LES DOUBLES NÉGATIONS EN PROGRAMMATION

Philémon Globléhi
Développeur Back-end Java

Problématique :

Les doubles négations (comme !isNotReady ou !user.isNotActive()) complexifient inutilement la lecture du code.

Pourquoi ça marche ?

- Lisibilité : isAvailable() > !isNotAvailable().
- Maintenance : Moins de risques de bugs liés à une logique inversée.
- Philosophie Code Propre : "Le code est un message à vos futurs collègues (ou à vous dans 6 mois) !"

Voici 3 nouveaux exemples pour les remplacer par des méthodes affirmatives et intuitives !

Exemple 1 : Gestion d'un compte utilisateur

Java

```java
// AVANT ❌
if (!user.isNotVerified()) {
    // "Si l'utilisateur n'est PAS non-vérifié" → 🤯
}

// APRÈS ✅
if (user.isVerified()) {
    // Clair et direct !
}
```

Exemple 2 : Validation de formulaire

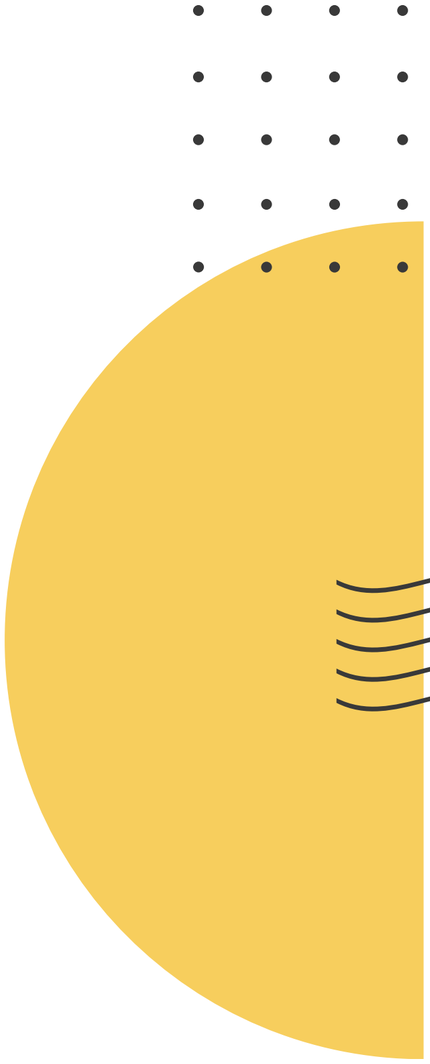
Java

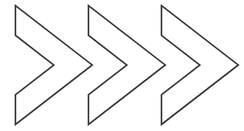
```java
// AVANT ❌
if (!form.hasNoErrors()) {
    System.out.println("Le formulaire est invalide");
}

// APRÈS ✅
if (form.hasErrors()) {
    System.out.println("Le formulaire est invalide");
}


```

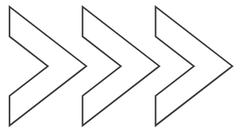
Exemple 3 : Contrôle d’accès à une ressource

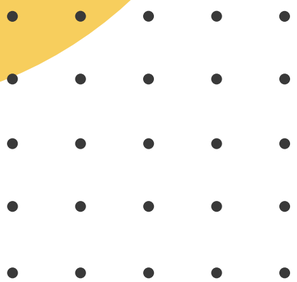
Java

```java
// AVANT ❌
if (!service.isNotAvailable()) {
    // "Si le service n'est PAS indisponible" → 🤔
}

// APRÈS ✅
if (service.isAvailable()) {
    // Tout est dit en 2 mots !
}


```









Merci

Philémon Globléhi, Back-end Developer & DevOps Lover
Linkedin: https://www.linkedin.com/in/philemon-globlehi/