ISEN ALL IS DIGITAL! OUEST | yncréa

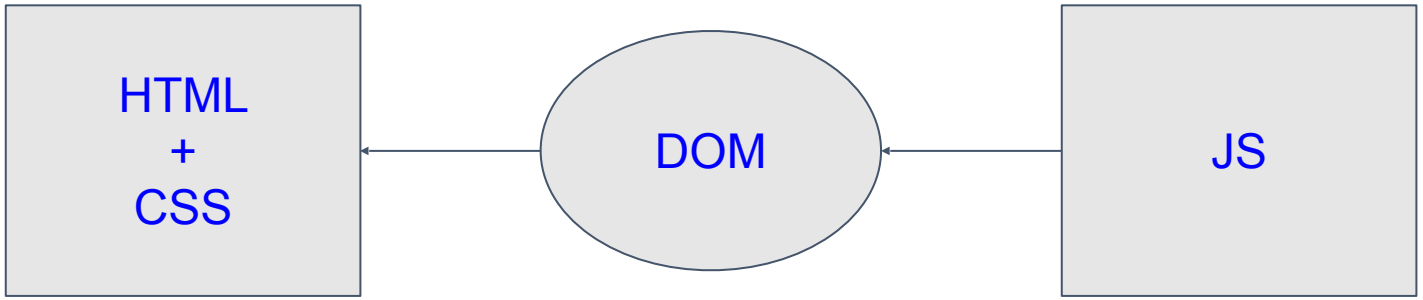
JavaScript
- Arbre DOM -

Groupe des étudiants : CIR1

Définition et exemples

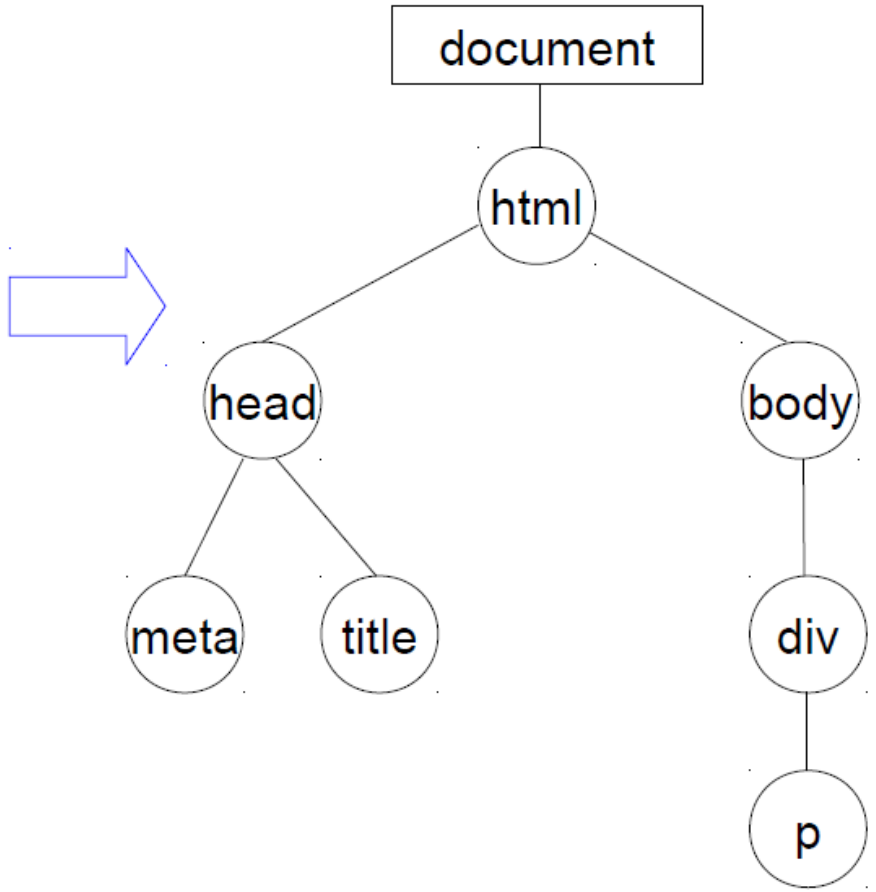
Définition

- En mémoire, une page web est représentée sous la forme d'un arbre, appelé arbre DOM (Document Object Model)
- Une fois l'arbre construit en mémoire, on peut le **consulter** et le **modifier** dynamiquement avec JavaScript
- La racine de l'arbre DOM est toujours stockée dans la variable globale document



