La gestion des événements en JavaScript

Dark Theme

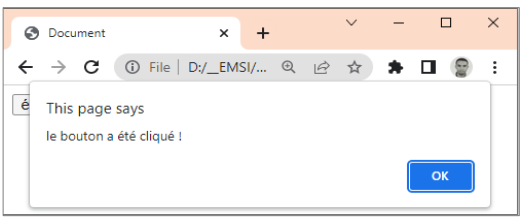
Les événement en JavaScript

ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

- Les événements sont des actions qui se produisent dans le système que vous programmez. Le système envoie un signal lorsque des événements se produisent, ce qui nous permet de faire quelque chose à ce moment-là.

- En générale les événements permettent de déclencher une fonction pour une action spécifique, comme par exemple le clic ou le survol d'un élément, le chargement du document HTML ou encore l'envoi d'un formulaire.

```
<button id="btn">événement en JS</button>
<script>
    var btn = document.getElementById("btn");
    btn.onclick = function() {
        alert("le bouton a été cliqué !");
    };
</script>
```



Les événement en JavaScript

ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

Principaux événements du DOM

Événement DOM	Description
click	Bouton de la souris enfoncé puis relâché sur un élément.
dblclick	Deux foix l'événement click
mouseover	Souris au-dessus d'un élément.
mouseout	Souris sort d'un élément.
mousedown	Bouton de la souris enfoncé, pas relâché, sur un élément.
mouseup	Bouton de la souris relâché sur un élément.
mousemove	Souris en mouvement au-dessus d'un élément.
keydown	Touche clavier enfoncée, pas relâchée, sur un élément.
keyup	Touche clavier relâchée sur un élément.
keypress	Touche clavier enfoncée et relâchée sur un élément.
focus	L'élément reçoit, gagne, le focus. Quand un objet devient l'élément actif du document.
blur	Elément perd le focus.
change	Changement de a valeur d'un élément de formulaire.
select	Sélection du texte d'un élémen, mis en srubrillance.
submit	Envoi d'un formulaire
reset	Réinitialisation d'un formulaire

Types d'évènements

- Evénements de la souris
- Evénements du clavier
- Evénements de formulaire
- Evénements de frame ...

Liste complète des événements :
https://www.w3schools.com/jsref/dom_obj_event.asp

Les événement en JavaScript

ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

Affecter une fonction à un événement

Il existe différentes manières d'affecter une fonction à l'événement d'un objet.

- Soit à l'aide de gestionnaires d'événements « on-event »

```
<élément onevent='code JavaScript'>
```

- Ou bien, de créer des écouteur d'événement (listener) avec La méthode addEventListener()

```
élément.addEventListener('event', function (){…});
```

Le meilleure moyen est souvent addEventListener()
Avec la méthode "on-event", chaque objet ne peut avoir qu'un seul gestionnaire d'événement pour un événement donné.

En ligne « on-event»

```
<button onclick="bgChange()">Changer la couleur du fond</button>
<script>
    function bgChange() {
        var rndCol = 'rgb(' + Math.floor(Math.random() * 256) + ','
            +Math.floor(Math.random() * 256) + ','
            +Math.floor(Math.random() * 256) + ')';
        document.body.style.backgroundColor = rndCol;
    }
</script>
```

Avec La méthode addEventListener()

```
<script>
    var btn = document.querySelector('button');
    function bgChange(){
        var rndCol = 'rgb('+Math.floor(Math.random()*256)+','
                          +Math.floor(Math.random()*256)+','
                          +Math.floor(Math.random()*256)+')';
        document.body.style.backgroundColor = rndCol;
    }
    btn.addEventListener('click', bgChange)
</script>
```

On-event

Les gestionnaires d'événements "on-event" sont nommées selon l'événement lié : *onclick*, *onkeypress*, *onfocus*, *onsubmit*, etc.

