ALOUANI Fadi

BTS SIO 1
groupe 1
B1
17/11/25

Atelier 13 : Javascript 2/2

Sommaire

Introduction

Dans le cadre de mes premiers travaux pratiques en JavaScript, j'ai découvert les bases du langage et la manière d'intégrer des scripts dans une page HTML. J'ai appris à utiliser la balise <script>.
Ce TP marque le début de mon apprentissage du fonctionnement et des règles fondamentales de JavaScript.

Navigateur 3

Exercice :

Question 1

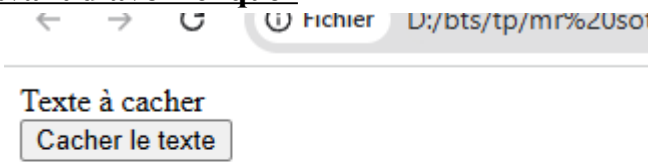
HTML :

```html
<div id="text">Texte à cacher</div>
<button id="hideButton">Cacher le texte</button>

<script>
  const button = document.getElementById('hideButton');
  const text = document.getElementById('text');

  button.addEventListener('click', () => {
    text.style.display = 'none';
  });
</script>
```

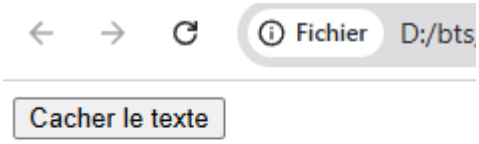
ALOUANI Fadi

Comme demander j'ai créer un bouton permettant de faire disparaître

avant d'avoir cliquer



après

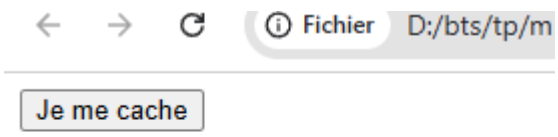


Question 2

Voici maintenant une version avec un bouton qui se cache tout seul

```
<button id="selfHideButton">Je me cache</button>

<script>
  const selfHideButton = document.getElementById('selfHideButton');

  selfHideButton.addEventListener('click', function() {
    this.style.display = 'none'; // 'this' fait référence au bouton
  });
</script>
```

avant avoir cliquer



après

Question 3

addEventListener() et removeEventListener() utilisent la même fonction de référence pour ajouter/supprimer un handler.

Comme on passe une fonction fléchée anonyme () => alert("1") aux deux méthodes, ce sont deux fonctions différentes en mémoire.

Donc removeEventListener() ne trouve pas la fonction à supprimer onclick est une propriété différente qui s'ajoute aux handlers existants

Au clic : les deux alertes s'affichent ("1" d'abord, puis "2")

ALOUANI Fadi

Exercice :

