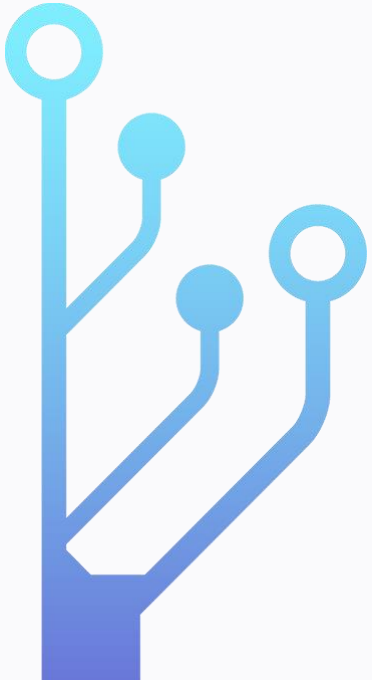
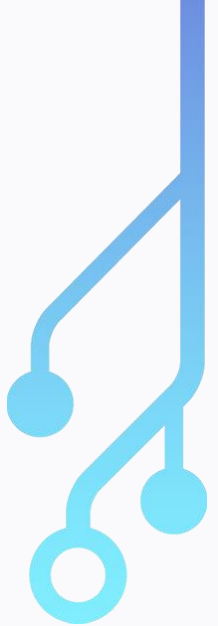
FORMATION DEV WEB

FRONT II : JAVASCRIPT

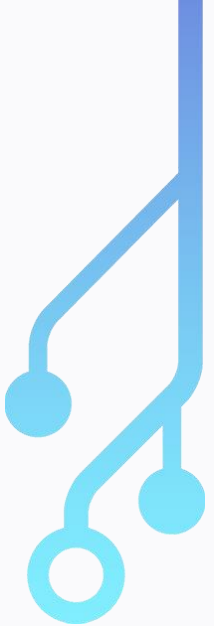




Contexte

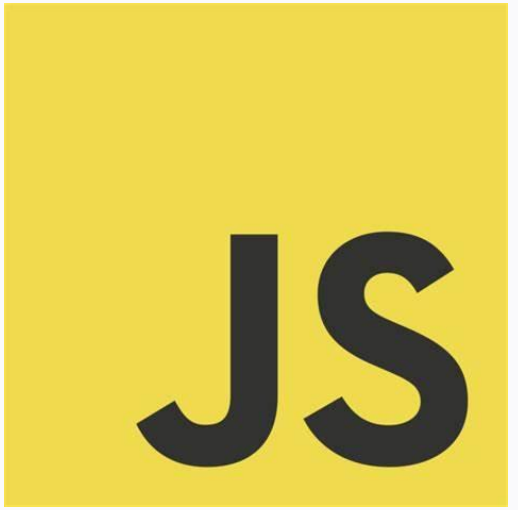
HTML & CSS (Formation Front I) -> faire de beaux trucs, mais ce n'est pas interactif

Le **javascript** entre en scène !



Javascript (JS)

- **Langage de programmation** (top 10 des plus utilisés !)
- Scripts exécutés dans la page du navigateur, qui peuvent interagir avec la page
- Typage dynamique, similaire à Python, mais en plus laxiste
- Rien à voir avec Java !

