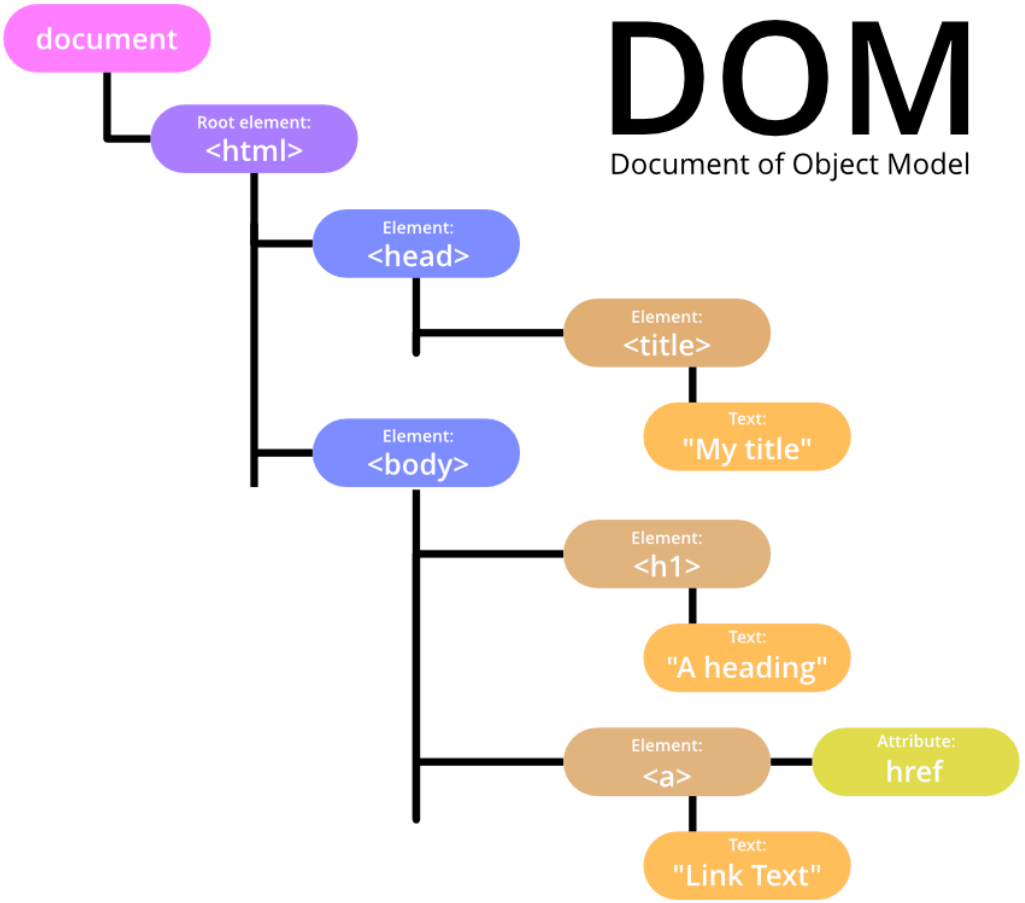
DOM

Définition: Le DOM* est comme l'ossature d'une page web. C'est une représentation hiérarchique (sous forme d'arbre*) des éléments de la page qui permet à JavaScript d'interagir dynamiquement* avec le contenu. En d'autres termes, le DOM permet à votre code JavaScript de lire, changer, et manipuler la structure et le style d'une page web en temps réel.

Chaque branche de cet arbre est composée de nœuds* qui vont contenir des objets.

Le DOM est l'une des API* les plus utilisées sur le WEB parce qu'elle autorise du code exécuté dans un navigateur à accéder et interagir avec chaque nœud dans le document. Les nœuds peuvent être créés, déplacés et modifiés.



