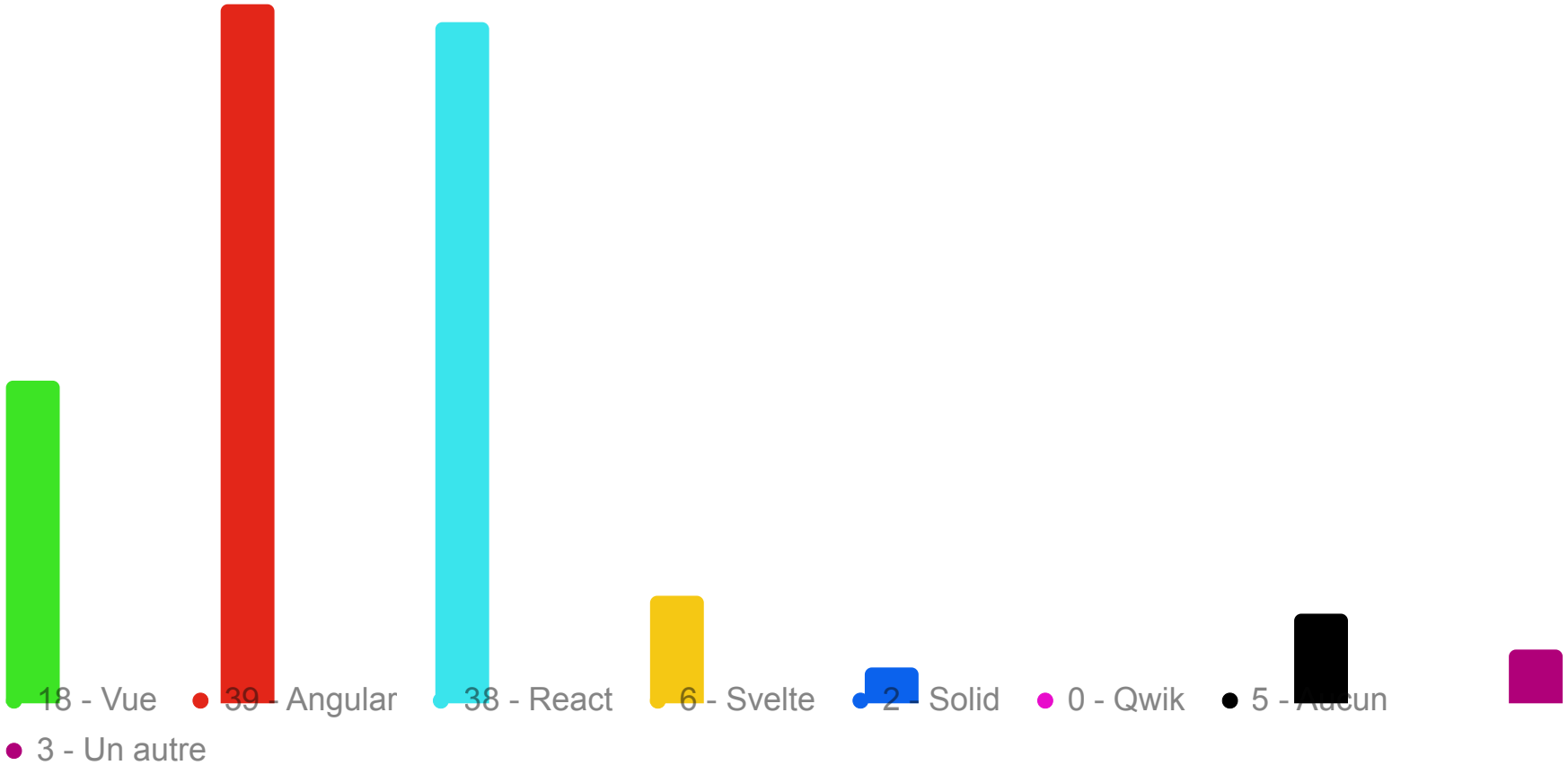
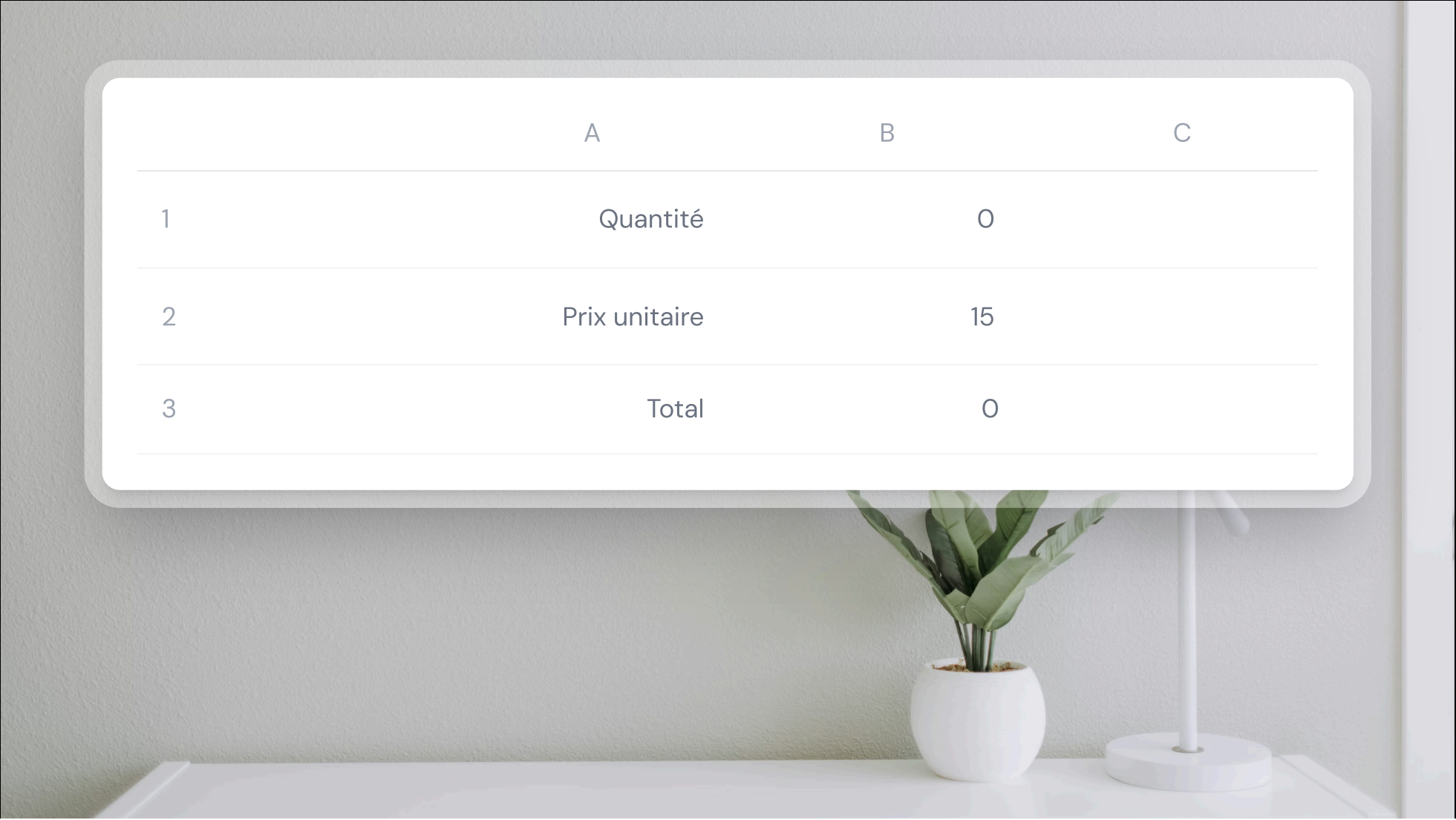
La réactivité et les signaux : démystifions la magie du frontend

