

Opgaver - Array hjælpe metoder

Opgave (forEach - 1)

```
let names = ['HenriK', 'JAMshid', 'AndERS', 'EBBe', 'pER', 'MichAel', 'PETER'];

names.forEach((name, index, array) => {
    //.....
})

console.log(names);
```

Udfyld funktionen i **forEach**-metoden således at alle navne i *names* konverteret til lowercase, undtagen 'HenriK' og 'pER' der skal konverteres til uppercase, se:

```
▶ (7) ["HENRIK", "jamshid", "anders", "ebbe", "PER", "michael", "peter"]  
▶
```

Hint: du kan benytte `toLowerCase()` og `toUpperCase()`. 'OR' – operatoren i JavaScript er som i C#: '`||`'

Opgave (forEach - 2)

```
let cars = [
    {brand: 'VW', model: 'Passat', fuel: 'diesel', owner_tax: 5550 },
    {brand: 'VW', model: 'Passat', fuel: 'gasoline', owner_tax: 460},
    {brand: 'VW', model: 'Passat', fuel: 'hybrid', owner_tax: 150},
    {brand: 'BMW', model: '320i', fuel: 'diesel', owner_tax: 4280},
    {brand: 'BMW', model: '320i', fuel: 'gasoline', owner_tax: 430},
    {brand: 'BMW', model: '320i', fuel: 'hybrid', owner_tax: 210},
    {brand: 'Tesla', model: 'S', fuel: 'electric', owner_tax: 0 }
]

var increaseOwnerTax = function(cars, fuel, taxPct){
    // .....
}

increaseOwnerTax(cars, 'diesel', 50);
increaseOwnerTax(cars, 'hybrid', 5);
console.log(cars);
```

Udfyld funktionen `increaseOwnerTax(cars, fuel, taxPct)`, så den gennemløber et array af Car-objekter og hæver `owner_tax` med `taxPct` hvis bilens brændstoftype matcher `fuel`, se:

```

▼ (7) [..., {}, {}, {}, {}, {}, {}] ⓘ
▶ 0: {brand: "VW", model: "Passat", fuel: "diesel", owner_tax: 8325}
▶ 1: {brand: "VW", model: "Passat", fuel: "gasoline", owner_tax: 460}
▶ 2: {brand: "VW", model: "Passat", fuel: "hybrid", owner_tax: 157.5}
▶ 3: {brand: "BMW", model: "320i", fuel: "diesel", owner_tax: 6420}
▶ 4: {brand: "BMW", model: "320i", fuel: "gasoline", owner_tax: 430}
▶ 5: {brand: "BMW", model: "320i", fuel: "hybrid", owner_tax: 220.5}
▶ 6: {brand: "Tesla", model: "S", fuel: "electric", owner_tax: 0}
  length: 7
▶ __proto__: Array(0)

```

Opgave (map - 1)

Givet følgende array: cars , af car objekter:

```

let cars = [
  {brand: 'VW', model: 'Passat', fuel: 'diesel', owner_tax: 5550 },
  {brand: 'VW', model: 'Passat', fuel: 'gasoline', owner_tax: 460},
  {brand: 'VW', model: 'Passat', fuel: 'hybrid', owner_tax: 150},
  {brand: 'BMW', model: '320i', fuel: 'diesel', owner_tax: 4280},
  {brand: 'BMW', model: '320i', fuel: 'gasoline', owner_tax: 430},
  {brand: 'BMW', model: '320i', fuel: 'hybrid', owner_tax: 210},
  {brand: 'Tesla', model: 'S', fuel: 'electric', owner_tax: 0 }
]

```

- a) Benyt **map**-metoden til at mappe arrayet cars til et array: carModels der indeholder alle modellerne:

```

▶ (7) ["Passat", "Passat", "Passat", "320i", "320i", "320i", "S"]

```

- b) Benyt **map**-metoden til at mappe arrayet cars til et array: carBrands der indeholder objekter med key: 'Mærke' og value: brand:

```

▼ (7) [..., {}, {}, {}, {}, {}, {}] ⓘ
▶ 0: {"Mærke": "VW"}
▶ 1: {"Mærke": "VW"}
▶ 2: {"Mærke": "VW"}
▶ 3: {"Mærke": "BMW"}
▶ 4: {"Mærke": "BMW"}
▶ 5: {"Mærke": "BMW"}
▶ 6: {"Mærke": "Tesla"}
  length: 7
▶ __proto__: Array(0)

```

Opgave (map - 2)

Givet følgende string *drivers*, der indeholder et array af JS-objekter i json-format:

```
// Data, fra: http://ergast.com/api/f1/2018/drivers.json?callback=drivers
var drivers = [
    {
        "driverId": "grosjean",
        "permanentNumber": "8",
        "code": "GRO",
        "url": "http://en.wikipedia.org/wiki/Romain_Grosjean",
        "givenName": "Romain",
        "familyName": "Grosjean",
        "dateOfBirth": "1986-04-17",
        "nationality": "French"
    },
    {
        "driverId": "hamilton",
        "permanentNumber": "44",
        "code": "HAM",
        "url": "http://en.wikipedia.org/wiki/Lewis_Hamilton",
        "givenName": "Lewis",
        "familyName": "Hamilton",
        "dateOfBirth": "1985-01-07",
        "nationality": "British"
    },
    {
        "driverId": "hulkenberg",
        "permanentNumber": "27",
        "code": "HUL",
        "url": "http://en.wikipedia.org/wiki/Nico_H%C3%BClkenberg",
        "givenName": "Nico",
        "familyName": "Hülkenberg",
        "dateOfBirth": "1987-08-19",
        "nationality": "German"
    },
    {
        "driverId": "kevin_magnussen",
        "permanentNumber": "20",
        "code": "MAG",
        "url": "http://en.wikipedia.org/wiki/Kevin_Magnussen",
        "givenName": "Kevin",
        "familyName": "Magnussen",
        "dateOfBirth": "1992-10-05",
    }
]
```

```
        "nationality": "Danish"
    }
]`;
```