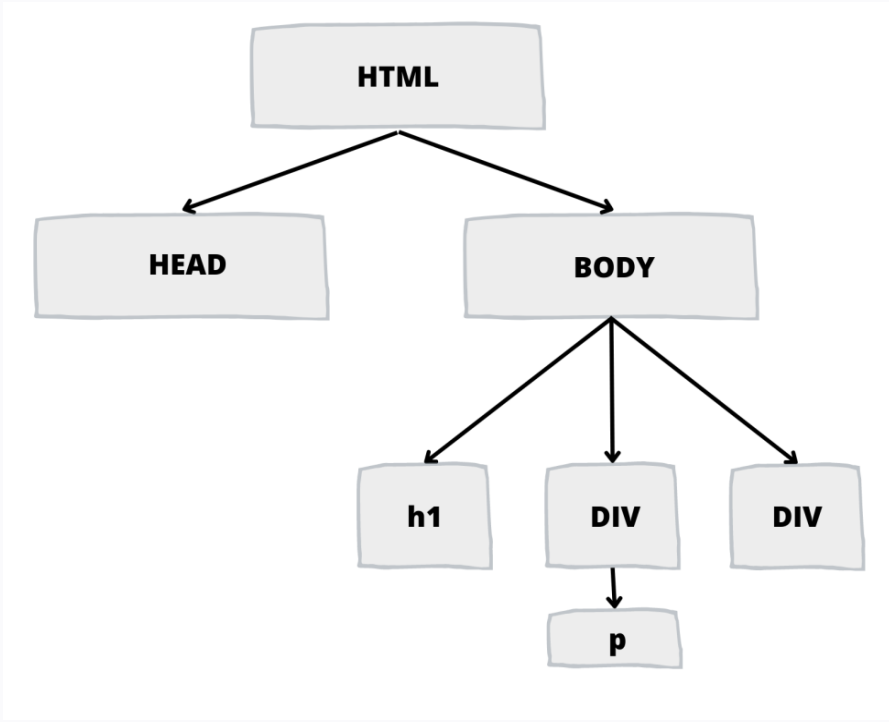


Rappel : le DOM

Le DOM est un assemblage de balises "emboitées" les unes dans les autres.

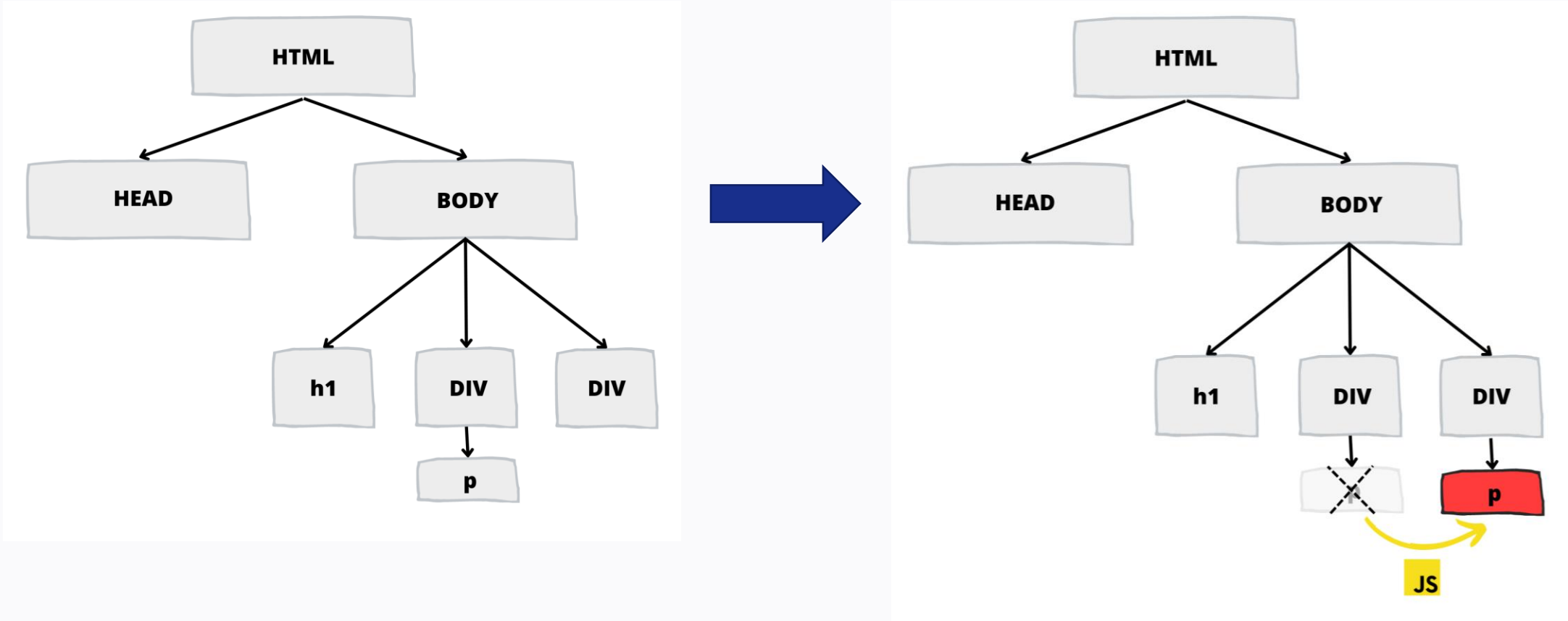
- **HEAD : contient les métadonnées**
- **BODY : contient les éléments visibles à l'écran**

```
<html>
    <head>
    </head>
    <body>
        <h1></h1>
        <div>
            <p></p>
        </div>
        <div></div>
    </body>
</html>
```