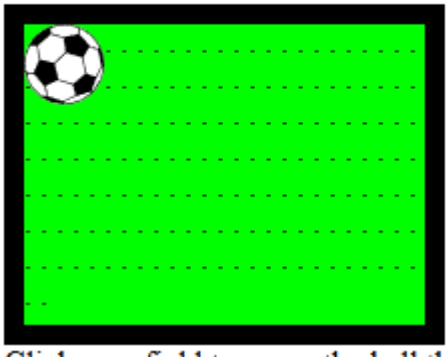
Question 1:ballon

Voici le script utiliser pour déplacer le ballon tout en suivant les contraintes

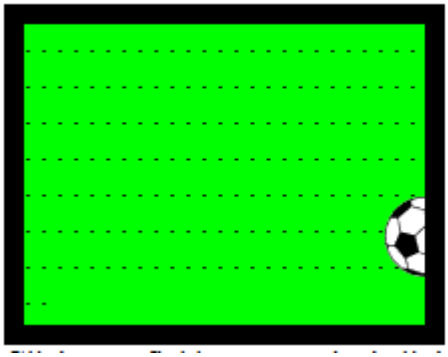
```
<script>
  const field = document.getElementById('field');
  const ball = document.getElementById('ball');

  // Position initiale du ballon au centre
  ball.style.left = field.offsetWidth / 2 - ball.offsetWidth / 2 + 'px';
  ball.style.top = field.offsetHeight / 2 - ball.offsetHeight / 2 + 'px';

  field.addEventListener('click', function(event) {
    // Coordonnées du clic par rapport à la fenêtre
    const clickX = event.clientX;
    const clickY = event.clientY;

    // Coordonnées du champ par rapport à la fenêtre
    const fieldRect = field.getBoundingClientRect();

    // Position souhaitée du centre du ballon
    // (en soustrayant le défilement pour les coordonnées absolues)
    let ballLeft = clickX - fieldRect.left - ball.offsetWidth / 2;
    let ballTop = clickY - fieldRect.top - ball.offsetHeight / 2;

    // Limites pour que le ballon reste dans le terrain
    // Ne pas dépasser le bord gauche
    ballLeft = Math.max(0, ballLeft);
    // Ne pas dépasser le bord droit
    ballLeft = Math.min(fieldRect.width - ball.offsetWidth, ballLeft);

    // Ne pas dépasser le bord haut
    ballTop = Math.max(0, ballTop);
    // Ne pas dépasser le bord bas
    ballTop = Math.min(fieldRect.height - ball.offsetHeight, ballTop);

    // Déplacer le ballon
    ball.style.left = ballLeft + 'px';
    ball.style.top = ballTop + 'px';
  });

  // Gérer le redimensionnement de la fenêtre
  window.addEventListener('resize', function() {
    // Réajuster la position si nécessaire
    const ballLeft = parseInt(ball.style.left) || 0;
    const ballTop = parseInt(ball.style.top) || 0;
    const maxLeft = field.offsetWidth - ball.offsetWidth;
    const maxTop = field.offsetHeight - ball.offsetHeight;

    ball.style.left = Math.min(ballLeft, maxLeft) + 'px';
    ball.style.top = Math.min(ballTop, maxTop) + 'px';
  });
</script>
```

Comme nous pouvons le constater, lorsque j'ai cliqué afin de déplacer le ballon de l'autre côté du terrain, l'action s'est déroulée correctement. Le ballon s'est déplacé comme prévu sans sortir des limites du terrain. Cela montre que les contraintes mises en place ont bien été respectées et que le système fonctionne conformément aux règles définies.

Click on a field to move the ball there.
The ball should never leave the field.



The ball should never leave the field.



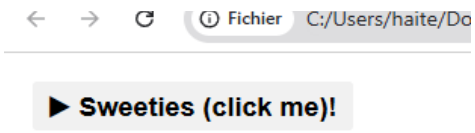
Question 2 : Menu glissant

L'HTML étant trop long, j'ai seulement mis le script, car je pense que c'est la partie la plus pertinente.

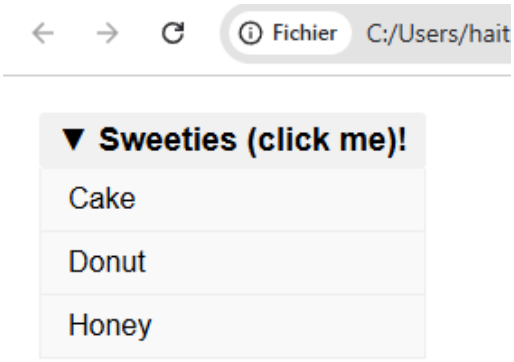
```
<script>
  const menu = document.getElementById('menu');
  const menuToggle = menu.querySelector('.menu-toggle');
  const menuContent = menu.querySelector('.menu-content');

  let isOpen = false;

  menuToggle.addEventListener('click', function(event) {
    event.stopPropagation(); // Empêche la fermeture immédiate

    if (!isOpen) {
      // Ouvrir le menu
      menu.classList.add('open');
      // Calculer la hauteur exacte pour l'animation
      const itemsHeight = Array.from(menuContent.children)
        .reduce((total, item) => total + item.offsetHeight, 0);
      menuContent.style.height = itemsHeight + 'px';
    } else {
      // Fermer le menu
      menu.classList.remove('open');
      menuContent.style.height = '0';
    }

    isOpen = !isOpen;
  });

  // Fermer le menu si on clique ailleurs
  document.addEventListener('click', function() {
    if (isOpen) {
      menu.classList.remove('open');
      menuContent.style.height = '0';
      isOpen = false;
    }
  });

  // Événements pour les items du menu
  const menuItems = menuContent.querySelectorAll('li');
  menuItems.forEach(item => {
    item.addEventListener('click', function() {
      alert('Selected: ' + this.textContent);
    });
  });
</script>
```

