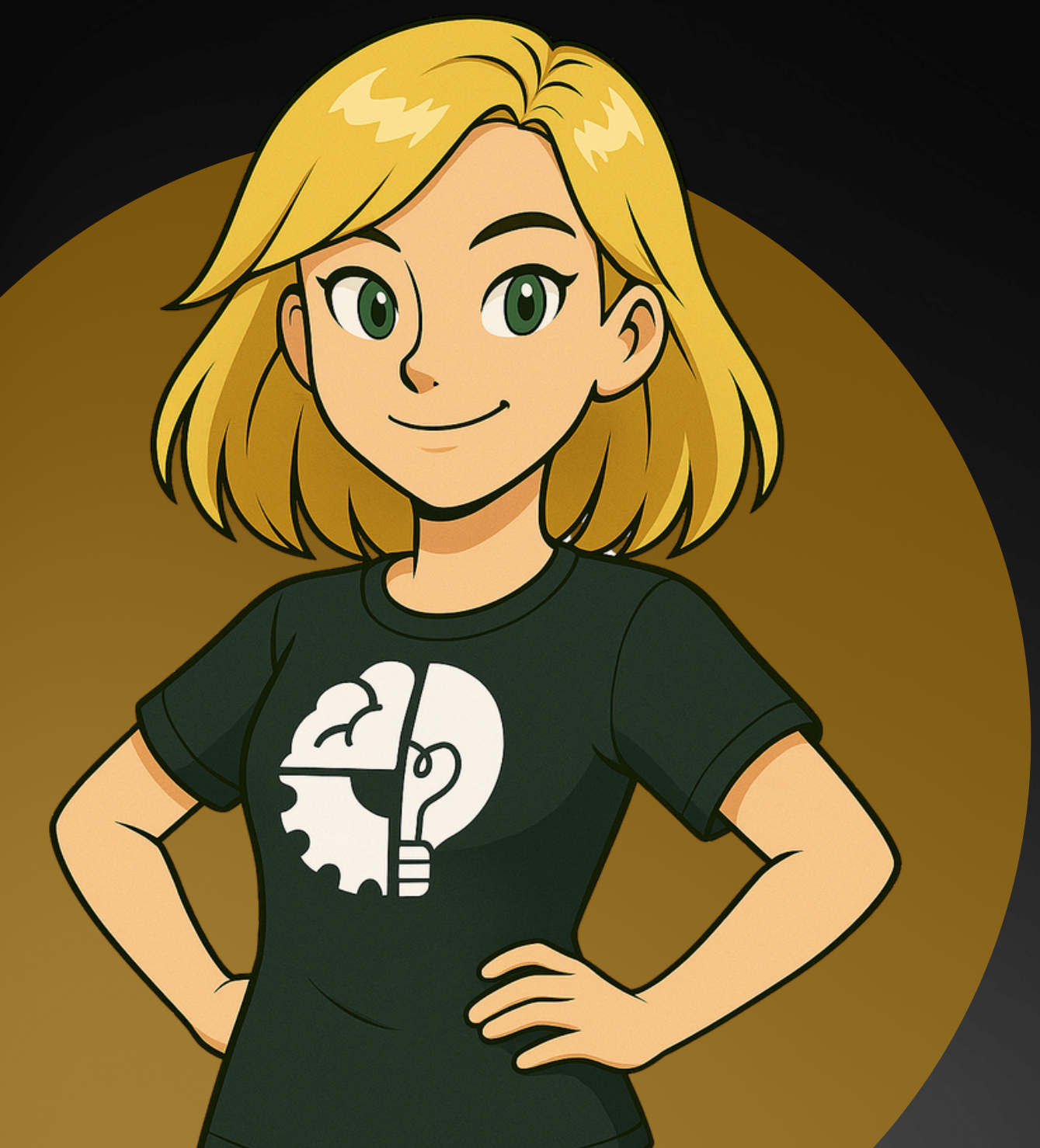
APPREND À CONTROLER TAKEUNTIL AVEC RXJS ET ANGULAR



Elodie TURLIER

Angular, IA, agilité & storytelling : construire des apps prêtes à survivre à l’apocalypse digitale

01

COMPRENDRE TAKEUNTIL

takeUntil est un opérateur ***RxJS*** essentiel pour gérer automatiquement le cycle de vie des observables.

- Complète un observable dès qu'un autre observable émet une valeur.
- Idéal pour prévenir les fuites mémoire dans Angular.

