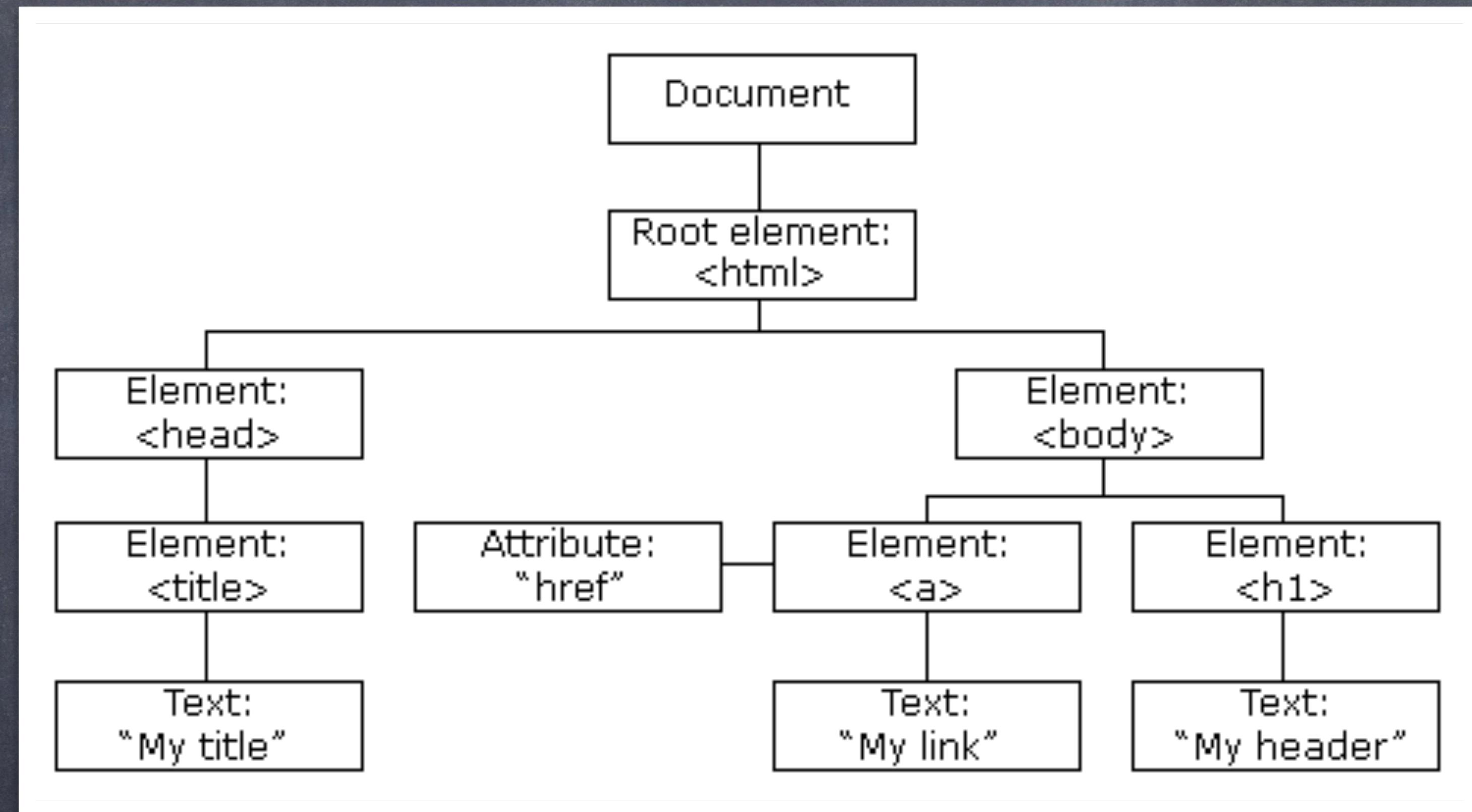


Javascript DOM

Représentation de la structure et du contenu
d'une page HTML sous la forme d'un arbre d'objets



- ✓ Modifier dynamiquement des éléments (balises) ou des attributs (class, style ...)
- ✓ Supprimer dynamiquement des éléments ou attributs
- ✓ Ajouter dynamiquement des éléments ou attributs
- ✓ Capturer des événements

Cibler des éléments

méthode	description
<code>getElementById(id)</code>	Retourne le noeud identifié par l'Id passé en argument
<code>getElementsByTagName(tag)</code>	Retourne la liste des noeuds (nodeList) identifiés par la balise passée en argument
<code>getElementByClassName(class)</code>	Retourne la liste des noeuds (nodeList) identifiés par la classe passée en argument
<code>getElementsByName(name)</code>	Retourne la liste des noeuds (nodeList) identifiés par l'attribut name passé en argument
<code>querySelector(selector)</code>	Retourne le premier noeud identifié par le sélecteur CSS passé en argument
<code>querySelectorAll(selector)</code>	Retourne la liste des noeuds (nodeList) identifiés par le sélecteur CSS passé en argument

propriété	description
innerHTML	Lit ou modifie le code html interne d'un balise
innerText	Lit ou modifie le texte (sans les balises) à l'intérieur d'une balise
outerHTML	Lit ou modifie le code d'une balise
outerText	En lecture se comporte comme innerText En écriture le nouveau contenu remplace la balise

```
<div id="innerHTML"><p>Hello</p><p>World</p></div>
<div id="innerText"><p>Hello</p><p>World</p></div>
<div id="outerHTML"><p>Hello</p><p>World</p></div>

<script>
    const innerHTMLDiv = document.getElementById("innerHTML");
    const innerTextDiv = document.getElementById("innerText");
    const outerHTMLDiv = document.getElementById("outerHTML");

    // Lecture
    console.log("innerHTML lecture");
    console.log(innerHTMLDiv.innerHTML);
    console.log("-----");

    console.log("innerText lecture");
    console.log(innerTextDiv.innerText);
    console.log("-----");

    console.log("outerHTML lecture");
    console.log(outerHTMLDiv.outerHTML);

    console.log("=====")
    // Ecriture
    console.log("innerHTML Ecriture");
    innerHTMLDiv.innerHTML = "<h1>Hello</h1>";
    console.log(innerHTMLDiv.innerHTML);
    console.log("-----");

    console.log("innerText Ecriture");
    innerTextDiv.innerText = "<h1>Hello</h1>";
    console.log(innerTextDiv.innerText);
    console.log("-----");

    console.log("outerHTML Ecriture");
    outerHTMLDiv.outerHTML = "<h1>Hello</h1>";
    console.log(outerHTMLDiv.outerHTML);
</script>
```

innerHTML lecture

<p>Hello</p><p>World</p>

innerText lecture

Hello

World

outerHTML lecture

<div id="innerHTML"><p>Hello</p><p>World</p></div>

=====

innerHTML Ecriture

<h1>Hello</h1>

innerText Ecriture

<h1>Hello</h1>

outerHTML Ecriture

<div id="innerHTML"><h1>Hello</h1></div>



Une page html

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>DOM</title>
</head>
<body id="page">

