

# Fiche d'investigation de fonctionnalité

**Fonctionnalité :** Filtrage par recherche de mots clefs sur un tableau de recettes.

**Problématique :** Il est impératif de trouver la méthode la plus optimisée pour afficher un grand nombre de recettes le plus rapidement possible.

**Option 1 :** Utilisation de la méthode `forEach()` + `some()`.  
Par cette méthode, on itère pour chaque élément du tableau. On va parcourir l'ensemble du tableau et vérifier dans chaque élément si il répond aux conditions.

**Inconvénients :**

- Impossible d'arrêter la boucle. Elle commence et s'arrête seulement à la fin.
- La boucle ne retourne aucune valeur, elle ne crée aucun nouveau tableau.
- On ne peut pas traiter le même tableau une fois dans la boucle.
- Un peu lent.

**Avantages :**

- Très simple à utiliser.
- Facile à lire.
- Compatible avec les anciennes version d'EcmaScript

**Option 2 :** Utilisation de la méthode `filter()` + `some()`.  
Par cette méthode, on va retourner un nouveau tableau selon la condition en paramètre. Lorsque la condition est remplie (`true`), la méthode ajoute la valeur au nouveau tableau.

**Inconvénients :**

- Peut être gourmand sur les très grands tableaux.
- N'est pas compatible avant la version EcmaScript 5.

**Avantages :**

- Plus concis.
- Plus rapide.
- Crée automatiquement un nouveau tableau.

**Option 3 :** Utilisation de la méthode `for()` + `for()` (nested).  
On itère sur chaque élément du tableau via une boucle `for` dans laquelle on va effectuer plusieurs tests.

**Inconvénients :**

- plus verbeux que les autres méthodes.
- N'est pas déclarative. (au premier coup d'oeil on ne devine pas forcément son utilité).
- Peut être plus lente qu'une méthode de programmation fonctionnelle dans certains cas.

**Avantages :**

- Contrôle précis. (départ et fin de la boucle avec `"i"`), choix des conditions plus large,
- Possibilité d'arrêter la boucle avec `break`.
- Hautement retro-compatible, puisque `for()` existe depuis la création de JS.

**Conclusion :**

Puisqu'il faut choisir la méthode la plus optimisée, l'option 2 serait à retenir. Mais un test de plus grand ampleur avec un tableau beaucoup plus grand devrait être conduit pour s'assurer que la méthode `filter()` est toujours la plus optimisée.