



Adieu RxJS ! Vive les
Signals ! Oh wait...

Anthony Pena



Anthony Pena

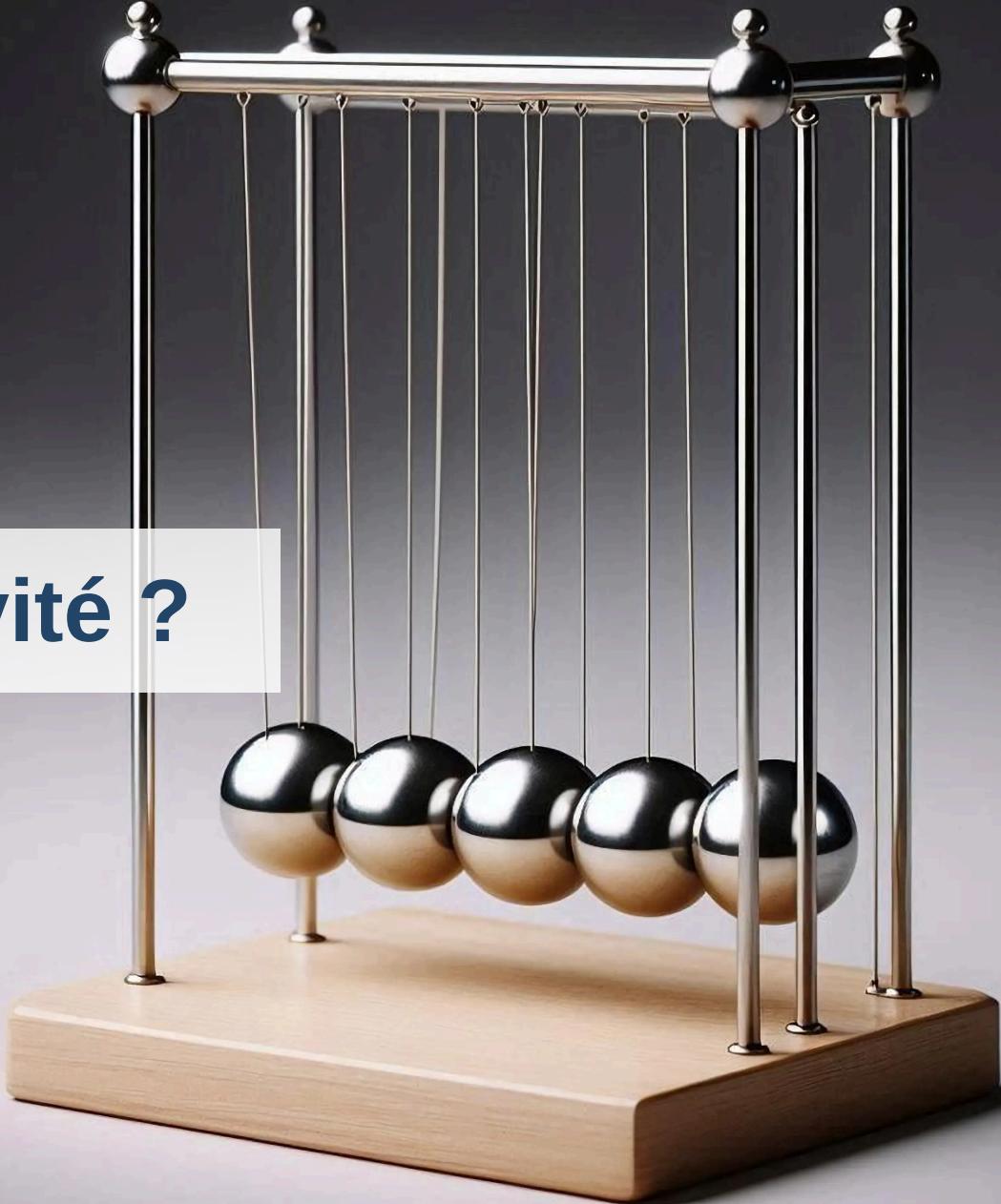
Développeur Web Fullstack @ **[sf=ir]**



A detailed painting of a caveman with long hair and a beard, wearing a fur vest, crouching on the ground and using a hammerstone to shape a smaller rock. He is surrounded by large ferns and tall trees. In the background, other cavemen are visible. A white rectangular box covers the upper left portion of the scene.

Avant les frameworks

La Réactivité ?