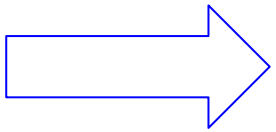
Exemple 1

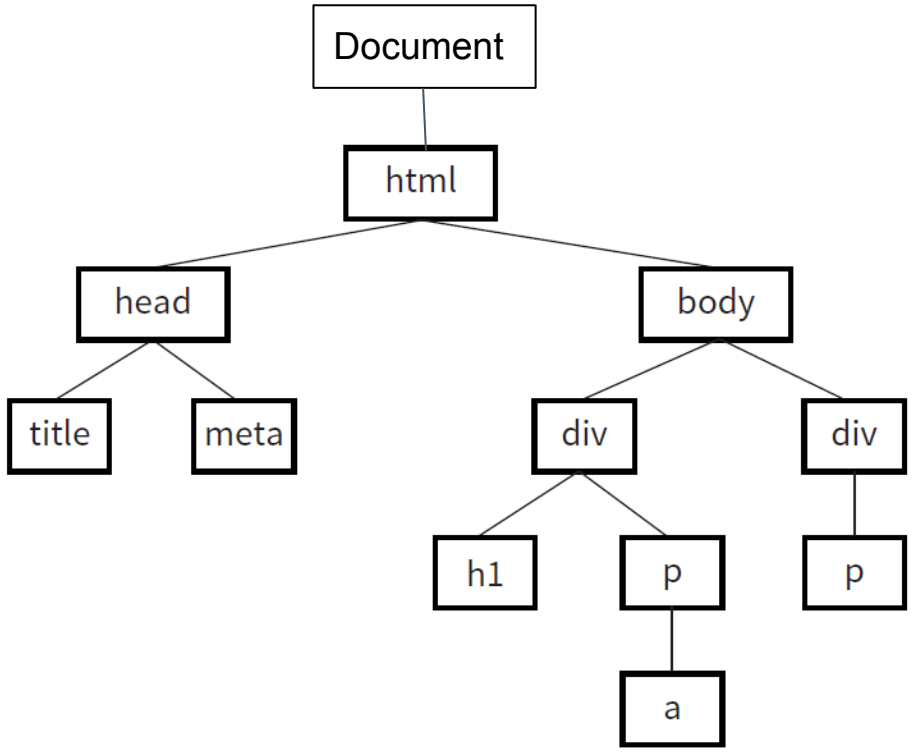
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>A web page</title>
  </head>
  <body>
    <div>
      <p>Paragraphe</p>
    </div>
  </body>
</html>
```



Exemple 2

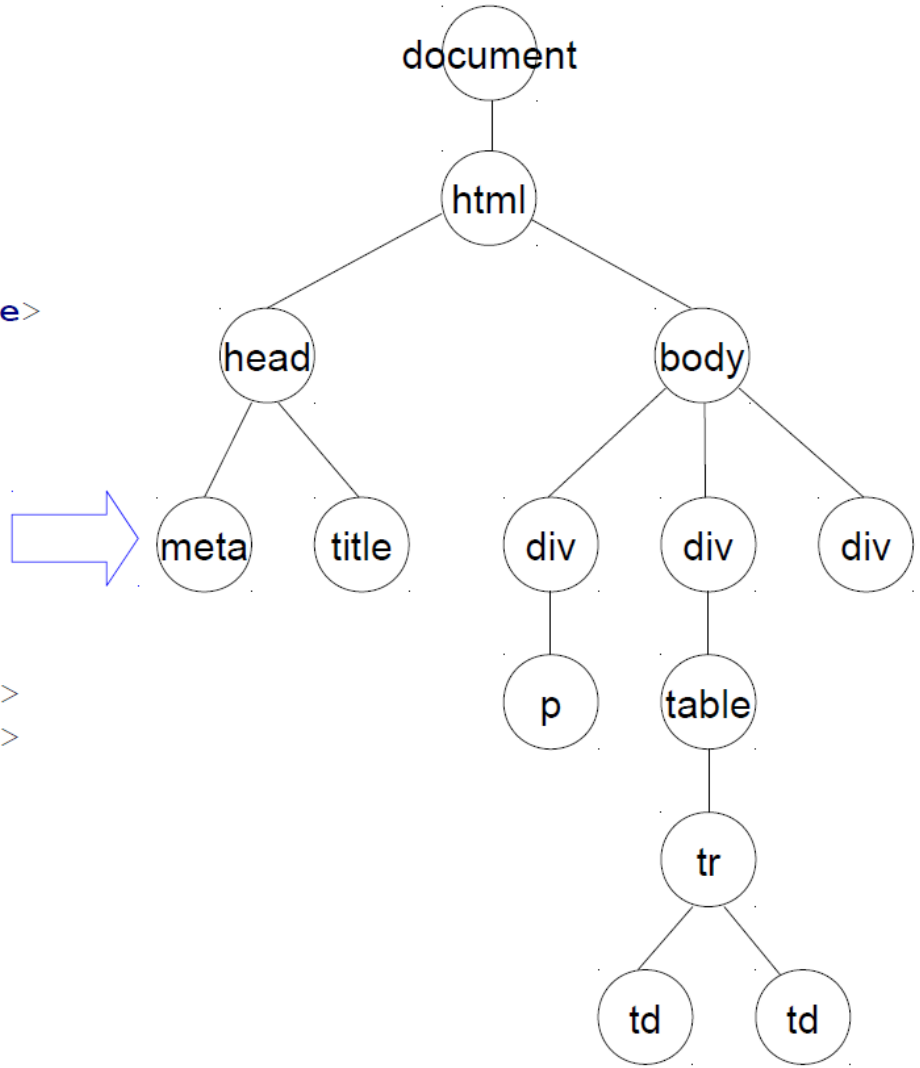
```
<!DOCTYPE html>
<html>
<head>
  <title>Document title</title>
  <meta charset="utf-8">
</head>
<body>
  <div>
    <h1>Heading</h1>
    <p>Paragraph text with a <a href="foo.html">link</a> here.</p>
  </div>
  <div>
    <p>More text here.</p>
  </div>
</body>
</html>
```





Exemple 3

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>A web page</title>
  </head>
  <body>
    <div>
        <p>Paragraphe</p>
    </div>
    <div>
      <table>
        <tr>
          <td>Colonne 1</td>
          <td>Colonne 2</td>
        </tr>
      </table>
    </div>
    <div>Div</div>
  </body>
</html>
```



