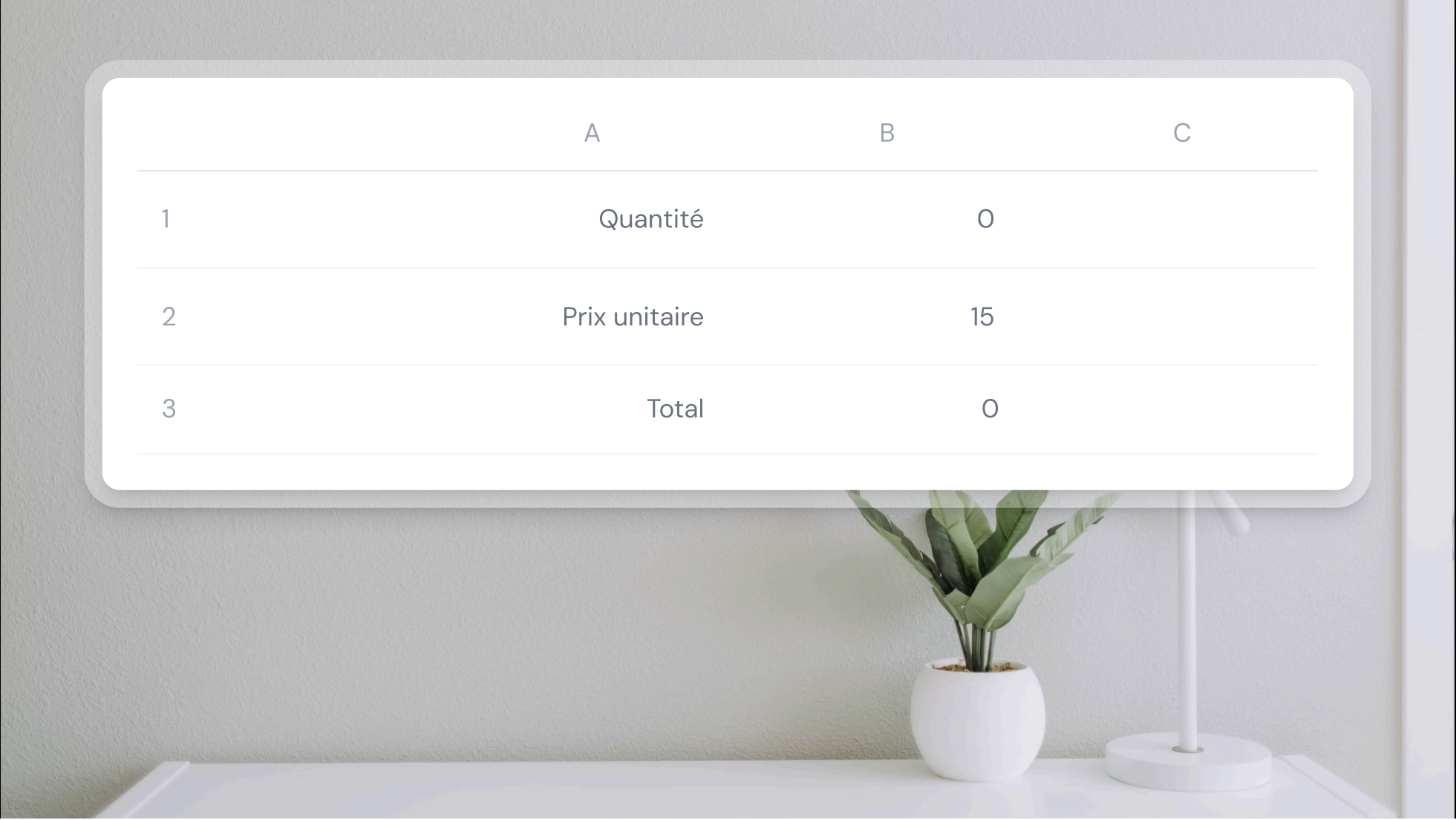
La réactivité et les signaux : démystifions la magie du frontend

```
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

Agent Builder chez Takima

Laravel | Vite | Vue | Nuxt | Java | Spring

Devoxx France 2026: Prioritizing People Over Talks