<h2>Les langues</h2>
<ul id="languages">
    <li class="item">Français</li>
    <li class="item">Anglais</li>
    <li class="item">Japonais</li>
</ul>

<h2>Les pays</h2>
<ul id="countries">
    <li class="item">France</li>
    <li class="item">Royaume Uni</li>
    <li class="item">Japon</li>
</ul>
</body>
</html>
```

```
const h2 = document.querySelectorAll("h2");
// Remplacement de la balise
h2[0].outerHTML = "<h5>Liste de langues</h5>";
// Modification du contenu de la balise
h2[1].innerHTML = "<em>Liste des pays</em>";

const countries = document.getElementById("countries");
// Suppression des éléments de la liste
countries.innerHTML = "";

const languagesItems = document.querySelectorAll("#languages li");
// Ajout d'une étoile à chaque élément
for (item of languagesItems){
    item.innerText += "*";
}
```

- ✓ De récupérer la liste des pays dans un tableau Javascript
- ✓ De mettre en italique (em) les éléments de classe item qui commencent par fr
- ✓ D'effacer tout le contenu de la page
- ✓ Dupliquer la liste des pays et la placer au début de la page
- ✓ Trier la liste des langues

sélecteur.style.propriété

La propriété CSS est camélisée
font-size devient fontSize

```
// Modification des titres
const h2 = document.querySelectorAll("h2");
for(let title of h2){
    title.style.fontSize = "68px";
}
// Modification de la liste des pays
const countries = document.getElementById("countries");
countries.style.listStyleType = "square";
```

- ✓ De mettre en rouge les trois pays
- ✓ De mettre en gras le premier élément de chaque liste
- ✓ De faire disparaître la liste des langues

<code>setAttribute(name, value)</code>	change l'attribut name et lui affecte la valeur value
<code>getAttribute(name)</code>	retourne la valeur de l'attribut dont le nom est passé en argument
<code>hasAttribute(name)</code>	retourne true si l'élément possède l'attribut dont le nom est passé en argument
<code>removeAttribute(name)</code>	supprime un attribut
<code>attributes</code>	tableau indicé de tous les attributs d'un élément

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>Attributs HTML</title>
</head>
<body>
    <form>
        <div class="form-group">
            <label for="password">Mot de passe</label>
            <input type="password" name="pwd" id="password">
            <p class="message"></p>
        </div>
        <div class="form-group">
            <label for="password">Mot de passe</label>
            <input type="text" name="login" id="login" required>
            <p class="message"></p>
        </div>
        <div class="form-group">
            <label for="languages">Mot de passe</label>
            <select name="languages" id="languages">
                <option>Javascript</option>
                <option>Python</option>
                <option>C++</option>
                <option>Dart</option>
            </select>
        </div>
    </form>
</body>
</html>
```

```


<ul id="js-ressources">
    <li>
        <a href="https://developer.mozilla.org/">
            MDN
        </a>
    </li>
    <li>
        <a href="http://jsforcats.com/">
            Javascript for cats
        </a>
    </li>
    <li>
        <a href="https://eloquentjavascript.net/">
            Eloquent Javascript
        </a>
    </li>
    <li>
        <a href="ma-page.html">
            Eloquent Javascript
        </a>
    </li>
</ul>

</body>
</html>
```

```
// Afficher le mot de passe
const passwordInput = document.getElementById("password");
passwordInput.setAttribute("type", "text");

// Obtenir les options de la liste déroulante
const languageList = document.querySelectorAll("#languages option");
const list = [];
for(let item of languageList){
    list.push(item.innerText);
}
console.log(list);

// Ajouter un attribut target aux liens externes uniquement
const links = document.querySelectorAll("a");
for (let item of links){
    if(item.getAttribute(href).startsWith("http")){
        item.setAttribute("target", "_blank");
    }
}
```

- ✓ De récupérer dans un tableau la liste des liens
- ✓ De changer la source de l'image
- ✓ De supprimer l'attribut required du champ de login

className	retourne une chaîne de caractère correspondant à la valeur de l'attribut class de l'élément
classList	retourne un tableau des classes affectées à l'élément (DOMTokenList)
classList.contains(className)	retourne true si l'élément possède la classe passée en argument
classList.add(className)	ajoute la classe passée en argument
classList.remove(className)	supprime la classe passée en argument
classList.toggle(className)	bascule la classe passée en argument, si elle existe elle est supprimée sinon elle est ajoutée

Les événements

```
element.onEventName = eventHandler
```

Une fonction (callback) associée à un événement
et qui ne se déclenche que lorsque l'événement survient

```
const bt = document.getElementById("bouton");
bt.onclick = function () {
    alert("cliqué");
}
```

```
const bt = document.getElementById("bouton");

function btClickHandler(){
    alert("cliqué");
}

bt.onclick = btClickHandler;
```

onclick	clic de la souris
ondblclick	double clic
onmouseup	bouton de la souris relâché
onmousedown	bouton de la souris enfoncé
onmouseover	le pointeur de la souris est sur l'élément
onmouseout	le pointeur de la souris quitte l'élément
onmousemove	le pointeur de la souris bouge
oncontextmenu	clic avec le bouton droit

onfocus	focus sur un champ de saisie
onblur	perte du focus
onchange	modification d'un contrôle de formulaire
onsubmit	soumission d'un formulaire
onreset	annulation de la saisie d'un formulaire
onscroll	déplacement de la barre de défilement d'un élément
onload	chargement d'une ressource (window, image, link...)
onkeydown	quand on appuie sur une touche
onkeyup	quand on relâche une touche
onkeypress	tant qu'une touche est enfoncée
onload	la fin du chargement de la page

```
// Avant de cibler des éléments on attend le chargement
// du DOM on peut donc placer ce code avant body
window.onload = function(){
    const bt = document.getElementById("bt");
    console.log(bt);
}
```

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>Events</title>

    <script>
        window.onload = function(){
            console.log("chargement terminé");

            const disableBt = document.getElementById("disableButton");
            const buttonList = document.querySelectorAll("#buttons button");

            disableBt.onclick = function(){
                for(item of buttonList){
                    item.setAttribute("disabled", "");
                }
            }
        }
    </script>
</head>
<body>
    <button id="disableButton">Désactiver</button>
    <div id="buttons">
        <button>bt 1</button>
        <button>bt 2</button>
        <button>bt 3</button>
    </div>
</body>
</html>
```

Essayer de mettre en place
un comportement de bascule

quand on clique sur le bouton désactiver,
une première fois son texte passe à activer
et il désactive les boutons

La seconde fois le texte repasse à
désactiver et on active les boutons
(suppression de l'attribut disabled)

Stockage de données dans une balise

