JavaScript

Flavien Bigi

lesentrecodeurs.com

flavien@lesentrecodeurs.com

hAppywAy#8369

Le plan

1. DOM les éléments
2. Array
3. Object
4. Dataset
5. Local/Session Storage et cookie

DOM

create / remove element

```
containerEl.innerHTML = "<div>une div</div>";
const newEl = document.createElement("div");
containerEl.append(newEl);
newEl.remove();
```

Exercice toaster

DOM

parentNode, childNodes, children, firstChild, lastChild, firstElementChild, lastElementChild

```
<div id="parent">
    <p name="child1">enfant 1</p>
    <input name="child2" />
</div>
```

Essayez l'ensemble des methodes et expliciter les différences entre childNodes et children, first/lastChild et first/lastElementChild

DOM

```
<div id="parent">
    <p name="child1">enfant 1</p>
    <input name="child2" />
    <!-- Comment -->
</div>
```

```
console.log("children", parentEl.children);
console.log("childNodes", parentEl.childNodes);
console.log("nextSibling", parentEl.childNodes[2].nextSibling);
console.log("previousSibling", parentEl.childNodes[2].previousSibling);
console.log("lastChild", parentEl.lastChild);
console.log("lastElementChild", parentEl.lastElementChild);
console.log("name", parentEl.childNodes[3].name);
console.log("nodeName", parentEl.childNodes[1].nodeName);
console.log("parentNode", parentEl.childNodes[3].parentNode);
console.log("nodeName", parentEl.childNodes[3].nodeName);
```

```
children
▼ HTMLCollection(2) [p, input, child1: p, child2: input]
  ▶ 0: p
  ▶ 1: input
  ▶ child1: p
  ▶ child2: input
    length: 2
  ▶ [[Prototype]]: HTMLCollection
childNodes
▼ NodeList(7) [text, p, text, input, text, comment, text]
  ▶ 0: text
  ▶ 1: p
  ▶ 2: text
  ▶ 3: input
  ▶ 4: text
  ▶ 5: comment
  ▶ 6: text
    length: 7
  ▶ [[Prototype]]: NodeList
nextSibling ▶ input
previousSibling ▶ p
lastChild ▶ #text
lastElementChild ▶ input
name child2
nodeName P
parentNode ▶ div#parent
nodeName INPUT
```

Array : create

instanciation d'un tableau

```
const anArray = [1, "Metz", {school: "MNS", group: "DW2"}];
const anArrayOld = new Array(1, "Metz", {school: "MNS", group: "DW2"});
console.log(anArray, anArrayOld); // [1, "Metz", {school: "MNS", group: "DW2"}]
```

Array : méthodes

Parcours d'un tableau

```
countries.forEach(function (country :string ) {
  console.log(country.toUpperCase());
});

console.log(countries.map(function (country :string ) {
  return country.toUpperCase();
}).join(", "));
```

Array : méthodes

Tri et concaténation

```
const countries = ["France", "Italie", "Espagne"];
countries.sort(); // mutable, procedure qui modifie la variable
console.log(countries);
```

```
const americanCountries = ["Canada", "États-Unis", "Mexique"];
console.log(countries.concat(americanCountries));
// [ 'Espagne', 'France', 'Italie', 'Canada', 'États-Unis', 'Mexique' ]
```

Array : méthodes

Dépiler la tête ou la queue

```
console.log(americanCountries.shift());
console.log(americanCountries);
console.log(americanCountries.pop());
console.log(americanCountries);
// Canada
// [ 'États-Unis', 'Mexique' ]
// Mexique
// [ 'États-Unis' ]
```

Array : méthodes

Ajouter un élément et connaître sa position

```
console.log(americanCountries.indexOf("Canada"));
// 0
```

```
countries.push("Allemagne");
console.log(countries);
// [ 'Espagne', 'France', 'Italie', 'Allemagne' ]
```

Array : méthodes

Prendre une section d'un tableau

```
americanCountries = ["Canada", "États-Unis", "Mexique"];
console.log(americanCountries.slice(0,2));
console.log(americanCountries);
// [ 'Canada', 'États-Unis' ]
// [ 'Canada', 'États-Unis', 'Mexique' ]
```

