

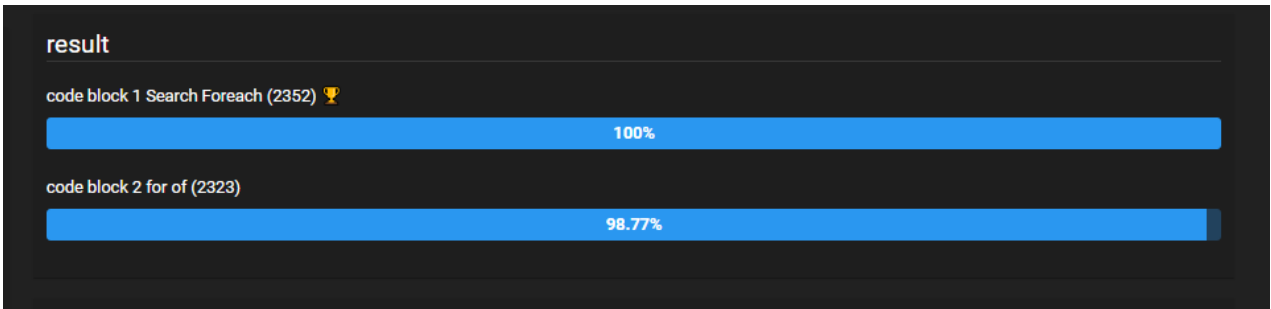


<b>Fonctionnalité :</b> Recherche de recettes et/ou par ingrédients, appareils, ustensiles		<b>Fonctionnalité</b>
<b>Problématique :</b> Filtrer les recettes dans la barre de recherche principale, rechercher par <b>3 critères</b> (Ingrédients, appareils, ustensiles)		
<b>Option n° 1 :</b> <b>ForEach</b> méthode native  Méthode spécifique pour les itérations (ancienne)		
<b>Avantages :</b> <ul style="list-style-type: none"> <li>- Itère sur tous les éléments du tableau</li> <li>- Pas besoin de faire un array.from</li> <li>- Convertit un objet Set en un []</li> <li>- Plus rapide</li> </ul>		<b>Inconvénients:</b> <ul style="list-style-type: none"> <li>- Pas une syntaxe es6</li> <li>- Dépend aussi du navigateur</li> </ul>
Saisie de 3 caractères pour lancer la recherche principale		

<b>Option n° 2 :</b> <b>- For (let ..of )</b> Une instruction à exécuter à chaque itération - Javascript doit analyser avant, sur quelle méthode il va devoir itérer - Syntaxes es6		
<b>Avantages :</b> <ul style="list-style-type: none"> <li>- Syntaxe ES6</li> <li>- Facile à lire</li> </ul>		<b>Inconvénients :</b> <ul style="list-style-type: none"> <li>- Création d'une variable pour le contenu du []</li> </ul>
Saisie de 3 caractères pour lancer la recherche principale		

<b>Block 1</b> 	
--	--

## Set up Block ( configuration )

Code qui sera utilisé toujours par les 2 tests blocs 1 et 2 -

Ne changera pas - Il est appelé pour exécuter les 2 blocs de test.

- Const recipes - l'objet à parcourir

```
Setup block (useful for function initialization, it will be run before every test, and is not part of the benchmark.)  
  
1  
2  
3 ▸ const recipes = [  
4 ▸   {  
5     id: 1,  
6     name: "Limonade de Coco",  
7     servings: 1,  
8     ingredients: [  
9     {  
10      ingredient: "Lait de coco",  
11      quantity: 400,  
12      ...}
```

- Charge la classe RecipeDto (mon dataProvider va charger mon [] dans des objets Dto

```
class RecipeDto {  
  constructor(recipeData) {  
    this.recipeData = recipeData;  
    this.name = recipeData.name;  
    this.servings = recipeData.servings;  
    this.time = recipeData.time;  
    this.description = recipeData.description;  
    this.ustensils = new Set();  
    this.appliances = new Set();  
    this.ingredients = new Set();  
    this.ingredientsData = new Set(); // collection
```