A quoi va servir JavaScript ?

JS va nous permettre de modifier l'organisation du DOM.



Où écrit-on du JavaScript ?

⚠ **LA BASE EST UN FICHIER HTML**

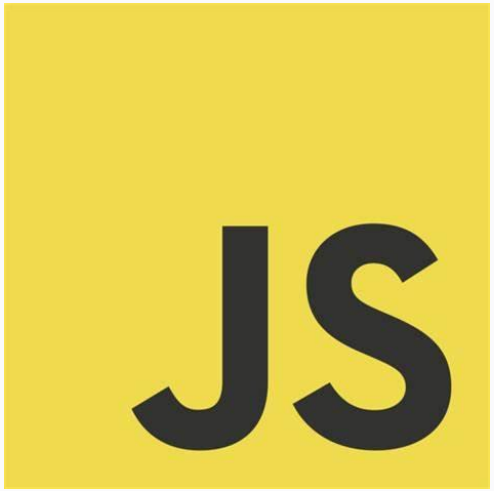
RAPPEL CSS

- <style>...</style> dans le fichier HTML
- On lie le fichier html à un fichier css **<link rel="stylesheet" href="style.css">**

JAVASCRIPT

- <script>...</script> dans le fichier HTML
- On lie le fichier html à un fichier JS **<script type="text/javascript" src="script.js"></script>**

LA SYNTAXE



Syntaxe – Types de valeur

Similarités avec python :

- Nombres : **10**, **0.1**, **1e3**
- Chaînes de caractères **''Hello world!''**
- Listes **[1.0, ''bye'']**
- Booléens (sans majuscules) **true**, **false**
- Valeurs spéciales **undefined**, **null**
- Objets (≈ dictionnaires Python)

```
{
        key: value,
        ''via'': ''rézo'',
        0: true,
}
```

Syntaxe - variables

Variable locale et mutable **: let variable = 0;**

Variable locale et immutable : **const variable = 0;**

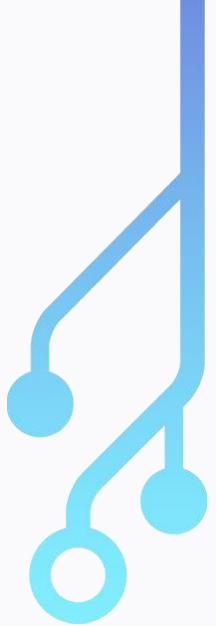
Il existe aussi une syntaxe avec **« var »** et une syntaxe pour les **variables globales**.

Changer les valeurs de la variable :

```
let variable = 0;

variable = 1;

variable++;
```

⚠ On met des points virgules à la fin de chaque ligne ! ⚠

Syntaxe – opérations de base



- Opérations numériques de base (**+, -,** etc.) sauf le **//**
- Appel de fonctions, indexation de liste ou objets