ALOUANI Fadi

Comme demandé dans l'exercice voici un menu déroulant qui s'ouvre et se replie au clic



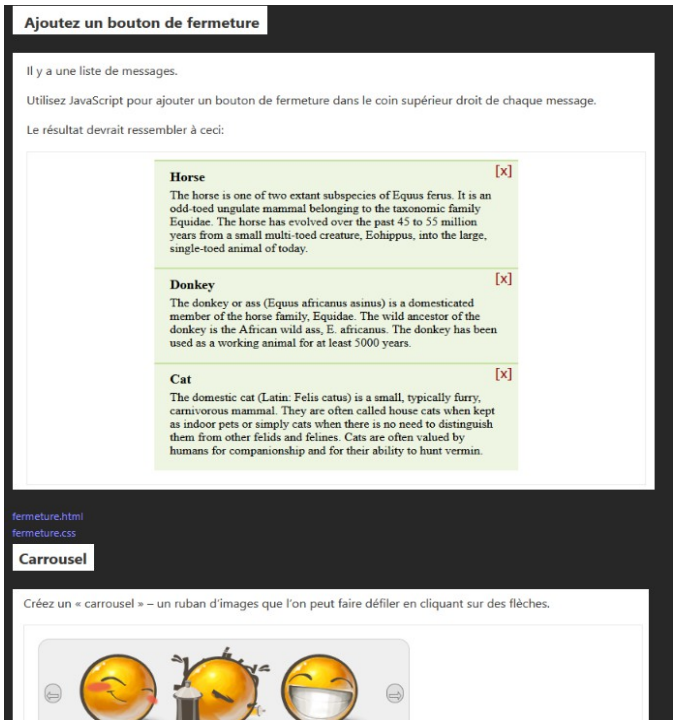
Voici le menu une fois dérouler



Ce que m'affiche la page après avoir cliquer sur une des sucrerie :

ALOUANI Fadi

Exercice :



Question 1 : Fermeture

(j'ai seulement mis la partie script de l'HTML)

```
<script>
  // Sélectionner tous les panneaux
  const panes = document.querySelectorAll('.pane');

  // Pour chaque panneau
  panes.forEach(pane => {
    // Créer le bouton de fermeture
    const closeButton = document.createElement('button');
    closeButton.className = 'remove-button';
    closeButton.textContent = '[x]';
    closeButton.title = 'Fermer ce message';

    // Positionner le bouton dans le coin supérieur droit
    pane.style.position = 'relative'; // Important pour le positionnement absolu

    // Ajouter le bouton au panneau
    pane.appendChild(closeButton);

    // Ajouter l'événement de clic pour supprimer le panneau
    closeButton.addEventListener('click', function() {
      pane.style.transition = 'opacity 0.3s ease';
      pane.style.opacity = '0';

      // Attendre la fin de l'animation avant de supprimer
      setTimeout(() => {
        pane.remove();
      }, 300);
    });
  });
</script>

</body>
```