The Unexpected Rollercoaster of Feelings at MiXiT 2026

From Idea to Victory: the 2026 Mistral AI Paris Hackathon

2025: A Turning Point from Slow Steps to Clear Action

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

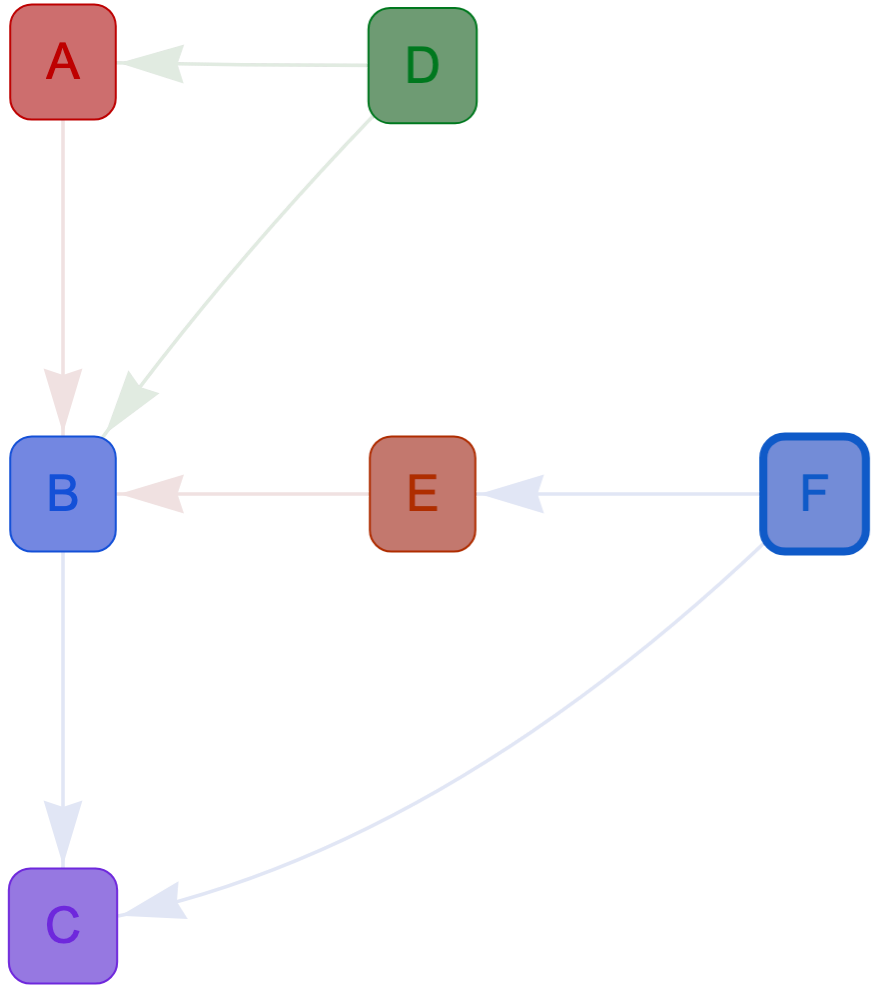
const quantity = signal(0)