1. Le Document : Point d'Entrée du DOM
    - Le DOM commence par le "document", qui est l'objet racine*. Bien que techniquement pas un élément HTML, le document représente la page web entière et sert de point d'accès pour manipuler le contenu de la page.
    - Il est le niveau le plus élevé dans le DOM et permet l'accès à tous les autres éléments (nœuds) de l'arbre DOM.
2. L'Élément <html> : La Racine de l'Arbre HTML
    - Dans la structure de l'arbre DOM, l'élément <html> est considéré comme la racine de l'arbre HTML.
    - C'est le premier élément sous le "document" et contient deux principaux sous-éléments : <head> et <body>.
3. Les Branches : Éléments HTML
    - Chaque élément HTML dans le DOM est comme une branche de cet arbre. Ces branches représentent des éléments tels que , <p>, <header>, <footer>, etc.
    - Ces branches peuvent avoir leurs propres sous-branches, reflétant la nature imbriquée des éléments HTML.
4. Les Feuilles : Contenu et Éléments Finaux
    - Les feuilles sont les éléments qui ne contiennent pas d'autres éléments HTML. Elles peuvent être des éléments de texte, des images (<img>), des liens (<a>), etc.
    - Ces éléments sont les points terminaux de l'arbre et portent le contenu réel de la page (texte, images, etc.).
5. Interaction et Modification via JavaScript
    - JavaScript permet d'interagir avec le DOM. On peut utiliser JavaScript pour ajouter, modifier ou supprimer des éléments (branches ou feuilles), changer des styles, ou répondre à des événements*.
    - Par exemple, on peut modifier le texte d'un paragraphe, ajouter une nouvelle section ou réagir à un clic sur un bouton.
6. Navigation dans l'Arbre DOM
    - La structure arborescente du DOM facilite la navigation et la sélection d'éléments spécifiques.
    - On peut utiliser des méthodes JavaScript pour "descendre" ou "monter" dans l'arbre, comme getElementById, querySelector, ou en accédant aux propriétés parent/enfant des éléments

Ci dessous, voyons la structure d'un document html source représentée sous forme d'arbre composé de noeud et de feuilles
Chaque nœud est lié aux autres par une relation parent-enfant. Le premier nœud de l'arbre est la racine, tandis que les nœuds contiennent des feuilles.

- STRUCTURE :

    <!doctype html>
    <html lang="en"> // la balise <html> est le *parent* (:root) / racine
    <head> // La balise <head> est une *branche*
      <title>My first web page</title> // La balise <title> est un enfant & le contenu de <title> est la *feuille*
    </head>
    <body> // La balise <body> est une autre/seconde *branche*
      <h1>Hello, world!</h1> // La balise <h1> est un *enfant* et 'Hello, world!' est la *feuille*
      How are you? // est un *enfant* et 'How are you?' est la *feuille*
    </body>
    </html>
  - La structure d'objet du DOM est représentée par ce qu'on appelle une "*arborescence de noeuds*" (**node tree**)