ALOUANI Fadi

Le CSS :

```css
body {
  margin: 10px auto;
  width: 470px;
}

h3 {
  margin: 0;
  padding-bottom: .3em;
  font-size: 1.1em;
}

p {
  margin: 0;
  padding: 0 0 .5em;
}

.pane {
  background: #edf5e1;
  padding: 10px 20px 10px;
  border-top: solid 2px #c4df9b;
  position: relative; /* Important pour positionner le bouton */
  margin-bottom: 10px;
  transition: opacity 0.3s ease; /* Animation pour la disparition */
}

.remove-button {
  font-size: 110%;
  color: darkred;
  position: absolute;
  top: 10px;
  right: 10px;
  width: 24px;
  height: 24px;
  border: none;
  background: transparent;
  cursor: pointer;
  padding: 0;
  line-height: 24px;
  text-align: center;
}

.remove-button:hover {
  color: #ff0000;
  font-weight: bold;
}
```

Résultat :

The button code (may need to adjust CSS):

Horse [x]

The horse is one of two extant subspecies of Equus ferus. It is an odd-toed ungulate mammal belonging to the taxonomic family Equidae. The horse has evolved over the past 45 to 55 million years from a small multi-toed creature, Eohippus, into the large, single-toed animal of today.

Donkey [x]

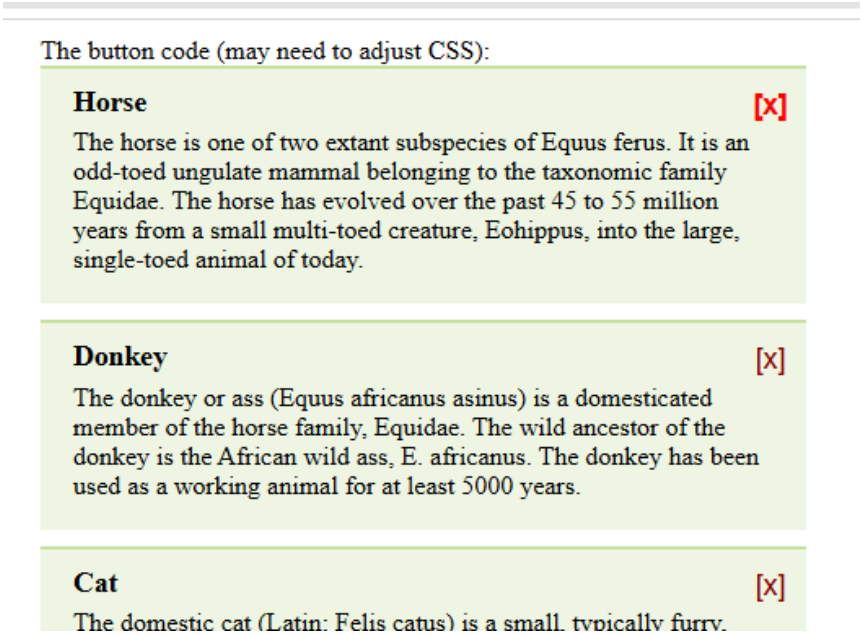
The donkey or ass (Equus africanus asinus) is a domesticated member of the horse family, Equidae. The wild ancestor of the donkey is the African wild ass, E. africanus. The donkey has been used as a working animal for at least 5000 years.

Cat [x]

