mixins

Mixing it up

```
class Image extends UIElement {
  // ...
}

const myImage = new Image()

Object.assign(myImage, Clickable)
```

Prototypes

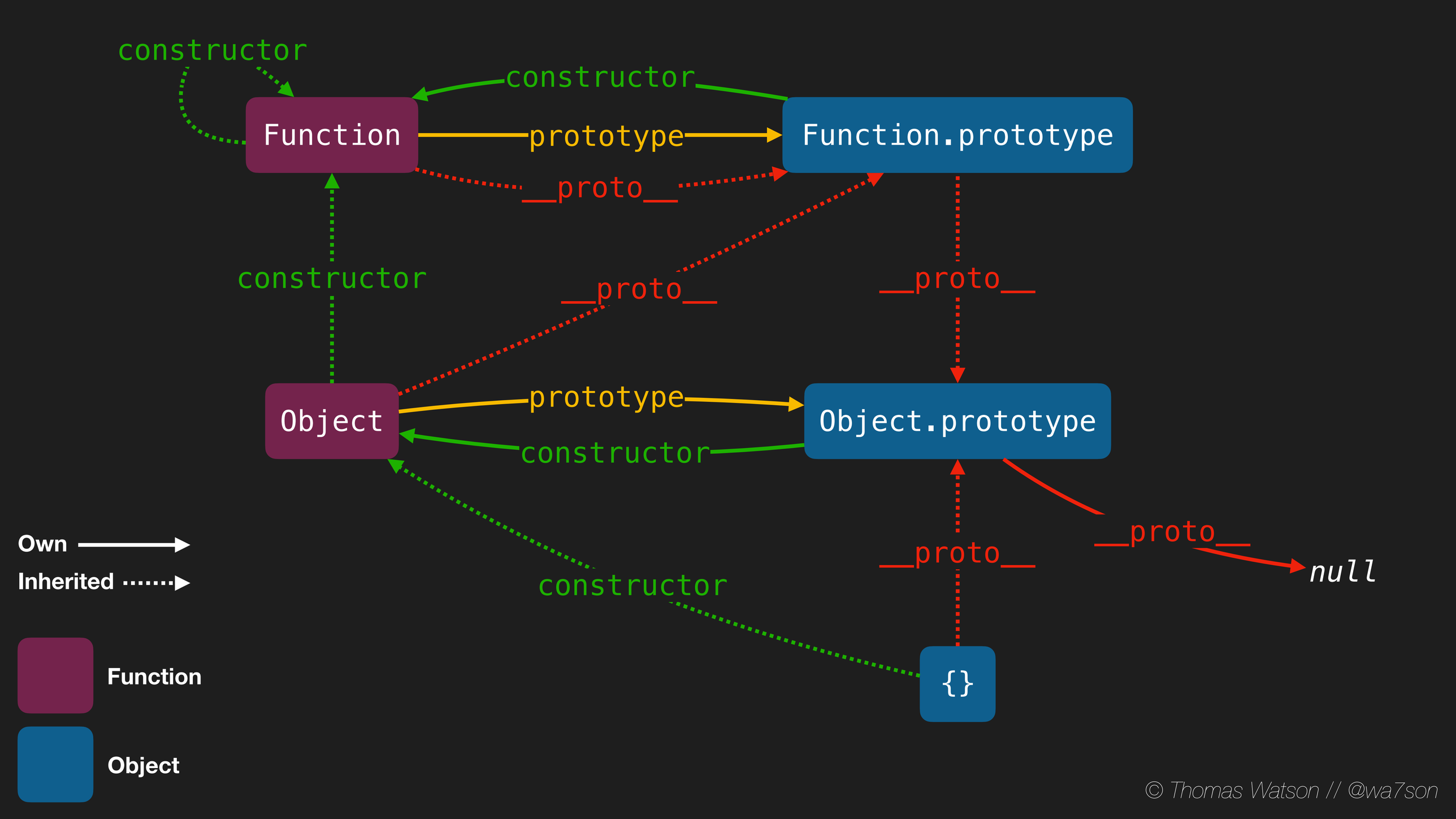