Quel framework pour le frontend ?



Powered by Inalia

```
ui = fn(state)
```



	A	B	C
1	Quantité	0	
2	Prix unitaire	15	
3	Total	0	


import { computed, effect, signal } from 'alien-signals'
const quantity = signal(0)
const price = signal(15)
const total = computed(() => quantity() * price())
effect(() => {
console.log(`Total: ${total()}`)
})
//
Total: 0



Estéban Soubiran

Laravel Vite Vue Nuxt

Ingénieur chez Takima

2025: A Turning Point from Slow Steps to Clear Action

Starting a New Chapter: New Ambitions and a New Job

Markdown Exit Breaks the Rules With Async Rendering

Erase CLS by Automatically Optimizing Images with Vite

soubiran.dev @soubiran_ Barbapapazes Estéban S Barbapapazes Estéban's Open Source

```
import { computed, effect, signal } from 'alien-signals'

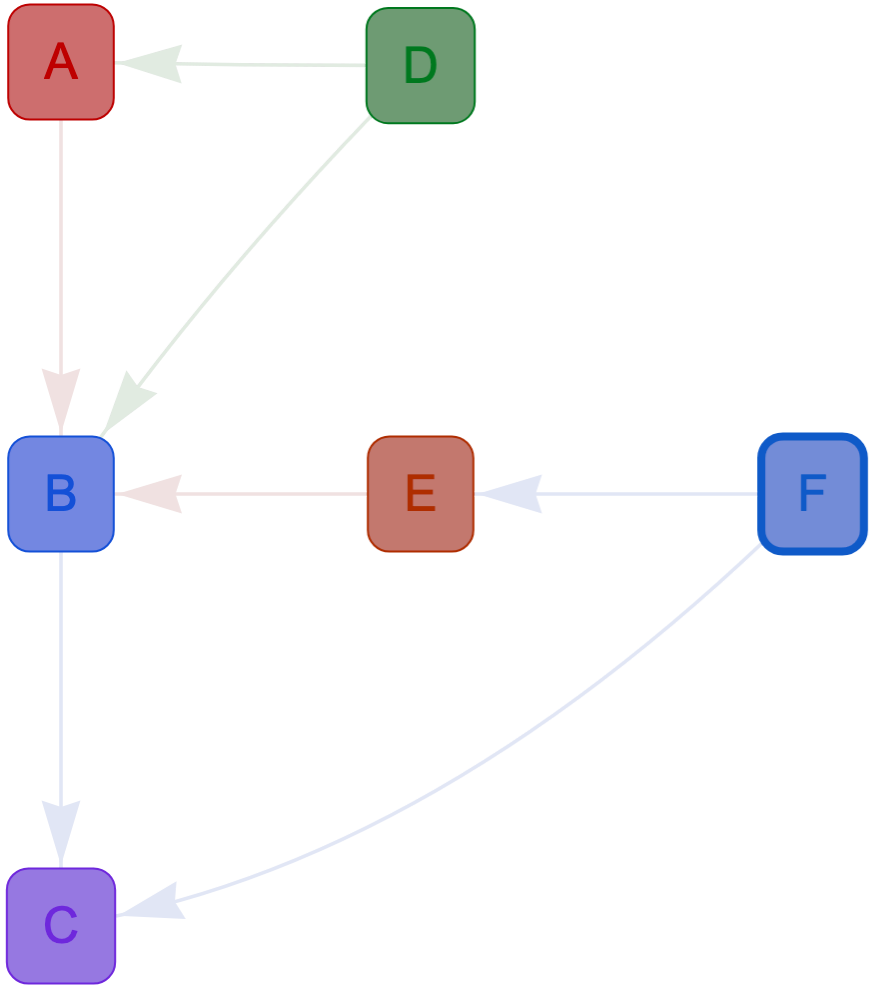
const quantity = signal(0)
const price = signal(15)

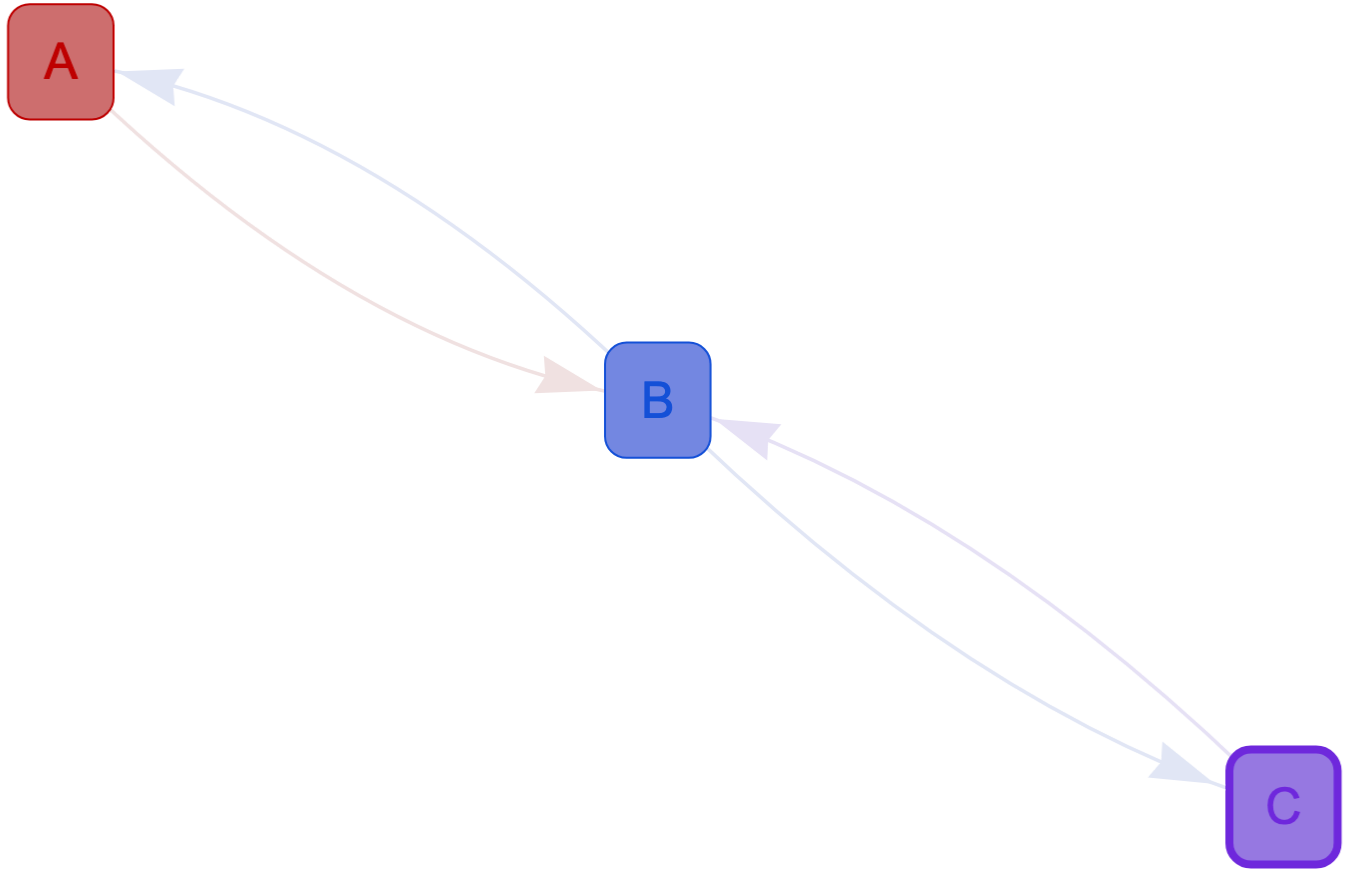
const total = computed(() => quantity() * price())

effect(() => {
  console.log(`Total: ${total()}`)
})

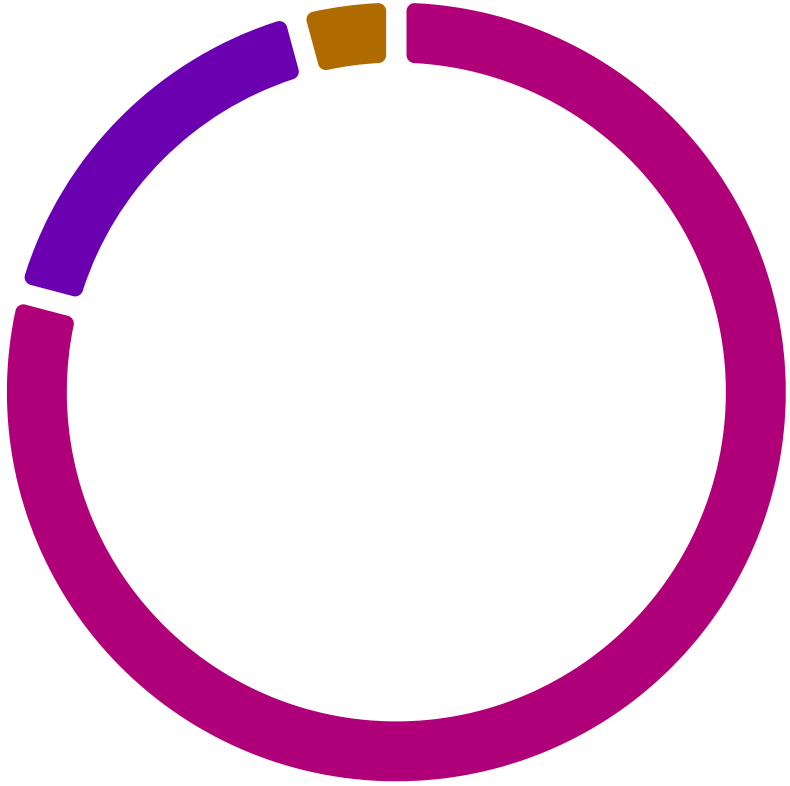
quantity(3)
```

```
Total: 0
Total: 45
```