"A declarative programming model for updating based on changes to state."

Kristen / pzuraq

<https://www.pzuraq.com/blog/what-is-reactivity>



Angular.JS et ses watchers



Angular 2 et Zone.js

```
@Component({
  template: `
    <p>{{ text }}</p>
  `,
})
export class PlaygroundComponent {
  text = "";

  ngOnInit() {
    setInterval(() => this.text += '!', 1_000)
  }
}
```

```
@Component({
  template: `
    <input (change)="setText($event)"/>
    <p>{{ text }}</p>
  `,
})
export class PlaygroundComponent {
  text = "";

  setText(event: Event) {
    this.text = (event.target as HTMLInputElement).value;
  }
}
```

```
@Component({
  template: `
    <input [(ngModel)]="text"/>
    <p>{{ text }}</p>
  `
})
export class PlaygroundComponent {
  text = "";

  ngOnInit() {
    setInterval(() => this.text += '!', 1_000)
  }
}
```

```
@Component({
  selector: 'app-root',
  standalone: true,
  template: `
    <p>{{ text }}</p>
  `,
})
export class PlaygroundComponent {
  text = "waiting...";

  ngOnInit() {
    this.asyncHello().then(text => this.text = text);
  }

  private asyncHello(): Promise<string> {
    // ...
  }
}
```

```
@Component({
  template: `
    <p>{{ text | async }}</p>
  `,
})
export class PlaygroundComponent {
  text = this.asyncHello();

  private asyncHello(): Promise<string> {
    // ...
  }
}
```

```
@Component({
  template: `
    <app-alert>Alert</app-alert>
    <p>{{ text | async }}</p>
    <app-card />
    <app-card />
  `,
})
export class PlaygroundComponent {
  @Input() text: string;
}
```

```
@Component({
  template: `
    <app-alert>Alert</app-alert>
    <p>{{ text | async }}</p>
    <app-card />
    <app-card />
  `,
})
export class PlaygroundComponent {
  @Input() text: string;
  @Output() textChange = new EventEmitter<string>();
}
```

```
@Component({
  template: `
    <app-alert>Alert</app-alert>
    <p>{{ text | async }}</p>
    <app-card />
    <app-card />
  `
})
export class PlaygroundComponent {
  @Input() text: string;
  @Output() textChange = new EventEmitter<string>();
  @ViewChild(Alert) cards: Alert;
}
```

```
@Component({
  template: `
    <app-alert>Alert</app-alert>
    <p>{{ text | async }}</p>
    <app-card />
    <app-card />
  `,
})
export class PlaygroundComponent {
  @Input() text: string;
  @Output() textChange = new EventEmitter<string>();
  @ViewChild(Alert) cards: Alert;
  @ViewChildren(CustomCard) cards: QueryList<CustomCard>;
}
```

C'est cool tout ça non ?

Oui mais Zone.js

Angular 17 et les Signals

CODE SLIDE : reprendre la demo edition de variable mais en Signal

**CODE SLIDE : reprendre la demo edition de
champ de texte mais en Signal**

**CODE SLIDE : reprendre la demo promesse
mais en Signal**

CODE SLIDE : montrer effect

CODE SLIDE : montrer computed

Signals <3



tc39 / proposal-signals

Type to search



Code



Issues

101



Pull requests

11



Actions



Security



Insights

proposal-signals

Public

generated from [tc39/template-for-proposals](#)

8 Branches



Go to file



Add file



Code



littledan Simpler logo (#229)

b608efa · 4 days ago

146 Commits



README



Code of conduct



MIT license



Security



JavaScript Signals standard proposal

Stage 1 ([explanation](#))

TC39 proposal champions: Daniel Ehrenberg, Yehuda Katz, Jatin Ramanathan, Shay Lewis, Kristen Hewell Garrett, Dominic Gannaway, Preston Sego, Milo M, Rob Eisenberg

Original authors: Rob Eisenberg and Daniel Ehrenberg

Bientôt un standard

This document outlines an early proposal for a standardization effort for JavaScript, similar to the Promises/A+ effort which preceded the Promises standardization by TC39 in ES2015. Try it for yourself, using [a polyfill](#).

Similarly to Promises/A+, this effort focuses on aligning the JavaScript ecosystem. If this alignment is successful, a standard could emerge based on that experience. Several framework authors are collaborating here on https://github.com/tc39/proposal-signals



About

A proposal to add signals to JavaScript.