```
window.onload = function(){
    const bt = document.getElementById("bouton");
    bt.onclick = function(){
        // récupération du nombre de clics
        let nbClicks = bt.getAttribute("data-click-count") || 0;
        // la couleur du bouton
        let color = "green";
        // incrémentation du nombre de clicks
        nbClicks = parseInt(nbClicks) + 1;

        // Définition de la nouvelle couleur
        if(nbClicks > 4){
            color = "red";
        } else if(nbClicks > 3){
            color = "orange";
        } else if (nbClicks > 2){
            color = "yellow";
        }

        console.log(nbClicks);
        // Changement de couleur du bouton
        bt.style.backgroundColor = color;
        // Ecriture du nombre de clicks dans le bouton
        bt.setAttribute("data-click-count", nbClicks);
    }
}
```

Ecouteurs d'événements

- ✓ Même effet que les gestionnaires d'événements mais syntaxe différente
- ✓ Un élément peut avoir plusieurs écouteurs pour le même type événement

Ecouter un événement

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Event listener</title>
  <script>

    function init(){
      document.getElementById("bt1")
        .addEventListener("click", function(){
          alert("cliqué");
        });
    }

    window.addEventListener('load', init);
  </script>
</head>
<body>

<button id="bt1">Cliquez</button>

</body>
</html>
```

pas de "on" avant le nom de l'événement ici,
on utilise **click** et non **onclick**

Créer une page avec un bouton et un paragraphe de texte, au clic sur le bouton masquer le paragraphe

Créer une page avec une image, au survol de l'image changer celle ci pour une autre image, à la sortie du curseur restaurer l'ancienne image

Créer une page et trois feuilles de style différentes.
Sur la page ajouter trois boutons qui permettront de changer la feuille de style

L'objet Event

- ✓ Un objet passé en argument des gestionnaires d'évènements
- ✓ Indique le contexte de l'évènement

```
const bt1 = document.getElementById("bt1");
bt1.onclick = function (event) {
    console.log(event);
};
```

```
▼ PointerEvent {isTrusted: true, pointerId: 1, width: 1, height: 1, pressure: 0, ...} ⓘ
  altKey: false
  altitudeAngle: 1.5707963267948966
  azimuthAngle: 0
  bubbles: true
  button: 0
  buttons: 0
  cancelBubble: false
  cancelable: true
  clientX: 49
  clientY: 19
  composed: true
  ctrlKey: false
  currentTarget: null
  defaultPrevented: false
  detail: 1
  eventPhase: 0
  fromElement: null
  height: 1
  isPrimary: false
  isTrusted: true
  layerX: 49
  layerY: 19
  metaKey: false
  movementX: 0
  movementY: 0
  offsetX: 40
  offsetY: 9
  pageX: 49
  pageY: 19
  ▶ path: (5) [button#bt1, body, html, document, Window]
    ↳ pointerId: 1
    ↳ pointerType: "mouse"
    ↳ pressure: 0
    ↳ relatedTarget: null
    ↳ returnValue: true
    ↳ screenX: 1193
    ↳ screenY: 130
    ↳ shiftKey: false
  ▶ sourceCapabilities: InputDeviceCapabilities {firesTouchEvents: false}
  ▶ srcElement: button#bt1
  ▶ tangentialPressure: 0
  ▶ target: button#bt1
  ▶ tiltX: 0
  ▶ tiltY: 0
  ▶ timeStamp: 6441.799999952316
  ▶ toElement: null
  ▶ twist: 0
  ▶ type: "click"
  ▶ view: Window {window: Window, self: Window, document: document, name: '', location: Location, ...}
  ▶ which: 1
  ▶ width: 1
  ▶ x: 49
  ▶ y: 19
  ▶ [[Prototype]]: PointerEvent
```

target	l'élément qui a déclenché l'événement
type	le nom de l'événement
timeStamp	horodatage de l'événement (en ms depuis le 01/01/1970)

Quelques propriétés de PointerEvent

clientX, clientY	Les coordonnées X et Y absolue par rapport à la fenêtre visible
pageX, pageY	Les coordonnées X et Y par rapport au document
button	le bouton de la souris qui a été enfoncé (0 : gauche, 1 : droit, 2 : roue)
which	le bouton de la souris qui a été enfoncé (0 : aucun, 1 : gauche, 2 : roue, 3 : droit)
detail	le nombre de clics successif dans un faible laps de temps
altKey, ctrlKey, shiftKey	true si la touche ALT, CTRL ou SHIFT était enfoncée lors du clic

```
const buttons = document.querySelectorAll("button");
for(let item of buttons) {
    item.onclick = function (evt) {
        evt.target.innerText += " cliqué";
    };
}
```

Créer une page et trois feuilles de style différentes.
Sur la page ajouter trois boutons qui permettront de
changer la feuille de style (déjà fait)

Peut-on faire ça avec une seule fonction ?

Essayer de changer l'apparence du bouton actif
(celui qui correspond à la feuille de style en cours)

```
<table>
  <tr>
    <td>Java</td>
    <td><button>Sélectionner</button></td>
  </tr>
  <tr>
    <td>Python</td>
    <td><button>Sélectionner</button></td>
  </tr>
  <tr>
    <td>C++</td>
    <td><button>Sélectionner</button></td>
  </tr>
</table>
```

Changer la couleur de fond
de la ligne (tr)
quand on clique
sur un bouton

```
const buttons = document.querySelectorAll("table button");

clickHandler = function(even) {
    // On sélectionne le parent du parent
    // donc le tr
    even.path[2].style.backgroundColor = "yellow";
}

// Affectation de clickHandler à tous les boutons
for(let item of buttons) {
    item.onclick = clickHandler;
}
```

```
window.onmousemove = function (evt) {  
    console.log("clientX = " + evt.clientX);  
    console.log("pageX = " + evt.pageX);  
  
    console.log("clientY = " + evt.clientY);  
    console.log("pageY = " + evt.pageY);  
};
```

Insérer une image dans la page et
déplacer cette image
avec le curseur de la souris

```
position: absolute;  
left: 5px;  
top: 5px;
```

```
const link = document.querySelector("a");

link.onclick = function (event) {
    // désactive le comportement par défaut
    event.preventDefault();
    alert(event.target.href);
}
```

Désactiver le clic droit sur les images

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>Events</title>

    <script>
        window.onload = function () {
            const imgList = document.querySelectorAll("img");

            const imgClickHandler = function (event) {
                // désactive le comportement par défaut
                event.preventDefault();
                alert("téléchargement des images interdit");
            }

            for(let item of imgList){
                item.oncontextmenu = imgClickHandler;
            }
        }
    </script>
</head>
<body>

