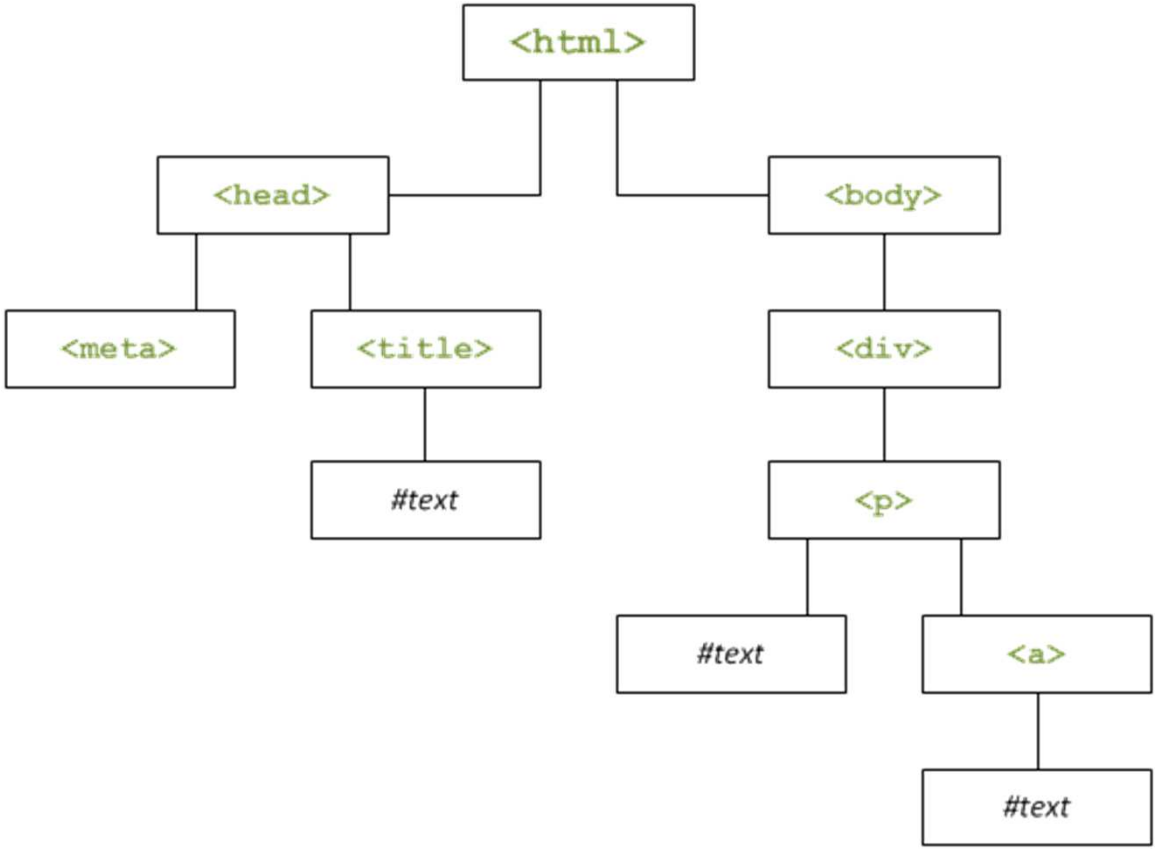
Javascript Avancé partie 2

Le Document Object Model

- Document Object Model (abrégé DOM) est une interface de programmation pour les documents XML et HTML
- Permet d'accéder au code XML et/ou HTML
- Permet de modifier, ajouter, déplacer ou supprimer des éléments HTML
- DOM-1 : unification par W3C, spécifie ce qu'est le DOM et comment il est représenté
- DOM-2 : introduction de la méthode getElementById()

Structure du DOM

- La page Web est vue comme un arbre




```
<!DOCTYPE html>
<html>
  <head>
      <meta charset="utf-8" />
      <title>Le titre de la
page</title>
  </head>
  <body>
      <div>
          <p>Un peu de texte
<a>et un lien</a></p>
      </div>
  </body>
</html>
```