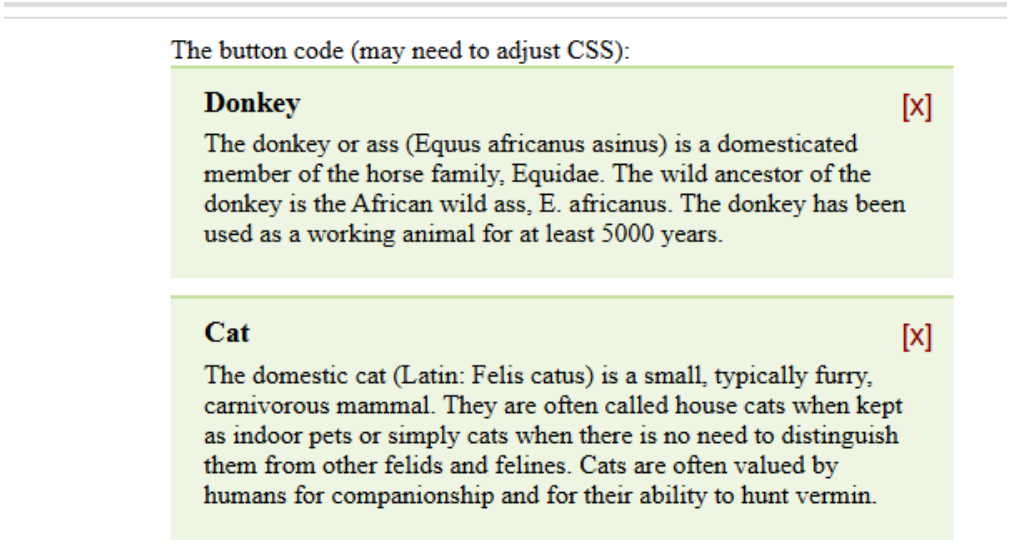
The domestic cat (Latin: Felis catus) is a small, typically furry, carnivorous mammal. They are often called house cats when kept as indoor pets or simply cats when there is no need to distinguish them from other felids and felines. Cats are often valued by humans for companionship and for their ability to hunt vermin.

ALOUANI Fadi

On peut noter que le css a fonctionner correctement le style est similaire à celui de l'exemple.
J'ai néanmoins ajouté un hover pour que la croix de fermeture change de couleur.



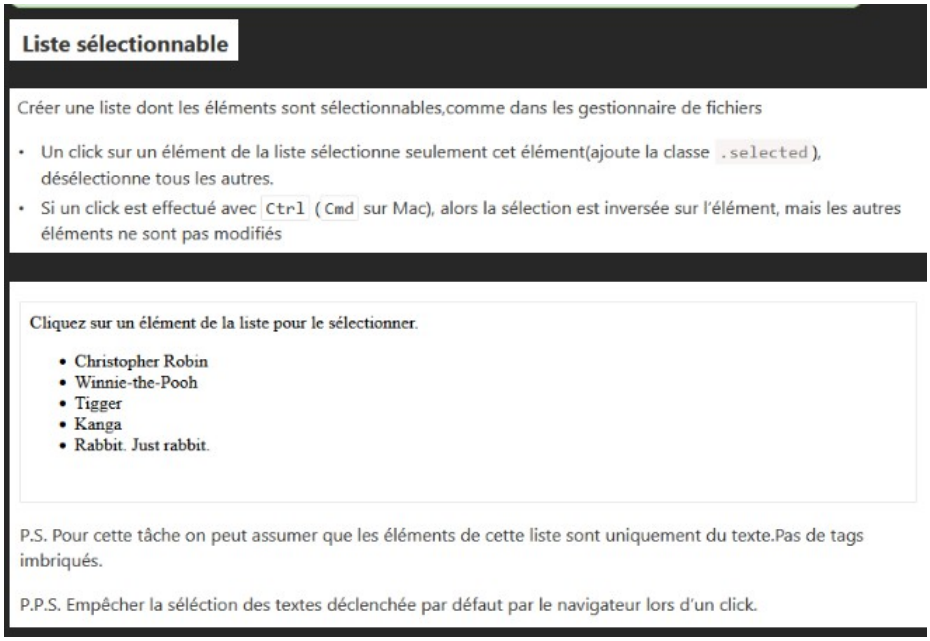
J'ai fermer l'onglet Horse cela fonctionne donc correctement.