</body>
</html>
```

Propagation des événements

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>Events</title>

    <style>
        #inner {
            width: 200px;
            height: 200px;
            background-color: darkgreen;
        }

        #outer {
            padding: 100px;
            width: 500px;
            height: 500px;
            background-color: darkred;
        }
    </style>
</head>
<body>
    <div id="outer">
        <div id="inner"></div>
    </div>
</body>
</html>
```

```
<script>
    window.onload = function () {
        const inner = document.getElementById("inner");
        const outer = document.getElementById("outer");

        inner.onclick = function (evt) {
            evt.stopPropagation();
            alert("inner");
        };

        outer.onclick = function (evt) {
            alert("outer");
        };
    }
</script>
```

Événements clavier

Javascript DOM L'objet KeyboardEvent

```
KeyboardEvent {isTrusted: true, key: 'a', code: 'KeyQ', location: 0, ctrlKey:  
false, ...} ⓘ  
  altKey: false  
  bubbles: true  
  cancelBubble: false  
  cancelable: true  
  charCode: 0  
  code: "KeyQ"  
  composed: true  
  ctrlKey: false  
  currentTarget: null  
  defaultPrevented: false  
  detail: 0  
  eventPhase: 0  
  isComposing: false  
  isTrusted: true  
  key: "a"  
  keyCode: 65  
  location: 0  
  metaKey: false  
▶ path: (4) [body, html, document, Window]  
  repeat: false  
  returnValue: true  
  shiftKey: false  
▶ sourceCapabilities: InputDeviceCapabilities {firesTouchEvents: false}  
▶ srcElement: body  
▶ target: body  
  timeStamp: 136239.2999995232  
  type: "keyup"  
▶ view: Window {window: Window, self: Window, document: document, name: '', ...}  
  which: 65  
▶ [[Prototype]]: KeyboardEvent
```

```
// Touche enfoncée  
window.onkeydown = function (event) {  
    console.log(event);  
}  
// Touche relâchée  
window.onkeyup = function (event) {  
    console.log(event);  
}  
  
// Touche pressée  
// l'événement se déclenche  
// plusieurs fois  
window.onkeypress = function (event) {  
    console.log(event);  
}
```

```
window.onload = function () {
    const blockElement = document.getElementById("block");
    // Touche enfoncée
    window.onkeydown = function (event) {
        if(event.key === "o" && event.ctrlKey){
            // Affiche le contenu avec CTRL + o
            blockElement.style.display= "block";
        } else if(event.key === "f" && event.ctrlKey){
            // Masque le contenu avec CTRL + f
            blockElement.style.display= "none";
        }
    }
}
```

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>square</title>

    <style>
        #square {
            position: absolute;
            width: 50px;
            height: 50px;
            background-color: palegreen;
        }
    </style>
</head>
<body>
    <div id="square"></div>
</body>
</html>
```

```
<script>
    window.onload = function () {
        const square = {
            target : document.getElementById("square"),
            x: 5,
            y: 5,
            show : function () {
                this.target.style.left = this.x + "px";
                this.target.style.top = this.y + "px";
            }
        }

        window.onkeydown = function (event) {
            if(event.key === "ArrowRight"){
                square.x += 1;
            } else if(event.key === "ArrowLeft"){
                square.x -= 1;
            }

            square.show();
        }
    }
</script>
```

Délégation d'événements

Tester ce code

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>Fabrique de boutons</title>

    <style>
        .btn {
            margin: 10px;
        }
    </style>
    <script>
        window.onload = function () {
            const bt = document.querySelector("button");
            const body = document.querySelector("body");
            let numberOfButtons = 1;
            bt.onclick = function (event) {
                const newButton = event.target.cloneNode();
                numberOfButtons++;
                newButton.innerHTML = "bouton " + numberOfButtons;
                body.appendChild(newButton);
            }
        }
    </script>
</head>
<body>
    <button class="btn">spawn</button>
</body>
</html>
```

- ✓ Le clic sur spawn génère un nouveau bouton
- ✓ Les nouveaux boutons eux ne font rien quand on clique dessus
- ✓ Il n'existaient pas quand on a défini la cible et l'événement

Solution

Placer l'événement sur le parent du bouton

```
window.onload = function () {
    const bt = document.querySelector("button");
    const body = document.querySelector("body");
    let numberOfButtons = 1;

    // Evénement sur tous les clics sur body
    body.onclick = function (event) {
        // On ne réagit que si la cible
        // est un élément de type button
        if(event.target.localName === "button"){
            const newButton = event.target.cloneNode();
            numberOfButtons++;
            newButton.innerHTML = "bouton " + numberOfButtons;
            body.appendChild(newButton);
        }
    }
}
```

Les formulaires

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>Form</title>

    <script>
        window.onload = function () {
            const form = document.getElementById("form");
            const wordInput = document.getElementById("word");

            form.onsubmit = function (even) {
                // empêcher l'envoi du formulaire
                even.preventDefault();

                // Récupération de la saisie
                const word = wordInput.value;

                alert(word);
            }
        }
    </script>
</head>
<body>
    <form method="post" id="form">
        <input type="text" placeholder="un mot" id="word">
        <button type="submit">Valider</button>
    </form>
</body>
</html>
```

On doit cibler tous les inputs
ce qui va vite devenir fastidieux
sur un gros formulaire

Traitement d'un formulaire

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Form</title>
  <script>
    window.onload = function () {
      const form = document.getElementById("form");
      const wordInput = document.getElementById("word");

      form.onsubmit = function (even) {
        // empêcher l'envoi du formulaire
        even.preventDefault();

        // Récupération de toutes les saisies
        const data = new FormData(even.target);
        // boucle sur les saisies
        for (item of data.entries()){
          console.log(item);
        }
      }
    }
  </script>
</head>
<body>
  <form method="post" id="form">
    <input type="text" placeholder="un mot" name="word">
    <input type="text" placeholder="prénom" name="firstName">
    <input type="text" placeholder="nom" name="name">
    <input type="number" placeholder="age" name="age">
    <button type="submit">Valider</button>
  </form>
</body>
</html>
```

- E ["word", "stylo"] (2)
- E ["firstName", "Sébastien"] (2)
- E ["name", "Maloron"] (2)
- E ["age", "51"] (2)

Retourne une suite de tableaux ordinaux
avec en clef 0 le nom de la saisie
et en clef 1 sa valeur

```
function getFormDataAsObject(form) {
    const object = {};
    const data = new FormData(form).entries();
    for(let input of data){
        const key = input[0];
        const val = input[1];
        object[key] = val;
    }
    return object;
}
```

