

Fonctions de tableaux

Incitation à la programmation fonctionnelle

Introduction

Fonctions de tableau (Array Functions)

Les fonctions de tableau sont des méthodes intégrées à JavaScript qui permettent de manipuler facilement les tableaux. Elles offrent des fonctionnalités pratiques pour ajouter, supprimer, rechercher et modifier des éléments dans un tableau.

Ces fonctions peuvent se chaîner les unes à la suite des autres (one liner)

Utilisation

Fonctions de tableaux

Voici quelques unes des fonctions de tableau couramment utilisées :

- `push()`: Ajoute un ou plusieurs éléments à la fin du tableau
- `pop()`: Supprime et renvoie le dernier élément du tableau
- `splice()`: Modifie le contenu du tableau en supprimant, remplaçant ou ajoutant des éléments
- `indexOf()`: Recherche un élément donné dans le tableau et renvoie son index
- `forEach()`: Exécute une fonction donnée sur chaque éléments du tableau

Utilisation

Push



```
// Exemple d'utilisation de push()  
let fruits = ['pomme', 'banane'];  
fruits.push('orange');  
console.log(fruits); // Affiche ['pomme', 'banane', 'orange']
```

Utilisation

Pop



```
// Exemple d'utilisation de pop()  
let fruits = ['pomme', 'banane', 'orange'];  
let dernierFruit = fruits.pop();  
console.log(dernierFruit); // Affiche 'orange'
```

Utilisation


Splice



```
// Exemple d'utilisation de splice()  
let fruits = ['pomme', 'banane', 'orange'];  
fruits.splice(1, 1, 'fraise', 'kiwi');  
console.log(fruits); // Affiche ['pomme', 'fraise', 'kiwi', 'orange']
```

Utilisation

IndexOf



```
// Exemple d'utilisation de indexOf()  
let fruits = ['pomme', 'banane', 'orange'];  
let indexOrange = fruits.indexOf('orange');  
console.log(indexOrange); // Affiche 2
```

Utilisation forEach



```
// Exemple d'utilisation de forEach()  
let fruits = ['pomme', 'banane', 'orange'];  
fruits.forEach(function(fruit) {  
    console.log(fruit);  
});
```


Introduction

Programmation fonctionnelle

La programmation fonctionnelle est un paradigme de programmation qui se concentre sur l'utilisation de fonctions pures et l'évitement des effets de bord. Les fonctions de tableau en programmation fonctionnelle sont conçues pour faciliter la transformation des tableaux sans modifier l'état global.

Introduction

Programmation fonctionnelle - Prédicats

En programmation fonctionnelle on utilise des prédicats. Les prédicats sont des fonctions qui renvoient une valeur booléenne. Par exemple, la fonction `isEven()` est un prédicat qui renvoie `true` si un nombre est pair et `false` sinon.

Utilisation

Fonctions de tableaux - programmation fonctionnelle

Voici quelques unes des fonctions de tableau utilisée en programmation fonctionnelle :

- `map()`: Applique une fonction a chaque éléments du tableaux et retourne un nouveau tableau contenant le résultat
- `filter()`: Filtre les éléments d'un tableau en fonction d'un prédicat et retourne une nouveau tableau avec les éléments correspondants
- `reduce()`: Réduits un tableau a une seule valeur en appliquant une fonction accumulatrice a chaque éléments

Utilisation

Map



```
// Exemple d'utilisation de map()  
let nombres = [1, 2, 3, 4, 5];  
let carres = nombres.map(x => x * x);  
console.log(carres); // Affiche [1, 4, 9, 16, 25]
```

Utilisation

Filter



```
// Exemple d'utilisation de filter()  
let nombres = [1, 2, 3, 4, 5];  
let nombresPairs = nombres.filter(x => x % 2 === 0);  
console.log(nombresPairs); // Affiche [2, 4]
```

Utilisation

Reduce



```
// Exemple d'utilisation de reduce()  
let nombres = [1, 2, 3, 4, 5];  
let somme = nombres.reduce((acc, cur) => acc + cur, 0);  
console.log(somme); // Affiche 15
```



```
const strings = ['Hello', 'World', 'How', 'Are', 'You?'];  
  
const totalLength = strings.reduce((acc, currentString) => accumulator + currentString.length, 0);  
  
console.log(totalLength); // affiche : 21
```

Conclusion

Les fonctions de tableau en programmation fonctionnelle sont un outil puissant pour la transformation et la manipulation des tableaux. Elles favorisent la modularité, la réutilisabilité et la lisibilité du code. En utilisant les fonctions de tableau appropriées, vous pouvez effectuer des opérations complexes sur les tableaux de manière élégante et déclarative.

