



JavaScript

Les événements de l'utilisateur

Définition

- Un événement est une **action** faite par l'utilisateur et suivie en général par une **réaction** exécutant un code.
- Exemple:
 - Le clic sur un bouton;
 - Le survol d'un paragraphe;
 - La saisie d'un texte dans un formulaire.

Définition

- Un gestionnaire d'évènements est divisé en deux parties :
 - une partie qui sert à écouter le déclenchement de l'évènement;
 - une partie gestionnaire en soi qui est le code à exécuter dès que l'évènement se produit.

La méthode `addEventListener()`

- La méthode **`addEventListener()`** est utilisée pour attacher un gestionnaire d'événements à un élément HTML.
- Nous pouvons ajouter plusieurs gestionnaires d'événements à un élément sans écraser ceux qui sont existants.
- La méthode **`addEventListener`** a trois arguments:
 - le nom de l'évènement qu'on souhaite prendre en charge
 - la fonction à exécuter en cas de déclenchement de cet évènement.
 - La valeur booléenne qui précise si l'évènement est exécuté en phase de bouillonnement **`False`** ou de capture **`true`**.(argument facultatif et la valeur par défaut est **`False`**)

La méthode `addEventListener()`

- **Événements de souris :**
 - **click** : lorsque la souris clique sur un élément.
 - **contextmenu** : lorsque la souris fait un clic droit sur un élément.
 - **mouseover/ mouseout** : lorsque le pointeur de la souris survole / quitte un élément.
 - **mousedown/ mouseup** : lorsque le bouton de la souris est appuyé/relâché sur un élément.
 - **mousemove** : lorsque la souris est déplacée l'intérieur de l'élément.

La méthode addEventListener()

- **Exemple 1 : appliquer un événement sur un élément**

```
<body>
<p> Cliquez sur le bouton suivant pour voir l'effet.</p>
<button id = "btn"> Cliquez sur moi </button>
<p id = "para"></p>

<script>
document.getElementById("btn").addEventListener("click", effet);
function effet() {
document.getElementById("para").innerHTML = "Vous avez cliqué sur le bouton du haut";
}
</script>
</body>
```

Cliquez sur le bouton suivant pour voir l'effet.

Cliquez sur moi

Vous avez cliqué sur le bouton du haut

La méthode addEventListener()

- **Exemple 2 : appliquer plusieurs événements de type différent sur le même élément**

```
<body>
  <p> Cliquez sur le bouton suivant pour voir l'effet.</p>
  <button id = "btn"> Cliquez sur moi </button>
  <p id = "para"></p>
  <p id = "para1"></p>
  <script>
    function effet1() {
      document.getElementById("para").innerHTML = "Vous avez cliqué sur le bouton du haut";
    }
    function effet2() {
      document.getElementById("para1").innerHTML = "Vous avez quitté le bouton du haut";
    }
    document.getElementById("btn").addEventListener("click", effet1);
    document.getElementById("btn").addEventListener("mouseout", effet2);
  </script>
</body>
```

Cliquez sur le bouton suivant pour voir l'effet.

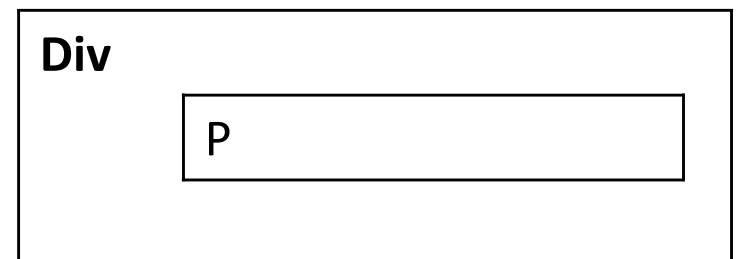
Cliquez sur moi

Vous avez cliqué sur le bouton du haut

Vous avez quitté le bouton du haut

La méthode `addEventListener()`




- **Exemple 3** : capture ou bouillonnement
- Supposons que nous avons :
 - un élément `div` et un élément `paragraphe` à l'intérieur
 - Appliqué l'événement "**click**" aux éléments précédents en utilisant la méthode **`addEventListener()`**.
- Maintenant, si on clique sur l'élément `paragraphe`, quel événement est déclenché en premier ?



La méthode `addEventListener()`

- **Exemple 3** : capture ou bouillonnement
- Dans **Bouillonnement**, l'événement de `<P>` est déclenché en premier, puis l'événement de `<div>`.
 - ↳ L'événement de l'élément interne est déclenché en premier, puis l'événement de l'élément externe.
- Dans **Capturer**, l'événement de `<div>` est déclenché en premier, puis l'événement de `<P>`.
 - ↳ L'événement de l'élément externe est déclenché en premier, puis l'événement de l'élément interne.