Delegates

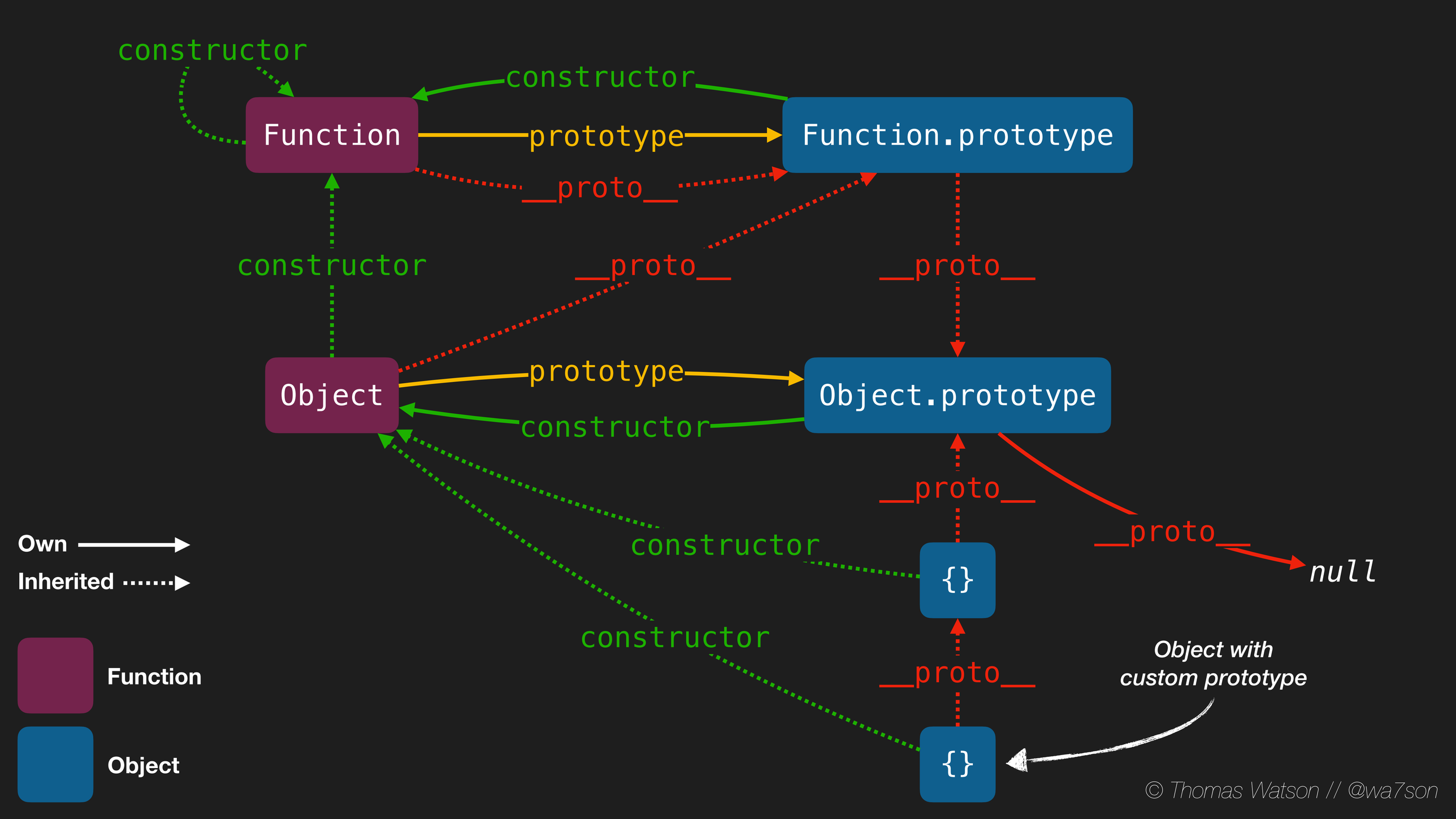
- Shared properties / methods
- Don’t find what you’re looking for on the object? Look at its prototype!
- Conserve memory