A
D
B
E
F
C


A
B
C

Vous suivez encore ? 😲



- 44 - Oui, évidemment
- 9 - Oui, enfin je crois
- 2 - Non, là c'est trop

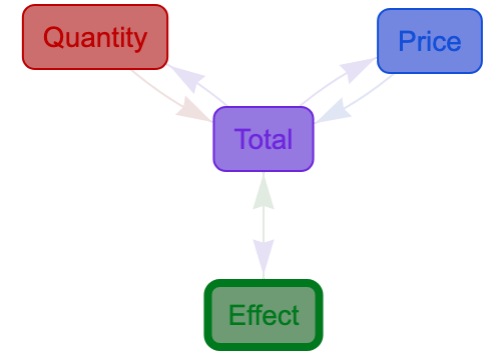
Powered by Inalia

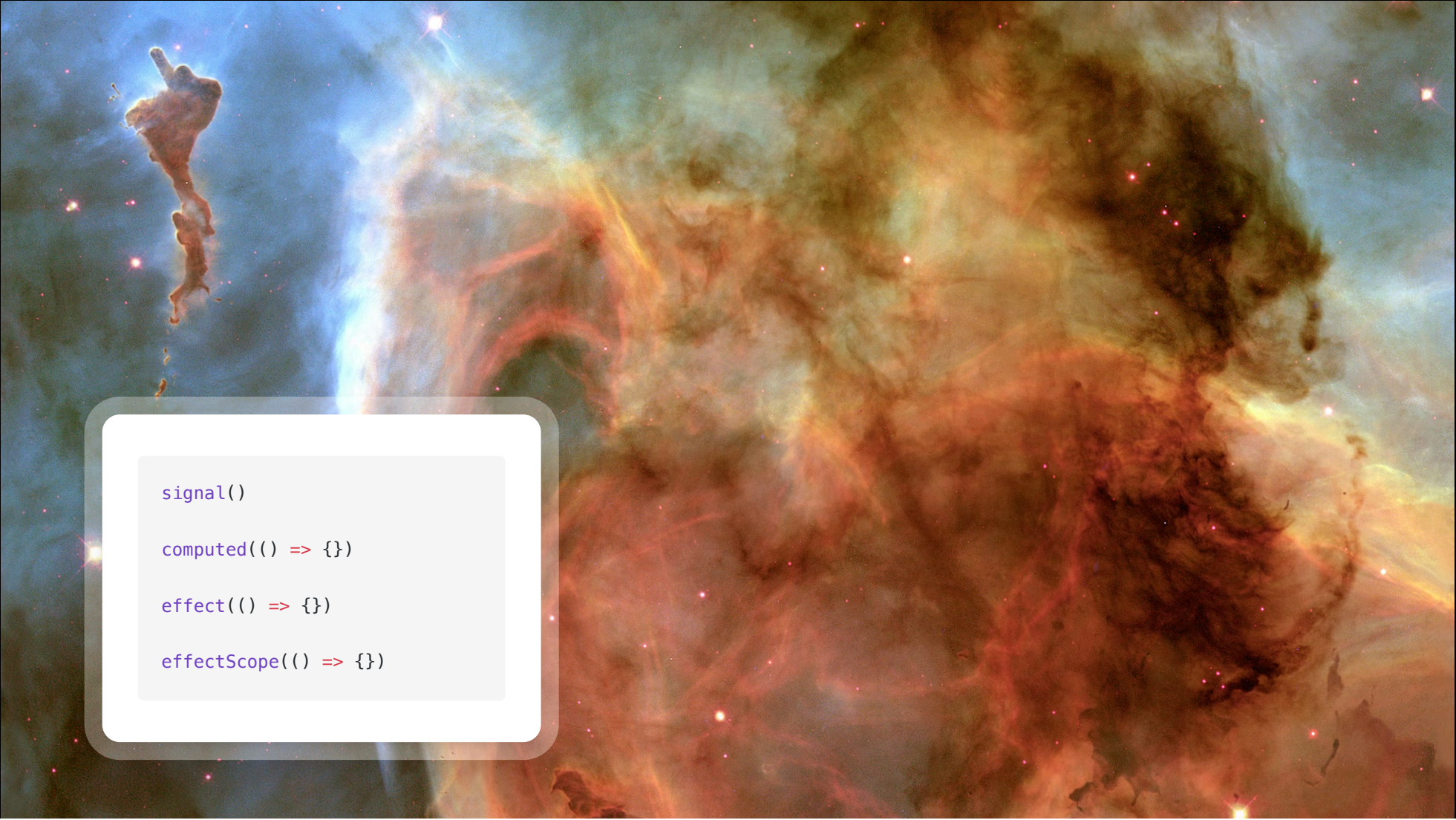
```