Accès aux éléments du DOM

Raccourcis pratiques

- La variable globale document est de type Document. Elle contient des attributs qui pointent **directement** sur certains éléments de la page :

document.body	élément correspondant à la balise <body> de la page
document.head	élément correspondant à la balise <head> de la page
document.images	éléments correspondant aux balises <img> de la page
document.links	éléments correspondant aux balises <a> de la page
document.title	éléments correspondant aux balises <title> de la page

Recherche par propriété

- La variable globale document est de type Document. Elle contient des fonctions permettant de rechercher des éléments par **propriété** :

`getElementById(id)`	renvoie l'élément qui possède l'identiant spécifié
`getElementsByTagName(tag)`	renvoie les éléments ayant le nom de balise spécifié
`getElementsByClassName(class)`	renvoie les éléments ayant le nom de classe CSS spécifié

Recherche par sélection CSS

- La variable globale document est de type Document. Elle contient des fonctions permettant de rechercher des éléments par **sélection CSS (id, classe, nom, ...)** :

`querySelector(selector)`	renvoie le premier élément trouvé correspondant au sélecteur CSS spécifié
`querySelectorAll(selector)`	renvoie tous les éléments correspondant au sélecteur CSS spécifié

Exemple 1

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>A web page</title>
  </head>
  <body>
    <div id="i1" class="c2">
      <p>Paragraphe</p>
    </div>
    <div class="c1">
      <table>
        <tr class="c2">
           <td >Colonne 1</td>
           <td>Colonne 2</td>
        </tr>
      </table>
    </div>
    <div id="i2" class="c2">Div</div>
  </body>
</html>
```

```
document.querySelector("#i1");
```

ou

```
document.getElementById("i1");
```

Exemple 2

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>A web page</title>
  </head>
  <body>
    <div id="i1" class="c2">
      <p>Paragraphe</p>
    </div>
    <div class="c1">
      <table>
        <tr class="c2">
            <td >Colonne 1</td>
            <td>Colonne 2</td>
        </tr>
      </table>
    </div>
    <div id="i2" class="c2">Div</div>
  </body>
</html>
```

```
document.querySelectorAll(".c2");
```

ou

```
document.getElementsByClassName("c2");
```

Exemple 3

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>A web page</title>
  </head>
  <body>
    <div id="i1" class="c2">
      <p>Paragraphe</p>
    </div>
    <div class="c1">
      <table>
        <tr class="c2">
          <td >Colonne 1</td>
          <td>Colonne 2</td>
        </tr>
      </table>
    </div>
    <div id="i2" class="c2">Div</div>
  </body>
</html>
```