On peut spécifier un "on-event" pour un événement particulier de différentes manières :

- Avec un attribut HTML : *<button onclick="maFnct()">*
- En utilisant la propriété correspondante en JavaScript : *Element.onclick = maFnct;*

En JavaScript, afin d'affecter la fonction bonjour() et non son résultat, on n'ajoute pas les parenthèses après le nom de la fonction.

```
Element.onclick = bonjour;  // affecte la fonction bonjour().
Element.onclick = fonction(); //  affecte le résultat de la fonction bonjour().
```


addEventListener

La méthode *addEventListener()* permet de définir une fonction à appeler chaque fois que l'événement spécifié est détecté sur l'élément ciblé.

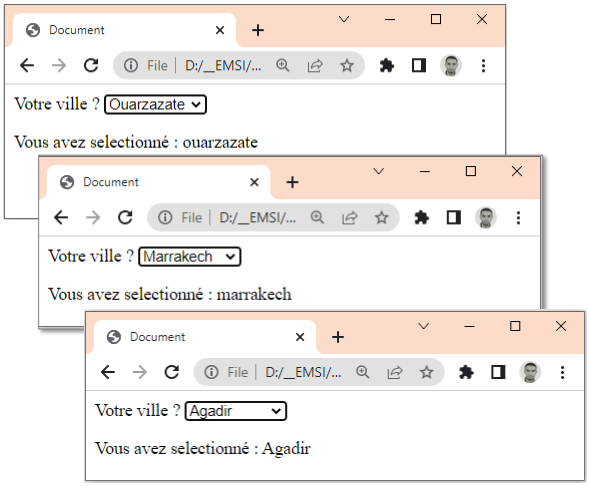
Syntaxe : `ElementCible.addEventListener("nomEvenement", nomFonction);`

```
<button>Clique moi !</button>
<script>
    // Récupère le 1er boutons du document
    const bouton = document.querySelector("button");
    // Ajoute événement click avec une fonction avec paramètre event
    bouton.addEventListener("click", function(event) {
        // Récupère l'élément qui a envoyé l'événement, la cible
        let cible = event.target;
        // Modifie la taille du texte de la cible
        cible.style.fontSize = "2em";
        // Affiche le contenu texte de la cible
        alert(cible.innerText); // Clique moi !
    });
</script>
```

Les événement en JavaScript {change}

- Exemple :

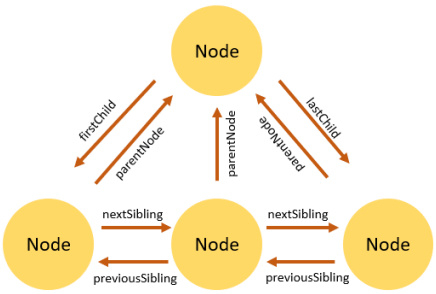
```
<label>Votre ville ?</label>
<select name="villes" id="villes">
    <option value="ouarzazate">Ouarzazate</option>
    <option value="marrakech">Marrakech</option>
    <option value="casablanca">Casablanca</option>
    <option value="Agadir">Agadir</option>
    <option value="Tanger">Tanger</option>
</select>
<p>Vous avez selectionné : <span id="demo"></span></p>
<script>
    var select = document.getElementById("villes");
    select.addEventListener("change", function() {
        var val = document.getElementById("villes").value;
        document.getElementById("demo").innerHTML = val;
    });
</script>
```



ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

JavaScript et HTML DOM

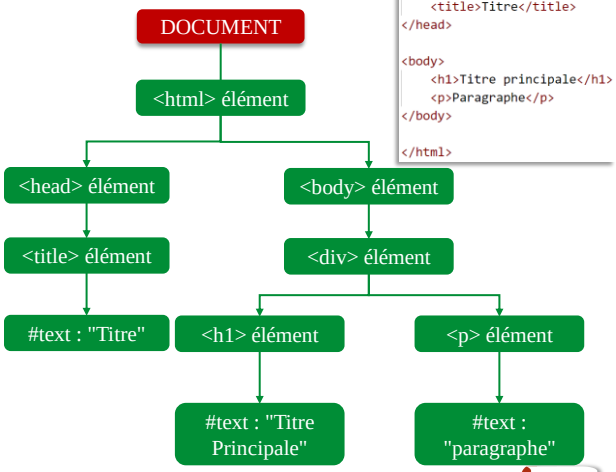
- C'est quoi Le DOM
- JS DOM : Les sélecteurs et les Nœuds
- JS DOM : Navigation
- JS DOM : CSS ou Style
- JS DOM : Les effets



Introduction au DOM

- comme vous le savez déjà, le squelette d'un document HTML est constitué d'éléments
- lorsqu'un document HTML est chargé ; le navigateur crée un modèle d'objet de document ou HTML DOM.
- Le DOM est créé par le navigateur lorsqu'un document HTML est chargé.
- il s'agit d'une représentation arborescente d'une structure HTML
- L'arbre DOM contient tous les éléments d'un document

```
<!DOCTYPE html>
<html>

<head>
    <title>Titre</title>
</head>

<body>
    <h1>Titre principale</h1>
    <p>Paragraphe</p>
</body>

</html>
```



JS DOM : Les sélecteurs

Nous pouvons sélectionner n'importe quel élément de l'arbre DOM en utilisant l'objet document.

- Pourquoi sélectionne-t-on des éléments ?
  - ✓ pour obtenir leur contenu
  - ✓ pour changer leur contenu
  - ✓ pour leur donner un style
  - ✓ pour obtenir ou modifier leurs attributs
  - ✓ pour les supprimer
  - ✓ ....

Les éléments les plus hauts peuvent être sélectionnés directement à partir des propriétés de l'objet du document.

- `document.documentElement` sélectionne `<html>`
- `document.head` sélectionne `<head>`
- `document.body` sélectionne `<body>`

```
// obtenir la valeur de l'attribut lang de <html>
alert(document.documentElement.getAttribute('lang'));

// obtenir le contenu du balise <title> dans le <head>
alert(document.head.firstElementChild.innerHTML);

// changer la couleur du fond de l'élément <body>
document.body.style.backgroundColor = 'green'
```

JS DOM : Les sélecteurs

Sélectionner un élément par ID

`document.getElementById()` sélectionne l'élément dont l'attribut id correspond à une chaîne spécifiée, c'est la meilleure méthode à utiliser pour sélectionner un seul élément.

```
<p id="demo">
    Lorem ipsum, dolor sit amet consectetur adipisicing elit.
    obcaecati? Facere voluptate officiis ut ad possimus.
</p>
```

```
document.getElementById('demo').innerHTML =
    "Bonjour Je suis Ameksa Mohammed";
```

Sélectionner un élément par nom de classe

`document.getElementsByClassName()` sélectionne tous les éléments avec le nom de classe donné. Cet sélecteur retourne un tableau d'objets.

```
<p id="demoClass">
    Lorem ipsum, dolor sit amet consectetur adipisicing elit.
    obcaecati? Facere voluptate officiis ut ad possimus.
</p>
<div id="demoClass">
    <ul>
        <li>JS</li>
        <li>Unix</li>
    </ul>
</div>
```

```
// sélectionner les éléments par nom de classe
var resultat = document.getElementsByClassName('demoClass');
// Maintenant, nous pouvons accéder à chaque élément sélectionné
for (let i = 0; i < resultat.length; i++) {
    resultat[i].style.color = 'green';
}
```