import { computed, effect, signal } from 'alien-signals'

const quantity = signal(0)
const price = signal(15)

const total = computed(() => quantity() * price())

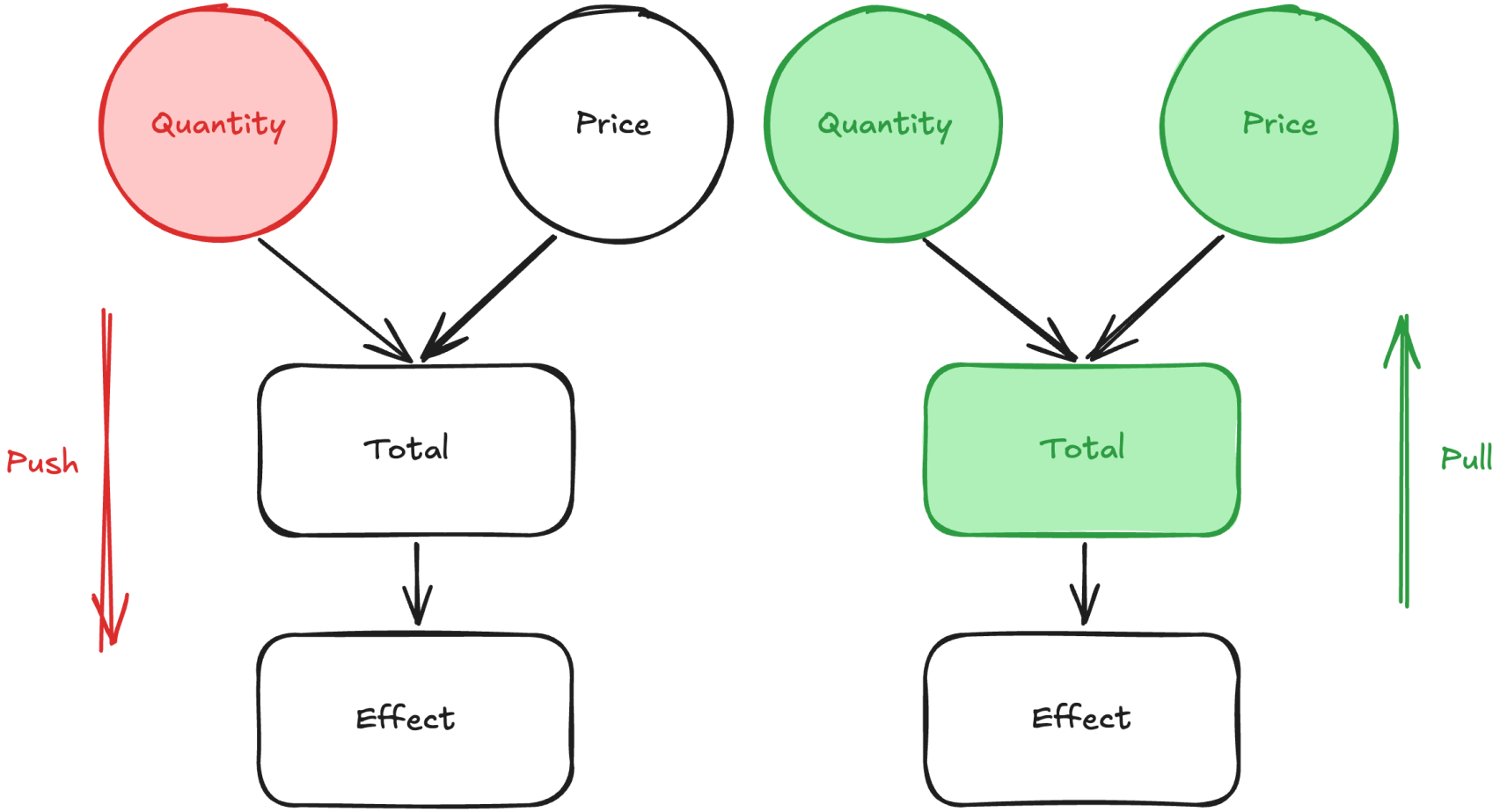
effect(() => {
  console.log(`Total: ${total()}`)
})
```