Exercice array class

Array : méthodes

Chercher dans un array avec find, filter et findIndex

```
const _europeanCountries = countries.map((country :string ) => ({
  name: country,
  continent: "europe"
}));
const _americanCountries = americanCountries.map((country :string ) => ({
  name: country,
  continent: "america"
}));
console.log(_europeanCountries, _americanCountries);
// [
//   { name: 'Espagne', continent: 'europe' },
//   { name: 'France', continent: 'europe' },
//   { name: 'Italie', continent: 'europe' },
//   { name: 'Allemagne', continent: 'europe' }
// ]
// [
//   { name: 'Canada', continent: 'america' },
//   { name: 'États-Unis', continent: 'america' },
//   { name: 'Mexique', continent: 'america' }
// ]
const _worldCountries = _europeanCountries.concat(_americanCountries);
console.log(_worldCountries);
// [
//   { name: 'Espagne', continent: 'europe' },
//   { name: 'France', continent: 'europe' },
//   { name: 'Italie', continent: 'europe' },
//   { name: 'Allemagne', continent: 'europe' },
//   { name: 'Canada', continent: 'america' },
//   { name: 'États-Unis', continent: 'america' },
//   { name: 'Mexique', continent: 'america' }
// ]
```

Array : méthodes

```
console.log(_worldCountries.filter(country => country.continent === "europe"));
// [
//   { name: 'Espagne', continent: 'europe' },
//   { name: 'France', continent: 'europe' },
//   { name: 'Italie', continent: 'europe' },
//   { name: 'Allemagne', continent: 'europe' }
// ]
console.log(_worldCountries.find(country => country.name === "France"));
// { name: 'France', continent: 'europe' }
const franceIndex = _worldCountries.findIndex(country => country.name === "France")
// 1
console.log(franceIndex);
_worldCountries.splice(franceIndex, 1)
console.log(_worldCountries);
// [
//   { name: 'Espagne', continent: 'europe' },
//   { name: 'Italie', continent: 'europe' },
//   { name: 'Allemagne', continent: 'europe' },
//   { name: 'Canada', continent: 'america' },
//   { name: 'États-Unis', continent: 'america' },
//   { name: 'Mexique', continent: 'america' }
// ]
```

Array : méthodes

Les objets

Pourquoi les objets ?

- Meilleure organisation du code
- Facilite le travail d’équipe
- Maintenabilité

Respect des principes S.O.L.I.D

Les objets

Initialisation

```
const isAnObject = ["oui", "non"];
console.log(typeof isAnObject); // object not array
console.log(typeof {}); // object
console.log(typeof null); // object
```

Les objets

Initialisation

```
const isAnObject = ["oui", "non"];
console.log(typeof isAnObject); // object not array
console.log(typeof {}); // object
console.log(typeof null); // object
```

Les objets

Assignation de valeurs à des attributs de l'objet

```
const isAnotherObject = {
  key: "value"
};
console.log(isAnotherObject);
// { key: 'value' }
isAnotherObject.secondKey = 10;
console.log(isAnotherObject);
// { key: 'value', secondKey: 10 }
delete isAnotherObject.secondKey;
console.log(isAnotherObject);
// { key: 'value' }
```

Les objets

Modification d'un objet

```
console.log(isAnObject[0]);
// oui
isAnObject.bam = 2;
console.log(isAnObject);
//   [ 'oui', 'non', bam: 2 ]
// isAnObject.0 = "pourquoi pas"; // error
isAnObject[0] = "pourquoi pas";
console.log(isAnObject);
// [ 'pourquoi pas', 'non', bam: 2 ]
isAnObject[isAnotherObject["key"]] = "Vraiment ?";
console.log(isAnObject);
// [ 'pourquoi pas', 'non', bam: 2, value: 'Vraiment ?' ]
```

Les objets

Système de gestion d'une école

- **Ecole**
- **Sessions scolaires**
- **Élèves**
- **Professeurs**

Les objets

Exemple

```
function OldSchool(name) {
  this.name = name;
  this.getName = function () {
    return this.name;
  }

  this.createSession = function (name, year) {
    return new Session(name, year);
  }
}
```

“this” fait référence au context, à l’objet manipulé lui même.

Les objets

```
class School {
  #name;
  #sessions = [];

  constructor(name) {
    this.setName(name);
  }

  getName () {
    return this.#name;
  }

  setName (name) {
    this.#name = name;
  }

  createSession(name, year) {
    const session = new Session(name, year);
    // const session = {name, year};
    this.#sessions.push(session)
    return session;
  }

  getSessions () {
    return this.#sessions;
  }
}
```

pour les attributs privés
Ils ne sont pas accessible
en dehors de la classe

Les objets

Exemple

```
class Session {
  #name;
  #year;

  constructor(name, year) {
    this.setName(name);
    this.setYear(year);
  }

  setName (name) {
    this.#name = name;
  }

  getName () {
    return this.#name;
  }

  setYear (year) {
    this.#year = year;
  }

  getYear () {
    return this.#year;
  }
}
```

Les objets

```
class Person {
  #firstName;
  #lastName;
  #birthday;

  constructor(firstName, lastName, birthday) {
    this.#firstName = firstName;
    this.#lastName = lastName;
    this.#birthday = birthday;
  }
}
```