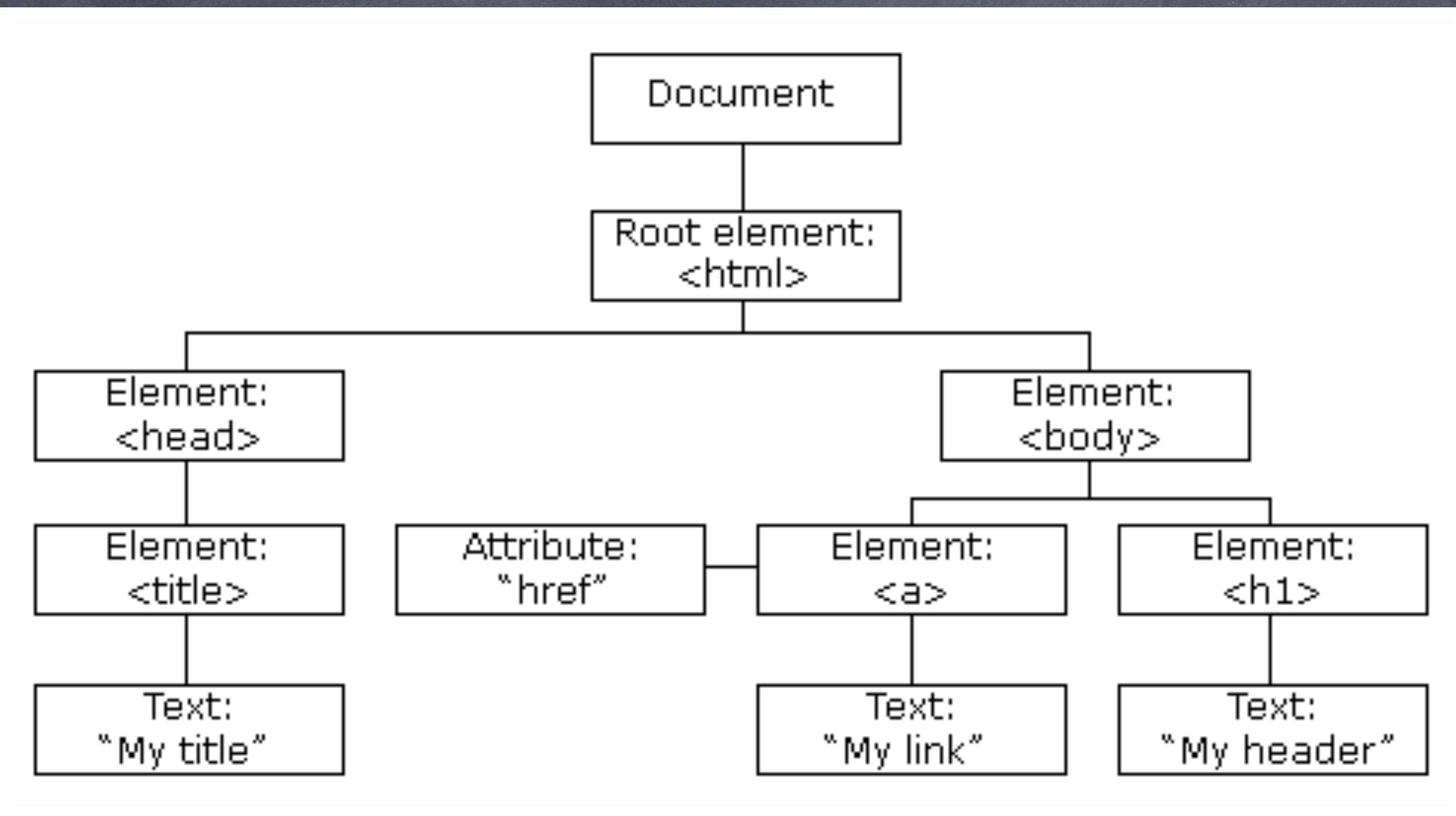
```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>Form</title>
    <script>
        window.onload = function () {
            const form = document.getElementById("form");
            const wordInput = document.getElementById("word");

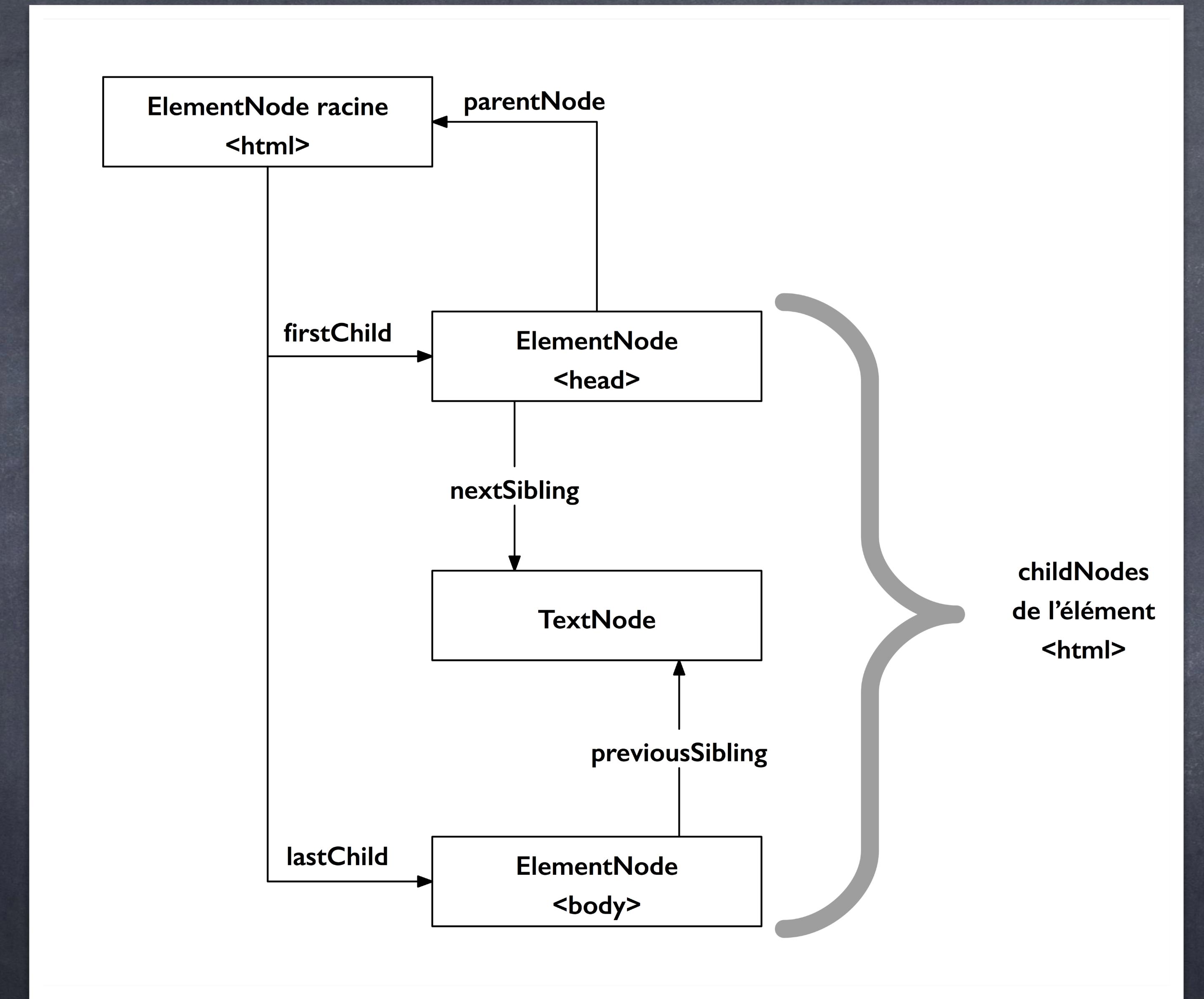
            form.onsubmit = function (even) {
                // empêcher l'envoi du formulaire
                even.preventDefault();

                // Récupération de toutes les saisies
                const data = getFormDataAsObject(event.target);
                console.log(data);
            }
        }
    </script>
</head>
<body>
    <form method="post" id="form">
        <input type="text" placeholder="un mot" name="word">
        <input type="text" placeholder="prénom" name="firstName">
        <input type="text" placeholder="nom" name="name">
        <input type="number" placeholder="age" name="age">
        <button type="submit">Valider</button>
    </form>
</body>
</html>
```

```
{
    word: "stylo",
    firstName: "Sébastien",
    name: "Maloron",
    age: "50"
}
```