ALOUANI Fadi

Navigateur 4

Exercice :



HTML :

```html
<!DOCTYPE HTML>
<html>

<head>
  <meta charset="utf-8">
  <style>
    .selected {
      background: #0f0;
    }

    li {
      cursor: pointer;
      padding: 5px;
      margin: 2px 0;
      border-radius: 3px;
      list-style-position: inside;
    }

    /* Empêcher la sélection de texte */
    ul {
      user-select: none;
      -webkit-user-select: none;
      -moz-user-select: none;
      -ms-user-select: none;
      padding-left: 20px;
    }

    body {
      font-family: Arial, sans-serif;
    }
  </style>
</head>

<body>

  Cliquez sur un élément de la liste pour le sélectionner.
  <br>
  <em>(Maintenez Ctrl/Cmd pour sélectionner multiple)</em>

  <ul id="ul">
    <li>Christopher Robin</li>
    <li>Winnie-the-Pooh</li>
    <li>Tigger</li>
    <li>Kanga</li>
    <li>Rabbit. Just rabbit.</li>
  </ul>
```

ALOUANI Fadi

SUITE :

```
</ul>

<script>
  const ul = document.getElementById('ul');

  // Gérer la sélection des éléments
  ul.addEventListener('click', function(event) {
    // Vérifier si on a cliqué sur un élément LI
    if (event.target.tagName !== 'LI') return;

    const li = event.target;

    // Si Ctrl (ou Cmd sur Mac) est pressé
    if (event.ctrlKey || event.metaKey) {
      // Inverser la sélection de cet élément seulement
      li.classList.toggle('selected');
    } else {
      // Sans Ctrl : désélectionner tout et sélectionner cet élément
      const allItems = ul.querySelectorAll('li');
      allItems.forEach(item => {
        item.classList.remove('selected');
      });
      li.classList.add('selected');
    }

    // Empêcher la sélection de texte
    event.preventDefault();
  });

  // Empêcher la sélection de texte via d'autres moyens
  ul.addEventListener('mousedown', function(event) {
    // Si on clique sur un LI, empêcher la sélection de texte
    if (event.target.tagName === 'LI') {
      event.preventDefault();
    }
  });

  // Empêcher la sélection par glisser-déposer
  ul.addEventListener('dragstart', function(event) {
    if (event.target.tagName === 'LI') {
      event.preventDefault();
    }
  });

  // Optionnel : ajouter un feedback visuel pour la sélection multiple
  ul.addEventListener('mouseover', function(event) {
    if (event.target.tagName === 'LI' && (event.ctrlKey || event.metaKey)) {
      event.target.style.outline = '2px dashed #999';
    }
  });

  ul.addEventListener('mouseout', function(event) {
    if (event.target.tagName === 'LI') {
      event.target.style.outline = 'none';
    }
  });
</script>
```

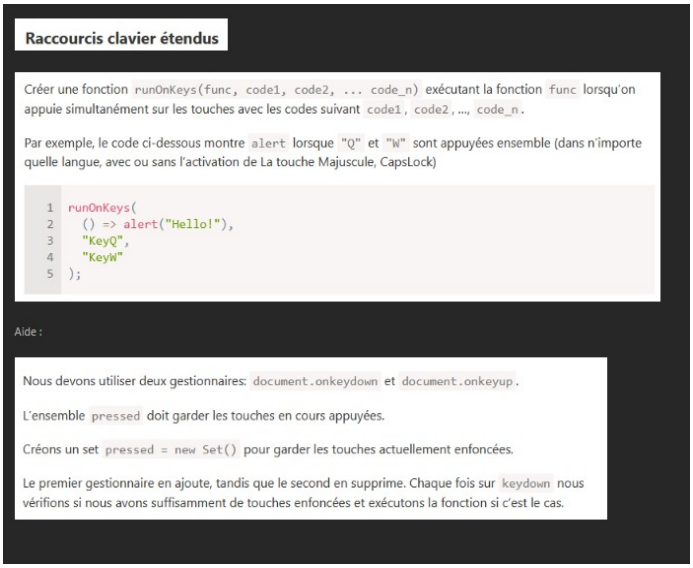
RESULTAT :

J'ai sélectionner les deux premier nom pour tester la fonction sélection multiple avec ctrl .

