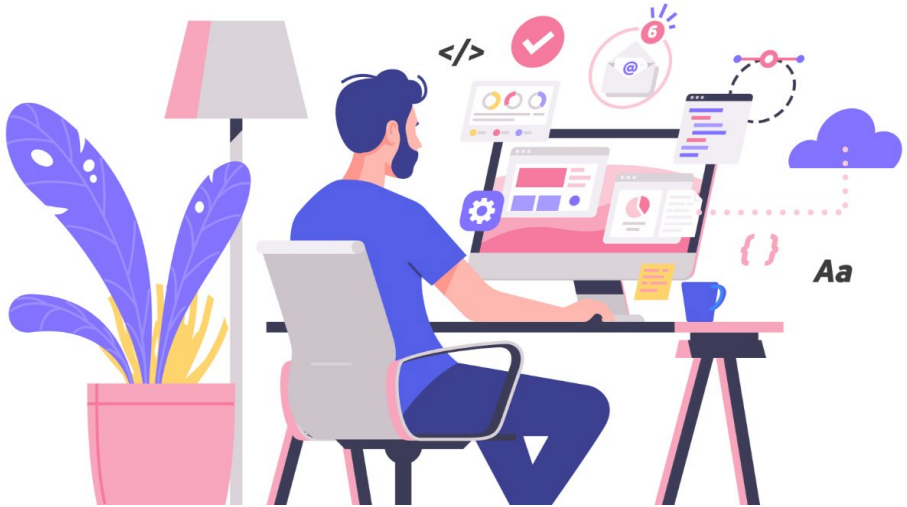
Javascript



Objectifs

- Comprendre le **lien** entre les **attributs HTML** et les **propriétés des objets du DOM**

Attributs HTML vs Propriétés DOM

Pour rappel, lorsqu'un navigateur charge une page HTML, il en génère une copie sous la forme d'un objet Javascript possédant de nombreux noeuds et objets : c'est le **DOM**.

Exemple : si la page HTML contient le code suivant :

```
<input type="text" id="username">
```

Le **DOM** va alors posséder un noeud qui correspondra à un objet **HTMLInputElement**

Cela signifie que le navigateur **convertit automatiquement les attributs des éléments HTML en propriétés des objets du DOM**

Attributs HTML vs Propriétés DOM

```
<input type="text" id="username">
```

L'élément HTML **input** possède 2 attributs :

- L'attribut **type** dont la valeur est "**text**"
- L'attribut **id** dont la valeur est "**username**"

L'objet **HTMLInputElement** généré possédera quant-à-lui des **propriétés** correspondantes à ces 2 attributs :

- **input.type** dont la valeur est "**text**"
- **input.id** dont la valeur est "**username**"

Créer ou modifier un attribut : setAttribute()

element.setAttribute(name, value) :

- Si l'élément **possède déjà** l'attribut : permet de **changer** la valeur de l'attribut
- Si l'élément **ne possède pas** l'attribut : permet **d'ajouter** l'attribut et sa valeur à l'élément

```
<img src="" id="first-image">
```

```
let image = document.querySelector('#first-image');
// Modifier l'attribut src avec une nouvelle valeur
image.setAttribute('src', 'https://....');
// Créer l'attribut alt avec une valeur
image.setAttribute('alt', 'chaton mignon');
```

JS

Supprimer un attribut : removeAttribute()

element.removeAttribute(name)

```
<a href="https://www.youtube.com"
    target="_blank"
    id="js">JavaScript Tutorial
</a>
```

```
let link = document.querySelector('#js');
link.removeAttribute("target");
```

```
<a href="https://www.youtube.com" id="js">JavaScript Tutorial </a>
```

Récupérer la valeur d'un attribut : getAttribute()

element.getAttribute(name) :

- Retourne un **string** si l'attribut existe, sinon **null**

```
<a href="https://www.youtube.com"
    target="_blank"
    id="js">JavaScript Tutorial
</a>
```

```
let link = document.querySelector('#js');
let target = link.getAttribute('target');
console.log(target);
```

```
_blank
```

Attributs HTML vs Propriétés DOM : synchronisation

Les attributs et les propriétés sont **synchronisés**

Cela signifie que si un **attribut** change, sa **propriété** correspondante sera **automatiquement mise à jour côté JS.**

Agir sur les propriétés est plus simple que sur les attributs. On préférera donc toujours le sens **propriété → attribut**

⚠️ Il y a des exceptions : les attributs des éléments d'un **formulaire** (form, input, textarea…) ne sont pas synchronisés avec les changements de leurs propriétés. On passera, dans ce cas précis, dans le sens **attribut → propriété**

Attributs HTML vs Propriétés DOM : synchronisation

Exemple 1 ✅

```html
<input type="text" id="username" tabindex="1">
```

```js
let input = document.querySelector('#username');

// attribute -> property
input.setAttribute('tabindex', 2);
console.log(input.tabIndex);  // 2


// property -> attribute
input.tabIndex = 3;
console.log(input.getAttribute('tabIndex')); // 3
```

Pas de soucis avec l'attribut/propriété **tabindex**

Attributs HTML vs Propriétés DOM : synchronisation

Exemple 2 ❌

```html
<input type="text" id="username" tabindex="1">
```

```js
// attribute -> property: OK
input.setAttribute('value','guest');
console.log(input.value);  // guest


// property -> attribute: ne change pas
input.value = 'admin';
console.log(input.getAttribute('value')); // guest
```

L'attribut/propriété **value** ne changera pas

Pratiquons !