const price = signal(15)

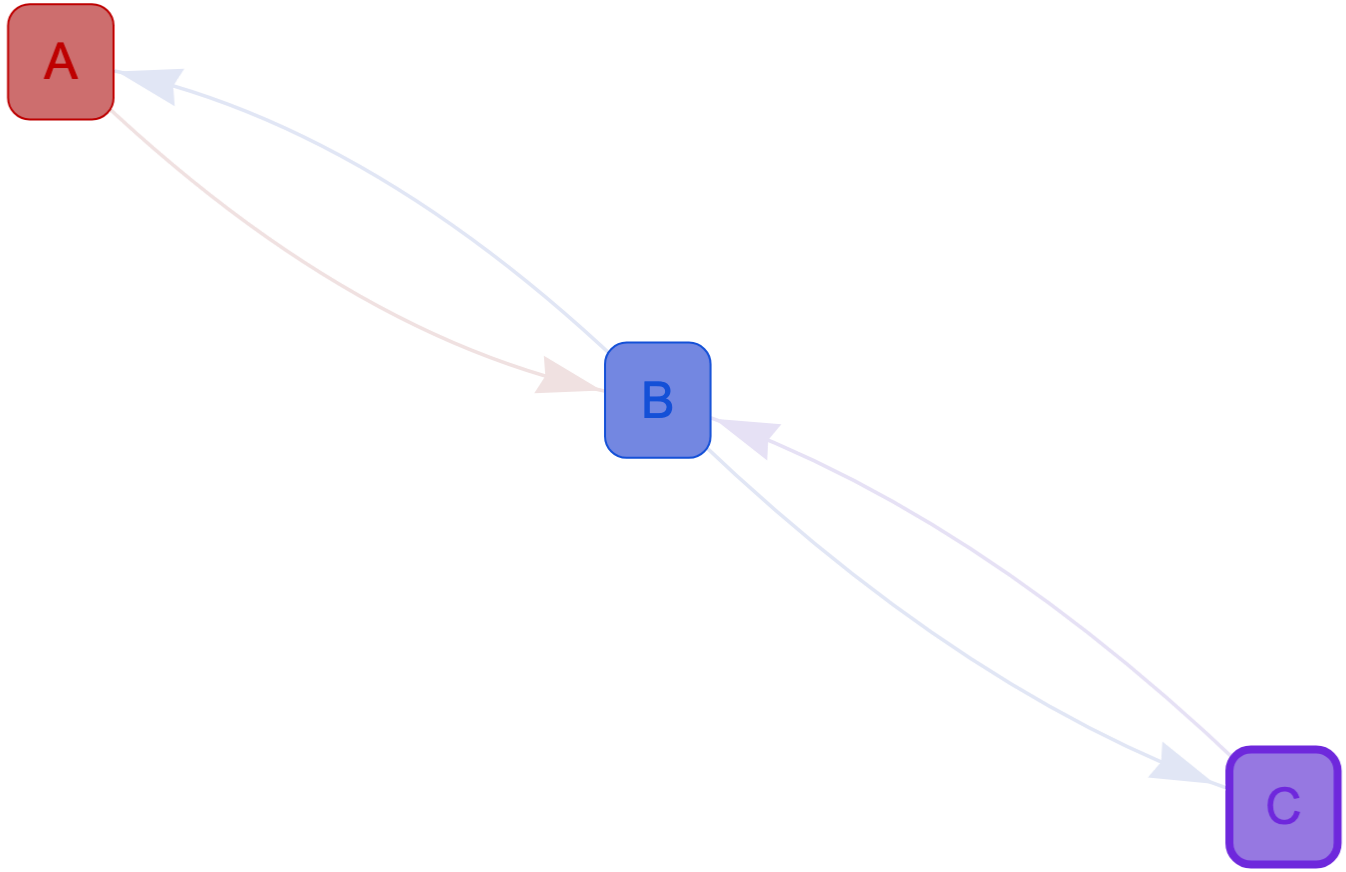
const total = computed(() => quantity() * price())

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

quantity(3)
```

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


A
D
B
E
F
C

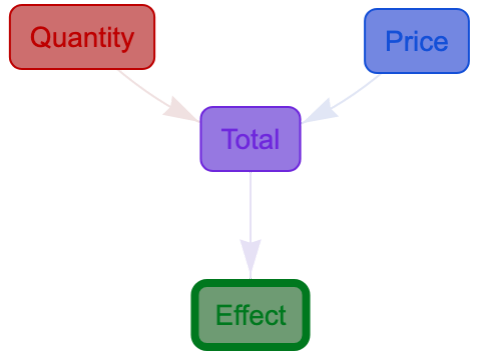