Accéder aux éléments

➢ La sélection d'élément du DOM HTML peut se faire via :

❑ L'ID de l'élément : renvoie un objet

```
var myElement = document.getElementById("intro");
```

❑ La Classe des éléments : renvoie un tableau d'objets

```
var x = document.getElementsByClassName("intro");
```

❑ Le nom de l'élément : renvoie un tableau d'objets

```
var x = document.getElementsByTagName("p");
```

❑ Un sélecteur CSS : renvoie un tableau d'objets

```
var x = document.querySelectorAll("p.intro");
```

Accéder aux attributs d'un élément

```
<a id="lien" href="http://www.google.com">Un lien modifié dynamiquement</a>
```

- L'objet Element possède deux méthodes :

- getAttribute() permettant de récupérer un attribut

```
var link = document.getElementById('lien');
var href = link.getAttribute('href');
console.log(href);
```

- setAttribute() permettant d'éditer un attribut

```
link.setAttribute('href', 'http://www.yahoo.com');
```

- Avec la propriété

```
href = link.href; // récupération via la propriété
link.href = 'http://www.yahoo.com'; // Affectation via la propriété
```

Accéder à l'attribut class

```
<style>
    .blue {
        background: blue;
        color: white;
    }
</style>

<div id="divTexte">
  <p>Un peu de texte</p>
</div>
```

- Pour modifier l'attribut class d'un élément on utilise la propriété classeName

```
document.getElementById('divTexte').className = 'blue';
```

- Pour modifier le contenu de l'attribut class d'un élément on utilise la propriété classeList

```
document.getElementById('divTexte').classList.add('class')
document.getElementById('divTexte').classList.remove('class')
document.getElementById('divTexte').classList.toggle('class')
```

Accéder au code HTML

- La propriété innerHTML permet de récupérer ou modifier le code enfant d'un élément.

```
var div = document.getElementById('divTexte');
console.log(div.innerHTML);

div.innerHTML = '<em>Du texte en italique</em>';

div.innerHTML += '<strong>on ajoute du texte en gras</strong>';
```

- La propriété value permet de récupérer ou modifier la valeur d'un champ de formulaire.

```
<input type="text" id="lastname" name="lastname" />
```

```
document.getElementById("lastName").value
```

Manipuler le DOM

- Pour créer un nouvel élément HTML :

```
var para = document.createElement("p");
```

- Pour insérer un nouvel élément dans le DOM :

- Insertion en tant que dernier enfant de element :

```
element.appendChild(para);
```

- Insertion de para avant child dans element :

```
element.insertBefore(para,child);
```

- Insertion de texte dans un paragraphe :

```
var para = document.createElement("p");
var node = document.createTextNode("This is new.");
para.appendChild(node);
```

Exercice

Horloge :

1) Créez un paragraphe dans votre fichier HTML

2) Récupérez l'heure et l'afficher au format "H:m:s" dans ce paragraphe

3) Mettre l'affichage des secondes, minutes et heures à jour en utilisant la méthode setInterval ou setTimer

Les évènements

Les évènements permet d'interagir avec les utilisateurs et de déclencher des actions.

Par exemple on peut :
- détecter les mouvements de souris ou les click
- détecter les actions sur les touches du clavier
- détecter la validation d'un formulaire (vérification de saisie)

Listes des évènements

Voici la liste des événements principaux, ainsi que les actions à effectuer pour qu'ils se déclenchent :

Nom de l'événement	Action pour le déclencher
click	Cliquer (appuyer puis relâcher) sur l'élément
dblclick	Double-cliquer sur l'élément
mouseover	Faire entrer le curseur sur l'élément
mouseout	Faire sortir le curseur de l'élément
mousedown	Appuyer (sans relâcher) sur le bouton gauche de la souris sur l'élément
mouseup	Relâcher le bouton gauche de la souris sur l'élément
mousemove	Faire déplacer le curseur sur l'élément

