

Activités JavaScript

Les évènements input et change

Révision

Question 1

```
function isDigit(ch) {  
  let i = 0, valid = true;  
  while (i < ch.length && valid) {  
    valid = '0' <= ch[i] <= "9"; i++;  
  }  
  return valid;  
}  
console.log(isDigit("123")); // true  
console.log(isDigit("ABC")); // true
```

La fonction `isDigit(ch)` retourne toujours `true` indépendamment de la valeur de `ch`.

Comment corriger cette fonction en ne modifiant qu'une seule instruction ?

```
valid = ch[i] >= '0' && ch[i] <= '9';
```

Question 2

```
function somme(t) {  
  const s = 0;  
  for (let i = 0; i < t.length; i++) s += t[i];  
  return s;  
}  
console.log(somme([])); // 0  
console.log(somme([1,2,3])); // invalid assignment to const 's'
```

L'appel de la fonction `somme(t)` produit toujours l'erreur `invalid assignment to const 's'` lorsque le tableau `t` n'est pas vide.

Comment corriger cette erreur en ne modifiant qu'une seule ligne ?

```
let s = 0;
```

Question 3

```
function randint(a, b) {  
  return a + Math.random() * (b - a + 1);  
}  
console.log(randint(1, 6)); // 1.439311099391309  
console.log(randint(0, 9)); // 7.606718899286581
```

La fonction `randint(a, b)` est supposée retourner un entier aléatoire dans l'intervalle `[a, b]`. Or, lors du test elle retourne un réel.

Comment corriger ce problème ?

```
return Math.floor(a + Math.random() * (b - a + 1));
```

Question 4

```
function reverseString(ch) {  
  let i = 0, j = ch.length - 1;  
  while (j > i) {  
    const t = ch[i]; ch[i] = ch[j]; ch[j] = t;  
    i++; j--;  
  }  
  return ch;  
}  
console.log(reverseString("ahmed")); // "ahmed" au lieu de "demha"
```

La fonction `reverseString(ch)` a pour fonction d'inverser l'ordre des caractères d'une chaîne. L'opération est réalisée par l'instruction : `const t = ch[i]; ch[i] = ch[j]; ch[j] = t;`. Seulement, il y a un problème qui l'empêche de bien fonctionner.

Lequel ?

L'écriture `ch[i] = ch[j]` est incorrecte car une chaîne en JavaScript est immuable, il n'est pas possible de modifier une portion de la chaîne.

Utiliser la méthode `ch.substring()` pour corriger cette instruction.

```
ch = ch.substring(0, i) + ch[j] + ch.substring(i+1, j) + ch[i] + ch.substring(j+1);
```

Question 5

Soit le code HTML suivant :

```
<div><input type="text" id="nom"></div>  
<div id="res"></div>
```

et le code JavaScript suivant :

```
const inpNom = document.getElementById("Nom");  
const divRes = document.getElementById("res");  
  
function nomChanged() {  
  divRes.value = inpNom.value;  
}
```

Le code précédent a été reproduit afin d'afficher le contenu du champ du texte `inpNom` dans le div `divRes` lorsqu'on y tape un texte.

Bien que tout semble à première vue correct, la fonction `nomChanged()` n'est pas appelée lorsqu'on saisit quelque chose dans le champ du texte `inpNom`.

Comment résoudre ce problème et répondre à l'évènement `input` ?

```
inpNom.addEventListener("input", nomChanged);
```

Après correction du problème précédent, un nouveau surgit. L'erreur suivante s'affiche dans la console dès qu'on tape une touche du clavier : `inpNom is null`.

Pourquoi `inpNom` est-il `null` ?

Au lieu de taper l'id du champ de texte `nom` l'élève a tapé `Nom` qui ne correspond à aucun élément HTML. L'id est sensible à la casse.

Après correction du problème, le script n'est pas fonctionnel bien que la fonction `nomChanged()` soit appelée lorsqu'on tape un texte dans le champ. L'unique instruction de cette fonction semble contenir une erreur.