- a) Konverter tekst-strengen: *drivers* til et array: *f1Drivers* af JS-objekter.
b) Map arrayet: *f1Drivers* til et nyt array: *myDrivers* med objekter af følgende struktur:

{Kode : "MAG", Fornavn : "Kevin", Efternavn : "Magnussen"}

Resultatet skulle gerne se således ud:

```
console.log(myDrivers);
```

```
ArrayHelperMethods.js:183
▼ (4) [..., ..., ...]
  ▶ 0: {Kode: "GRO", Fornavn: "Romain", Efternavn: "Grosjean"}
  ▶ 1: {Kode: "HAM", Fornavn: "Lewis", Efternavn: "Hamilton"}
  ▶ 2: {Kode: "HUL", Fornavn: "Nico", Efternavn: "Hülkenberg"}
  ▶ 3: {Kode: "MAG", Fornavn: "Kevin", Efternavn: "Magnussen"}
    length: 4
  ▶ __proto__: Array(0)
>
```

Opgave (filter - 1)

Givet arrayet *cars* fra ovenstående eksempel. Find vha. **filter**-metoden, alle de hybrid biler der har en grøn afgift under 200.

Resultatet skal se ud som følgende:

```
Array(1) ⓘ
  ▶ 0: {brand: "VW", model: "Passat", fuel: "hybrid", owner_tax: 157.5}
    length: 1
  ▶ __proto__: Array(0)
>
```

Opgave (filter - 2)

- a) Lav en funktion *fuelCriterium* der har *car* og *fuel* som parameter og som returnere *true* hvis værdien af *car.fuel* er det samme som værdien af *fuel*, ellers returneres *false*.
- b) Givet arrayet *cars* fra ovenstående opgave. Find vha. **filter**-metoden og ovenstående prædikat-funktion *fuelCriterium*, alle de biler der har "gasoline" som *fuel*.

Resultatet skal se ud som følgende:

```
ArrayHelperMethods.js:221
▼ Array(2) ⓘ
  ► 0: {brand: "VW", model: "Passat", fuel: "gasoline", owner_tax: 460}
  ► 1: {brand: "BMW", model: "320i", fuel: "gasoline", owner_tax: 430}
    length: 2
  ► __proto__: Array(0)
>
```

Opgave (reduce - 1)

Benyt **reduce** til at finde summen af alle *distance* og gem resultatet i variablen *totalDistance*:

```
var trips = [{distance : 48}, {distance : 12}, {distance : 6}];
var totalDistance =
```

(Forventet resultat: 66)

Opgave (reduce - 2)

Givet følgende kodestump, - udfyld callback-funktionen, så den giver det forventede resultat.

(Hint: husk at returnere *accumulate* objektet *acc*)

```
var desk = [
  {type: 'sitting'},
  {type: 'standing'},
  {type: 'sitting'},
  {type: 'sitting'},
  {type: 'standing'}
];

var deskTypes = desk.reduce((acc, desk) => {
  ...
}, {sitting : 0, standing : 0});

console.log(deskTypes);
```

```
▼ {sitting: 3, standing: 2} ⓘ
  sitting: 3
  standing: 2
  ► __proto__: Object
>
```

Opgave (reduce/find - 3 - lidt svær)

Lav en funktion *unique* der kan fjerne alle dubletter fra et array.

fx givet arrayet [1, 1, 2, 3, 4, 4] vil funktionen returnere [1,2,3,4]

```
var numbers = [1, 1, 2, 3, 4, 4];

function unique(array) {
  ...
}

console.log(unique(numbers));
```

Opgave (reduce - 3 - svær)

En hyppigt anvendt opgave til jobsamtaler er det "ballancerede parentes problem". Lav en funktion `balanceParen`s der tager en streng af parenteser som parameter og undersøger om parenteserne er ballancerede. Fx `balanceParen("(())")` -> true hvor `balanceParen(")()()` -> false.

(Hint: reduce virker kun på array, derfor konverter strengen til et array af char -fx med `string.split("")`)
(Hint: "!" konvertere number til boolean)

Opgave (filter, map og reduce - 1)

Givet nedenstående array `personnel`. Find den samlede `shootingScore` for "force users" der har en `shootingScore` større end 60. (Hint: benyt **filter**, **map** og **reduce** - forventet resultat: 152)

```
var 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,  
},  
];
```

Opgave (every - 1)

Givet ovenstående array *personnel* (opgave: filter, map og reduce - 1). Benyt **every**-metoden til at undersøge om alle personer i *personnel* har en *pilotingScore* der er større end eller lig med 20.

Opgave (some - 1)

Givet ovenstående array *personnel* (opgave: filter, map og reduce - 1). Benyt **some**-metoden til at undersøge om der i *personnel* findes personer som har en *shootingScore* større end 80 og som ikke bruger Force.

Opgave (Ekstra)

Ud over de viste array hjælpe metoder (**forEach**, **map**, **filter**, **find**, **every**, **some** og **reduce**) findes der en lang række flere fx **sort**, **values**, **keys**

Undersøg hvilke metoder der findes og afprøv dem fx på arrayet *personnel*.

(Hint: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array)