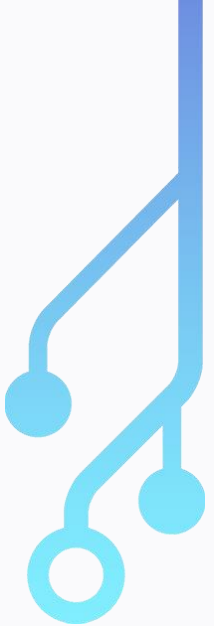
- **objet[“clé”]** ou **objet.clé**
- Existence d’une clé dans un objet : **”clé” in object**

- Opérations logiques : **!** (non), **&&** (et), **||** (ou)

⚠ Attention **aux doubles symboles** en Javascript ! ⚠

- Comparaisons **==, ===, !=, !==, <=, > etc.**

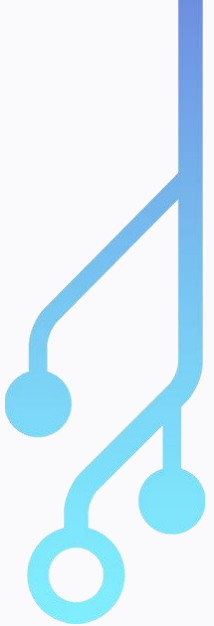
Syntaxe – les commentaires



```
// commentaire sur une ligne //

/* commentaire
qui
prend
beaucoup
de
lignes */
```

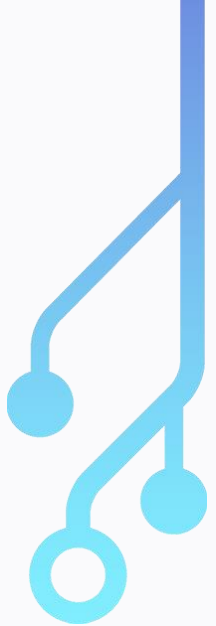
Syntaxe – les conditions

```
if (ma condition) {
        instructions;
        instructions;
} else if (mon autre condition) {
        instructions;
}
else {
        instructions;
}
```



Syntaxe – les conditions

```
if (ma condition) {
        instructions;
        instructions;
} else if (mon autre condition) {
        instructions;
}
else {
        instructions;
}
```



Syntaxe – les boucles

```
while (ma condition) {
        instructions;
}
```

```
for(let i = 0; i < 10; i++) { ... }
```

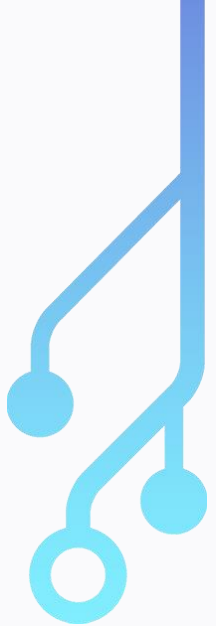
Il existe aussi des boucles pour parcourir facilement des listes, objets, ...

Syntaxe – les fonctions

```
const maFonction = (arg1, arg2) => {
        instructions;
        return valeur;
};
```

ou :

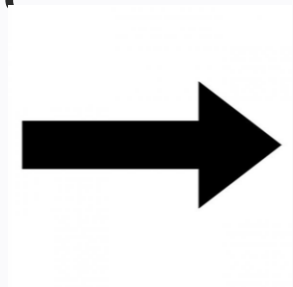
```
function maFonction (arg1, arg2) {
        instructions;
        return valeur;
};
```



Syntaxe – les objets

C'est comme en python :

```
class Book {
    constructor(title, author, pages) {
        this.title = title;
        this.author = author;
        this.pages = pages;
    }
}
```



```
let myBook = new Book("L'Histoire de Tao", "Will Alexander", 250);
//Cette ligne crée l'objet suivant :
{
    title: "L'Histoire de Tao",
    author: "Will Alexander",
    pages: 250
}
```

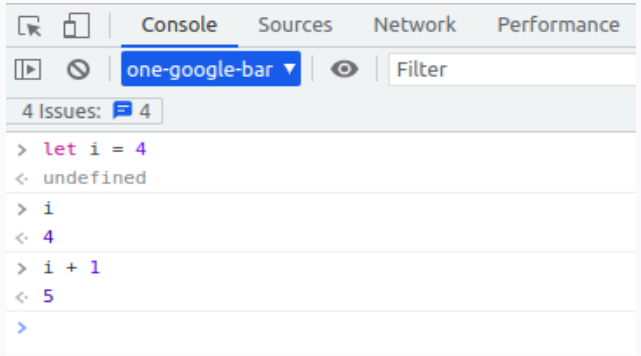
Syntaxe – les objets – Pourquoi ?