Parcourir le DOM





propriété	description
parentNode	Le noeud parent
firstChild	Le premier noeud enfant
lastChild	Le dernier noeud enfant
nextSibling	Le noeud frère suivant
previousSibling	Le noeud frère précédent
childNodes[]	Un tableau de l'ensemble des noeuds enfants (NodeList)
children[]	un tableau des noeuds enfant de type Element (HTMLCollection)

propriété	description
childElementCount	Le nombre de noeuds de type Element (balise)
firstElementChild	Le premier noeud enfant de type Element
lastElementChild	Le dernier noeud enfant de type Element
nextElementSibling	Le noeud frère suivant de type Element
previousElementSibling	Le noeud frère précédent de type Element

Ajout et suppression d'éléments au DOM

- ✓ Créer un élément détaché du DOM
- ✓ Modifier les propriété de cet élément
- ✓ Attacher cet élément au DOM

```
document.createElement("balise")
```

```
//création de l'élément
const item = document.createElement("li");

//Modification des attributs
item.textContent = "cerises";
item.style.backgroundColor = "red";
```

```
parent.appendChild(element)
```

```
const list = document.querySelector("#list");

//création de l'élément
const item = document.createElement("li");

//Modification des attributs
item.textContent = "cerises";
item.style.backgroundColor = "red";

//Ajout comme dernier enfant
list.appendChild(item);
```

```
parent.insertBefore(element, cible)
```

```
const list = document.querySelector("#list");

// Le premier enfant de la liste
const firstItem = list.firstChild;

// création de l'élément
const newItem = document.createElement("li");

// Ajout avant le premier enfant
list.insertBefore(newItem, firstItem);
```

```
parent.replaceChild(element, cible)
```

```
const list = document.querySelector("#list");

const firstItem = list.firstElementChild;

// Création de l'élément
const newItem = document.createElement("li");

// Remplacement du premier enfant
list.replaceChild(newItem, firstItem);
```

element.remove()

```
const list = document.querySelector("#list");
const lastItem = list.lastElementChild;

// Suppression du dernier enfant
lastItem.remove();
```

```
element.cloneNode(deep: Boolean)
```

```
const template = document.getElementById("template");
const list = document.getElementById("list");

// clone du li
const clone = template.cloneNode();
// suppression de l'id sur le clone
// car un id doit être unique
clone.removeAttribute("id");
// modification du texte
clone.innerHTML = "Poires";

// Ajout du clone à la liste
list.appendChild(clone);
```

```
<ul id="list">
    <li id="template">Pommes</li>
</ul>
```

Cloner un élément et ses enfants

```
const template = document.getElementById("template");
const tbody = document.querySelector("#table tbody");

// clone de la ligne
// true car on veut un clone profond
// l'élément et ses enfants
const clone = template.cloneNode(true);
// suppression de l'id sur le clone
clone.removeAttribute("id");

// modification du texte des deux enfants (les td)
clone.children[0].innerHTML = "Paul";
clone.children[1].innerHTML = "Eluard";

// Ajout du clone au tableau
tbody.appendChild(clone);
```

```
<table id="table">
  <thead>
    <tr>
      <th>Prénom</th>
      <th>Nom</th>
    </tr>
  </thead>
  <tbody>
    <tr id="template">
      <td>Victor</td>
      <td>Hugo</td>
    </tr>
  </tbody>
</table>
```

```
const template = document.getElementById("template");
const tbody = document.querySelector("#table tbody");
const btAdd = document.getElementById("addRow");

// Ajout d'une ligne au click
btAdd.onclick = function () {
    // clone de la ligne
    const clone = template.cloneNode(true);
    // suppression de l'id sur le clone
    clone.removeAttribute("id");
    // modification du texte des deux enfants (les td)
    clone.children[0].innerHTML = "Paul";
    clone.children[1].innerHTML = "Eluard";

    // Ajout du clone au tableau
    tbody.appendChild(clone);
}
```

```
<style>
    #template {
        display: none;
    }
</style>

<button id="addRow">Ajouter une ligne</button>

<table id="table">
    <thead>
        <tr>
            <th>Prénom</th>
            <th>Nom</th>
        </tr>
    </thead>
    <tbody>
        <tr id="template">
            <td>Victor</td>
            <td>Hugo</td>
        </tr>
    </tbody>
</table>
```

```
const template = document.getElementById("template");
const tbody = document.querySelector("#table tbody");
const btAdd = document.getElementById("addRow");

// On clone au chargement de la page
const clonedTemplate = template.cloneNode(true);
clonedTemplate.removeAttribute("id");
// Suppression du template du DOM
template.remove();

// Ajout d'une ligne au click
btAdd.onclick = function () {
    // clone du clone
    const newRow = clonedTemplate.cloneNode(true);
    // modification du texte des deux enfants (les td)
    newRow.children[0].innerHTML = "Paul";
    newRow.children[1].innerHTML = "Eluard";

    // Ajout du clone au tableau
    tbody.appendChild(newRow);
}
```

Créer un formulaire qui propose de saisir un nom et d'ajouter des numéro de téléphone