```
document.querySelectorAll("div");
```

ou

```
document.getElementsByTagName("div");
```

Exemple 4

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>A web page</title>
  </head>
  <body>
    <div id="i1" class="c2">
      <p>Paragraphe</p>
    </div>
    <div class="c1">
      <table>
        <tr class="c2">
           <td >Colonne 1</td>
           <td>Colonne 2</td>
        </tr>
      </table>
    </div>
    <div id="i2" class="c2">Div</div>
  </body>
</html>
```

```
document.querySelectorAll("div.c2");
```

Exemple 5

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>A web page</title>
  </head>
  <body>
    <div id="i1" class="c2">
      <p>Paragraphe</p>
    </div>
    <div class="c1">
      <table>
        <tr class="c2">
           <td >Colonne 1</td>
           <td>Colonne 2</td>
        </tr>
      </table>
    </div>
    <div id="i2" class="c2">Div</div>
  </body>
</html>
```

```
document.querySelectorAll("div+div");
```

Affichage du contenu d'un élément

HTML

```
<p id="i1"> hello world </p>
```

JS

```
let e = document.getElementById('i1');
console.log(e.innerHTML);
```

console

```
hello world
```

Modification d'éléments via l'arbre DOM

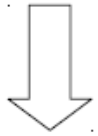
Attributs HTML

`element.hasAttribute(name)`	Renvoie `true` si `element` possède l'attribut `name`, `false` sinon.
`element.getAttribute(name)`	Renvoie la valeur de l'attribut `name` de `element` (`""` ou `null` s'il n'existe pas)
`element.setAttribute(name, value)`	Affecte `value` à l'attribut `name` de `element` (crée l'attribut s'il n'existait pas)
`element.removeAttribute(name)`	Supprime l'attribut `name` de `element`.

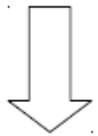
- Exemple :

```
<button id="ok">OK</button>
```

OK



```
document.querySelector("#ok").setAttribute("disabled", "true");
```



```
<button id="ok" disabled="true">OK</button>
```

OK

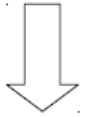
Autres propriétés

`element.id`	Identifiant de `element` (plutôt destiné à être utilisé lors de la création d'un élément)
`element.innerText` `element.textContent`	Contenu textuel de `element` (quelques différences de comportement, notamment dans la prise en compte du style)
`element.className`	Nom des classes CSS appliquées à `element`, séparées par des espaces.
`element.style`	Objet contenant les propriétés CSS de `element`. La propriété CSS `p` est manipulable via `element.style.p`

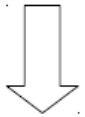
Autres propriétés