- Christopher Robin
- Winnie-the-Pooh
- Tigger
- Kanga
- Rabbit. Just rabbit.

ALOUANI Fadi

Exercice :



Raccourcis clavier étendus

Créer une fonction `runOnKeys(func, code1, code2, ... code_n)` exécutant la fonction `func` lorsqu'on appuie simultanément sur les touches avec les codes suivant `code1`, `code2`, ..., `code_n`.

Par exemple, le code ci-dessous montre `alert` lorsque `"Q"` et `"W"` sont appuyées ensemble (dans n'importe quelle langue, avec ou sans l'activation de La touche Majuscule, CapsLock)

```
runOnKeys(
  () => alert("Hello!"),
  "KeyQ",
  "KeyW"
);
```

Aide :

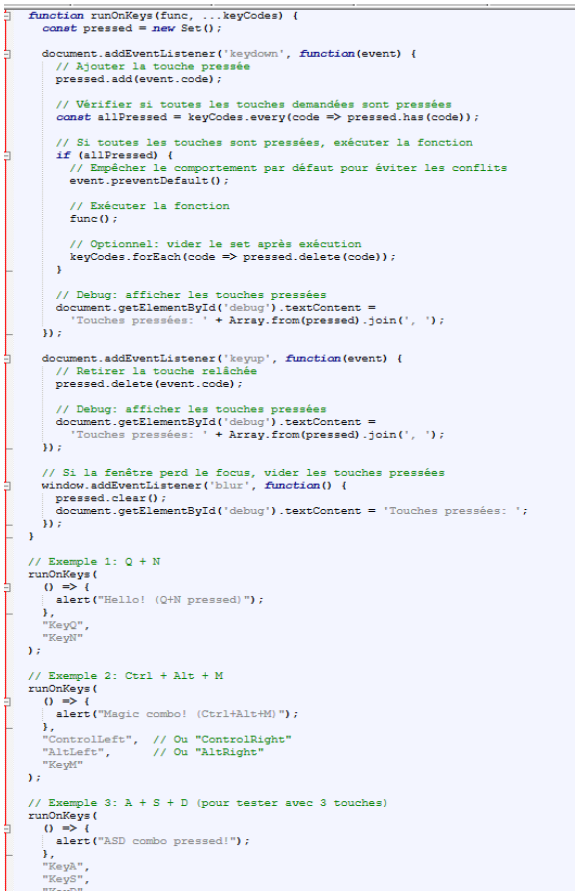
Nous devons utiliser deux gestionnaires: `document.onkeydown` et `document.onkeyup`.

L'ensemble `pressed` doit garder les touches en cours appuyées.

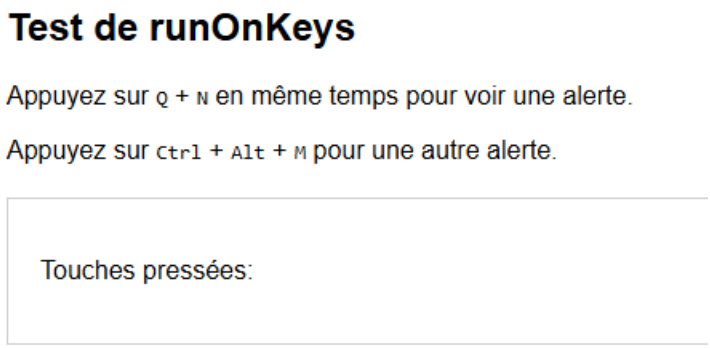
Créons un set `pressed = new Set()` pour garder les touches actuellement enfoncées.

Le premier gestionnaire en ajoute, tandis que le second en supprime. Chaque fois sur `keydown` nous vérifions si nous avons suffisamment de touches enfoncées et exécutons la fonction si c'est le cas.