- Avoir une structure de données plus organisée
- Faire des opérations dessus (des "méthodes") :
- Plein d'autres choses (mais ce n'est pas le but de cette formation)
- Faire des tableaux d'objets

```
// Le tableau qui permet d'enregistrer nos objets
Profils = [
    new Profil("chenow", "bruce", "miam les pates", "chenow.burce@viarezo.fr"),
    new Profil("hamza", "touize", "miam les pizza", "hamza.touize@viarezo.fr"),
    new Profil("miel", "des champs", "miam les patates", "miel.des_champs@cs.fr"),
]
```

La console késako ?

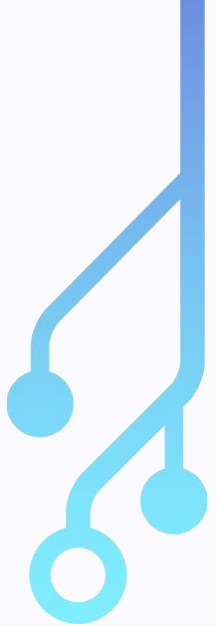
- **Taper directement du js**



Ctrl + Maj + J (Win/Linux)
⌘ + Option + J (MacOS)

- **Ecrire des fonctions**

Ctrl + Maj + K (Win/Linux)
⌘ + Option + K (MacOS)



J'ouvrirai une console en live pour afficher hello world !

🔢 Exercice (10 min) 🔢

Ecrire dans la console une fonction qui prend un paramètre en entrée (un nombre) et renvoie son carré.

Si plus de temps, **écrire dans la console** une fonction qui prend un paramètre en entrée (un nombre supposé être un âge) et renvoie un string « tu es majeur.e » ou « tu es mineur.e ».

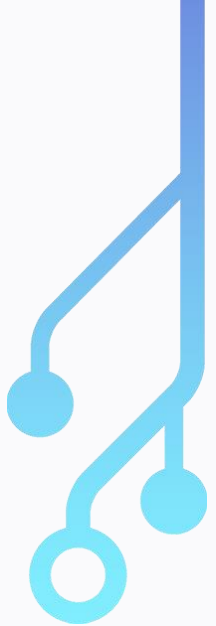
Où écrire mon javascript ? (Rappel du rappel)

On écrit le javascript dans un fichier .js

On l'inclut dans le header du fichier html :

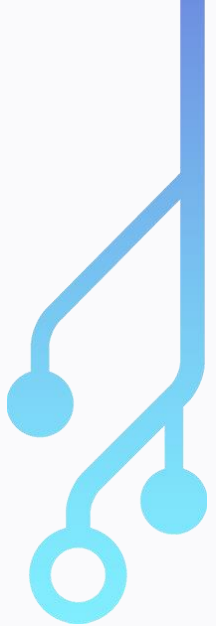
```
<!DOCTYPE html>
<html lang="fr">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Front II</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css" rel="stylesheet"
        integrity="sha384-iYQeCzEYFbKjA/T2uDLTpkwGzCiq6soy8tYaI1GyVh/UjpbCx/TYkiZhlZB6+fzT" crossorigin="anonymous">
    <link rel="stylesheet" href="index.css" type="text/css" />
    <script type="text/javascript" src="index.js"></script>
      chenow, 4 days ago • first
</head>
```

La console : oui, mais pour afficher quoi ?