Listes des évènements

Nom de l'événement	Action pour le déclencher
keydown	Appuyer (sans relâcher) sur une touche de clavier sur l'élément
keyup	Relâcher une touche de clavier sur l'élément
keypress	Frapper (appuyer puis relâcher) une touche de clavier sur l'élément
focus	« Cibler » l'élément
blur	Annuler le « ciblage » de l'élément
change	Changer la valeur d'un élément spécifique aux formulaires (input,checkbox, etc.)
select	Sélectionner le contenu d'un champ de texte (input, textarea, etc.)

Il existe aussi deux événements spécifiques à l'élément <form>, que voici :

Nom de l'événement	Action pour le déclencher
submit	Envoyer le formulaire
reset	Réinitialiser le formulaire

Utiliser les évènements

Avec le DOM-0 :

```
<span id="clickme">Cliquez-moi !</span>

<script>

    var element = document.getElementById('clickme');

    element.onclick = function() {
        alert("Vous m'avez cliqué !");
    };

</script>
```

Avec le DOM-2 :

```
<span id="clickme">Cliquez-moi !</span>

<script>
    var element = document.getElementById('clickme');

    element.addEventListener('click', function() {
        alert("Vous m'avez cliqué !");
    });
</script>
```

L’objet Event

Cet objet sert à fournir une multitude d'informations sur l'événement actuellement déclenché. Par exemple, on peut récupérer quelles sont les touches actuellement enfoncées, les coordonnées du curseur, l'élément qui a déclenché l'événement…

```
// L'argument « e » va récupérer une référence vers l'objet « Event »
element.onclick = function(e) {
    // Ceci affiche le type de l'événement (click, mouseover, etc.)
    alert(e.type);
};
```

```
// L'argument « e » va récupérer une référence vers l'objet « Event »
element.addEventListener('click', function(e) {
    // Ceci affiche le type de l'événement (click, mouseover, etc.)
    alert(e.type);
});
```

L’objet Event

Attributs et méthodes :

- event.type : le nom de l’évènement
- event.target : l'élément déclencheur de l'événement
- event.clientX : position horizontal de la sourie
- event.clientY : positon vertical de la sourie
- event.keyCode : code de la touche (différent si keyup / keydown ou keypress)
- event.preventDefault() : empêche lapropagation de l’évènement (Utile avec les formulaires)

Exercices

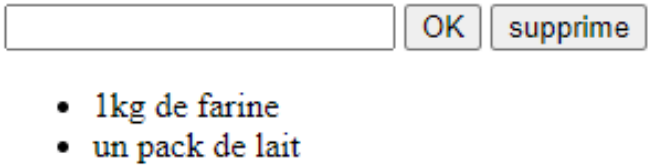
Navigation

1) Créez une navbar avec une liste à puce (ul, li, a) contenant 5 liens
2) Changer le style (background-color, color, etc.) du seul élément cliqué
3) Si un autre lien à été cliqué avant, réinitialiser son style d'origine

Exercices

Liste de course

1) Créez le HTML pour avoir le visuel suivant :



2) Au clic sur le bouton OK, ajouter le texte saisie dans la liste.

3) Au clic sur le bouton Supprime, supprimer le dernier élément de la liste.

Exercices

Formulaire

1) Créez un formulaire avec les champs :
- Prénom
- Nom
- Email
- Téléphone
- Password

2) Ajouter un bouton de Validation

Exercices

Formulaire

3) Au clic sur le bouton de validation vérifier :

- Prénom : Requis
- Nom : Requis
- Email : Requis, doit respecter un format de mail
- Téléphone : Requis, uniquement valeur numérique et de longueur maximale de 10
- Password : Requis, minimum 8 caractères + 1 caractère spécial (!, ?, #) (utiliser une regex)

4) Afficher les messages d'erreur dans une div