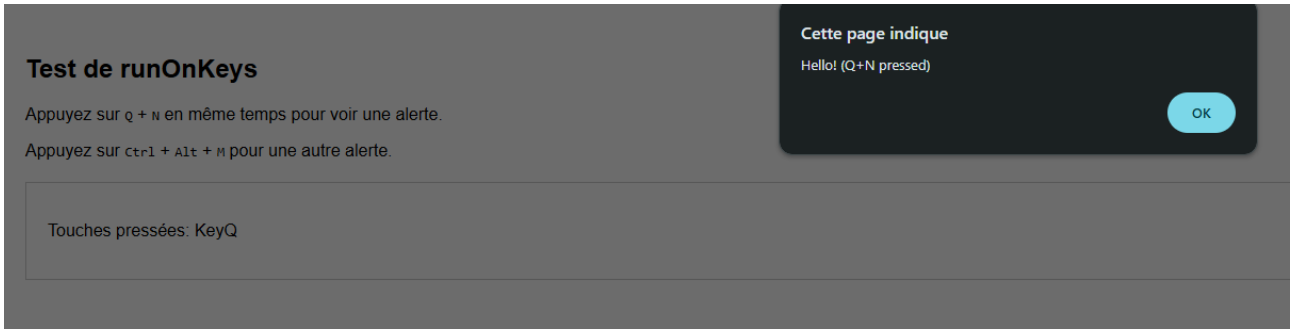
Voici la fonctions que j'ai utiliser (zoomer)



```
function runOnKeys(func, ...keyCodes) {
  const pressed = new Set();

  document.addEventListener('keydown', function(event) {
    // Ajouter la touche pressée
    pressed.add(event.code);

    // Vérifier si toutes les touches demandées sont pressées
    const allPressed = keyCodes.every(code => pressed.has(code));

    // Si toutes les touches sont pressées, exécuter la fonction
    if (allPressed) {
      // Empêcher le comportement par défaut pour éviter les conflits
      event.preventDefault();

      // Exécuter la fonction
      func();

      // Optionnel: vider le set après exécution
      keyCodes.forEach(code => pressed.delete(code));
    }

    // Debug: afficher les touches pressées
    document.getElementById('debug').textContent =
      'Touches pressées: ' + Array.from(pressed).join(', ');
  });

  document.addEventListener('keyup', function(event) {
    // Retirer la touche relâchée
    pressed.delete(event.code);

    // Debug: afficher les touches pressées
    document.getElementById('debug').textContent =
      'Touches pressées: ' + Array.from(pressed).join(', ');
  });

  // Si la fenêtre perd le focus, vider les touches pressées
  window.addEventListener('blur', function() {
    pressed.clear();
    document.getElementById('debug').textContent = 'Touches pressées: ';
  });
}

// Exemple 1: Q + N
runOnKeys(
  () => {
    alert("Hello! (Q+N pressed)");
  },
  "KeyQ",
  "KeyN"
);

// Exemple 2: Ctrl + Alt + M
runOnKeys(
  () => {
    alert("Magic combo! (Ctrl+Alt+M)");
  },
  "ControlLeft",  // Ou "ControlRight"
  "AltLeft",      // Ou "AltRight"
  "KeyM"
);

// Exemple 3: A + S + D (pour tester avec 3 touches)
runOnKeys(
  () => {
    alert("ASD combo pressed!");
  },
  "KeyA",
  "KeyS",
```



Apres avoir presser q et n voici le pop up ;



Conclusion

Ce TP m'a permis de comprendre les premiers éléments essentiels de JavaScript et d'apprendre à exécuter du code dans une page web. J'ai découvert les instructions, les points-virgules, les commentaires et le mode strict. Ces notions me serviront de base pour progresser vers des scripts plus complexes et développer mes compétences en programmation web.