JavaScript

CM3 : DOM

Mickaël Martin Nevot

V4.2.0

JavaScript

DOM

- *Document object model* (modèle objet de document) :
  - **Arborescence** des éléments d'une page « balisée » (HTML)
  - **API** qui permet aux scripts d'accéder ou de mettre à jour le contenu, la structure ou le style d'une page « balisée »
  - **Arbre** (DOM) : ← Cette organisation forme un arbre
    - **Nœud** (*node*) : élément constituant de l'arbre (balise, attribut, etc.)
    - Parent, enfant (fils) et frère : identique à une structure familiale
    - Racine : unique nœud n'ayant pas de parent
    - Feuille : nœud n'ayant pas de fils (attribut par exemple)

Objets du DOM

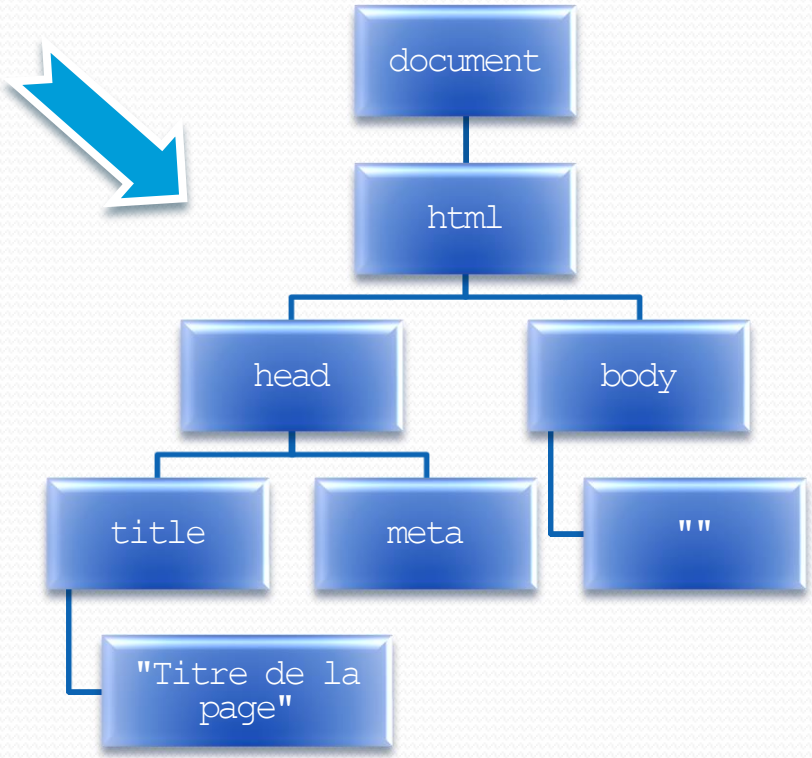
- Objet `window` :
  - Fenêtre du **navigateur**
  - Variable globale = attribut de `window`
  - En général, non spécifié ← `alert() = window.alert()`
- Objet `document` :
  - **Page Web** (fils de `window`)
  - Racine de l'arbre DOM

Arbre DOM

```
<!DOCTYPE html>
<html lang="fr">
    <head>
        <title>Titre de la page</title>
        <meta charset="utf-8"/>
    </head>
    <body>

    <!-- Ici votre site Web -->

    </body>
</html>
```



Propriétés du DOM

Propriété	Action
Informations	
`id`	Identifiant d'un nœud
`className`	Classe d'un nœud
`nodeName`	Nom du nœud
`nodeValue` (ou `data`)	Valeur/contenu du nœud
`nodeType`	Type du nœud (voir ci-après)
`offsetX`	`X` = Dimension (`Height`, `Width`)/position (`Left`, `Top`)
Accéder à un nœud	
`getElementById(id)`	Récupération d'un nœud par son identifiant
`getElementsByTagName(tag)`	Récupération des nœuds d'un type de balise
`getElementsByName(name)`	Récupération des nœuds d'un même nom
`getAttribute(attribute)`	Récupération de la valeur d'un attribut

Propriétés du DOM

Propriété	Action
Se déplacer dans l'arbre	
`hasChildNodes`	Existence de nœud enfant
`length`	Nombre de nœuds dans une liste
`childNodes`	Liste de nœuds enfants
`firstChild`	Premier nœud enfant
`lastChild`	Dernier nœud enfant
`nextSibling`	Prochain nœud d'un type (nœud de même niveau)
`parentNode`	Nœud parent
`previousSibling`	Nœud précédent d'un type (nœud de même niveau)