Elodie TURLIER
Angular, IA, agilité & storytelling

02

POURQUOI L’UTILISER ?

Son rôle principal est de compléter un observable source lorsqu'un observable notificateur émet une valeur, offrant ainsi un mécanisme robuste pour **prévenir les fuites mémoire** et **automatiser les désinscriptions**.

- **Désinscription automatique des abonnements.**
- **Simplifie la gestion des ressources.**
- **Évite les erreurs courantes liées au cycle de vie.**

Cas d’usage typiques :

- Gestion du cycle de vie des composants Angular
- Événements externes (clics, timers)

Elodie TURLIER
Angular, IA, agilité & storytelling

03

SYNTAXE

takeUntil suit la signature :

```
takeUntil<T>(notifier: Observable<any>): MonoTypeOperatorFunction<T>
```

Il surveille deux flux :

- **L'observable source** émet des valeurs.
- **L'observable notificateur** déclenche la complétion du source à sa première émission

```
interval(1000).pipe(
  takeUntil(timer(5000))
).subscribe(console.log);
// Émet 0,1,2,3,4 puis se complète
```

Elodie TURLIER
Angular, IA, agilité & storytelling

EXEMPLE

Gestion du cycle de vie des composants

Le principal cas d'usage dans Angular est la désinscription automatique lors de la destruction d'un composant :

```
private destroy$ = new Subject<void>();

ngOnInit() {
  this.dataService.getData().pipe(
    takeUntil(this.destroy$)
  ).subscribe(/* ... */);
}

ngOnDestroy() {
  this.destroy$.next();
  this.destroy$.complete();
}
```

- **Centralisation de la désinscription**
- **Prévient les fuites mémoire**

Elodie TURLIER
Angular, IA, agilité & storytelling

05

NOUVEAUTÉ
ANGULAR 16+

Angular 16 introduit ***takeUntilDestroyed***, simplifiant radicalement le code :

```
constructor() {
  this.dataService.getData().pipe(
    takeUntilDestroyed()
  ).subscribe(/* ... */);
}
```

- Plus de simplicité, moins de code boilerplate.
- Contexte de destruction détecté automatiquement.

Elodie TURLIER
Angular, IA, agilité & storytelling

COMPARAISON AVEC LES OPÉRATEURS SIMILAIRES

Opérateur	Mécanisme d'arrêt	Cas d'usage typique
`take(n)`	Après n émissions	Limitation explicite de résultats
`takeWhile`	Prédicat sur les valeurs émises	Arrêt conditionnel basé sur données
`takeUntil`	Émission externe (notificateur)	Gestion de cycle de vie Angular

Avantage clé de ***takeUntil*** : Réactivité immédiate à un événement externe, idéal pour ***ngOnDestroy***

Elodie TURLIER
Angular, IA, agilité & storytelling

07

PIÈGES COURANTS ET SOLUTIONS

Ordre des opérateurs

```
source$.pipe(
  takeUntil(this.destroy$),
  switchMap(() => inner$)
)
```

=> Les abonnements internes à switchMap ne sont pas nettoyés.

Solution : Placer takeUntil en dernier.

```
source$.pipe(
  switchMap(() => inner$),
  takeUntil(this.destroy$)
)
```

Elodie TURLIER
Angular, IA, agilité & storytelling

08

BONNES PRATIQUES

- Préférer ***takeUntilDestroyed*** dans Angular 16+
- Toujours placer ***takeUntil*** en fin de chaîne
- Utiliser un Subject dédié (***destroy$***) pré-complété dans ***ngOnDestroy***
- Combiner avec ***async pipe*** quand possible pour minimiser les subscriptions manuelles

L'évolution vers ***takeUntilDestroyed*** marque une simplification notable, mais repose sur les mêmes principes sous-jacents. Son adoption progressive, couplée à une compréhension approfondie de ***takeUntil***, garantit des applications Angular robustes et performantes

Elodie TURLIER
Angular, IA, agilité & storytelling

09

DÉBOGAGE ET OPTIMISATION

- Utiliser Chrome DevTools Memory Profiler pour surveiller la mémoire.
- Linting RxJS (***rxjs-no-unsafe-takeuntil***).
- Logs dans ***ngOnDestroy*** pour vérifier la désinscription.