```
<p id="p42" class="yellow">A paragraph</p>
```

A paragraph



```
document.querySelector("#p42").className = "green";
```



```
<p id="p42" class="green">A paragraph</p>
```

A paragraph

```
.green{
    background-color: green;
}
```

Création et suppression d'éléments via l'arbre DOM

Ajout et suppression de noeuds

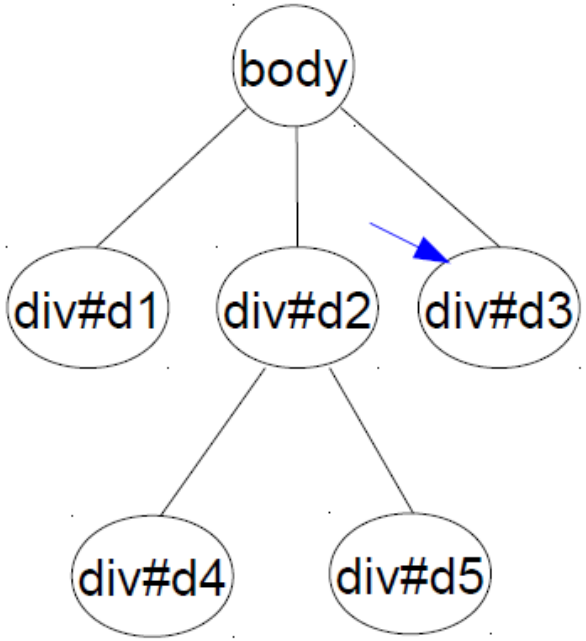
Appel	Effet
`node.removeChild(child)`	Retire `child` des enfants de `node`.
`node.appendChild(child)`	Ajoute `child` à la fin des enfants de `node`.
`node.replaceChild(newChild, oldChild)`	Remplace `oldChild` par `newChild` dans les enfants de `node`.
`node.insertBefore(child, referenceNode)`	Insère `child` dans les enfants de `node`, juste avant `referenceNode`.

Les noeuds en argument peuvent être :

- des noeuds existants (dans ce cas ils seront déplacés)
- des noeuds créés dynamiquement

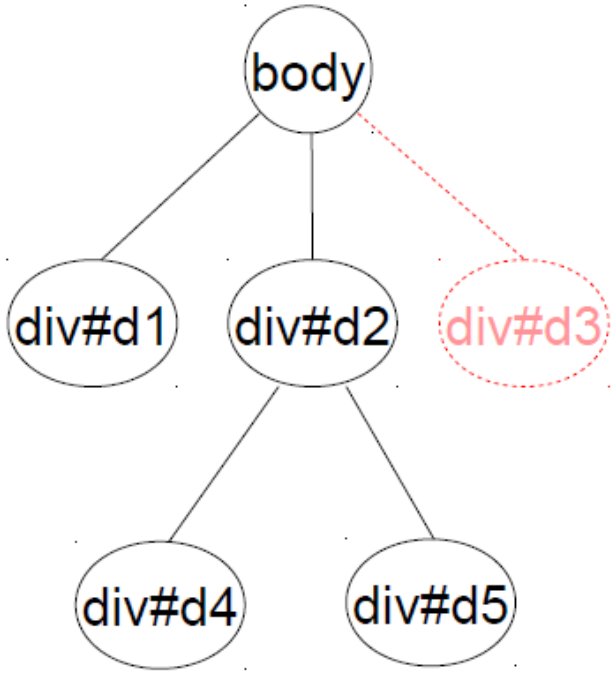
Ajout et suppression de noeuds

```
let d3 = document.querySelector("#d3");
```



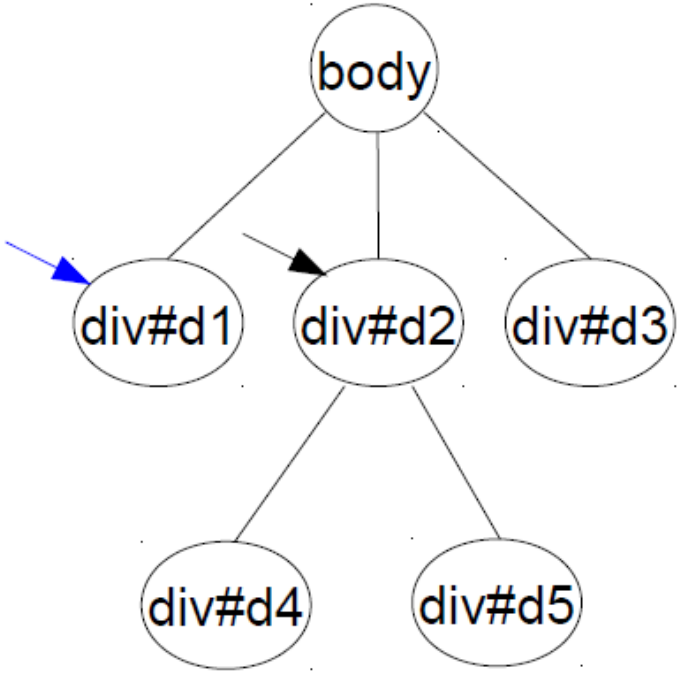
Ajout et suppression de noeuds

```
let d3 = document.querySelector("#d3");
document.body.removeChild(d3);
```



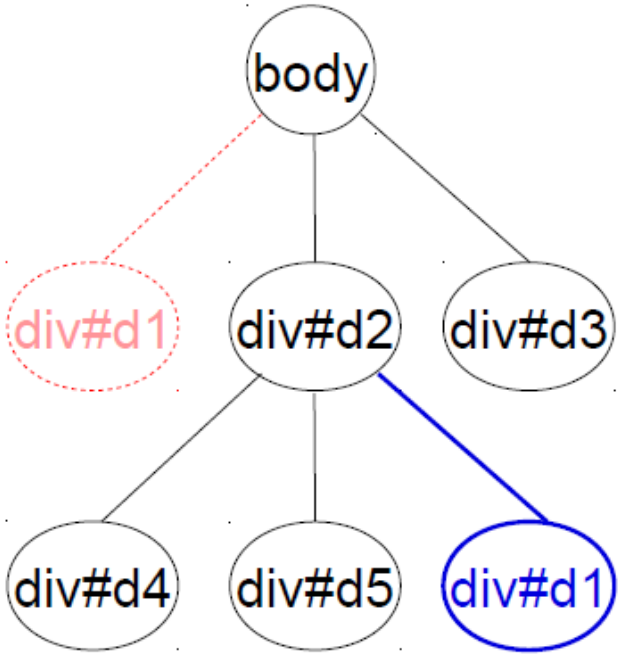
Ajout et suppression de noeuds