Laquelle ?

`divRes` n'a pas de propriété `value` car il s'agit d'un `<div>`. On devra soit utiliser `divRes.textContent` ou bien `divRes.innerHTML`

L'évènement `change`

L'évènement **`change`** se produit lorsque la sélection, l'état coché ou le contenu d'un élément ont changé.

Dans certains cas, cela ne se produit que lorsque l'élément perd le focus.

L'évènement **`change`** peut être utilisé avec les contrôles `<input>`, `<select>` et `<textarea>`.

Exemple

Code HTML

```
<div><input type="text" name="nom" id="nom" placeholder="Votre nom"></div>
<div id="message"></div>
```

Code JavaScript

```
const nom = document.getElementById("nom");
const msg = document.getElementById("message");

let nom_actuel = nom.value;

function nomValueChanged(e) {
  const nouv_nom = nom.value;
  msg.innerHTML += "Nom est changé de '" + nom_actuel + "' à '" + nouv_nom + "'<br>";
  nom_actuel = nouv_nom;
}

nom.addEventListener("change", nomValueChanged);
```

L'évènement input

L'évènement **input** est utile pour détecter le changement d'un élément <textarea> ou <input>.

A l'inverse de l'évènement **change** qui se déclenche uniquement lorsque l'élément perd le focus, l'évènement **input** se produit immédiatement après la modification.

Exemple

Code HTML

```
<div><input type="text" name="nom" id="nom" placeholder="Votre nom"></div>
<p>input Event</p>
<div id="msg-input"></div>
<p>change Event</p>
<div id="msg-change"></div>
```

Code JavaScript

```
const inpNom = document.getElementById("nom");
const divMsgInput = document.getElementById("msg-input");
const divMsgChange = document.getElementById("msg-change");

function inputEvent(e) {
  const nom = inpNom.value;
  divMsgInput.innerHTML += "Nom est contient '" + nom + "'
";
}

function changeEvent(e) {
  const nom = inpNom.value;
  divMsgChange.innerHTML += "Nom est contient '" + nom + "'
";
}

nom.addEventListener("input", inputEvent);
nom.addEventListener("change", changeEvent);
```

TP

Color Mixer

On demande de créer la page suivante :

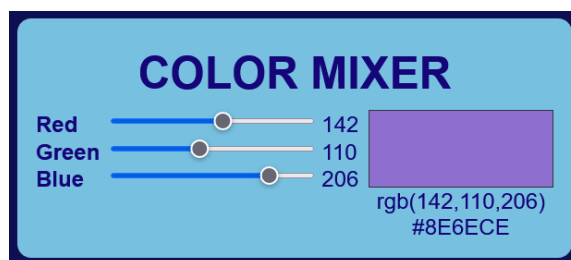


Figure 1, Color Mixer

On donne le code HTML suivant :

```
<div id="color-mixer">
  <h1>Color Mixer</h1>
  <div id="left">
    <div>
      <label for="red">Red</label>
      <input type="range" name="red" id="red" min="0" max="255" value="0"><span id="vred"></span>
    </div>
    <div>
      <label for="red">Green</label>
      <input type="range" name="green" id="green" min="0" max="255" value="0"><span id="vgreen"></span>
    </div>
    <div>
      <label for="blue">Blue</label>
      <input type="range" name="blue" id="blue" min="0" max="255" value="0"><span id="vblue"></span>
    </div>
  </div>
  <div id="right">
    <div id="box"></div>
    <div id="rgb-color"></div><div id="html-color"></div>
  </div>
</div>
```

Ainsi que le code CSS pour la mise en forme :

```
* { box-sizing: border-box; }

body {
  font-size: 18pt; font-family: sans-serif;
  background: linear-gradient(to bottom, #0E0E52 259px, #78C0E0) no-repeat fixed;
  color: #150578; height: 100%;
}

#color-mixer {
  width: 600px;
  padding: 20px; margin: auto;
  display: flex; flex-wrap: wrap; justify-content: space-between;
  background-color: #78C0E0;
  border-radius: 15px;
}

h1 {
  text-align: center; text-transform: uppercase;
  margin: 0.25em 0; width: 100%;
}

#left { width: 360px; }
#right { width: 200px; text-align: center; }

label { display: inline-block; width: 3em; font-weight: bold; }
input { width: 220px; }

#box { width: 200px; height: 84px; border: #333 solid 1px; }
```