La méthode addEventListener()

```
<div id="d1" style=" border: 3px solid  green; height:300px; ">
  <div id="d2" style=" border: 3px solid  blue; height:200px; ">
    <div id="d3" style=" border: 3px solid  red ; height:100px;">
      <h1>3 éléments div imbriqués</h1>
    </div>
  </div>
</div>
```

```
<script>
  let d1= document.getElementById("d1");
  let d2= document.getElementById("d2");
  let d3= document.getElementById("d3");
  //Le clic sur d1 déclenche son événement.
  d1.addEventListener("click", ()=>alert("Vous avez cliqué sur l'élément div 1"), false);
  //Le clic sur d2 déclenche son événement puis celui de d1 ---> bouillonnement
  d2.addEventListener("click", ()=>alert("Vous avez cliqué sur l'élément div 2"), true);
  //Le clic sur d3 déclenche l'événement de d2 puis celui de d3 ---> capture
  // ensuite celui de d1 ---> bouillonnement
  d3.addEventListener("click", ()=>alert("Vous avez cliqué sur l'élément div 3"), false);
</script>
```

la méthode `stopPropagation()`

- la méthode **`stopPropagation()`** stoppe la propagation d'un évènement après qu'un gestionnaire d'évènement a été atteint.
- Cela signifie que si la phase de bouillonnement est choisie, le gestionnaire de l'élément cible sera exécuté et les gestionnaires de ce même évènement attachés aux éléments parents seront ignorés.
- Dans le cas où c'est la phase de capture qui est choisie, le gestionnaire parent le plus lointain de l'élément cible sera exécuté et les autres seront ignorés.
-

la méthode stopPropagation()

- Exemple 4:

```
<div id="d1">
  Ceci est un élément div.
  <span id="s1"> Ceci est un élément span. </span>
</div>
<script>
  document.getElementById("d1").addEventListener("click", ()=>alert("Vous avez cliqué sur div"));
  document.getElementById("s1").addEventListener("click", stopParents);
  function stopParents(e) {
    alert("Vous avez cliqué sur l'élément span puis stoppé la propagation");
    e.stopPropagation()
  }
</script>
```

la méthode `stopImmediatePropagation()`

- la méthode **`stopImmediatePropagation()`** permet d'exécuter le premier gestionnaire d'événement dans le cas où un élément est associé à plusieurs gestionnaires du même événement.

la méthode stopImmediatePropagation()

```
<div id="d1">Ceci est un élément div.  
  <span id="s1"> Ceci est un élément span. </span>  
</div>  
<script>  
  let d1= document.getElementById("d1");  
  let s1= document.getElementById("s1");  
  d1.addEventListener("click", () => alert("Vous avez cliqué sur l'élément div"));  
  s1.addEventListener("click", stopParents);  
  s1.addEventListener("click", stopImmediate);  
  s1.addEventListener("click", () => alert("Vous avez cliqué sur span 3 fois"));  
  function stopParents(e) {  
    alert("Vous avez cliqué sur l'élément span puis stoppé la propagation");  
    e.stopPropagation()  
  }  
  function stopImmediate(e) {  
    alert("Vous avez cliqué sur l'élément span puis stoppé les autres événements click");  
    e.stopImmediatePropagation()  
  }  
</script>
```

La méthode `removeEventListener()`

- La méthode **`removeEventListener()`** supprime un gestionnaire d'évènement déclaré avec `addEventListener`.
- La méthode **`removeEventListener()`** est utile lorsqu'on veut retirer un gestionnaire d'évènement selon certains cas comme par exemple empêcher le déclenchement de l'évènement deux fois.

La méthode removeEventListener()

```
<p> Cliquez sur le bouton suivant pour voir l'effet.</p>
<button id="btn"> Cliquez sur moi </button>
<p id="p2"> vous avez cliqué 0 fois </p>
<script>
  let btn = document.querySelector('button');
  btn.addEventListener('click', affiche);
  btn.addEventListener('click', compteur);
  let compte = 0;
  function compteur() {
    ++compte;
    if (compte == 3) //On supprime le compteur au 3ème clic
      btn.removeEventListener('click', compteur);
    else
      document.getElementById('p2').textContent = "vous avez cliqué " + compte + " fois";
  }
  function affiche() {
    alert('Bouton cliqué');
  }
</script>
```


La méthode `addEventListener()`

- **Événements de clavier :**
 - **keydown/ keyup** : lorsqu'une touche du clavier est enfoncée / relâchée.
 - **keypress** : lorsqu'une touche du clavier est dans la position "pressée".
 - Si vous utilisez la combinaison de touches Maj+A, l'événement **keypress** détectera un A majuscule, par contre les événements **keyup** et **keydown** se seraient déclenchés deux fois (une fois pour la touche Maj et une seconde fois pour la touche A).

