Programación funcional en JS (map, reduce, filter)

Sistemas Web

mapfilter ×

```
1 // What you have
2 let officers = [
3   { id: 20, name: 'Captain Piett' },
4   { id: 24, name: 'General Veers' },
5   { id: 56, name: 'Admiral Ozzel' },
6   { id: 88, name: 'Commander Jerjerrod' }
7 ];
8 // What you need
9 [20, 24, 56, 88]
```

.forEach()

mapfilter ×

```
1  // What you have
2  let officers = [
3    { id: 20, name: 'Captain Piett' },
4    { id: 24, name: 'General Veers' },
5    { id: 56, name: 'Admiral Ozzel' },
6    { id: 88, name: 'Commander Jerjerrod' }
7  ];
8  // What you need
9  // [20, 24, 56, 88]
10
11 let officersIds = [];
12 officers.forEach(function (officer) {
13   officersIds.push(officer.id);
14 });
15
16 console.log(officersIds)
17
```

.map()

mapfilter ×

```
1  // What you have
2  let officers = [
3    { id: 20, name: 'Captain Piett' },
4    { id: 24, name: 'General Veers' },
5    { id: 56, name: 'Admiral Ozzel' },
6    { id: 88, name: 'Commander Jerjerrod' }
7  ];
8  // What you need
9  // [20, 24, 56, 88]
10
11 let officersIds = officers.map(function (officer) {
12   return officer.id
13 });
14 console.log(officersIds)
```

.map()

```
11 const officersIds = officers.map( officer => {
12   return officer.id
13 });
```

.map()

mapfilter* ×

```
1  // What you have
2  let officers = [
3    { id: 20, name: 'Captain Piett' },
4    { id: 24, name: 'General Veers' },
5    { id: 56, name: 'Admiral Ozzel' },
6    { id: 88, name: 'Commander Jerjerrod' }
7  ];
8  // What you need
9  // [20, 24, 56, 88]
10
11 const officersIds = officers.map( officer =>  officer.id);
```

.reduce()

We need to know the total years of experience of all of them.

reduce ×

```
const pilots = [
 {
   id: 10,
   name: "Poe Dameron",
   years: 14,
 },
 {
   id: 2,
   name: "Temmin 'Snap' Wexley",
   years: 30,
 },
 {
   id: 41,
   name: "Tallissan Lintra",
   years: 16,
 },
 {
   id: 99,
   name: "Ello Asty",
   years: 22,
 }
];
```

.reduce()

reduce ×

```
const pilots = [
 {
   id: 10,
   name: "Poe Dameron",
   years: 14,
 },
 {
   id: 2,
   name: "Temmin 'Snap' Wexley",
   years: 30,
 },
 {
   id: 41,
   name: "Tallissan Lintra",
   years: 16,
 },
 {
   id: 99,
   name: "Ello Asty",
   years: 22,
 }
];


let totalYears = pilots.reduce(function (accumulator, pilot) {
  return accumulator + pilot.years;
}, 0);
```

.reduce()

Let's see how this can be shortened with ES6's arrow functions:

```
const totalYears = pilots.reduce((acc, pilot) => acc + pilot.years,
0);
```

.reduce()

Now let's say I want to find which pilot is the most experienced one. For that, I can use reduce as well:

.reduce()

reduce ×

```
1  const pilots = [
2   {
3     id: 10,
4     name: "Poe Dameron",
5     years: 14,
6   },
7   {
8     id: 2,
9     name: "Temmin 'Snap' Wexley",
10    years: 30,
11  },
12  {
13    id: 41,
14    name: "Tallissan Lintra",
15    years: 16,
16  },
17  {
18    id: 99,
19    name: "Ello Asty",
20    years: 22,
21  }
22 ];
23
24
25 let mostExpPilot = pilots.reduce(function (oldest, pilot) {
26   let experience = oldest.years || 0
27    if (experience > pilot.years)
28          return oldest
29     else
30         return pilot;
31 }, {});
32
33 console.log(mostExpPilot)
```

.filter()

filter ×

```
1  const pilots = [
2    {
3      id: 2,
4      name: "Wedge Antilles",
5      faction: "Rebels",
6    },
7    {
8      id: 8,
9      name: "Ciena Ree",
10     faction: "Empire",
11   },
12   {
13     id: 40,
14     name: "Iden Versio",
15     faction: "Empire",
16   },
17   {
18     id: 66,
19     name: "Thane Kyrell",
20     faction: "Rebels",
21   }
22 ];
23
24
```

Say we want two arrays now: one for rebel pilots, the other one for imperials.

.filter()

filter ×

```
1  const pilots = [
2    {
3      id: 2,
4      name: "Wedge Antilles",
5      faction: "Rebels",
6    },
7    {
8      id: 8,
9      name: "Ciena Ree",
10     faction: "Empire",
11   },
12   {
13     id: 40,
14     name: "Iden Versio",
15     faction: "Empire",
16   },
17   {
18     id: 66,
19     name: "Thane Kyrell",
20     faction: "Rebels",
21   }
22 ];
23
24
25 // rebels
26 console.log(pilots.filter( pilot => pilot.faction == "Rebels"))
27
28 // empire
29 console.log(pilots.filter( pilot => pilot.faction == "Empire"))
```

Say we want two arrays now: one for rebel pilots, the other one for imperials.

filter + reduce

filterMapReduce

```
const personnel = [
  {
    id: 5,
    name: "Luke Skywalker",
    pilotingScore: 98,
    shootingScore: 56,
    isForceUser: true,
  },
  {
    id: 82,
    name: "Sabine Wren",
    pilotingScore: 73,
    shootingScore: 99,
    isForceUser: false,
  },
  {
    id: 22,
    name: "Zeb Orellios",
    pilotingScore: 20,
    shootingScore: 59,
    isForceUser: false,
  },
  {
    id: 15,
    name: "Ezra Bridger",
    pilotingScore: 43,
    shootingScore: 67,
    isForceUser: true,
  },
  {
    id: 11,
    name: "Caleb Dume",
    pilotingScore: 71,
    shootingScore: 85,
    isForceUser: true,
  },
```

Our objective: get the total score of force users only.