JS DOM : Les sélecteurs

Sélectionner les éléments par le nom de la balise

`document.getElementsByTagName()` sélectionne tous les éléments avec le nom de balise indiqué ; de même, il renvoie un tableau d'objets .

```
<h2>Sous titre 1</h2>
<p>
    Lorem ipsum, dolor sit amet consectetur adipisicing elit.
    obcaecati? Facere voluptate officiis ut ad possimus.
</p>
<h2>Sous titre 2</h2>
<p>
    Lorem ipsum, dolor sit amet consectetur adipisicing elit.
    obcaecati? Facere voluptate officiis ut ad possimus.
</p>
```

```
// sélectionner tout les éléments <h3>
var sousTitres = document.getElementsByTagName('h3');
// Maintenant, nous pouvons accéder à chaque élément
for (let i = 0; i < sousTitres.length; i++) {
    sousTitres[i].style.color = 'green';
}
```

Sélectionner un élément par l'attribut name

`document.getElementsByName()` sélectionne tous les éléments avec un nom « attribut name » donné. Cet sélecteur retourne un tableau d'objets.

```
<label for="pseudo">Username : </label>
<input type="text" name="pseudo" id="pseudo">
<button onclick="getVal()">Afficher la valeur</button>
```

```
function getVal() {
    var username = document.getElementsByName('pseudo')[0].value;
    alert('voici la valeur saisi' + username);
}
```

JS DOM : Les sélecteurs

ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

sélectionner des éléments à l'aide des sélecteurs css

`document.querySelectorAll()` sélectionne tous les éléments correspondant à un sélecteur bien spécifié . Ce dernier doit être un sélecteur css valide.

```
<p id="demo">
    Lorem ipsum, dolor sit amet consectetur adipisicing elit.
    obcaecati? Facere voluptate officiis ut ad possimus.
</p>
```

```
document.querySelectorAll('#demo')[0].innerHTML =
    "les sélecteurs JS";
```

```
<p id="demoClass">
    Lorem ipsum, dolor sit amet consectetur adipisicing elit.
    obcaecati? Facere voluptate officiis ut ad possimus.
</p>
<div id="demoClass">
    <ul>
        <li>JS</li>
        <li>Unix</li>
    </ul>
</div>
```

```
var el = document.querySelectorAll('.demoClass');
for (let i = 0; i < el.length; i++) {
    el[i].style.color = "green";
}
```


JS DOM : Les sélecteurs par des nœuds

ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

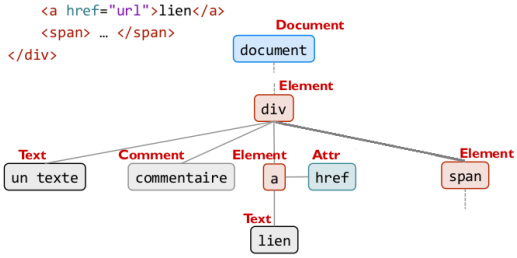
- dans l'arbre du DOM; tout est un nœud : les éléments, le texte, les attributs et même les commentaires sont considérés comme des nœuds.

- Pour se naviguer dans une arbre DOM à l'aide des nœuds il existe 3 notions à comprendre :
  1. Parent : est le parent du nœud "un texte"
  2. Child : "un texte" est l'enfant de
  3. Sibling : <a> et sont des siblings

- pour accéder aux nœuds dans le DOM, les propriétés suivantes de l'objet document sont utilisées :

  parentNode ; firstChild; childNodes[index]; previousSibling; nextSibling

```
<div>un texte
    <!-- commentaire -->
    <a href="url">lien</a>
    <span> … </span>
</div>
```



- Les propriétés pour accéder aux nœuds de type éléments :

  parentElement ; firstChildElement ; lastChildElement ; children[index] ; previousElementSibling ; nextElementSibling