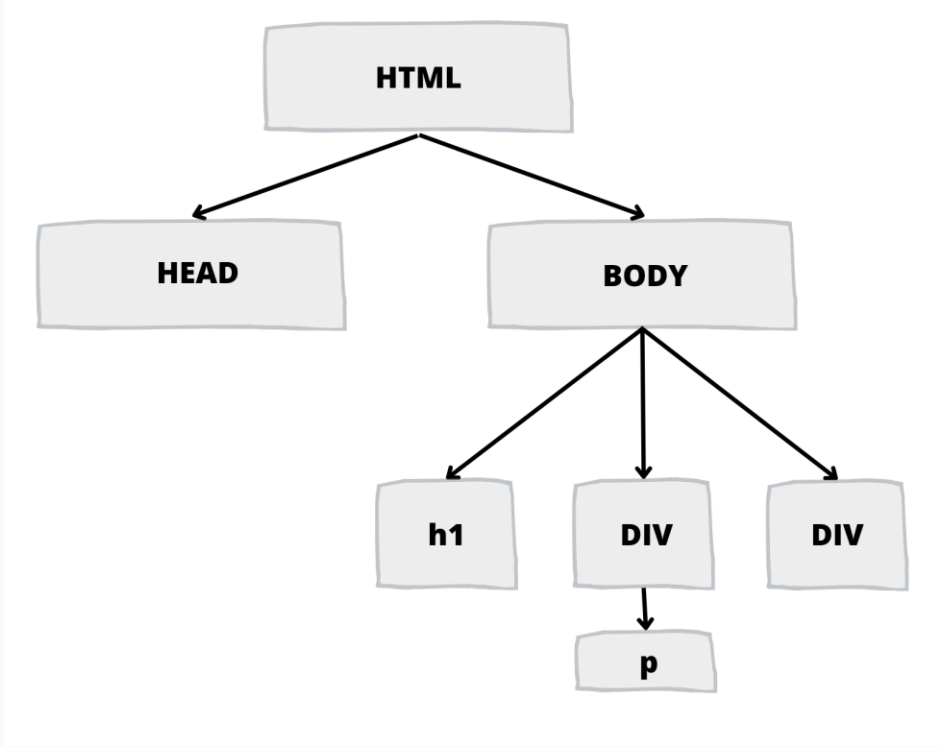
Présentation de mon dossier exemple

1234 Le deuxième TD (20 min) 1234



Compléter la fonction print_profil pour afficher les champs des profils (nom , prénom …) stockés dans le dictionnaire.

Modifier le DOM



Modifier le DOM

1ère étape : sélectionner les nœuds que l'on veut modifier.

- `const element = document.getElementsByTagName('div');`
- `const element1 = document.getElementById("identifiant");`
- `const element2 = document.querySelector("classe");`

Fichier HTML :

```
<div id="identifiant"></div>
<div class="classe"></div>
```

On peut aussi sélectionner les nœuds liés :

- `const parent = element.parentNode`
- `const children = element.children`

Modifier le DOM

On peut maintenant modifier les propriétés de l'élément sélectionné !

Quelques propriétés : *attributes, children, parentElement, classList, id, style, value...*

```
const element = document.getElementsById('myId");

element.innerText = "Je choisis le texte dedans.";

element.innerHTML = "<div>Je viens d'insérer une div avec du texte </div>";
```

Et il existe encore bien plus de fonctions, plus d'infos sur : https://developer.mozilla.org/

Modifier le DOM

On peut aussi rajouter ou supprimer depuis le Javascript des nœuds du DOM.

```
const newParagraphe = document.createElement("p");

newParagraphe.innerText="Encore un nouveau texte !";

body.appendChild(newParagraphe);
```

Ou

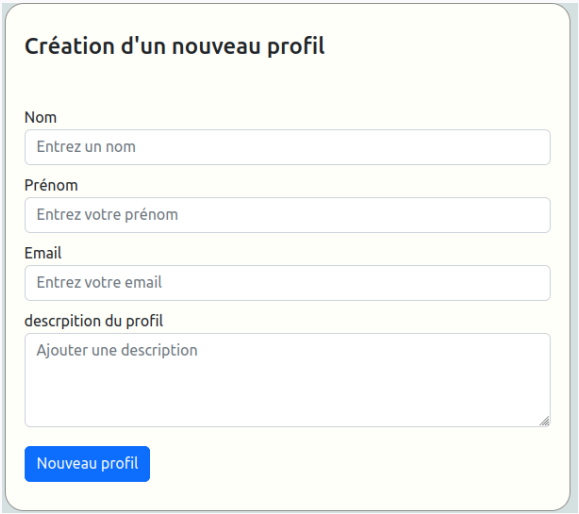
```
myElement.insertAdjacentHTML(position,  '<div>Nouvelle div</div>');
```