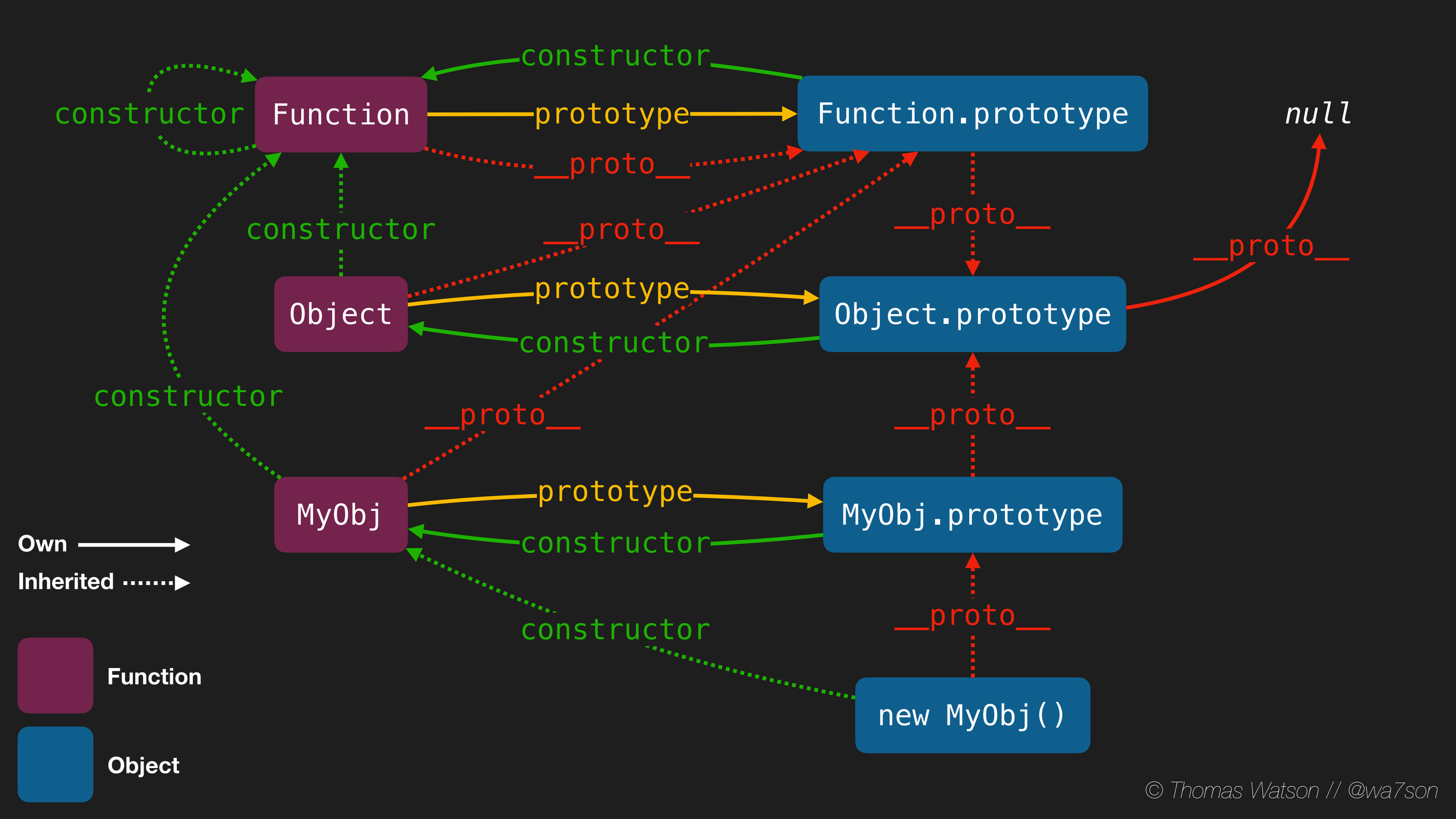
Cloning / Concatenation

- Default state / mixins

Time to get our hands dirty…








wa7son
Prototype Pollution

How to clone an object

```
function merge (target, source) {
  for (const [key, value] of Object.entries(source)) {
    target[key] = value
  }
  return target
}
```

