MAÎTRISE FILTER EN RXJS + ANGULAR

Apprends à filtrer efficacement tes flux pour coder plus vite et plus propre.

Elodie TURLIER

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

01

COMPRENDRE FILTER

***filter* = un videur de boîte**

Il laisse passer uniquement ce qui satisfait ta règle métier.

✓ Si true → la valeur continue

✗ Si false → elle est supprimée. Définitivement.

C’est l’outil indispensable pour :

- **Éviter les traitements inutiles**
- **Améliorer les performances**
- **Simplifier la logique**

Elodie TURLIER
Angular, IA, agilité & storytelling

02

SYNTAXE

```
observable$.pipe(
  filter((value, index?) => boolean)
)
```

- ***value*** : valeur courante du flux
- ***index*** : position de l'élément dans le flux (optionnel)
- Retourne ***true*** pour garder la valeur

Simple mais puissant.

Elodie TURLIER
Angular, IA, agilité & storytelling

03

CAS SIMPLES

Filtrer des nombres :

```
from([10, 20, 30, 40]).pipe(
  filter(x => x > 25)
)
```

Filtrer des objets :

```
filter(user => user.id === 2)
```

Résultat : tu gardes uniquement les éléments pertinents.
C’est une gestion plus ciblée.

Elodie TURLIER
Angular, IA, agilité & storytelling

CAS PRATIQUES ANGULAR

04

Filtrage dans un formulaire réactif :
Tu veux déclencher un appel API uniquement après que l'utilisateur ait saisi plus de 2 caractères ?

```
searchControl.valueChanges.pipe(
  debounceTime(300),
  filter(text => text.length > 2),
  distinctUntilChanged()
)
```

Réponse API filtrée :

```
http.get(...).pipe(
  filter(users => users.some(u => u.active))
)
```

Résultat : un code optimisé, sans surcharger l'API ou l'UI.

Elodie TURLIER
Angular, IA, agilité & storytelling

COMBINAISON AVEC D'AUTRES OPÉRATEURS

05

Avec map :

```
pipe(
  map(res => res.data),
  filter(data => data.length > 0)
)
```

Avec switchMap :

```
pipe(
  filter(e => e.clientX > 500),
  switchMap(() => this.loadSidebarContent())
)
```

Combine-les pour plus de réactivité et de performance.

Elodie TURLIER
Angular, IA, agilité & storytelling

06

MEILLEURES PRATIQUES

- Place *filter* **dès le début** pour éviter des traitements inutiles.

- Utilise un **type guard** pour assurer la bonne forme des données.

```
filter((user?): user is User => !!user)
```

Elodie TURLIER
Angular, IA, agilité & storytelling

PIÈGES FRÉQUENTS À ÉVITER 07

Ne pas muter dans *filter* :

```
// ❌ Mauvaise pratique
filter(user => {
  user.lastLogin = new Date();
  return user.active;
})
```

Solution : Utilise *tap* pour effectuer des actions sans mutation avant de filtrer.

Comparer strictement :

```
filter(id => id === this.selectedId) // ✅
```

Elodie TURLIER
Angular, IA, agilité & storytelling

08

PERFORMANCES

- Réduction des re-rendus.
- Économie des d’appels API.
- Combine avec *takeUntil* pour éviter les fuites :

```
.pipe(filter(...), takeUntil(this.destroy$))
```

Elodie TURLIER
Angular, IA, agilité & storytelling

09

COMPARATIF : IMPÉRATIF VS RÉACTIF

Approche	Exemple	Avantages
Impérative	`if (x) {...}` dans `subscribe`	Simple pour débuter
Réactive	`.pipe(filter(...))`	Code plus propre et réutilisable

Elodie TURLIER
Angular, IA, agilité & storytelling

EXERCICES PRATIQUES

10

1. Filtrage d'un tableau

Filtre les nombres divisibles par 3 et multiplie-les par 10 :

```
from([15, 32, 7, 44]).pipe(
  filter(x => x % 3 === 0),
  map(x => x * 10)
)
```