JS DOM : Les sélecteurs par des nœuds

ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

Parent et enfants

La propriété `parentElement` renvoie l'élément parent du nœud DOM

```
<div id="container">
    <h1>Titre</h1>
    <p id="demo">
        Lorem ipsum dolor sit amet consectetur adipisicing elit.
        Fugiat, fugit omnis quam excepturi optio corporis deleelk.
        nesciunt facilis accusamus facere ipsam ipsum praesentium.
    </p>
    <a href="www.ameksa.com">Aller à AMEKSA.COM </a>
</div>
```

```
// pour sélectionner l'élément <p>
var elementP = document.getElementById('demo');
/*
Maintenant nous allons sélectionner
l'élément parent du paragraphe <p>
et on change la couleur de son fond
*/
elementP.parentElement.style.backgroundColor = "gray";
```

JS DOM : Les sélecteurs par des nœuds

ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

Parent et enfants

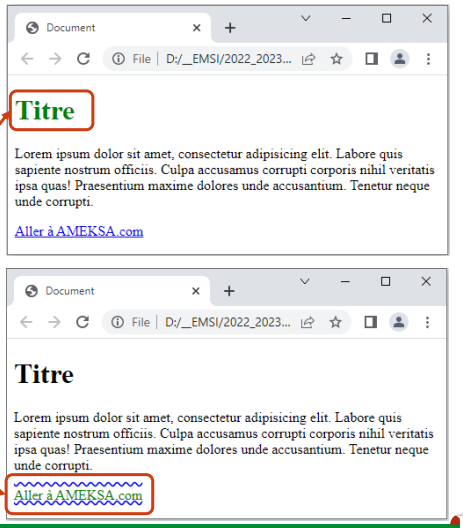
La propriété `firstElementChild` renvoie le premier élément enfant

La propriété `lastElementChild` renvoie le dernier élément enfant

```
<div id="container">
    <h1>Titre</h1>
    <p id="demo">
        Lorem ipsum dolor sit amet, consectetur adipisicing
        sapiente nostrum officiis. Culpa accusamus corrupti
        veritatis ipsa quas! Praesentium maxime dolores unde
        Tenetur neque unde corrupti.
    </p>
    <a href="www.ameksa.com">Aller à AMEKSA.com</a>
</div>
```

```
var container = document.getElementById('container');
container.firstElementChild.style.color = 'green';
```

```
var container = document.getElementById('container');
container.lastElementChild.style.color = 'green';
container.lastElementChild.style.textDecoration =
    "blue underline overline wavy";
```

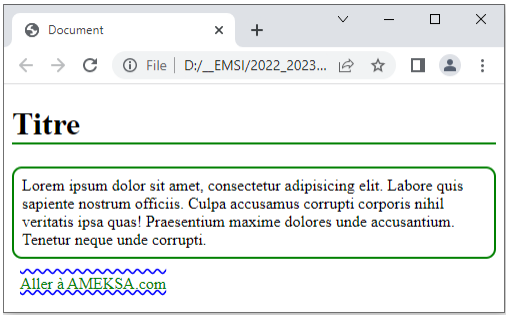


Parent et enfants

La propriété `children` renvoie une collection des éléments enfants de l'élément sélectionné. Nous pouvons accéder à chaque élément en utilisant la notation entre crochets [ ].

```
<div id="container">
    <h1>Titre</h1>
    <p id="demo">
        Lorem ipsum dolor sit amet, consectetur adipisicing
        sapiente nostrum officiis. Culpa accusamus corrupti
        veritatis ipsa quas! Praesentium maxime dolores unde
        Tenetur neque unde corrupti.
    </p>
    <a href="www.ameksa.com">Aller à AMEKSA.com</a>
</div>
```

```
var container = document.getElementById('container');
container.children[0].style.borderBottom = '2px solid green';
container.children[1].style.borderRadius = '10px';
container.children[1].style.padding = '8px';
container.children[1].style.border = '2px solid green';
container.children[2].style.color = 'green';
container.children[2].style.padding = '8px';
container.children[2].style.textDecoration =
    'blue underline overline wavy';
```