On demande de compléter le code JavaScript.

```
const inpRed = document.getElementById("red");
const spanRed = document.getElementById("vred");
const inpGreen = document.getElementById("green");
const spanGreen = document.getElementById("vgreen");
const inpBlue = document.getElementById("blue");
const spanBlue = document.getElementById("vblue");
const divBox = document.getElementById("box");
const divRgbColor = document.getElementById("rgb-color");
const divHtmlColor = document.getElementById("html-color");

function sliderMoved() {
  /* TODO */
}

function toHtmlColor(r, g, b) {
  /* TODO */
}

function toHexa(v) {
  /* TODO */
}

inpRed.addEventListener("input", sliderMoved);
inpGreen.addEventListener("input", sliderMoved);
inpBlue.addEventListener("input", sliderMoved);
sliderMoved();
```

Travail demandé

1. Compléter la fonction `toHexa(v)`. `v` est un entier entre 0 et 255.

Cette fonction convertit un nombre compris entre 0 et 255 de la base 10 à la base 16. Le résultat final doit toujours comporter deux chiffres hexadécimaux.

2. Compléter la fonction `toHtmlColor(r, g, b)`. `r`, `g` et `b` sont des entiers entre 0 et 255.

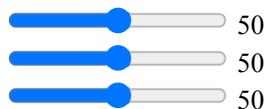
Cette fonction prend les composantes Rouge, Verte et Bleue d'une couleur et la convertit au format hexadécimal `#RRGGBB` chaque nombre est représenté sur deux chiffres hexadécimaux.

3. Compléter la fonction `sliderMoved()`. Cette fonction :

- Récupère les valeurs des trois sliders `inpRed`, `inpGreen` et `inpBlue`.
- Convertit ces valeurs au formats `rgb(r, g, b)` et `#RRGGBB`.
- Affiche les composantes `r`, `g` et `b` dans les spans correspondants `spanGreen`, `spanRed` et `spanBlue`.
- Modifie la couleur d'arrière plan du `divBox`.
- Affiche la couleur dans les `divRgbColor` et `divHtmlColor`.

Volume sliders

On demande de réaliser le slider suivant :



Le deuxième slider se déplace vers la position du premier lorsque cette dernière change.

Le troisième slider se déplace vers la position du premier lorsque ce slider a fini de se dépalcer.

On donne le code HTML de la page :

```
<div><input type="range" id="vol1" min="0" max="100" value="50"><span id="svol1"></span></div>
<div><input type="range" id="vol2" min="0" max="100" value="50"><span id="svol2"></span></div>
<div><input type="range" id="vol3" min="0" max="100" value="50"><span id="svol3"></span></div>
```

On donne aussi le code JavaScript incomplet :

```
const inpVol1 = document.getElementById("vol1");
const spanVol1 = document.getElementById("svol1");
const inpVol2 = document.getElementById("vol2");
const spanVol2 = document.getElementById("svol2");
const inpVol3 = document.getElementById("vol3");
const spanVol3 = document.getElementById("svol3");

function refreshPositions() {
  /* TODO 1 */
  // afficher la valeur de inpVol1 dans spanVol1
  // faire de même pour les autres sliders
}

function sliderMoved() {
  /* TODO 2 : inpVol2 doit être à la même position que inpVol1 */
  refreshPositions();
}

function sliderChangedPosition() {
  /* TODO 3 : inpVol3 doit être à la même position que inpVol1 */
  refreshPositions();
}

refreshPositions();
/* TODO 4 */
// Attacher l'évènement input à la fonction sliderMoved
// Attacher l'évènement change à la fonction sliderChangedPosition
```