Propriétés du DOM

Méthode	Action
Gestion des nœuds	
`createElement(elem)`	Création d'un nouveau nœud (`<div></div>`, etc.)
`createTextNode(text)`	Création d'un nœud texte
`appendChild(node)`	Insertion d'un nœud après le dernier enfant
`insertBefore(n1, n2)`	Insertion d'un nœud avant un autre nœud
`replaceChild(n1, n2)`	Remplacement d'un nœud par un autre
`removeChild(node)`	Suppression d'un nœud
`cloneChild(opt)`	Clonage d'un nœud
`setAttribute(att, val)`	Création ou modification d'un attribut
`innerHTML`	Lecture ou écriture du contenu d'un nœud
`style`	Modification du style (CSS) d'un nœud

Un style DOM ≈ style CSS (sauf – : par exemple, `text-align` → `textAlign`)

Types de nœud

Numéro	Type de nœud
1	**Nœud élément**
2	**Nœud attribut**
3	**Nœud texte**
4	Nœud pour passage CDATA (relatif au XML)
5	Nœud pour référence d'entité
6	Nœud pour entité
7	Nœud pour instruction de traitement
8	**Nœud pour commentaire**
9	Nœud document
10	Nœud type de document
11	Nœud de fragment de document
12	Nœud pour notation

Évènements du DOM

Évènement	Action détectée
Souris	
`click`	Clic de la souris
`dblclick`	Double-clic de la souris
`mouseover`	Survol de la souris
`mouseout`	Fin de survol de la souris
`mousedown`	Pression (sans relâchement) du bouton gauche de la souris
`mouseup`	Relâchement du bouton gauche de la souris
`mousemove`	Déplacement de la souris
Clavier	
`keydown`	Pression (sans relâchement) d'une touche du clavier
`keyup`	Relâchement d'une touche du clavier
`keypress`	Pression et relâchement d'une touche du clavier

Évènements du DOM

Évènement	Action détectée
Page/ fenêtre	
`error`	Erreur durant le chargement
`load`	Fin du chargement de la page
`unload`	Déchargement de la page (changement de page, etc.)
`resize`	Redimensionnement de la fenêtre
Formulaire	
`focus`	Prise de focus (« ciblage »)
`blur`	Perte de focus
`change`	Changement d'un élément de saisie d'un formulaire
`select`	Sélection d'un champ textuel d'un formulaire
`submit`	Action du bouton de soumission d'un formulaire
`reset`	Action du bouton de réinitialisation d'un formulaire

Objet Event

Propriété de Event	Description
`altkey`	Renvoie vrai si la touche `Alt` a été actionnée
`ctrlKey`	Renvoie vrai si la touche `Ctrl` a été actionnée
`shiftKey`	Renvoie vrai si la touche `Shift` a été actionnée
`keyCode`	Renvoie le code de touche actionnée
`clientX`	Abscisse de la souris
`clientY`	Ordonnée de la souris
`screenX`	Abscisses de l'évènement
`screenY`	Ordonnée de l'évènement

Lorsqu'un évènement se produit, un objet `Event` est créé et il est accessible depuis la fonction de traitement de l'évènement

Gestionnaires d'évènements

- Utilisation recommandée des gestionnaires d'évènements :

```
// Par exemple : <span id="id">Cliquez-moi !</span>.
let element = document.getElementById("id");

function addEvent(element, event, func) {
    if (element.addEventListener) {
        // Si notre élément possède la méthode addEventListener().
        // Mozilla, Chrome, Safari, Opera,...
        element.addEventListener(event, func, false);
    } else {
        // Si notre élément ne possède pas la méthode addEventListener().
        // Internet Explorer.
        element.attachEvent("on" + event, func);
    }
}
function myFunction(e) {
    alert("Clic : (" + e.clientX + ", " + e.clientY + ")");
}
addEvent(element, "click", myFunction);
```

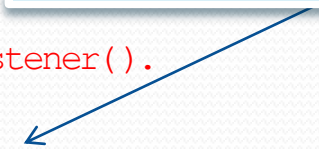
Gestionnaires d'évènements

- Utilisation recommandée des gestionnaires d'évènements :

```
// Par exemple : <span id="id">Cliquez-moi !</span>.
let element = document.getElementById("id");

function addEvent(element, event, func) {
    if (element.addEventListener) {
        // Si notre élément possède la méthode addEventListener().
        // Mozilla, Chrome, Safari, Opera,...
        element.addEventListener(event, func, false);
    } else {
        // Si notre élément ne possède pas la méthode addEventListener().
        // Internet Explorer.
        element.attachEvent("on" + event, func);
    }
}
function myFunction(e) {
    alert("Clic : (" + e.clientX + ", " + e.clientY + ")");
}
addEvent(element, "click", myFunction);
```

Capture (sens de propagation)



Certains événements appliqués à un élément parent peuvent se propager d'eux-mêmes aux éléments enfants, c'est le cas des événements `mouseover`, `mouseout`, `mousemove`, `click`

Crédits

Auteur

Mickaël Martin Nevot
mmartin.nevot@gmail.com

Carte de visite électronique

Relecteurs

Cours en ligne sur : www.mickael-martin-nevot.com