

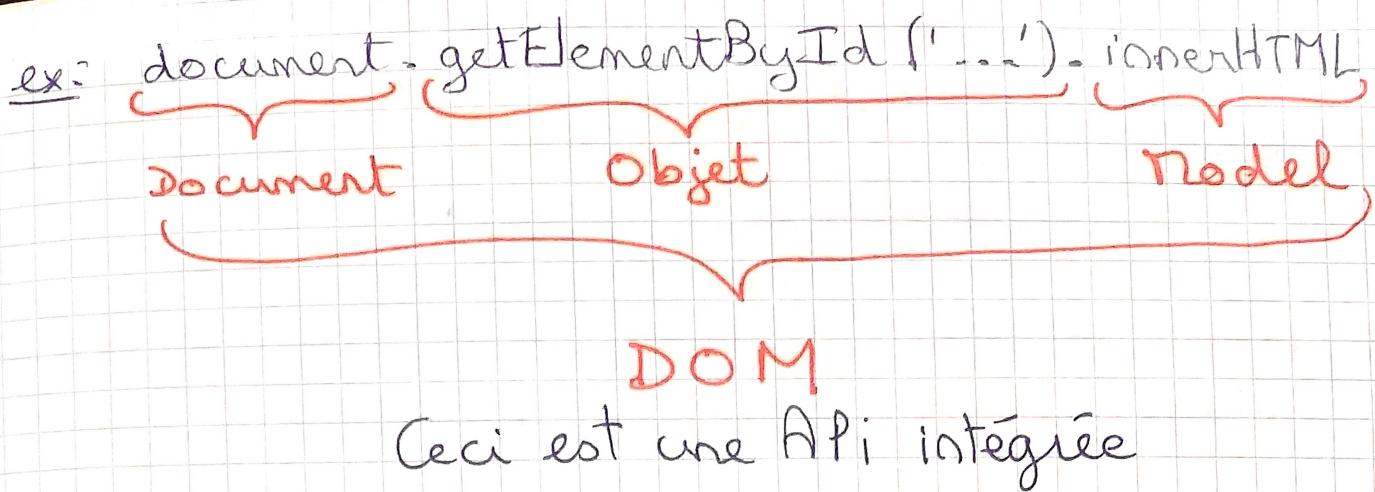
Le BOM

Info

API - Application Programming Interface . C'est un ensemble de codes grâce à laquelle un logiciel fournit des services à des clients. Permet à des personnes de faire des opérations complexes sans connaître la complexité . Tout comme on démarre une voiture avec une clé sans comprendre comment tout cela se déclenche.

API intégrées aux navigateurs : On utilise le DOM (Document Object Model), qui permet de manipuler le HTML et le CSS d'une page API Geolocation. L'API Canvas permet de dessiner et manipuler des graphiques dans une page . Par exemple pour faire des jeux .

API externes : Proposés par certains logiciels ou sites comme la suite d'API Google Maps qui permettent d'intégrer et manipuler des cartes dans nos pages web ou encore l'API Twitter qui permet d'afficher sur une liste de Tweets sur un site par exemple . Ou bien l'API YouTube qui permet d'intégrer des vidéos sur un site . Par exemple Bootstrap est une API Externe .



Le BOM (Browser Object Model) n'est pas un nom officiel. Le BOM est une sorte de SUPER API. Le BOM est une fenêtre window + l'interface + l'objet + ouvrir un onglet + etc. Window fait parti du BOM.

Ce qui appartient au BOM et qui sont des enfants de window:

- objet Navigateur, API Geolocation (emplacement, etc)
- objet History (cookies, etc)
- objet location (infos url)
- objet Screen (propriétés de l'écran)
- objet Document (le DOM)

Propriétés:

- | | |
|---------------|---------------------|
| - outerHeight | } Exterieur |
| - outerWidth | |
| - innerHeight | } Intérieur Visible |
| - innerWidth | |
- Retourne hauteur et largeur de la fenêtre invisible dans le navigateur

Ex: `document.getElementById('un').innerHTML = 'taille de la fenêtre (ext): ' + window.outerWidth + '*' + window.outerHeight,`

Les méthodes (fonctions) de Window

open(...) permet d'ouvrir une fenêtre, onglet ou iframe
ex: `window.open` dans le html

```
<button id='b1'> ... </button>
let b1 = document.getElementById('b1');
b1.addEventListener('click', openWindow);
function openWindow() {
    fenêtre = window.open('...');
```

}

On n'oubliera pas d'écrire la taille de la fenêtre avec 'width=500, height=500';

<code>window. size resizeBy</code>	fait passer la fenêtre définie
<code>window. resizeTo</code>	retaille la fenêtre en fonction de la taille qui on lui donne
<code>window. moveBy</code>	déplace la fenêtre en fonction des coordonnées qui on lui donne
<code>window. moveTo</code>	revient à la place initiale que je définie
<code>window. scrollBy</code>	Net un accesseur à la fenêtre
<code>window. scrollTo</code>	Remonte en haut de la page
<code>window. close</code>	Ferme la fenêtre

`addEventListener` signifie "ajouter un gestionnaire d'évènement"

`alert("...")` ou `window.alert("...")` affiche un pop-up.

`prompt("...")` affiche une barre de saisie

INFOS SUR L'UTILISATEUR

→ `navigator` "Données" données par l'utilisateur

`navigator.language` donne le nom du pays en abrégé

`navigator.cookieEnabled` a-t-il les cookies ?

`navigator.platform` Nom de la plateforme MAC ou Windows

Grâce à cela nous avons des infos sur les données de l'utilisateur.

`navigator.geolocation` Géolocalisation pour voir où se trouve l'utilisateur.

→ Voir le lien ci-dessous pour voir toutes les propriétés de navigateur : <https://developer.mozilla.org/fr/docs/Web/API/Navigator>

`getCurrentPosition` Donne l'latitude et longitude de l'utilisateur

`watchPosition` Chaque fois que la position de l'utilisateur change on peut voir où il est.

`clearWatch` Efface l'appel de la fonction de tracage

- Connaitre l'historique de site de l'utilisateur

`history.length` Donne le nombre de sites visités en cours

`history.go` Charger une page depuis l'historique

`history.back` Comme retour (comme la flèche ⬅)

`history.forward` Aller une page en avant (➡)

Parmis les DOM il y a aussi ces interfaces :

window (que l'on a en avant)

Event ex: actions de notre souris / Evènement

EventTarget (à venir plus tard)

Node Noeud (pas important) / départ d'une branche

Document (ce que l'on a vu avant)

Element (à voir plus tard)

ParentNode On atteind son parent

childNode On atteind son enfant

NonDocumentTypeChildNode

HTMLElement

NonElementParentNode

QuerySelector

QuerySelectorAll

ex: document.querySelector('p').style.color='blue';

Cible page ↑
sélectionne ↑
élément ↑
style CSS ↓
ordre ↓

GetElementById par le id

GetElementsByClassName par le class

GetElementsByName par les noms des balises

`getElementsByName` par le NAME par exemple dans la balise INPUT.

`outerHTML` récupère autour de l'élément HTML

`innerHTML` récupère le contenu de l'élément cible HTML

`innerText` récupère le contenu du texte visible

`textContent` représente le contenu du noeud et de ses enfants