A
B
C

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

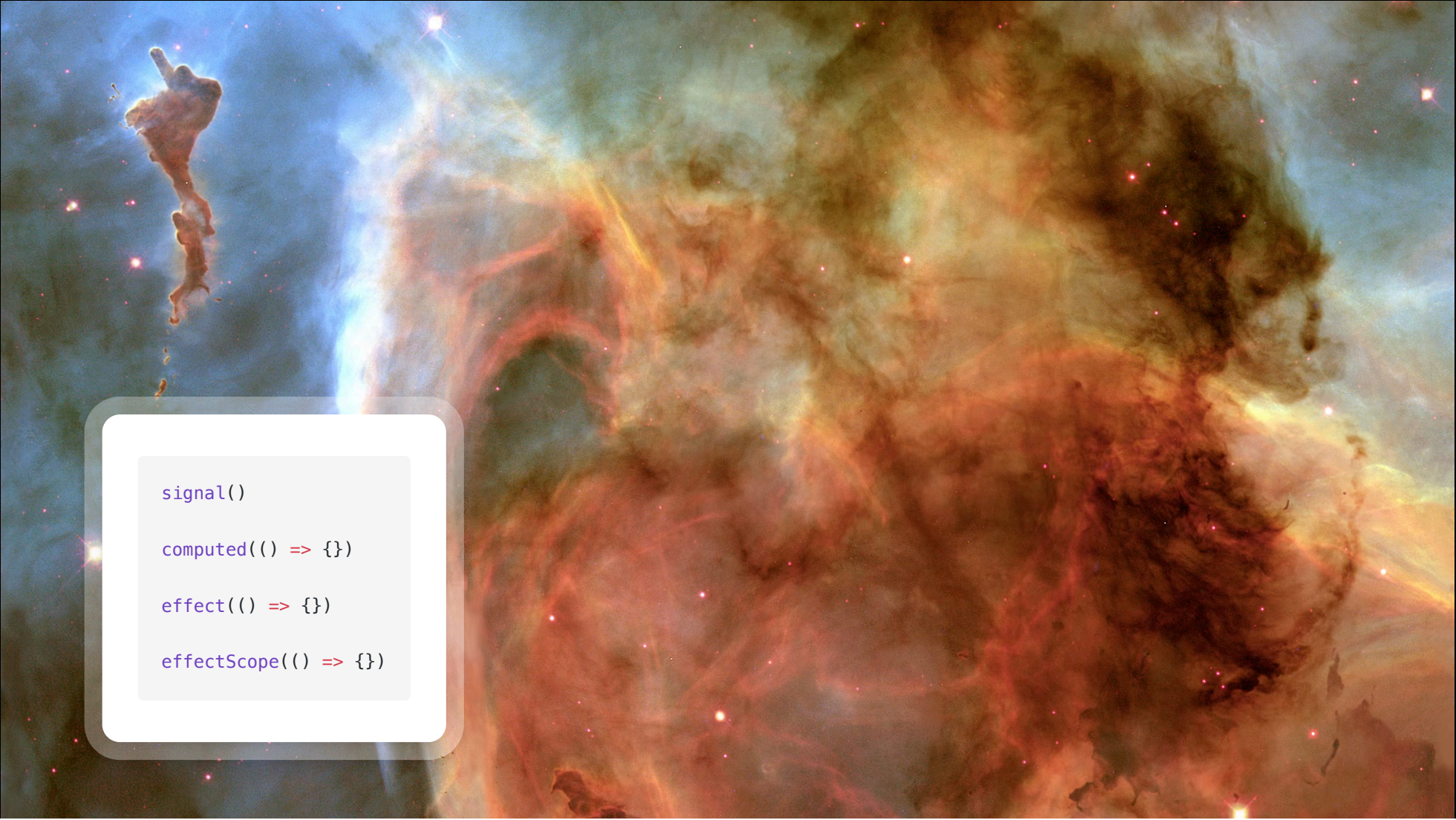
const quantity = signal(0)
const price = signal(15)

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

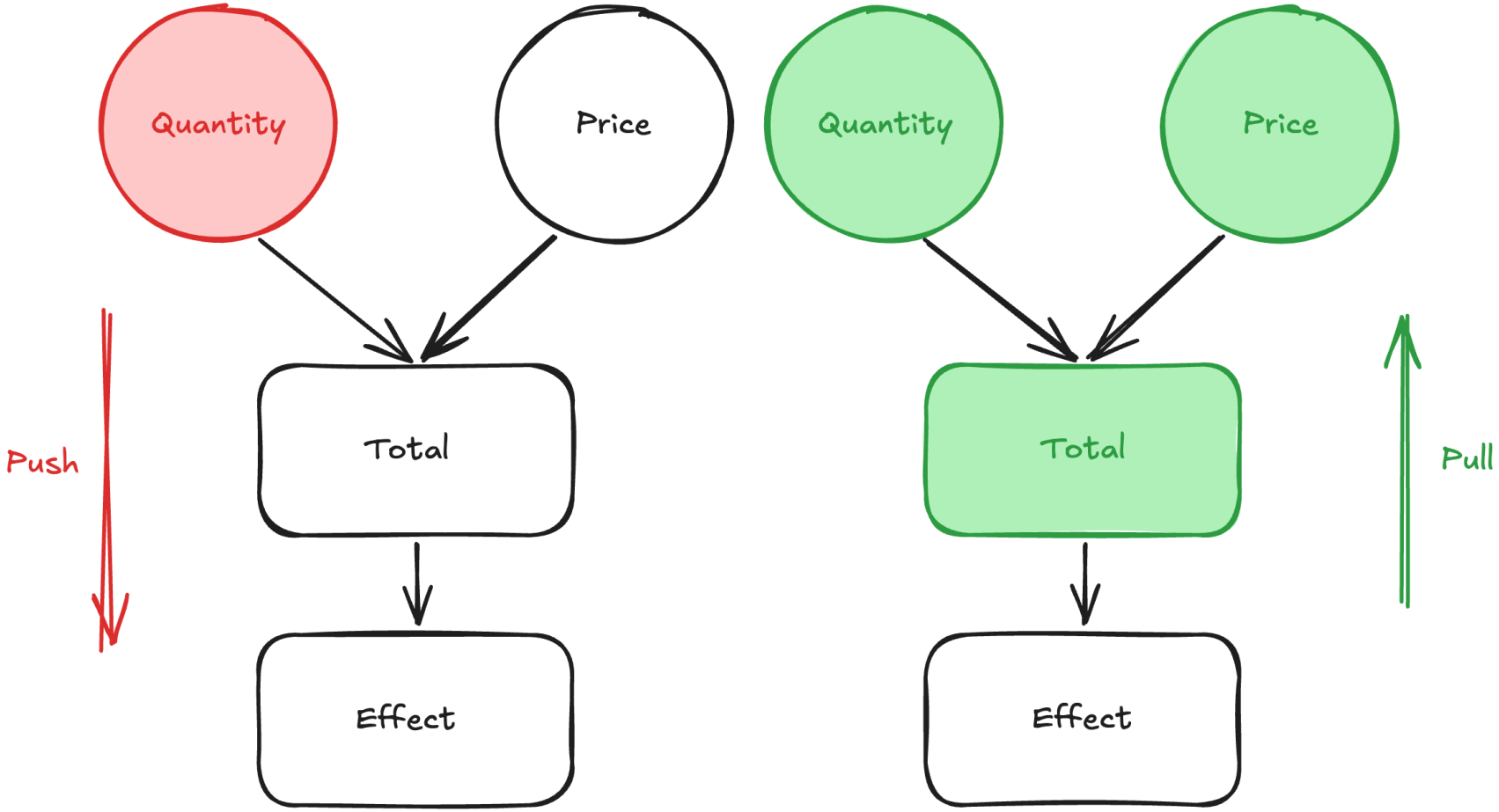
effect(() => {