How to clone an object

```
function merge (target, source) {
  for (const [key, value] of Object.entries(source)) {
    target[key] = value
  }
  return target
}

merge({ a: 1 }, { b: 2 })  // { a: 1, b: 2 }
```

How to clone an object

```
function merge (target, source) {
  for (const [key, value] of Object.entries(source)) {
    target[key] = value
  }
  return target
}
```

✅

```
merge({ a: 1 }, { b: 2 })  // { a: 1, b: 2 }
```

❌

```
merge(
  { a: { foo: 1 } },
  { a: { bar: 2 } }
)                          // { a: { bar: 2 } }
```

How to deep-clone an object

```
function deepMerge (target, source) {
  for (const [key, value] of Object.entries(source)) {
    if (isObject(value)) {
      const clone = deepMerge({}, value)
      if (isObject(target[key])) {
        deepMerge(target[key], clone)
      } else {
        target[key] = clone
      }
    } else {
      target[key] = value
    }
  }
  return target
}
```

How to deep-clone an object

```
function deepMerge (target, source) {
  for (const [key, value] of Object.entries(source)) {
    if (isObject(value)) {
      const clone = deepMerge({}, value)
      if (isObject(target[key])) {
        deepMerge(target[key], clone)
      } else {
        target[key] = clone
      }
    } else {
      target[key] = value
    }
  }
  return target
}
```

How to deep-clone an object

```
function deepMerge (target, source) {
  for (const [key, value] of Object.entries(source)) {
    if (isObject(value)) {
      const clone = deepMerge({}, value)
      if (isObject(target[key])) {
        deepMerge(target[key], clone)
      } else {
        target[key] = clone
      }
    } else {
      target[key] = value
    }
  }
  return target
}
```

How to deep-clone an object

```
function deepMerge (target, source) {
  for (const [key, value] of Object.entries(source)) {
    if (isObject(value)) {
      const clone = deepMerge({}, value)
      if (isObject(target[key])) {
        deepMerge(target[key], clone)
      } else {
        target[key] = clone
      }
    } else {
      target[key] = value
    }
  }
  return target
}
```

How to deep-clone an object

```
function deepMerge (target, source) {
  for (const [key, value] of Object.entries(source)) {
    if (isObject(value)) {
      const clone = deepMerge({}, value)
      if (isObject(target[key])) {
        deepMerge(target[key], clone)
      } else {
        target[key] = clone
      }
    } else {
      target[key] = value
    }
  }
  return target
}
```

```
deepMerge(
  { a: { foo: 1 } },
  { a: { bar: 2 } }
) // { a: { foo: 1, bar: 2 } }
```

The Attack

```
function isAdmin (user) {
  return user.admin === true
}
```

The Attack

```
function isAdmin (user) {
  return user.admin === true
}

const user = {}

isAdmin(user) // false
```

The Attack

```
function isAdmin (user) {
  return user.admin === true
}

const user = {}

isAdmin(user) // false

const payload = '{"__proto__":{"admin":true}}'
```