Utilisation

Fonctions de tableaux - programmation fonctionnelle

Voici quelques unes des fonctions de tableau supplémentaires :

- `flat()`: Permet de niveler un tableau `imbriqué` en un tableau d'un seul niveau. Retourne ce dernier
- `flatMap()`: Applique une fonction de transformation a chaque éléments du tableau (`map`) puis aplatit le résultat en un tableau d'une seule dimension (`flat`)

Utilisation

Flat



```
let tableauxImbriques = [[1, 2], [3, 4], [5, 6]];
let tableauAplatit = tableauxImbriques.flat();
console.log(tableauAplatit); // Affiche [1, 2, 3, 4, 5, 6]
```

Utilisation

FlatMap



```
let nombres = [1, 2, 3, 4];  
let carresDoubles = nombres.flatMap(x => [x * x, x * 2]);  
console.log(carresDoubles); // Affiche [1, 2, 4, 4, 9, 6, 16, 8]
```

Utilisation

Fonctions de tableaux - programmation fonctionnelle

- `every()`: verify si tous les éléments d'un tableau satisfont un prédicat.
Retourne true si oui sinon false
- `some()`: vérifie si au moins un élément du tableau satisfait un prédicat
- `find()`: retourne le premier élément du tableau qui satisfait un prédicat
- `findIndex()`: retourne l'index du premier élément qui satisfait un prédicat

Utilisation

Every



```
let nombres = [2, 4, 6, 8, 10];  
let tousPairs = nombres.every(x => x % 2 === 0);  
console.log(tousPairs); // Affiche true
```

Utilisation

Some



```
let nombres = [1, 3, 5, 7, 10];  
let auMoinsUnPair = nombres.some(x => x % 2 === 0);  
console.log(auMoinsUnPair); // Affiche true
```

Utilisation

Find



```
let personnes = [  
  { nom: 'Alice', age: 25 },  
  { nom: 'Bob', age: 30 },  
  { nom: 'Charlie', age: 35 }  
];  
let personne = personnes.find(p => p.age === 30);  
console.log(personne); // Affiche { nom: 'Bob', age: 30 }
```

Utilisation

FindIndex



```
let personnes = [
  { nom: 'Alice', age: 25 },
  { nom: 'Bob', age: 30 },
  { nom: 'Charlie', age: 35 }
];
let index = personnes.findIndex(p => p.age === 30);
console.log(index); // Affiche 1
```

Programmation fonctionnelle

One Liner



```
const personnes = [
  { nom: 'Alice', age: 25 },
  { nom: 'Bob', age: 30 },
  { nom: 'Dave', age: 40 },
  { nom: 'Charlie', age: 35 }
];

const personnesFiltrees = personnes
  .filter((personne) => personne.age > 30) // Filtrer les personnes dont l'âge est supérieur à 30
  .map((personne) => personne.nom) // Obtenir uniquement les noms des personnes filtrées
  .sort(); // Trier les noms par ordre alphabétique

console.log(personnesFiltrees); // Résultat : ['Charlie', 'Dave']
```


Conclusion

Fonctions de tableaux - programmation fonctionnelle

Les fonctions de tableau en JavaScript sont un outil puissant pour manipuler des tableaux. Elles offrent une grande flexibilité et facilitent le travail avec les données tabulaires. En utilisant les fonctions de tableau appropriées vous pouvez effectuer des opérations telles que l'ajout, la suppression, la recherche et la modification des éléments d'un tableau de manière efficace et concise.

En les utilisant judicieusement, vous pouvez améliorer la lisibilité et la maintenabilité de votre code JavaScript.

De plus les fonctions de tableau préviennent les effets de bord