```
let d1 = document.querySelector("#d1");
let d2 = document.querySelector("#d2");
```



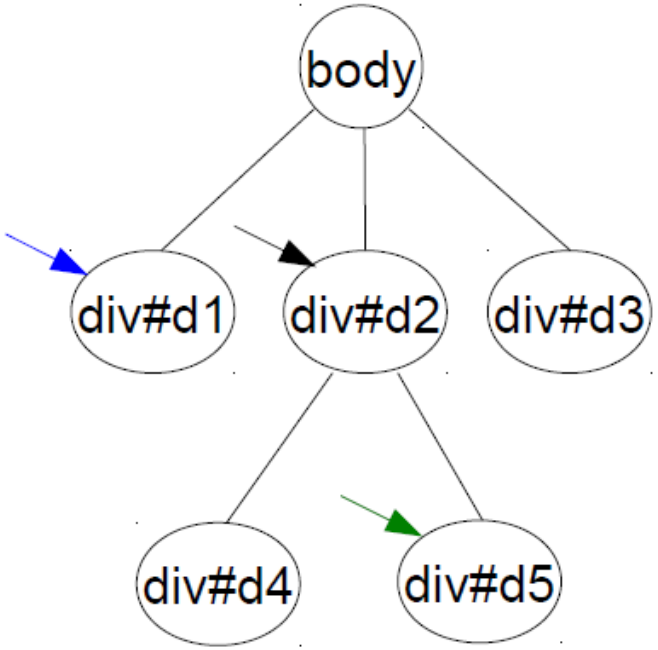
Ajout et suppression de noeuds

```
let d1 = document.querySelector("#d1");
let d2 = document.querySelector("#d2");
d2.appendChild(d1);
```



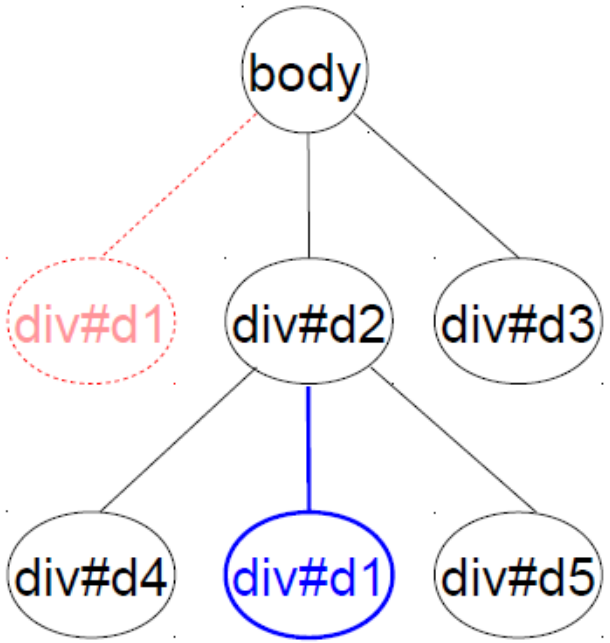
Ajout et suppression de noeuds

```
let d1 = document.querySelector("#d1");
let d2 = document.querySelector("#d2");
let d5 = document.querySelector("#d5");
```



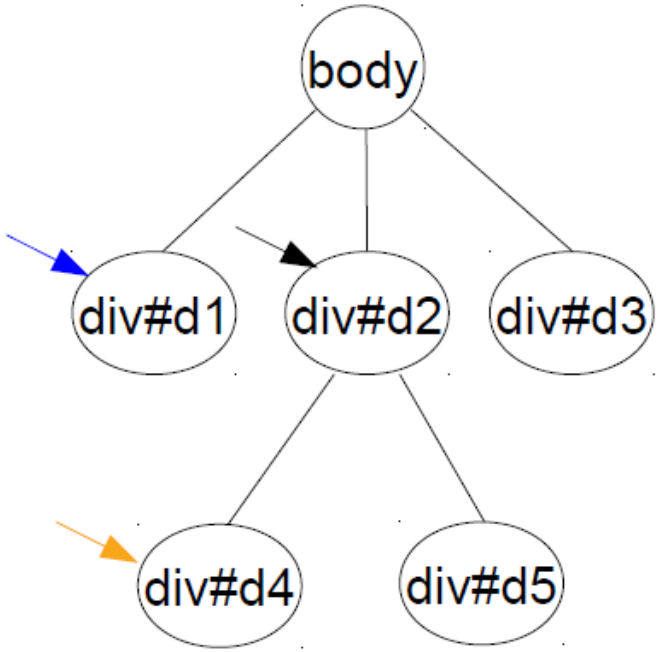
Ajout et suppression de noeuds