- Function replaceSpecialChars

```
▸ function replaceSpecialChars(str) {  
  return str  
    .replace(/[.,;:!*"()^]/g, "")  
    .replace(/[']/g, " ")  
    .replace(/[\d]/g, "")  
    .replace(/[éèêë]/g, "e")  
    .replace(/[îï]/g, "i")  
    .normalize("NFD")  
    .replace(/[\p{D}]/g, "")  
    .replace(/[\p{Z}]/g, " ")  
    .trim()  
}
```

- Classe RecipeDataProvider

```
class RecipeDataProvider {  
  constructor() {  
    this.recipes = new Set();  
    recipes.map((recipe) => {  
      this.recipes.add(new RecipeDto(recipe));  
    });  
  }  
}
```

## Boilerplate Block

Il lance la classe SearchParams (les critères de recherche)

avant les blocks 1 et 2

Il exécute ce bloc 2 fois (1 fois à chaque test)

```
boilerplate block (code will executed before every block and is part of the benchmark. use it for data initializing.)

1
2
3 ▾ class SearchParams {    ///! SIMULER UNE REQUETE HTTP  Example:
4 ▾   constructor() {
5       this.ingredients = new Set("jus de citron");
6       this.ustensils = new Set("cuillere");
7       this.appliances = new Set("saladier");
8       this.input = "creme";
9   }
10 ▾  isValid() {
11      return (
12          this.ingredients.size > 0 ||
```

On fait(simule) une requête Http (data Example)

jus de citron cuillère saladier avec les caractères dans Input : creme.

---

## BLOCK 1 Foreach

Class SearchService

Besoin d'un objet et de la fonction launch

Qui initialisera un [] vide qui contiendra les résultats de la recherche

Elle instancie un objet searchParam

```
class SearchService {
  constructor(recipes) {
    this.recipes = recipes;
  }

  launch() {
    this.recipesRecovered = new Set();
    ///! VA INITIALISER UN [] QUI CONTIENDRA LES RESULTATS DE LA RECHERCHE
    this.searchParams = new SearchParams();
    ///! ELLE INSTANCIE UN OBJET SEARCHPARAM

    if (this.searchParams.isValid()) {
      if (this.searchParams.isValid()) {
        this.recipes.forEach((recipe) => {
          let isRecovered =
            recipe.isValidSearchInput(this.searchParams.input) &&
            recipe.hasIngredients(this.searchParams.ingredients) &&
            recipe.hasUstensils(this.searchParams.ustensils) &&
            recipe.hasAppliances(this.searchParams.appliances);
          if (isRecovered) {
            this.recipesRecovered.add(recipe);
          }
        });
      }
    }
  }
}
```

- Instanciation du Provider
- Appel du searchService
- Exécution de la methode launch()

```

0  ///! INSTANCIATION DU PROVIDER
1  const recipeDataProvider = new RecipeDataProvider();
2
3  ///! L APPEL DE LA CLASSE SEARCHSERVICE
4  const searchService = new SearchService(recipeDataProvider.recipes);
5
6  ///!L APPEL DE LA METHODE LAUNCH()
7  searchService.launch();
8

```

---

## Block 2 For let... of

```

class SearchService {
  constructor(recipes) {
    this.recipes = recipes;
  }

  launch() {
    ///! va initialiser un array vide
    ///! qui apres, contiendra le resultat de la recherche
    this.recipesRecovered = new Set();
    ///! instancie un objet searchParams
    this.searchParams = new SearchParams();

    for (let recipe of this.recipes) {
      if (
        recipe.isValidSearchInput(this.searchParams.input) &&
        recipe.hasIngredients(this.searchParams.ingredients) &&
        recipe.hasUstensils(this.searchParams.ustensils) &&
        recipe.hasAppliances(this.searchParams.appliances)
      ) {
        this.recipesRecovered.add(recipe);
      }
    }
  }
}

/*****
*****/

///! INSTANCIATION DU PROVIDER
const recipeDataProvider = new RecipeDataProvider();

///! L APPEL DE LA CLASSE SEARCHSERVICE
const searchService = new SearchService(recipeDataProvider.recipes);

///!L APPEL DE LA METHODE LAUNCH()
searchService.launch();

```