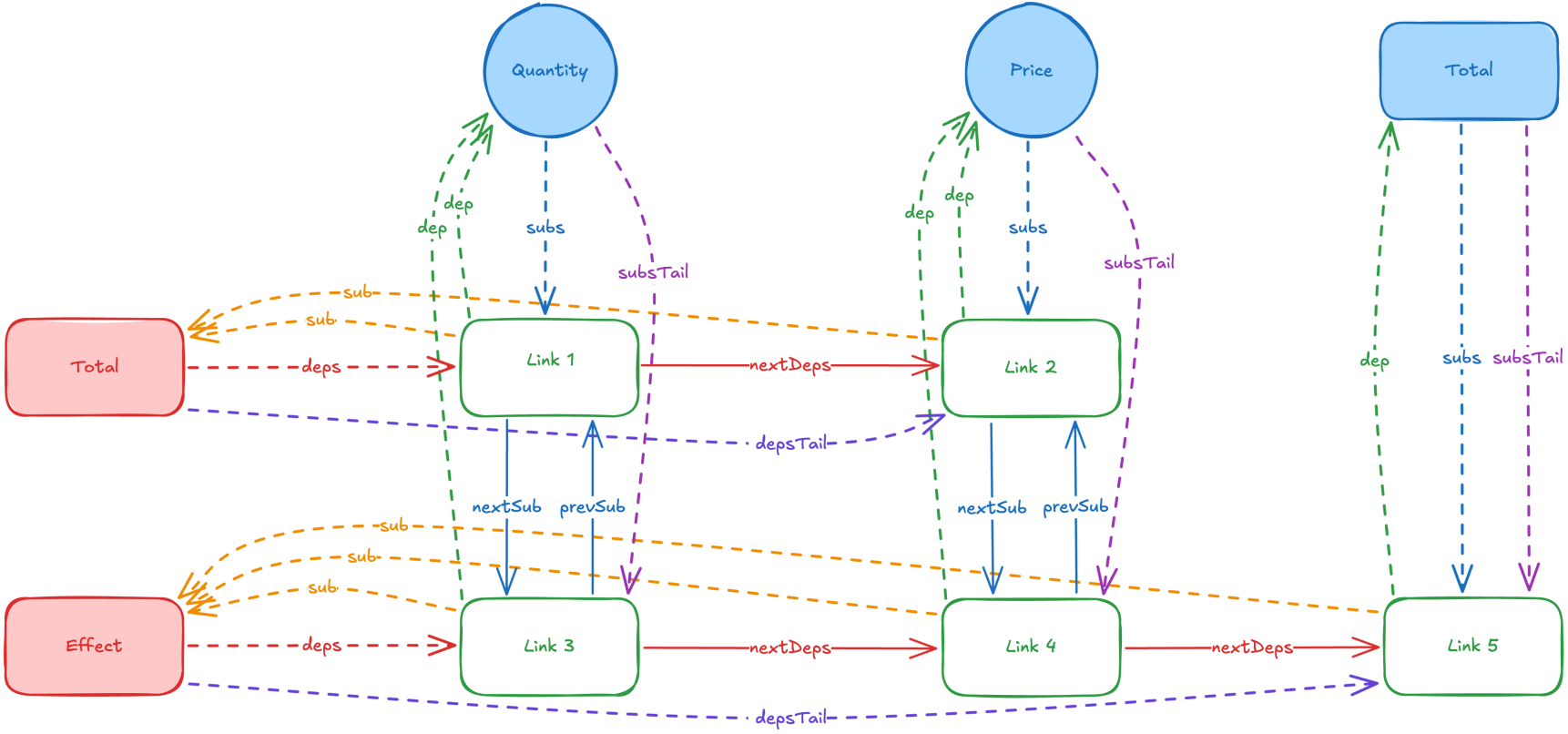

signal()
computed(() => {})
effect(() => {})
effectScope(() => {})