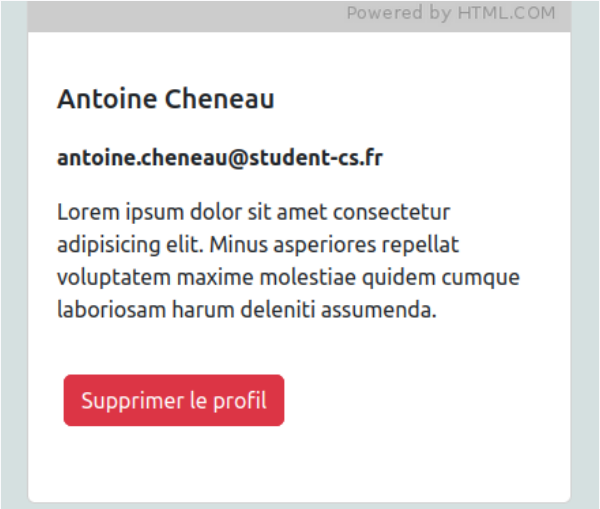
Suppression :

```
myElement.remove();
```

1234 Le troisième TD (20 min) 1234

Créer des fonctions JavaScript 'new_profile' et 'delete_profile' permettant d'ajouter et supprimer un profile lorsqu'on clique sur un bouton associé

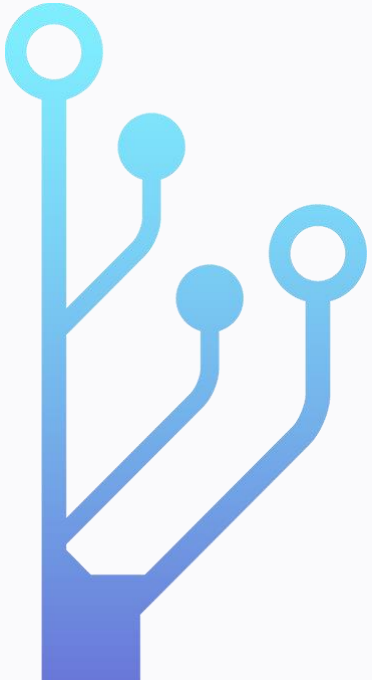


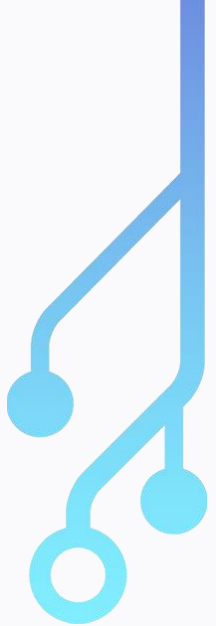


Indice : l'attribut HTML 'onclick' et l'opérateur JS 'this' pourraient vous aider...

LES EVENTS

Ou comment réagir aux actions de l'utilisateur





Les évènements

Pour exécuter nos fonctions, nous avons besoin de créer des événements. Ils permettront de réagir aux actions des utilisateurs.

Quelques types d'évènement utiles : **click, keydown, change, ...**

- `myElement.addEventListener("click", myFunction);`
- `myElement.addEventListener("change", myFunction);`

On peut aussi créer des balises HTML qui exécuteront du code JS lorsque l'événement spécifié est exécuté.

- `<div onclick='myFunction();'></div>`

1234 Le quatrième TD (20 min) 1234

Question ouverte :

Faire en sorte que les profils correspondants au motif de recherche entrée soient affichés en dessous de la barre de recherche.