Héritage avec le mot clé extends

```
class Student extends Person {
  #session;
  constructor(firstName, lastName, birthday) {
    super(firstName, lastName, birthday);
  }

  setSession(session) {
    this.#session = session;
  }
}

class Teacher extends Person {
  #sessions = [];
  constructor(firstName, lastName, birthday) {
    super(firstName, lastName, birthday);
  }

  setSession(session) {
    session.teacher = this;
    this.#sessions.push(session);
  }

  getSessions() {
    return this.#sessions;
  }
}
```

Les objets

```
const mns = new School("Metz Numeric School");
const ifa = new OldSchool("Institut Francais des Affaires");
console.log(mns.getName(), mns.name, ifa.getName());
// Metz Numeric School undefined Institut Francais des Affaires
mns.setName("MNS");
mns.name = "MNM";
console.log(mns.getName());
// MNS
const session = mns.createSession("DW1", 2022);
console.log(session.getName());
// DW1
console.log(mns.getSessions().map(function (session) {
  return session.getName();
}));
// [ 'DW1' ]
session.setName("DW2");
console.log(mns.getSessions().map(function (session) {
  return session.getName();
}));
// [ 'DW2' ]
```

Dataset

data-id sur une balise HTML

```
<button data-school-name="${schoolName}" class="addSession">Add</button>
```

```
const schoolName = schoolFormEl.dataset.schoolName;
```

En js, je peux récupérer le dataset avec “dataset”.
Attention, écriture camelCase !!

Local Storage

Vous pouvez sauver des données dans le navigateur client.
Local Storage est un système de gestion de données par clés, auxquels vous pouvez associer des valeurs en string

```
localStorage.setItem("key", "value");
const value = localStorage.getItem("key");
```

The syntax for removing the `localStorage` item is as follows:

```
localStorage.removeItem('myCat');
```

The syntax for removing all the `localStorage` items is as follows:

```
localStorage.clear();
```

Session Storage

Session Storage est un système de gestion de données par clés, auxquels vous pouvez associer des valeurs en string.
La différence avec le local storage est que la session storage est effacée une fois que la session est terminée, c'est à dire une fois que le navigateur client est fermé.

```
sessionStorage.setItem("key", "value");
const value = sessionStorage.getItem("key");
```

Pour stocker des objets, vous pouvez utiliser JSON.stringify puis les récupérer avec JSON.parse

Cookies

Par défaut, un cookie n’est accessible que d’un seul domaine.
Un cookie consite à une chaine de texte associé à une clé. On peut mettre une date d’expiration à ce cookie.
Pour récupérer un cookie en particulier, il faut “spliter” la chaine venant de document.cookie.

Exercice cookie

Cookies

```
document.cookie = "key1=Hello;";
document.cookie = "key2=World; expires=Thu, 01 Jan 2023 00:00:00 UTC;";
document.cookie = "key1=Foo; expires=Thu, 01 Jan 1970 00:00:00 UTC;"
console.log(document.cookie);
```

```
> document.cookie = "key1=Hello;";
<· 'key1=Hello;'
> document.cookie = "key2=World; expires=Thu, 01 Jan 2023 00:00:00 UTC;";
<· 'key2=World; expires=Thu, 01 Jan 2023 00:00:00 UTC;'
> document.cookie
<· 'key1=Hello; key2=World; _ga=GA1.2.1936481554.1644751507; _gid=GA1.2.197299065
   0.1644751507; _gat=1; ips4_ipsTimezone=America/New_York; ips4_hasJS=true; AWSA
   LB=UZ+A0yliC7x8OyqpKYrIvqbyj2faIiz4WOKIqFahV+qG74ujz6X4xOoPYBCAxwPjAaXSis0uU4m
   OE2yE4JlsVelLW/n3pInrEnj8DxjeI49Xq4aacYaRiOCgHcKA'
> document.cookie = "key1=Foo; expires=Thu, 01 Jan 1970 00:00:00 UTC;"
<· 'key1=Foo; expires=Thu, 01 Jan 1970 00:00:00 UTC;'
> document.cookie
<· 'key2=World; _ga=GA1.2.1936481554.1644751507; _gid=GA1.2.1972990650.164475150
   7; _gat=1; ips4_ipsTimezone=America/New_York; ips4_hasJS=true; AWSALB=UZ+A0yli
   C7x8OyqpKYrIvqbyj2faIiz4WOKIqFahV+qG74ujz6X4xOoPYBCAxwPjAaXSis0uU4mOE2yE4JlsVe
   lLW/n3pInrEnj8DxjeI49Xq4aacYaRiOCgHcKA'
```