La méthode `addEventListener()`

- **Événements liées au ciblage :**
 - **focus** : cet événement se déclenche lorsqu'on met le curseur dans un élément du formulaire ou la sélection d'un lien hypertexte.
 - **Blur** : cet événement se déclenche lorsqu'on quitte le cible.

L'objet Event

- L'objet **event** fournit des détails supplémentaires sur l'événement qui vient de se produire à l'aide de ses propriétés et ses méthodes.
- Les propriétés associée à l'objet event :
 - **type** qui retourne le type de l'événement déclenché (click, mouseover, mouseout...).
 - **clientX**: retourne la position (coordonnée) horizontale du pointeur de la souris par rapport à la fenêtre du navigateur.
 - **clientY**: retourne la position verticale du pointeur de la souris par rapport à la fenêtre du navigateur.
 - **button**: indique sur quel bouton de la souris on a appuyé. 0 pour le bouton gauche, 1 pour le bouton central et 2 pour la bouton droit.

L'objet Event

- Exemple 7

```
<body>
  <div></div>
  <script>
    let div = document.querySelector("div")
    div.style.height="200px";
    div.style.width="200px";
    div.style.border="solid 4px black";

    div.addEventListener("click", maFunction);
    div.addEventListener("dblclick", maFunction);
    div.addEventListener("contextmenu", maFunction);
    function maFunction(e) {
      div.innerHTML= " Tu as fais un : "+ e.type + "<br>"
        +"à la position : <br>"
        +"x = "+ e.clientX + "<br>"
        +"y = "+ e.clientY + "<br>"
        +"le bouton utilisé est : "+e.button
    }
  </script>
</body>
```

Tu as fais un : click
à la position :
x = 154
y = 154
le bouton utilisé est :0

L'objet Event

- **altKey** : indique si la touche ALT du clavier est activé. Elle retourne une valeur booléenne.
- **shiftKey** : indique si la touche SHIFT est activé. Elle retourne une valeur booléenne.
- **ctrlKey** : indique si la touche CTRL est activé. Elle retourne une valeur booléenne.
- **metaKey** : indique si la touche WINDOW est activé. Elle retourne une valeur booléenne.
- **keyCode** : retourne le code ASCII de la touche activée.
- **key** : retourne le caractère de la touche activée.

L'objet Event

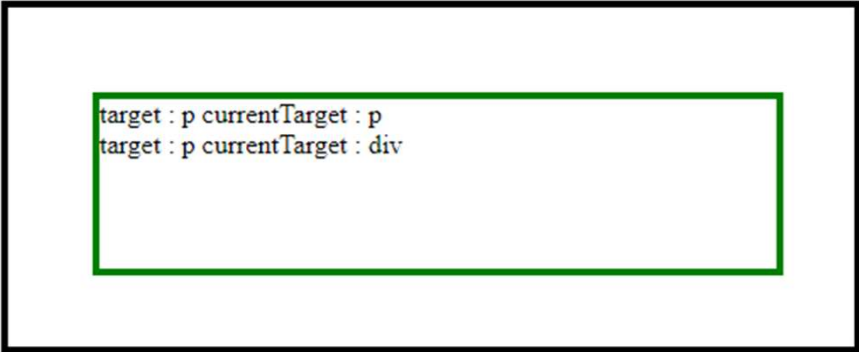
- Exemple 8

```
document.body.addEventListener('keydown' , e => {  
  div.innerHTML =  
    '-- alt: ' + event.altKey +  
    ' <br>-- ctrl: ' + e.ctrlKey +  
    ' <br>-- meta: ' + e.metaKey +  
    ' <br>-- shift: ' + e.shiftKey +  
    ' <br>-- code: ' + e.key +  
    ' <br>-- key: ' + e.keyCode;  
})
```

```
-- alt: false  
-- ctrl: false  
-- meta: false  
-- shift: true  
-- code: J  
-- key: 74
```

L'objet Event

- **target**: retourne l'élément sur lequel l'événement a eu lieu.
- **currentTarget**: retourne l'élément sur lequel l'événement est en cours de traitement.
- Exemple 9



target : p currentTarget : p
target : p currentTarget : div

```
<body>
  <div id="div">
    <p id="p"></p>
  </div>
  <script>
    let div = document.querySelector("div")
    div.style.height="200px";
    div.style.width="500px";
    div.style.border="solid 4px black";

    let p = document.querySelector("p")
    p.style.margin="50px";
    p.style.height="100px";
    p.style.width="400px";
    p.style.border="solid 4px green";

    div.addEventListener("click",maFonction);
    p.addEventListener("click",maFonction);
    function maFonction(e){
      p.innerHTML += "target : " + e.target.id
      + " currentTarget : " + e.currentTarget.id + "<br>"
    }
  </script>
</body>
```