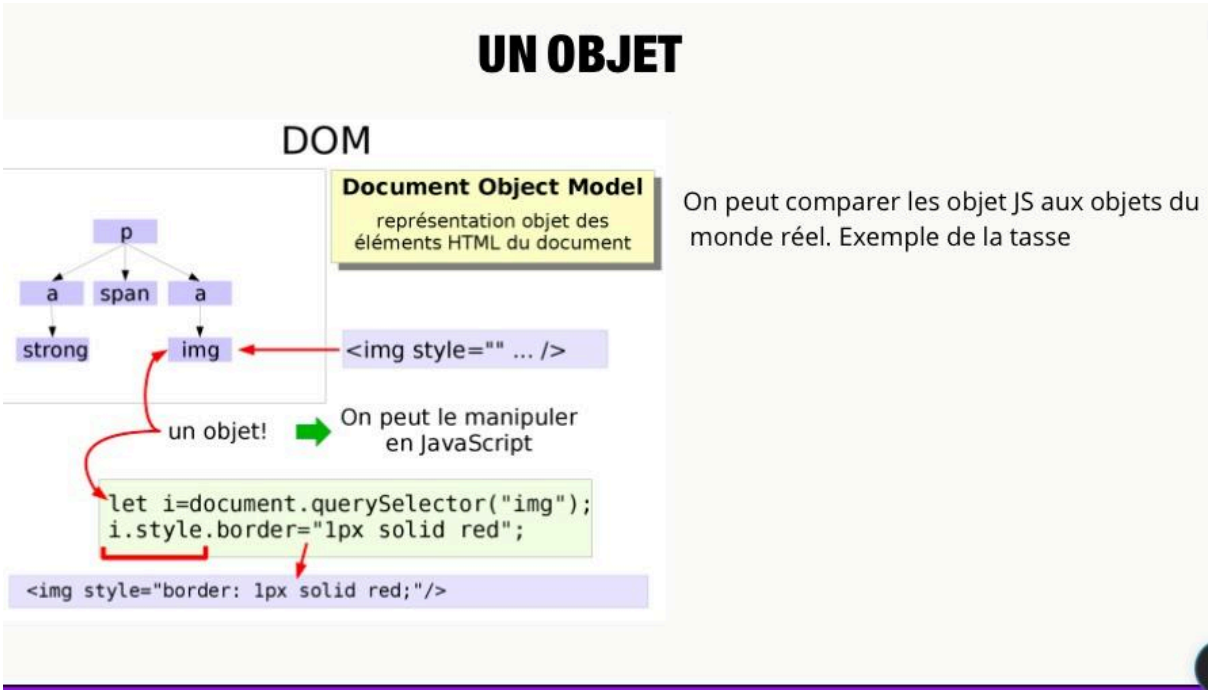
```
// On va sélectionner le document
// On log dans la console afin d'avoir un affichage complet de l'arborescence
// de la structure HTML en partant de la racine (document)
// Ceci est la représentation du DOM
console.log(document);
```

code plié:

```
▶#document (http://127.0.0.1:5500/index.html)
Live reload enabled.
```

code déplié:

```
▼#document (http://127.0.0.1:5500/index.html)
   <!DOCTYPE html>
   <html lang="fr">
   ▼<head>
      <meta charset="UTF-8">
      <meta name="viewport" content="width=device-width, initial-scale=1.0">
      <link rel="stylesheet" href="styles.css">
      <title>YOANN - KEVIN - JS</title>
    </head>
   ▼<body cz-shortcut-listen="true"> flex
     ▶<div class="container">…</div>
      <script src="test.js"></script>
      <!-- Code injected by live-server -->
     ▶<script>…</script>
     ▶<daily-companion-app>…</daily-companion-app>
     ▶<deepl-input-controller>…</deepl-input-controller>
    </body>
   </html>
Live reload enabled.
>
```



définition: Un objet, c'est un élément qui peut être modifié (par CSS, JS, etc..)

GLOSSAIRES:

- **Arbre**: c'est une représentation d'un élément composé de nœud de façon hiérarchique.

- **Noeud** : Dans le contexte du DOM, un nœud est un point unique dans l'arbre des nœuds du DOM. Parmi les différentes choses qui sont des nœuds, on trouve le document lui-même, les éléments, le texte et les commentaires, "il existe 12 nœuds différents" voici les 4 principaux.

- **Racine:** c'est la base du document (ex: html) ensuite on parlera d'éléments qui sont des enfants directs. En d'autres termes, ils sont imbriqués dans celui donné. Par exemple, <head> et <body> sont des enfants de l'élément <html> et ensuite de tous les éléments imbriqués dans l'élément donné, y compris les enfants, leurs enfants, etc. (Par exemple, ici <body> a des enfants et <ul>)

- **API :** (Application Programming Interfaces soit « interface de programmation d'application » en français) sont des constructions disponibles dans les langages de programmation pour permettre aux développeuses et développeurs de créer plus facilement des fonctionnalités complexes. Elles fournissent une abstraction sur les parties de code plus complexes, fournissant ainsi une syntaxe plus facile à utiliser à la place.