```
let d1 = document.querySelector("#d1");
let d2 = document.querySelector("#d2");
let d5 = document.querySelector("#d5");
d2.insertBefore(d1, d5);
```



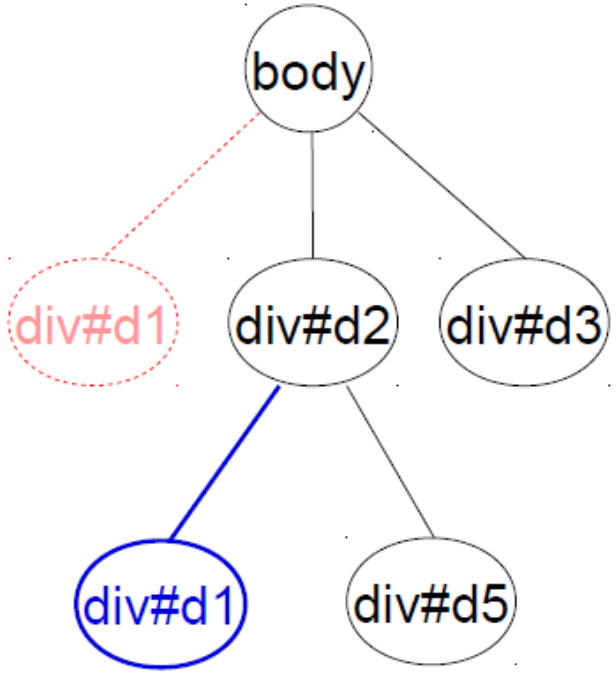
Ajout et suppression de noeuds

```
let d1 = document.querySelector("#d1");
let d2 = document.querySelector("#d2");
let d4 = document.querySelector("#d4");
```



Ajout et suppression de noeuds