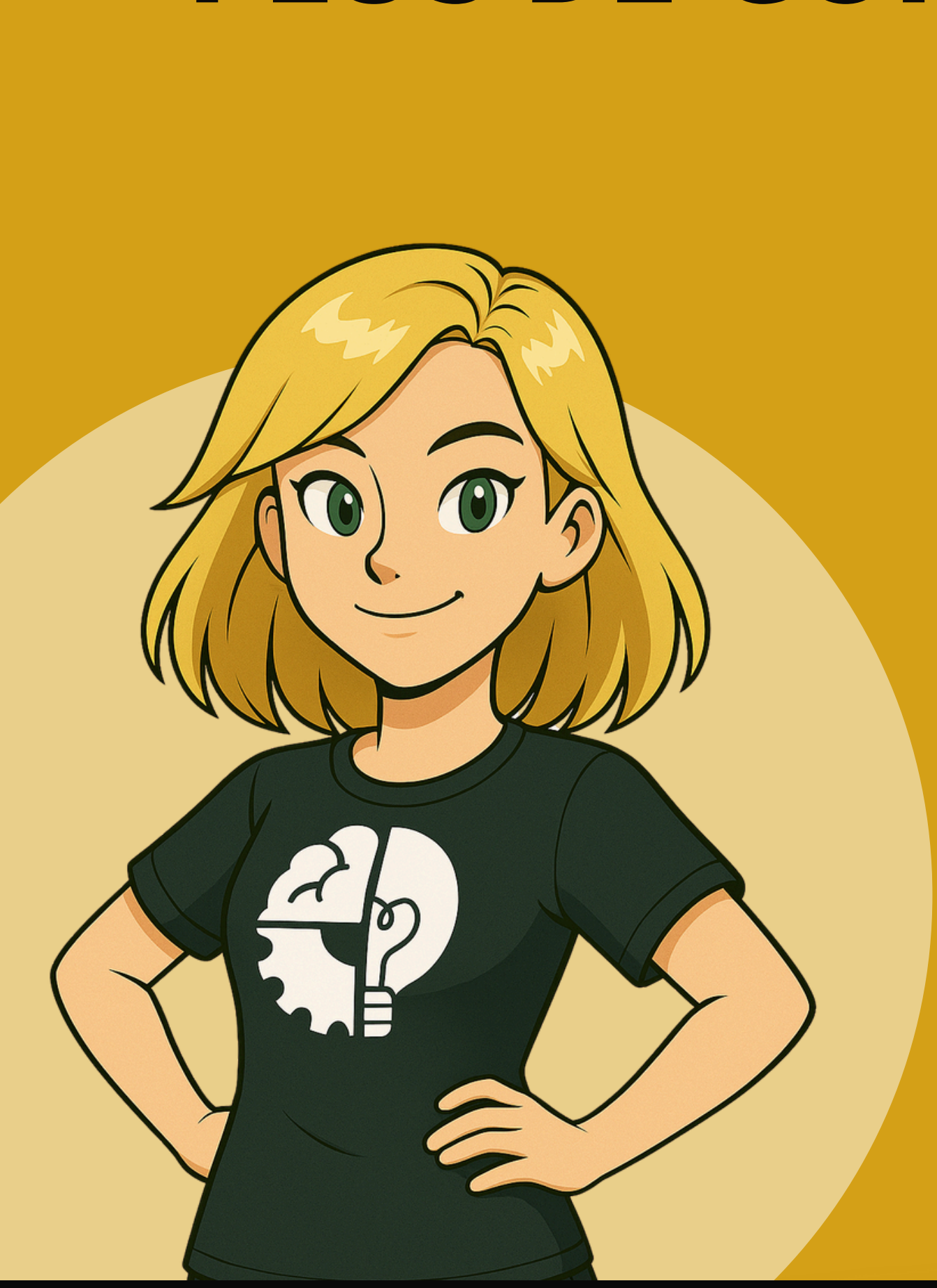
2. Vérification de permissions utilisateurs

Filtre les actions selon les permissions de l'utilisateur :

```
userActions$.pipe(
  filter(action => this.authService.hasPermission(action))
)
```

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