Elodie TURLIER
Angular, IA, agilité & storytelling

10

BONNES PRATIQUES

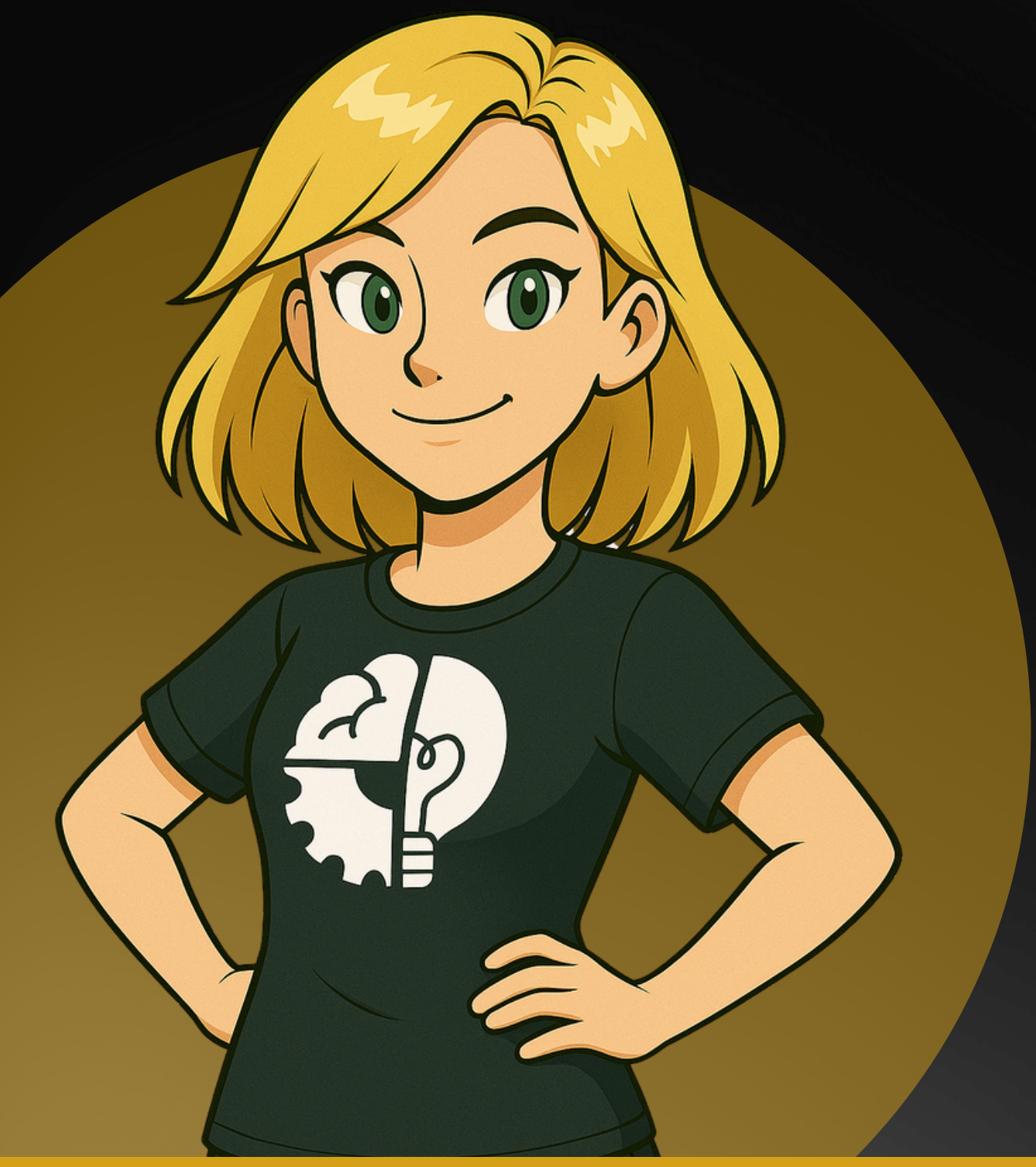
takeUntil reste indispensable pour :

- La gestion propre des abonnements Angular
- La composition complexe de flux réactifs
- L'intégration avec les patterns modernes (Signaux, SSR)
- Prévention efficace des fuites mémoire.
- Améliore les performances et maintient la clarté du code.

Pour en savoir plus, rendez-vous sur la **documentation officielle**: https://www.learnrxjs.io/learn-rxjs/operators/filtering/takeuntil

Elodie TURLIER
Angular, IA, agilité & storytelling

ABONNE TOI POUR PLUS DE CONTENU



Elodie TURLIER

Angular, IA, agilité & storytelling : construire des apps prêtes à survivre à l’apocalypse digitale

Enregistre ou partage si ça t’a été utile