L'objet Event

- **Méthodes associées à event**
 - **stopPropagation()**
 - **stopImmediatePropagation()**
 - **preventDefault()**: Permet d'annuler l'action prévue d'un événement.
 - exemple 10

```
<body>
  <a href="https://www.google.com/">Cliquez ici pour ouvrir Google </a>
  <script>
    document.querySelector("a").addEventListener("click", maFonction)
    function maFonction(e){
      e.preventDefault() //Empêcher l'ouverture de la page web google
    }
  </script>
</body>
```


Formulaire

- La propriété **value** : Récupère ou change le contenu des champs d'un formulaire.
- Les propriétés booléennes :
 - **required** : si le champ doit recevoir une valeur ou non
 - **Disabled** : si le champ est désactivé ou non
 - **Checked** : si un champ radio ou checkbox est coché ou non
 - **readOnly** : si le champ possède la propriété lecture seul ou non

Formulaire

- Exemple :11

```
<form>
  Parlez-vous ? <br><br>
  <input type="checkbox" value="Coréen" /> <label>Coréen</label><br>
  <input type="checkbox" value="Français" /> <label>Français</label><br>
  <input type="checkbox" value="Anglais" /> <label>Anglais</label><br>
  <input type="checkbox" value="Arabe" /> <label>Arabe</label> <br><br>
  <input type="button" name="button" value="Je fais le choix" /> <br><br>
  <label>Votre choix: </label><input type="text" name="text" value="Rien!" /> <br>
  <input type="submit" value="Envoyer">
</form>
<script>
  let box = document.querySelectorAll('input[type=checkbox]');
  let btn = document.getElementsByName("button");
  let txt = document.querySelector('input[type=text]');
  btn[0].addEventListener("click", action)
  function action() {
    box[0].disabled = true; // désactiver la case n°1
    box[1].checked = true; // cocher la case n°2
    box[3].required = true; // cocher cette case est obligatoire
    txt.value = box[1].value // mettre Français dans input text
    txt.readOnly = true; //lire seulement le contenu
  }
</script>
```

Formulaire

- La propriété **selectedIndex** : indique le rang de l'élément sélectionné dans la balise <select>.
- La propriété **selected** : indique si une option de la balise <select> est sélectionnée ou non
- Exemple : 12

```
<select id="list">
  <option>Sélectionnez votre sexe</option>
  <option value="h" >Homme</option>
  <option selected>Femme</option>
</select> <br>
<p></p>
<script>
let list = document.getElementById('list');
let p = document.querySelector('p');
for(i=0; i<list.length; i++)
p.innerHTML += list[i].selected + "<br>";

list.addEventListener('change', ()=>
p.innerHTML ="votre choix: "+ list[list.selectedIndex].value);
</script>
```

Formulaire

- La méthode **submit()** permet d'envoyer un formulaire sans l'intervention de l'utilisateur ;
- La méthode **reset()** permet de réinitialiser tous les champs d'un formulaire.

Formulaire

- Exemple 13

```
<form>
  <select>
    <option>Sélectionnez votre sexe</option>
    <option>Homme</option>
    <option>Femme</option>
  </select>
  <input type="submit" value="Envoyer">
</form>
<script>
  let list = document.querySelector("select")
  let form = document.querySelector("form")
  form.addEventListener("submit", envoi)
  function envoi(e) {
    if (!list[0].selected)
      form.submit()
    else {
      list.style.border = "4px solid red";
      e.preventDefault();
    }
  }
</script>
```

Formulaire

- **Exercice 1**
- Réaliser le formulaire ci-dessous sachant que :
 - La couleur de l'arrière plan du premier champ devient jaune quand il reçoit le curseur
 - Pendant la saisie de la confirmation, le deuxième champ devra être encadré en rouge tant que le mot de passe n'est pas identique, sinon en vert
 - Avant d'envoyer les données vérifiez la conformité de deux mots de passe



Saisir le Mot de passe :

Confirmer le Mot de passe :

Formulaire

Exercice 2

Soit le formulaire suivant:

```
<form>
  Je m'appelle : <input type="text"> <br><br>
  J'habite à :
  <select>
    <option value="Pas de sélection">Pas de sélection</option>
    <option value="Tiznit">Tiznit</option>
    <option value="Agadir">Agadir</option>
    <option value="Sidi ifni">Sidi ifni</option>
  </select><br><br>
  Tu as choisir :<label></label> <br><br>
  <input type="submit" value="Envoyer">
</form>
```

En utilisant JavaScript, Créez les événements suivants :

- Lorsque le champ input perd le curseur, il devient encadré en rouge si l'utilisateur n'a rien saisi ou en vert dans le cas contraire.
- A chaque changement sur <select>, La balise <label> affiche l'option sélectionnée.
- Annulez l'envoi du formulaire si le champ input est vide ou la première option de <select> est sélectionnée.