Gérer la suppression

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
  <link rel="stylesheet"
    href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css">
</head>
<body class="container-fluid">
<div class="row justify-content-center">
  <div class="col-md-6 mt-4">
    <form id="form">
      <div class="mb-3">
        <label for="name" class="form-label">Votre nom</label>
        <input type="text" name="name" id="name" class="form-control">
      </div>
      <button type="button" id="addTel" class="btn btn-info mb-1 mt-3">
        Ajouter un téléphone
      </button>
      <div id="telList" class="p-3">
        <div class="mb-2 row" id="template">
          <div class="col-md-9">
            <input type="tel" name="tel[]"
              class="form-control" placeholder="votre téléphone">
          </div>
          <div class="col">
            <button type="button" class="btn btn-danger">
              Supprimer
            </button>
          </div>
        </div>
      </div>

      <div class="mt-4">
        <button class="btn btn-primary w-100">Valider</button>
      </div>
    </form>
  </div>
</div>
</body>
</html>
```

Votre nom

Ajouter un téléphone

votre téléphone

Supprimer

Valider

Applications

Au clique sur un titre afficher/masquer
les textes en dessous

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Liste de choses</title>
  <link rel="stylesheet"
        href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css">
</head>
<body class="container-fluid">
<div class="row justify-content-center mt-5">
  <div class="col-md-8 col-lg-6">
    <h2>Chapitre 1</h2>
    <div class="ms-4">
      <p>
        Lorem ipsum dolor sit amet, consectetur adipisicing elit. Debitis et molestias placeat possimus ratione
        suscipit totam, voluptas. Distinctio incident ipsa maxime quaerat quisquam? Doloribus, eius fuga optio
        repudiandae sit tempora.
      </p>
      <p>
        Lorem ipsum dolor sit amet, consectetur adipisicing elit. Debitis et molestias placeat possimus ratione
        suscipit totam, voluptas. Distinctio incident ipsa maxime quaerat quisquam? Doloribus, eius fuga optio
        repudiandae sit tempora.
      </p>
    </div>
    <h2>Chapitre 2</h2>
    <div class="ms-4">
      <p>
        Lorem ipsum dolor sit amet, consectetur adipisicing elit. Debitis et molestias placeat possimus ratione
        suscipit totam, voluptas. Distinctio incident ipsa maxime quaerat quisquam? Doloribus, eius fuga optio
        repudiandae sit tempora.
      </p>
    </div>
    <h2>Chapitre 3</h2>
    <div class="ms-4">
      <p>
        Lorem ipsum dolor sit amet, consectetur adipisicing elit. Debitis et molestias placeat possimus ratione
        suscipit totam, voluptas. Distinctio incident ipsa maxime quaerat quisquam? Doloribus, eius fuga optio
        repudiandae sit tempora.
      </p>
    </div>
  </div>
</div>
</body>
</html>
```

Chapitre 1

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Debitis et molestias placeat possimus ratione suscipit totam, voluptas. Distinctio incident ipsa maxime quaerat quisquam? Doloribus, eius fuga optio repudiandae sit tempora.

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Debitis et molestias placeat possimus ratione suscipit totam, voluptas. Distinctio incident ipsa maxime quaerat quisquam? Doloribus, eius fuga optio repudiandae sit tempora.

Chapitre 2

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Debitis et molestias placeat possimus ratione suscipit totam, voluptas. Distinctio incident ipsa maxime quaerat quisquam? Doloribus, eius fuga optio repudiandae sit tempora.

Chapitre 3

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Debitis et molestias placeat possimus ratione suscipit totam, voluptas. Distinctio incident ipsa maxime quaerat quisquam? Doloribus, eius fuga optio repudiandae sit tempora.

Tester si un mot de passe et sa confirmation
sont identiques, afficher un message d'erreur
si ce n'est le cas

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
  <link rel="stylesheet"
    href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css">
</head>
<body class="container-fluid">
<div class="row justify-content-center">
  <div class="col-md-6 mt-4">
    <div class="alert alert-danger" id="error" style="display:none"></div>
    <form id="form">
      <div class="mb-3">
        <label for="password" class="form-label">Mot de passe</label>
        <input type="password" name="password" id="password" class="form-control">
      </div>
      <div class="mb-3">
        <label for="passwordConfirm" class="form-label">Confirmation Mot de passe</label>
        <input type="password" name="passwordConfirm" id="passwordConfirm" class="form-control">
      </div>
      <div class="mt-4">
        <button class="btn btn-primary w-100">Valider</button>
      </div>
    </form>
  </div>
</div>
</body>
</html>
```

Mot de passe

Confirmation Mot de passe

Valider

Masquer ou afficher les éléments
d'une liste au clic sur un titre

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Arborescence</title>
  <style>
    li h3{
      cursor: pointer;
    }
  </style>
</head>
<body>

<ul>
  <li>
    <h3>Légumes</h3>
    <ul>
      <li>
        <h3>Légumes racine</h3>
        <ul>
          <li>Carottes</li>
          <li>Panais</li>
          <li>Radis</li>
        </ul>
      </li>
      <li>
        <h3>Légumes feuilles</h3>
        <ul>
          <li>Salade</li>
          <li>Blettes</li>
          <li>Epinards</li>
        </ul>
      </li>
    </ul>
  </li>
  <li>
    <h3>Fruits</h3>
    <ul>
      <li>
        <h3>Fruits à noyau</h3>
        <ul>
          <li>Prunes</li>
          <li>Cerises</li>
          <li>Abricots</li>
        </ul>
      </li>
      <li>
        <h3>Agrumes</h3>
        <ul>
          <li>Citrons</li>
          <li>Oranges</li>
          <li>Pamplemousses</li>
        </ul>
      </li>
    </ul>
  </li>
</ul>
</body>
</html>
```