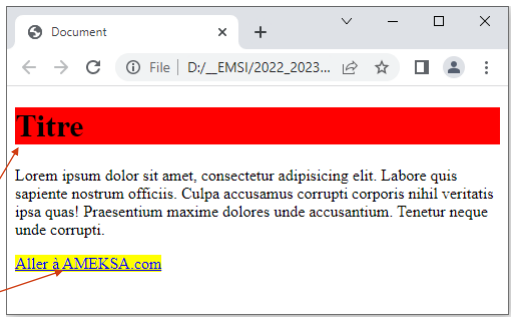


Siblings

- `previousElementSibling` renvoie l'élément précédent par rapport à un élément spécifié dans une liste des enfants.
- `nextElementSibling` renvoie l'élément suivant par rapport à un élément spécifié dans une liste des enfants.

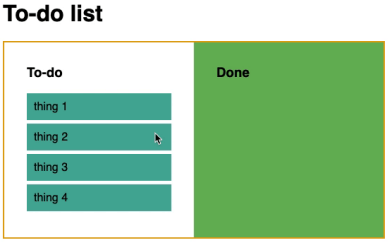
```
<div id="container">
    <h1>Titre</h1>
    <p id="demo">
        Lorem ipsum dolor sit amet, consectetur adipisicing
        sapiente nostrum officiis. Culpa accusamus corrupti
        veritatis ipsa quas! Praesentium maxime dolores unde
        Tenetur neque unde corrupti.
    </p>
    <a href="www.ameksa.com">Aller à AMEKSA.com</a>
</div>
```

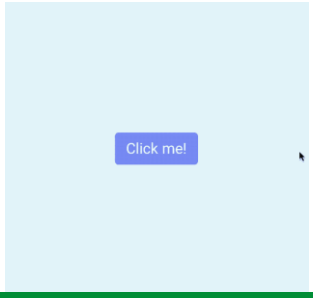
```
var pElement = document.getElementById('demo');
pElement.previousElementSibling.style.backgroundColor = 'red';
pElement.nextElementSibling.style.backgroundColor = 'yellow';
```



Ce que l'on peut faire aux éléments HTML en utilisant le DOM :

- Obtenir le contenu des éléments
- Remplacer le contenu des éléments
- Créer des éléments (nœuds)
- Insertion dans les éléments
- Suppression des éléments
- Obtenir la valeur d'un attribut
- Définir ou modifier la valeur d'un attribut

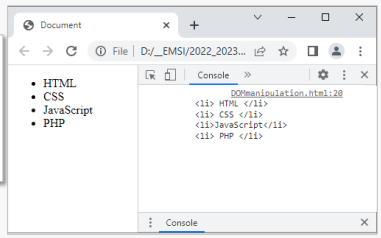




Obtenir / modifier le contenu d'un élément HTML

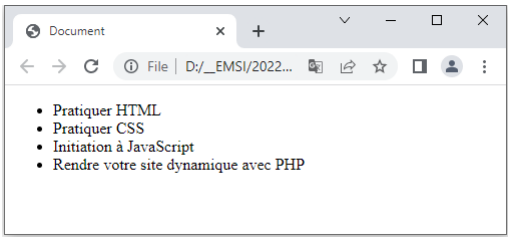
la propriété `innerHTML` renvoie le contenu d'un élément HTML ; elle peut également être utilisée pour remplacer son contenu.

```
<ul id="list">
    <li> HTML </li>
    <li> CSS </li>
    <li>JavaScript</li>
    <li> PHP </li>
</ul>
```




```
var list = document.getElementById('list');
console.log(list.innerHTML);
```

```
var list = document.getElementById('list');
list.innerHTML = '<li> Pratiquer HTML </li>' +
    '<li> Pratiquer CSS </li>' +
    '<li> Initiation à JavaScript </li>' +
    '<li> Rendre votre site dynamique avec PHP </li>'
```



Création des éléments (nœuds)

Avant de pouvoir insérer des éléments (nœuds) dans le DOM, nous devons d'abord apprendre à les créer en JavaScript, pour créer un élément, on utilise la méthode `document.createElement(élément)`.