```
let d1 = document.querySelector("#d1");
let d2 = document.querySelector("#d2");
let d4 = document.querySelector("#d4");
d2.replaceChild(d1, d4);
```



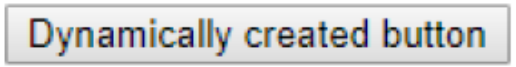
Création dynamique d'élément

- document.createElement(tagName) permet de créer un élément HTML de type tagName

- Exemple:

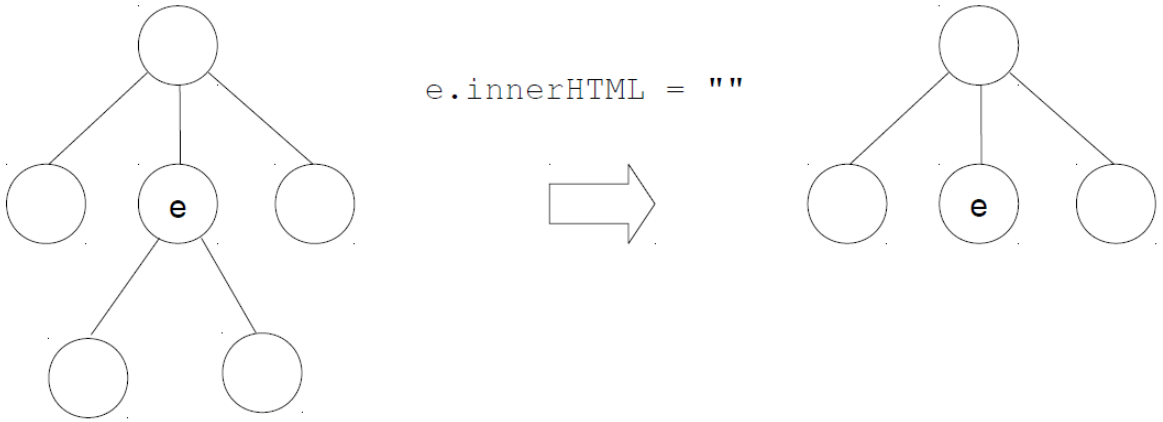
```
let button = document.createElement("button");
button.id = "uniqueId";  //pour retrouver ce bouton par la suite
button.innerText = "Dynamically created button";
```

Créera le bouton :

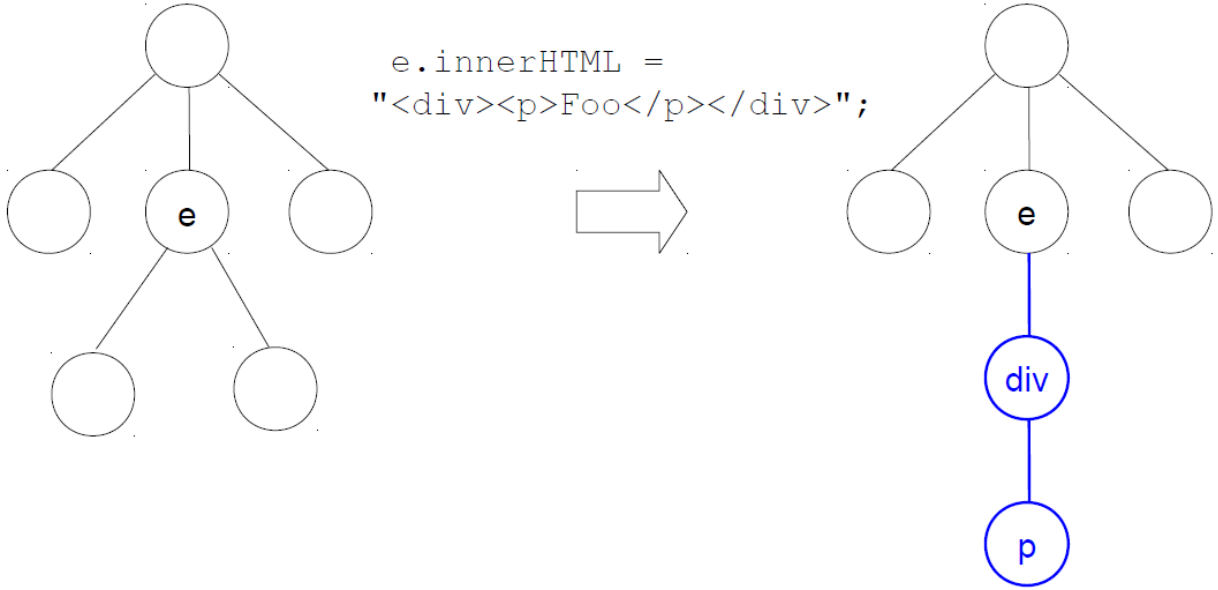


innerHTML

- Suppression

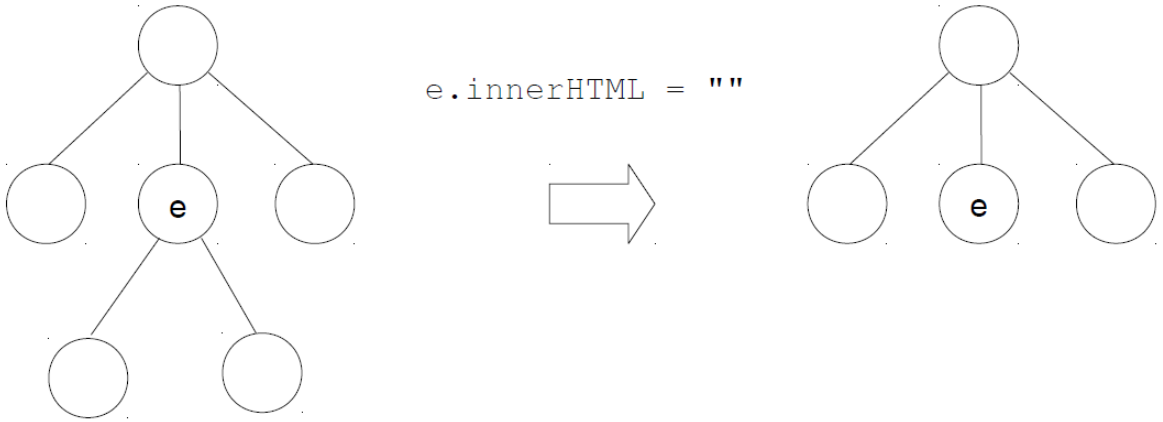


- Ajout



innerHTML

- Suppression



- Ajout

Utilisation d'un gabarit de chaîne