- Légumes

- Légumes racine

- Carottes
 - Panais
 - Radis

- Légumes feuilles

- Salade
 - Blettes
 - Epinards

- Fruits

- Fruits à noyau

- Prunes
 - Cerises
 - Abricots

- Agrumes

- Citrons
 - Oranges
 - Pamplemousses

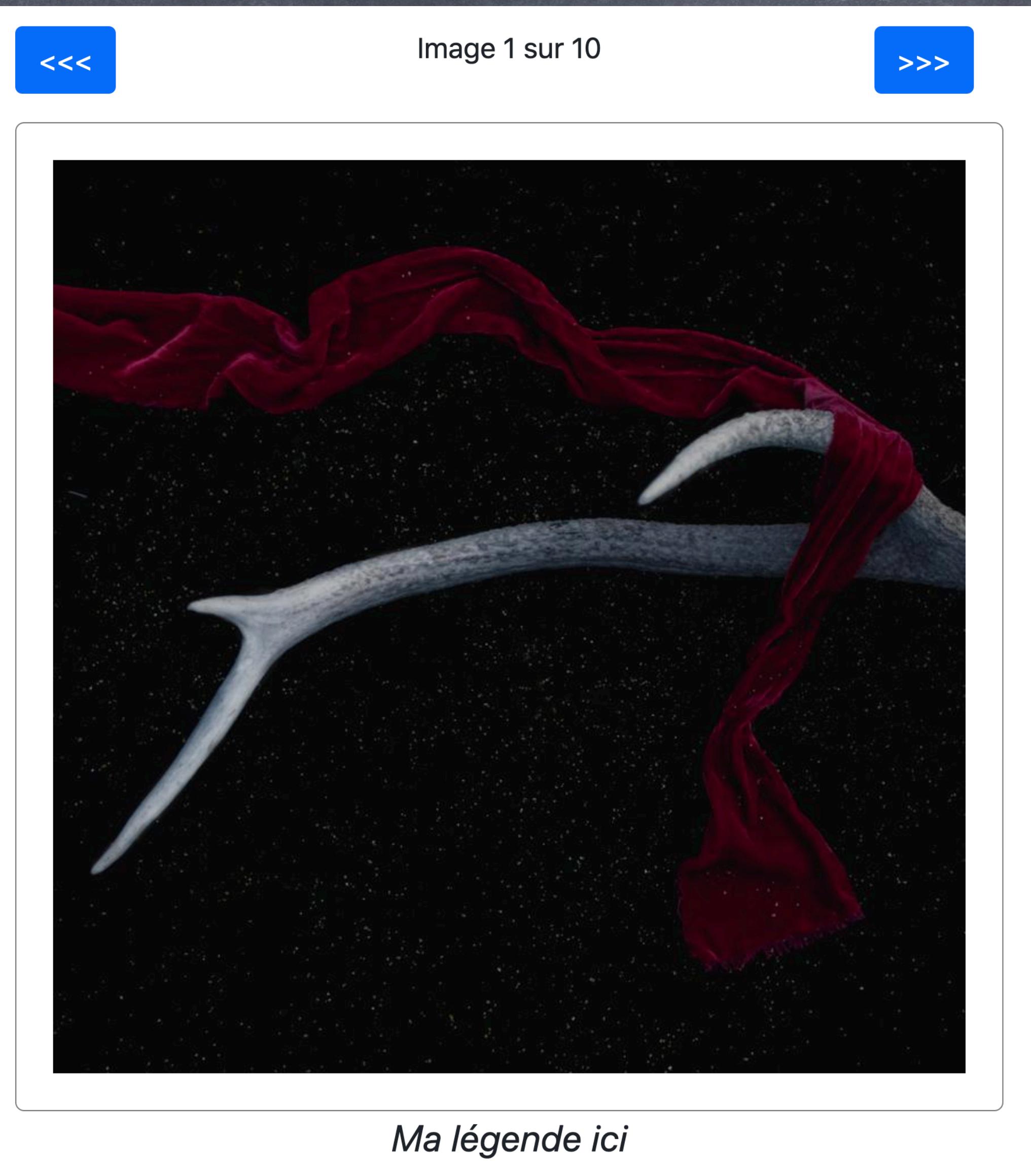
```
[  
  {  
    title: "Légumes", children: [  
      {  
        title: "Légumes racines", children: [  
          "Carottes", "Panais", "Radis"]  
        },  
        {  
          title: "Légumes feuilles", children: [  
            "Salades", "Blettes", "Epinards"]  
          }  
        ]  
      },  
      {  
        title: "Fruits", children: [  
          {  
            title: "Fruits à noyau", children: [  
              "Prunes", "Cerises", "Abricots"]  
            },  
            {  
              title: "Légumes feuilles", children: [  
                "Citrons", "Oranges", "Pamplemousses"]  
              }  
            ]  
          }  
        ]  
      }]
```

Afficher l'arbre en utilisant la structure de données suivante et en conservant l'interactivité

A partir d'un tableau contenant le nom des fichiers et la légende, créer une application qui affiche une image et permet de passer à l'image suivante ou précédente en cliquant sur des boutons

```
const photos = [
    {fileName: "photo-01.jpg", legend: "Une belle photo de vacance"},  
    {fileName: "photo-02.jpg", legend: "Mon chat qui se repose"},  
    {fileName: "photo-03.jpg", legend: "Ma fille avec son copain"},  
];
```

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>Galerie photo</title>
    <link rel="stylesheet"
        href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css">
    <style>
        .photo {
            width: 100%;
            padding: 20px;
            border: 1px solid gray;
            border-radius: 5px;
        }
        figcaption {
            text-align: center;
            font-style: italic;
            font-size: 1.3em;
        }
    </style>
</head>
<body class="container-fluid">
<div class="row justify-content-center mt-5">
    <div class="col-md-8 col-lg-6">
        <div class="row justify-content-between mb-3">
            <div class="col-2 ">
                <button class="btn btn-primary"> précédent </button>
            </div>
            <div class="col-6 text-center">Image 1 sur 10</div>
            <div class="col-2 text-right">
                <button class="btn btn-primary"> suivant </button>
            </div>
        </div>
        <figure>
            
            <figcaption>Ma légende ici</figcaption>
        </figure>
    </div>
</div>
</body>
</html>
```



Créer un application qui affiche une liste de choses dans une table html et permet l'ajout et/ou la suppression de nouveaux éléments

Ajouter un bouton qui trie les éléments

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Liste de choses</title>
  <link rel="stylesheet"
    href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css">
</head>
<body class="container-fluid">
<div class="row justify-content-center mt-5">
  <div class="col-md-8 col-lg-6 text-center">
    <h1 class="mb-4">Liste de choses</h1>
    <form method="post">
      <div class="mb-3 text-start">
        <label for="item" class="form-label">Le truc à ajouter</label>
        <input type="text" name="item" id="item" class="form-control">

      </div>
      <button type="submit" class="btn btn-primary w-100">Valider</button>
      <hr class="mb-4">
    </form>

    <table id="list" class="table text-start">
      <thead>
        <tr>
          <th>Element</th>
          <th></th>
        </tr>
      </thead>
      <tbody>
        <!-- elements à remplacer par la saisie -->
        <tr>
          <td class="align-middle">Un frigidaire</td>
          <td class="text-end"><button class="btn btn-danger delete">Supprimer</button></td>
        </tr>
        <tr>
          <td class="align-middle">Un évier en fer</td>
          <td class="text-end"><button class="btn btn-danger delete">Supprimer</button></td>
        </tr>
        <!-- fin des elements -->
      </tbody>
    </table>
  </div>
</div>
</body>
</html>
```

Liste de choses

Le truc à ajouter

Valider

Element

Un frigidaire	Supprimer
Un évier en fer	Supprimer