Une fois l'élément créé, nous voulons ajouter un contenu à l'intérieur.
Pour ce faire, nous devons créer un nœud et l'insérer dans l'élément créé précédemment.

Si on veut créer un élément paragraphe et ajouter du texte dans cet élément.
On utilise `document.createElement(p)`
Puis, `document.creatTextNode("texte à ajouter")` pour créer un nœud de texte.
Et finalement, la méthode `appendChild()` pour insérer le nœud dans un élément.

```
<div id="demo"></div>
<script>
    var para = document.createElement("p");
    var texte = document.createTextNode("Contenu du para");
    para.appendChild(texte);
    document.getElementById("demo").appendChild(para);
</script>
```

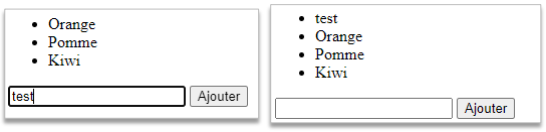

Insertion dans des éléments

En général, nous voulons seulement insérer un élément et non pas remplacer complètement son contenu.

- La méthode `ParentNode.prepend()` insère un élément (nœud) avant le premier enfant du ParentNode.
- La méthode `ParentNode.append()` insère un élément (nœud) après le dernier enfant du ParentNode.

```
<ul id="list">
    <li>Orange</li>
    <li>Pomme</li>
    <li>Kiwi</li>
</ul>
<input type="text" id="fruit">
<input type="button" value="Ajouter" id="btn">
```

```
var list = document.getElementById('list');
var fruit = document.getElementById('fruit');
var btn = document.getElementById('btn');

btn.addEventListener('click', function() {
    let fruitContent = fruit.value;
    let li = document.createElement("li");
    let texte = document.createTextNode(fruitContent);
    li.appendChild(texte);
    list.prepend(li)
    fruit.value = ''
});
```



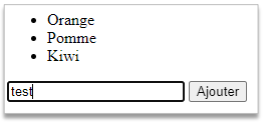
Insertion dans des éléments

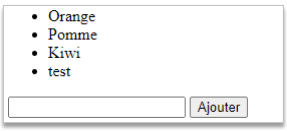
En général, nous voulons seulement insérer un élément et non pas remplacer complètement son contenu.

- La méthode `ParentNode.prepend()` insère un élément (nœud) avant le premier enfant du ParentNode.
- La méthode `ParentNode.append()` insère un élément (nœud) après le dernier enfant du ParentNode.

```
<ul id="list">
    <li>Orange</li>
    <li>Pomme</li>
    <li>Kiwi</li>
</ul>
<input type="text" id="fruit">
<input type="button" value="Ajouter" id="btn">
```

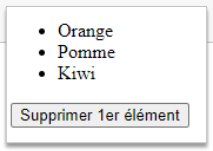
```
var list = document.getElementById('list');
var fruit = document.getElementById('fruit');
var btn = document.getElementById('btn');

btn.addEventListener('click', function() {
    let fruitContent = fruit.value;
    let li = document.createElement("li");
    let texte = document.createTextNode(fruitContent);
    li.appendChild(texte);
    list.append(li)
    fruit.value = ''
});
```





Enlever des éléments

Pour supprimer un élément, on utilise la méthode `node.remove()`. Cette méthode supprime le nœud de l'arbre auquel il appartient.