The Attack

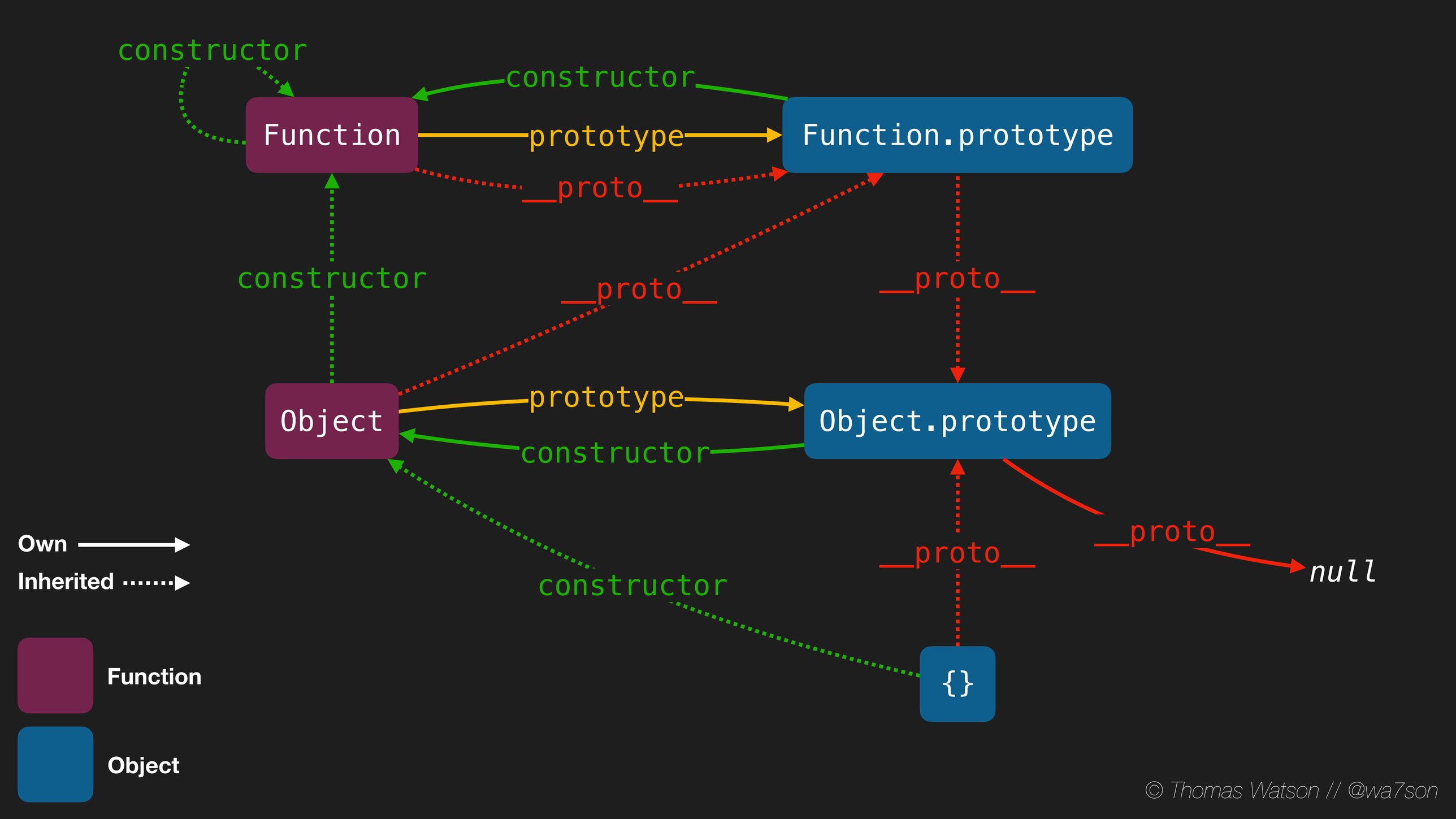
```
function isAdmin (user) {
  return user.admin === true
}

const user = {}

isAdmin(user) // false

const payload = '{"__proto__":{"admin":true}}'

deepMerge({}, JSON.parse(payload))
```

The Attack

```
function isAdmin (user) {
  return user.admin === true
}

const user = {}

isAdmin(user) // false

const payload = '{"__proto__":{"admin":true}}'

deepMerge({}, JSON.parse(payload))

isAdmin(user) // true
```

The Antidote

- Look out for `__proto__` in untrusted user input
- Look out for `constructor.prototype` in untrusted user input