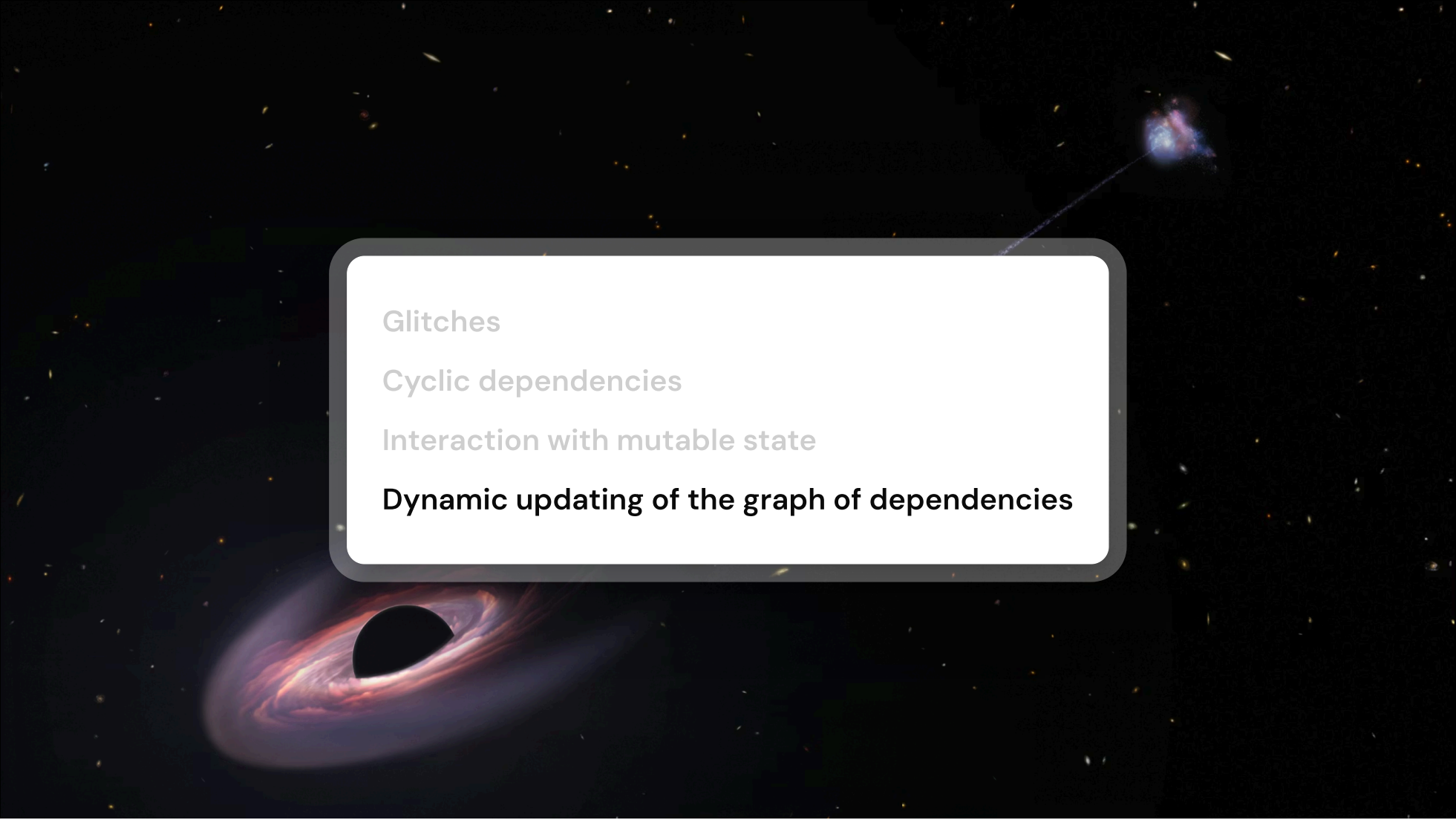


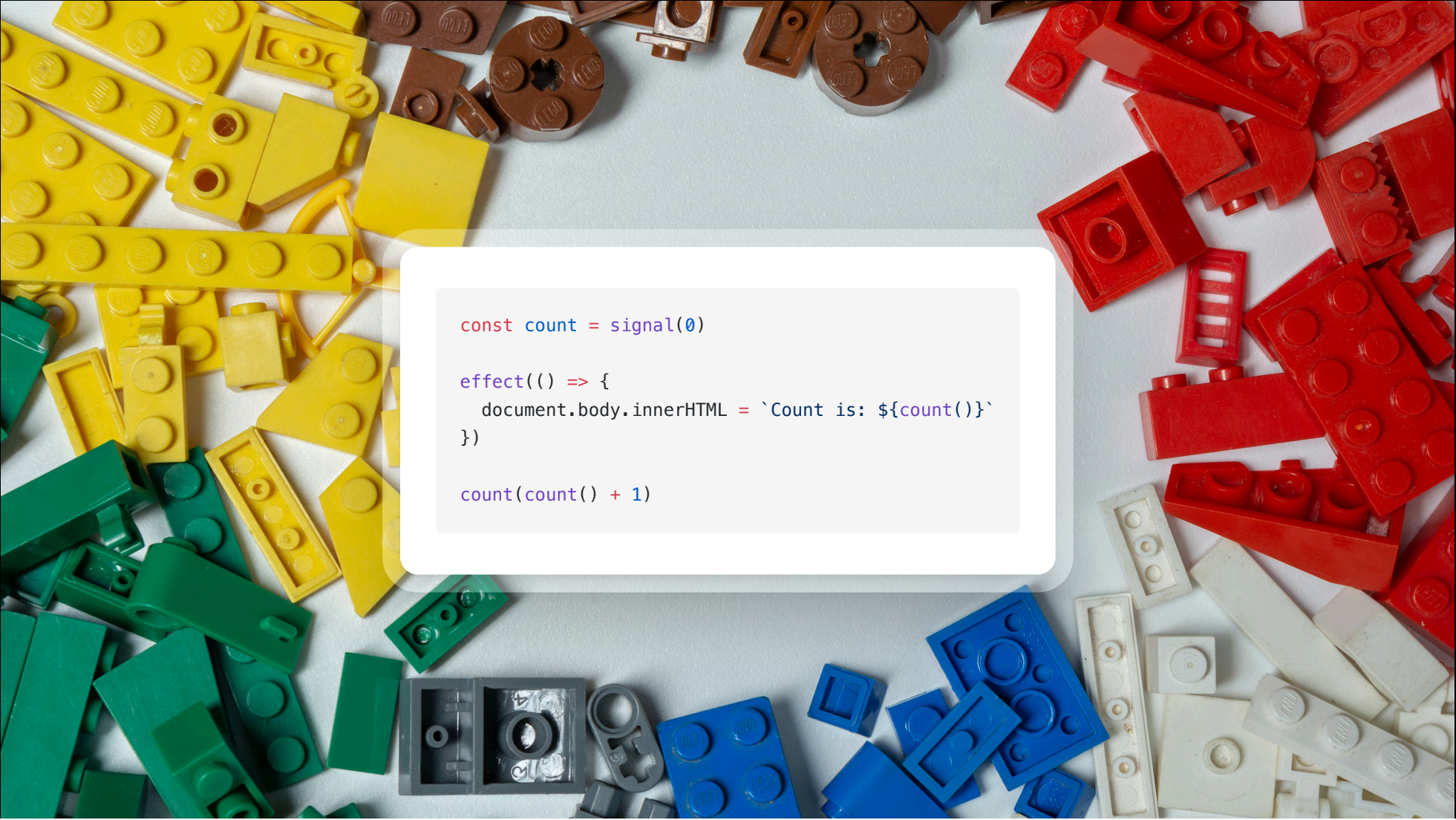
```ts
export function signal<T>(): WriteableSignal<T | undefined>
export function signal<T>(oldValue: T): WriteableSignal<T>
export function signal<T>(oldValue?: T): WriteableSignal<T | undefined> {
  return signalGetterSetter.bind({
    currentValue: oldValue,
    subs: undefined,
    subsTail: undefined,
  }) as WriteableSignal<T | undefined>
}
```


Quantity
Price
Total
Effect
Push
Quantity
Price
Total
Effect
Pull

```ts
function signalGetterSetter<T>(this: Signal<T>, ...value: [T]): T | void {
  if (value.length) {
    if (this.currentValue !== (this.currentValue = value[0])) {
      const subs = this.subs
      if (subs !== undefined) {
        propagate(subs)
        if (!batchDepth) {
          processEffectNotifications()
        }
      }
    }
  }
  else {
    if (activeSub !== undefined) {
      link(this, activeSub)
    }
    return this.currentValue
  }
}
```


Quantity
Price
Total
dep
dep
subs
subsTail
dep
dep
subs
subsTail
dep
subs
subsTail
sub
sub
Total
deps
Link 1
nextDeps
Link 2
depsTail
nextSub
prevSub
nextSub
prevSub
sub
sub
sub
Effect
deps
Link 3
nextDeps
Link 4
nextDeps
Link 5
depsTail


Glitches
Cyclic dependencies
Interaction with mutable state
Dynamic updating of the graph of dependencies


const count = signal(0)
effect(() => {
document.body.innerHTML = `Count is: ${count()}`
})
count(count() + 1)


instance.scope.on()
const effect = (instance.effect = new ReactiveEffect(componentUpdateFn))
instance.scope.off()

Estéban Soubiran
esteban@soubiran.dev

Looking for more?

- Explore the **Alien Signals** source code
- Watch **Reactivity across frameworks**
- Read **Unveiling the Magic: Exploring Reactivity Across Various Frameworks**
- Stay curious and keep learning!