Logical Assignment Operators

In a nutshell

- `a ||= b`
- `a &&= b`

Why?

Convenience mainly.

```
group[key] ||= 0
```

vs

```
group[key] = group[key] || 0
```

Nice with for loops, too

```
let flag = false

let flag2 = false

for (const x of array) {

  flag &&= check(x)

  flag2 ||= check(x)

}
```

Why?

- “Paves cow paths”
- Easily understandable
  - Mostly, we’ll talk about short-circuiting

But also

- Can you imagine yourself using ??=?
  - Sidestep null vs undefined arguments

- It’ll be inconsistent when it’s the only logical with assignment
  - Annoyingly so. WAT

Now, short circuiting

```
a ||= b
```

```
a = a || b // (simple)
```

```
a || (a = b) // (short-circuiting)
```

Short Circuiting, why?

- It’s nice
- Follows what Ruby and CoffeeScript do
  - Why fight it?

- There are things to consider like Proxy with defaults (like Ruby Hash)

Stage 1?