  console.log(`Total: ${total()}`)
})
```

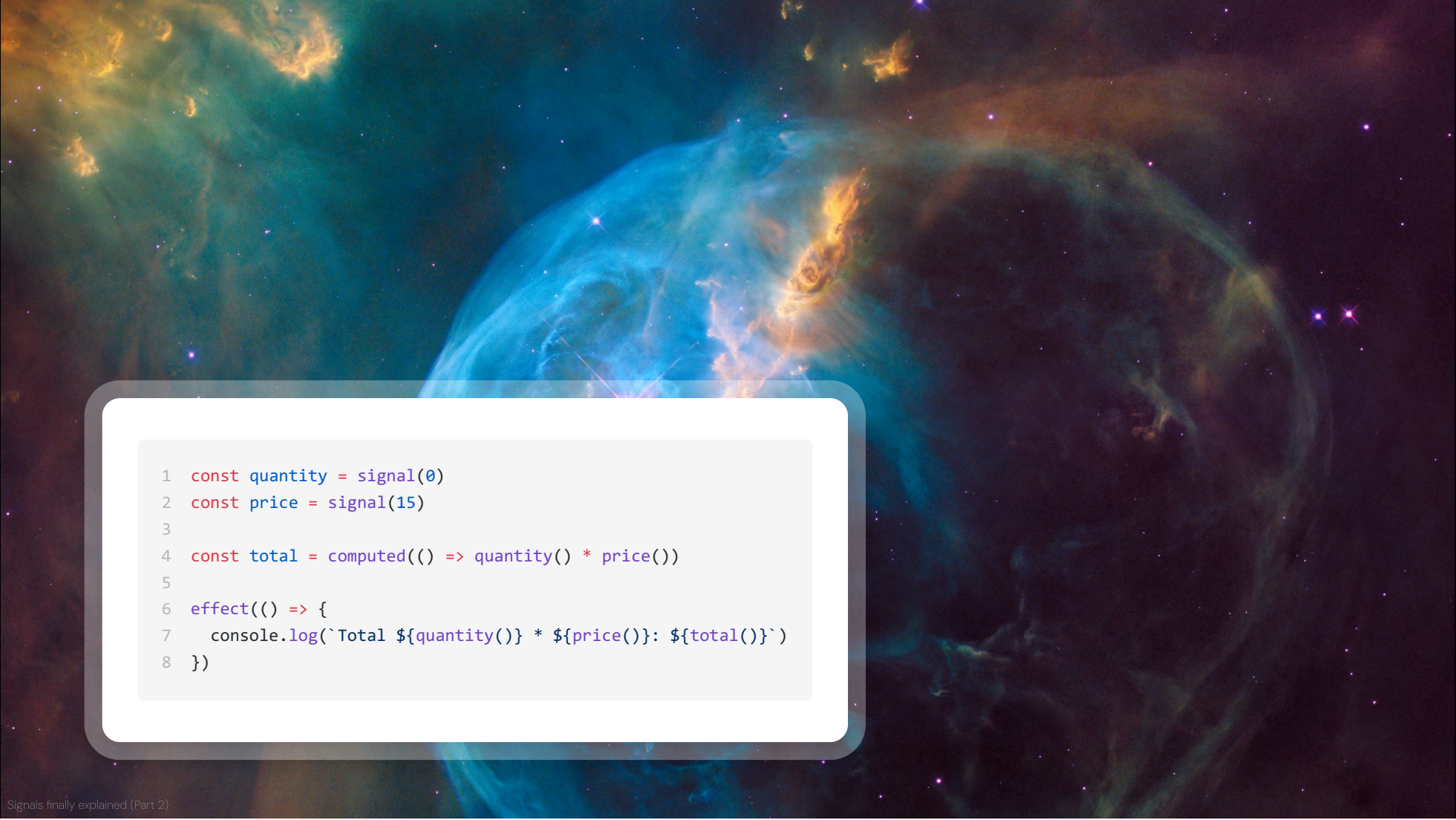


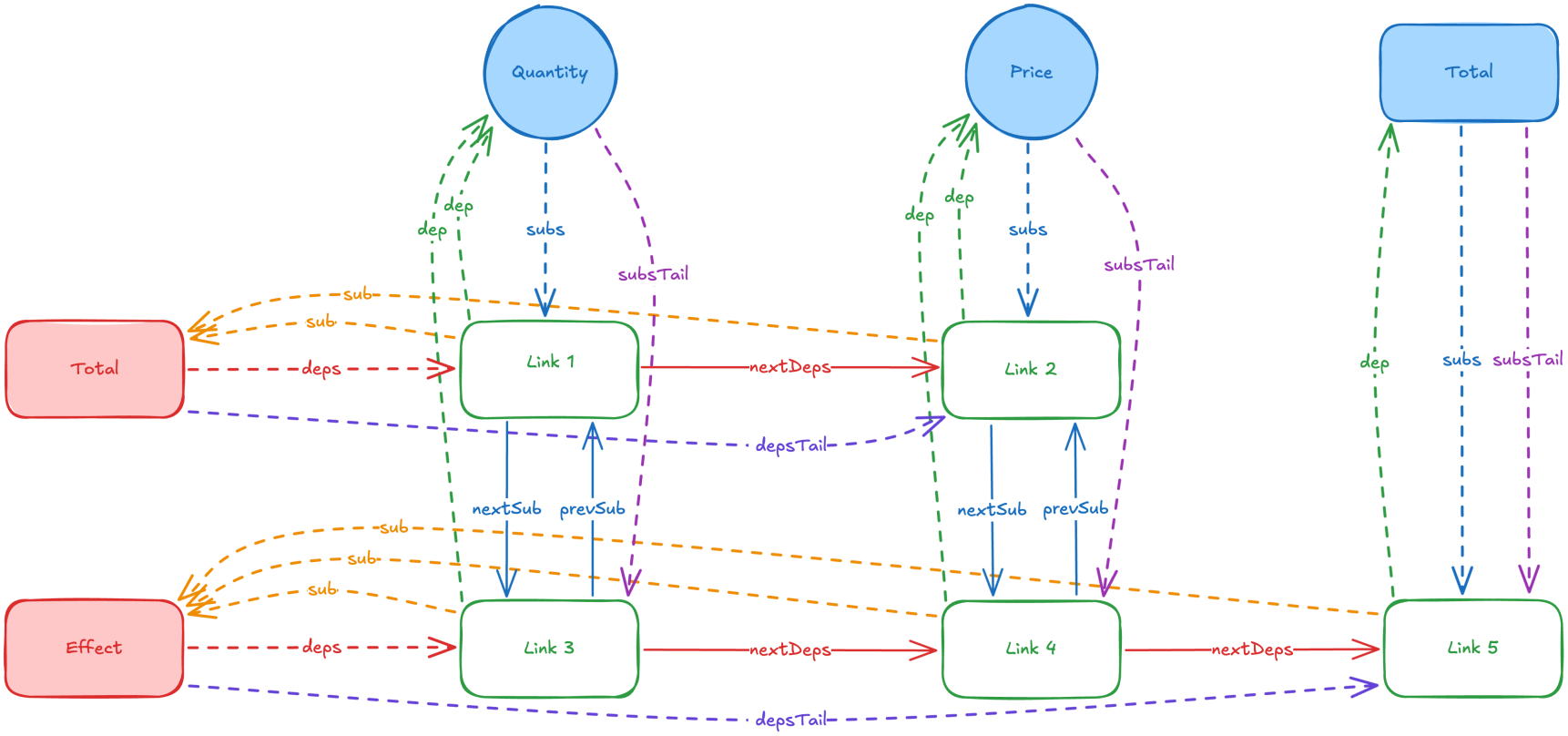
Le lien entre les graphes et la réactivité

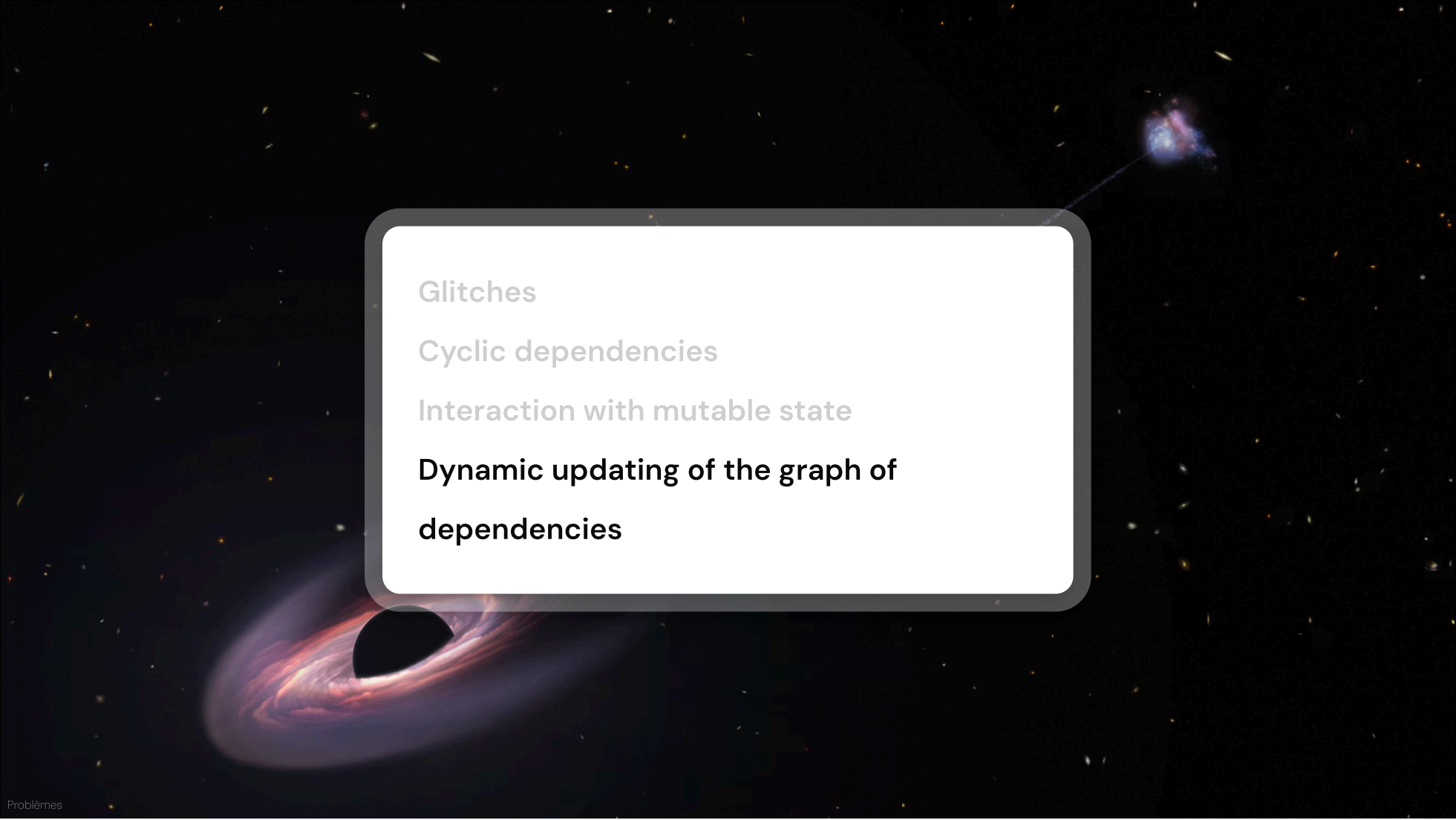

signal()
computed(() => {})
effect(() => {})
effectScope(() => {})

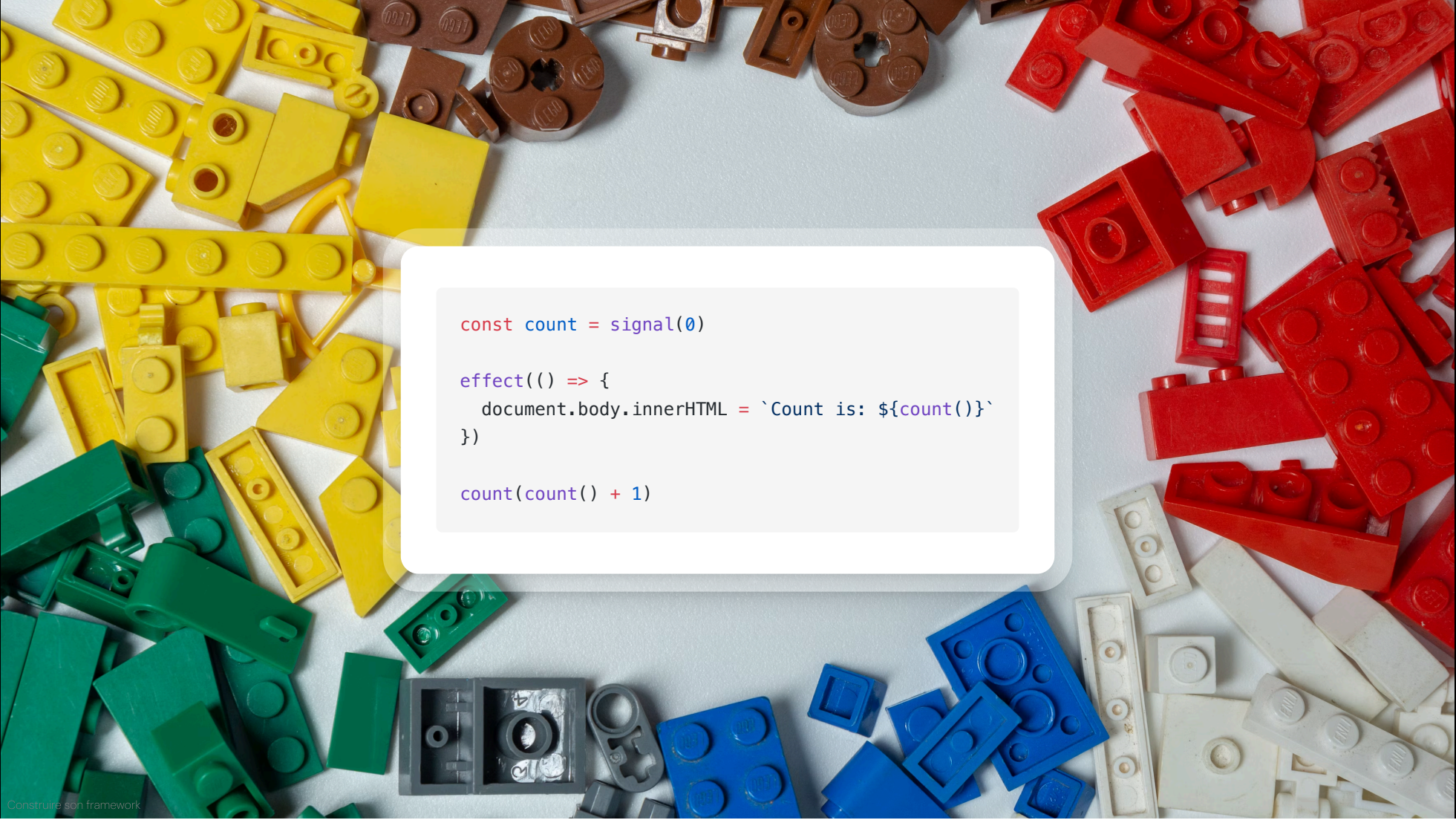
```ts
function signalGetterSetter<T>(this: Signal<T>, ...value: [T]): T | void {
  if (value.length) {
    if (this.currentValue !== (this.currentValue = value[0])) {
      const subs = this.subs
      if (subs !== undefined) {
        propagate(subs)
        if (!batchDepth) {
          processEffectNotifications()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


Quantity
Price
Push
Total
Effect
Quantity
Price
Total
Pull
Effect


const quantity = signal(0)
const price = signal(15)
const total = computed(() => quantity() * price())
effect(() => {
console.log(`Total ${quantity()} * ${price()}: ${total()}`)
})


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()
```

C'était

La réactivité et les signaux : démystifions la magie du frontend

Pour **Asynconf** (June 27, 2026)

Présenté par  Estéban Soubiran

⬇️ Votre feedback ⬇️

example.com