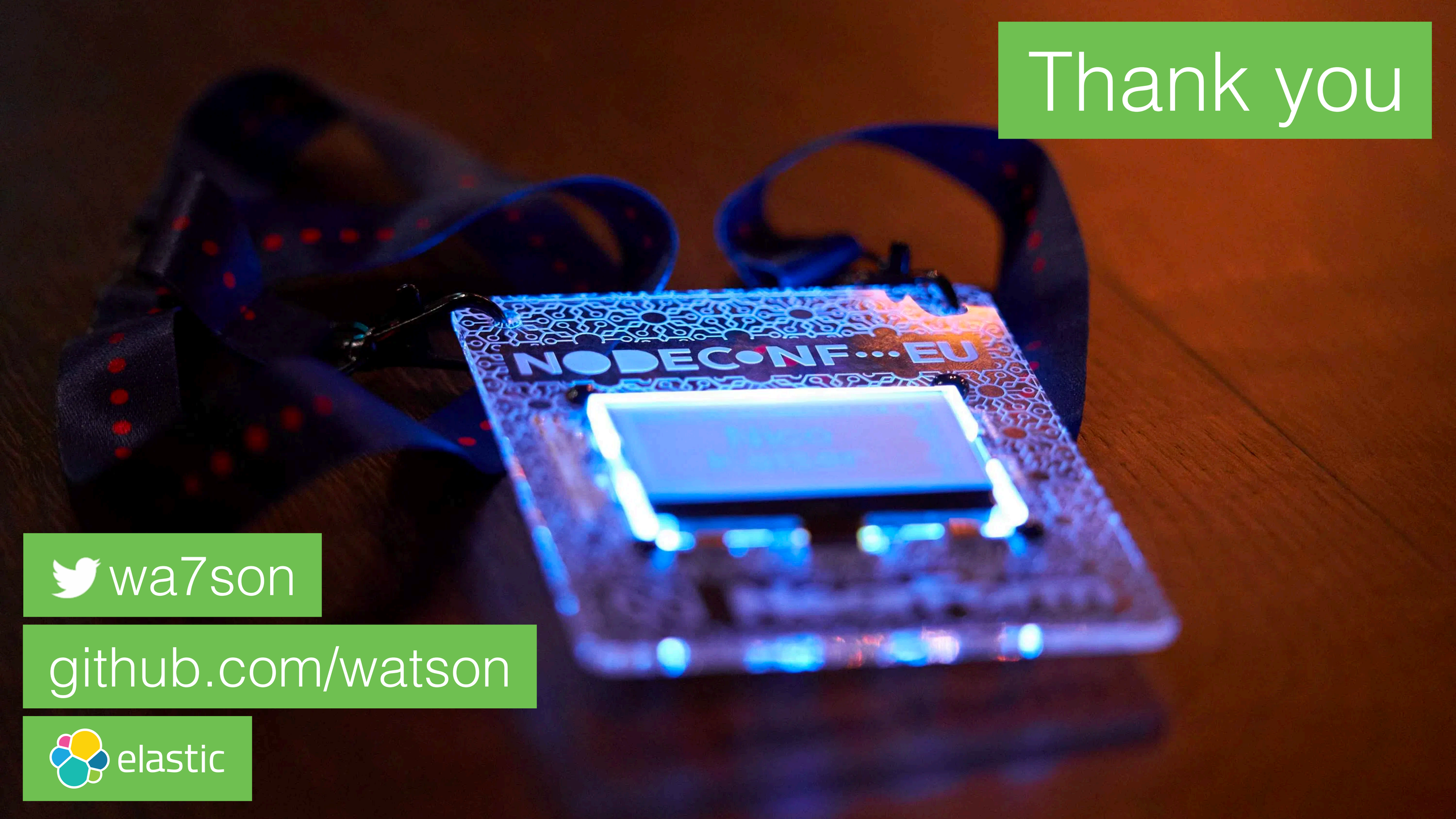


The Antidote

- Look out for `__proto__` in untrusted user input
- Look out for `constructor.prototype` in untrusted user input

```
npm install secure-json-parse
```

The Antidote



Thank you

wa7son

github.com/watson

elastic