- **Dynamiquement**: Quand on parle de “dynamique” dans js, c’est de créer une interactivité avec entre les pages web.

- **DOM** (description : *Document Object Model*)
    1. **Document :** Le DOM concerne un document, tel qu'une page web affichée dans un navigateur. Une page web commence par une balise, “!DOCTYPE” suivi de la balise <html> dans laquelle se trouve le reste du document. Le DOM représente le document affiché par une structure en arbre, comportant des nœuds (branches et feuilles).
    2. **Objet :** En programmation, un objet est un conteneur qui comporte des propriétés et des méthodes – qui sont des variables et des actions concernant ce qu'il représente. Les objets du DOM peuvent représenter une fenêtre, un document, une phrase, un style…
    3. **Modèle :** Un modèle sert à représenter quelque chose, comme le plan d'une ville. Le DOM représente le document qui se trouve dans le navigateur.

- **Événement**:
    Les évènements (aussi appelé “event”) sont des éléments actifs générés par les éléments DOM qui peuvent être manipulés par un code Javascript.

SÉLECTEUR:

définition: c’est une méthode Javascript pour cibler un élément dans un autre fichier afin de pouvoir modifier son style.

- Sélectionne un élément par son ID
    - let monElement = document.getElementById('monId');

- Sélectionne tous les éléments avec une classe spécifique
    - let elements = document.getElementsByClassName('maClasse');

- Sélectionne le premier élément correspondant à un sélecteur CSS
    - let premierElement = document.querySelector('.premier');

Manipulation du Contenu :

- Affiche et récupère le contenu d'un élément
    - monElement.innerHTML = 'Nouveau Contenu';

- Modifie un attribut
    - monElement.setAttribute('src', 'nouveau-src.jpg')

Modifier le Style :

- Change la couleur du texte
    - monElement.style.color = 'red';

- Modifie la taille de la police
    - monElement.style.fontSize = '18px';

Ajouter/Supprimer des Éléments :

- Crée un nouvel élément
    - let nouvelElement = document.createElement('div')

ÉVÉNEMENTS:

La capture d'un événement (dit aussi “event”) consiste à exécuter une action lorsque l'événement surveillé se produit dans le document.

Les événements capturables du DOM sont :

page et fenêtre :

- onabort — s'il y a une interruption de chargement
- onerror — en cas d'erreur pendant le chargement de la page
- onload — après la fin du chargement de la page
- onbeforeunload — se produit juste avant de décharger la page en cours (par changement de page, en quittant)
- onunload — se produit lors du déchargement de la page (par changement de page, en quittant)
- onresize — quand la fenêtre est redimensionnée

souris :

- onclick — sur un simple clic
- ondblclick — sur un double clic
- onmousedown — lorsque le bouton de la souris est enfoncé, sans forcément le relâcher
- onmousemove — lorsque le curseur est déplacé
- onmouseout — lorsque le curseur sort de l'élément
- onmouseover — lorsque le curseur se trouve sur l'élément
- onmouseup — lorsque le bouton de la souris est relâché

- onscroll — lorsque le scroll de la souris est utilisé

clavier :

- onkeydown — lorsqu'une touche est enfoncée
- onkeypress — lorsqu'une touche est pressée et relâchée
- onkeyup — lorsqu'une touche est relâchée

formulaire :

- onblur — à la perte du focus
- onchange — à la perte du focus, si la valeur a changé
- onfocus — lorsque l'élément obtient le focus (ou devient actif)
- onreset — lors de la remise à zéro du formulaire via un bouton ou une fonction reset()
- onselect — quand du texte est sélectionné
- onsubmit — quand le formulaire est validé via un bouton ou une fonction submit()

Cela permet de créer une animation/intéraction sur notre page web.
Exemple: https://developer.mozilla.org/fr/docs/Web/API/Element/click_event avec l’event “onclick”