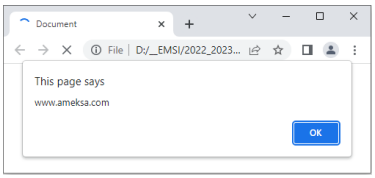

```
<ul id="list">
    <li>Orange</li>
    <li>Pomme</li>
    <li>Kiwi</li>
</ul>
<input type="button" value="Supprimer 1er élément" id="btn">
```

```
var list = document.getElementById('list');
btn.addEventListener('click', function() {
    let premElement = list.firstElementChild;
    premElement.remove();
});
```

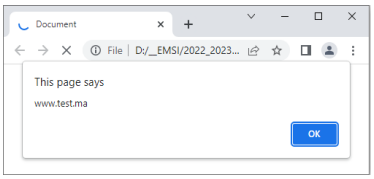
JS DOM et HTML

Récupérer et modifier la valeur d'un attribut

- Pour obtenir la valeur d'un attribut, on utilise la méthode `node.getAttribute(nomAttribut)`. Cette méthode renvoie la valeur d'un attribut spécifié.



- Pour changer la valeur d'un attribut, on utilise la méthode `node.setAttribute(nomAttribut, valeur)`. Si l'attribut spécifié existe, cette méthode va changer son valeur.




```
<a href="www.ameksa.com" id="demo">Allez à AMEKSA.com</a>
```

```
var e = document.getElementById("demo");
var valeurHref = e.getAttribute("href");
alert(valeurHref);
```

```
var e = document.getElementById("demo");
e.setAttribute("href", "www.test.ma");
var valeurHref = e.getAttribute("href");
alert(valeurHref);
```


JS DOM et CSS

styliser les éléments avec JS

- pour modifier les styles d'un élément, utilisez l'objet style d'un nœud.

Syntaxe :

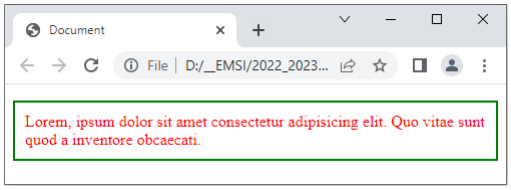
`node.style.propriété = "valeur"`

Exemple :

```
<p id="demo">
    Lorem, ipsum dolor sit amet consectetur adipisicing
    elit. Quo vitae sunt quod a inventore obcaecati.
</p>
```

```
var para = document.getElementById("demo");
para.style.color = 'red';
para.style.border = '2px solid green';
para.style.padding = '10px';
```

- Les propriétés avec un trait d'union comme : text-decoration ; background-color ; doivent être écrites en camel-case.
  - text-decoration => `textDecoration`
  - background-color => `backgroundColor`



effets de transition

```html
<h1>Effet de transition</h1>
<div id="demo"></div>
<button id="btnDisp">Disparaître</button>
<button id="btnOuv">Ouverture</button>
```

```css
<style>
    div#demo {
        width: 300px;
        height: 300px;
        background: #173459;
        opacity: 1;
        display: block;
        transition: all 1s;
    }
</style>
```

```js
var btnDisp = document.getElementById('btnDisp'),
    btnOuv = document.getElementById('btnOuv');

function fadeOut() {
    var d = document.getElementById('demo');
    d.style.opacity = 0;
    setTimeout(function() {
        d.style.display = 'none';
    }, 1000);
}

function fadeIn() {
    var d = document.getElementById('demo');
    d.style.display = 'block';
    setTimeout(function() {
        d.style.opacity = 1;
    }, 10);
}
btnDisp.addEventListener('click', fadeOut);
btnOuv.addEventListener('click', fadeIn);
```


effets de glissement

```html
<ul id="demo">
    <li>HTML</li>
    <li>CSS</li>
    <li>JavaScript</li>
    <li>PHP</li>
</ul>
<div>
    <button id="up">glisser vers le haut</button>
    <button id="down">glisser vers le bas</button>
</div>
```

```css
ul#demo {
    max-height: 100px;
    overflow-y: hidden;
    transition: all 1;
}
```

```js
var list = document.getElementById('demo'),
    btnUp = document.getElementById('up'),
    btnDown = document.getElementById('down');
btnUp.addEventListener('click', function() {
    list.style.maxHeight = '0px';
});
btnDown.addEventListener('click', function() {
    list.style.maxHeight = '100px';
})
```