[Readme](#)

[MIT license](#)

[Code of conduct](#)

[Security policy](#)

[Activity](#)

[Custom properties](#)

[3k stars](#)

[96 watching](#)

[54 forks](#)

[Report repository](#)

Contributors 20



Zone.js c'est fini ?

CODE SLIDE : demo Signal input()

CODE SLIDE : demo Signal output()

CODE SLIDE : demo Signal viewchild()

Oups j'ai oublié RxJS dans tous ça 🙄 (non)

Mais au fait... C'est quoi RxJS ?



RxJS

ReactiveX for JavaScript



ReactiveX

An API for asynchronous programming
with observable streams

A man with a beard and glasses is sitting in a brown armchair, looking through a pair of binoculars. He is wearing a blue and white plaid shirt. A small potted plant sits on the floor next to his chair. In the background, a large window looks out onto a lush green field where a bird is perched on a branch. The room has light-colored wooden floors and furniture, including a sofa and a coffee table.

Parlons Observable

**CODE SLIDE : demo Signal + Observable
toSignal() toObservable()**

Observables

Composable functions with guarantees.

Use observables for cancellation and event coordination.

- + Generally stateless
process event and clean up

- + Multiple values
a collection of events or values over time

- + Push N values

- + Convert to Signal? Yes

<https://twitter.com/BenLesh/status/1775207971410039230>

Signals

A dependency graph of values and logic around notification, invalidation, computation, and memoization.

Use signals for state management.

- + Maintain state
retain things in memory

- + Single value
a value that changes over time

- + Read 1 value

- + Convert to Observable? Avoidable
the event that set the signal can also notify the observable



CityJS
thisdot.co

Mais du coup les Signals...

Quelques cas concrets

HttpClient et interceptors

// TODO slide code avec HttpClient

// TODO slide code avec interceptor

HttpClient et interceptors

// TODO logo RxJS

État interne des composants

Reactive Forms

Services

Global state management (NgRx, NgXs)

The NgXs way

The NgXs way

// TODO ajouter un bout de code montrant le mapping Signal

The NgRx way

The NgRx way

// TODO ajouter un bout de code montrant la creation d'un Signal Store

The NgRx way

// TODO ajouter un bout de code montrant le mapping Signal dans un composant

The NgRx way

// TODO ajouter un bout de code montrant l'utilisation d'RxJS dans le store



En résumé



Les Signals c'est pour gérer les états et la réactivité dans les composants

A wide-angle photograph of a modern industrial factory. The floor is filled with workers wearing yellow hard hats and high-visibility vests. There are several conveyor belt systems moving various colored boxes (pink, blue, green) through the facility. In the background, there are tall metal shelving units and robotic arms. The lighting is bright, coming from large windows along the sides.

RxJS est là pour gérer tous vos flux de données



Ben Lesh ✅

@BenLesh · [Follow](#)

A Angular folks,

Some Signals vs Observables info...

1. Use signals for state management, not observables.
2. Use observables for cancellation and other event coordination.
3. DO NOT TRY TO USE SIGNALS LIKE RXJS. It's a bad/silly idea.

They are complimentary technologies.

1/

7:05 PM · Apr 2, 2024



563



Reply



Copy link

[Read 19 replies](#)



Anthony Pena

Développeur Web Fullstack @

[sf≡ir]

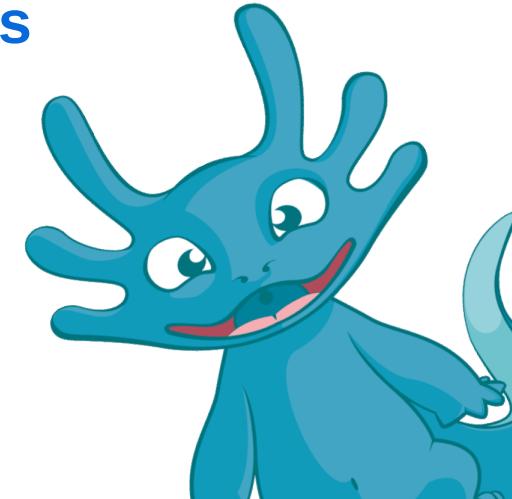
 @_Anthony_Pena_

 @kuroidoruido

 @penaanthon

<https://k49.fr.nf>

<https://github.com/kuroidoruido/talks>



Thank you!