UAA12 JavaScript (js)

TP1 – Découverte : Modeler une page web avec js

1 Introduction

Avant d'entrer dans le vif du sujet, un petit historique :

- ✓ JavaScript a été créé en dix jours par Brendan Eich en 1995. Malgré son nom, JavaScript n'a rien à voir avec le langage Java, même si Brendan Eich affirme s'être inspiré de nombreux langage, dont Java, pour le mettre au point.
  Après des débuts chaotiques, il est devenu incontournable dans le développement web.
- ✓ JavaScript a été conçu pour être **exécuté directement par le navigateur et côté client**. Quand tu ouvres une page web contenant du JS, il sera exécuté par votre machine et non par le serveur. C'est très important, car cela permet au **serveur de limiter sa charge**. 100 clients en même temps ? Il sert une fois la page par client et les calculs sont effectués chez ceux-ci.
- ✓ HTML5 est une norme qui voit le jour en 2014 décrivant le web moderne. Par abus de langage, HTML5 désigne le trio 'html, css, js'.
- ✓ Depuis quelques années JavaScript a beaucoup évolué et il existe de nombreux projets majeurs :
  - possibilité d'exécuter un **serveur** qui fonctionne en JS : c'est **node.js** (2009)
  - possibilité de créer des applications de bureau qui fonctionnent sur un serveur **node.js** et s'exécutent dans "chromium" (partie open source de Google Chrome). Par exemple : atom qui utilise la technologie **electron.**
- ✓ JavaScript est le langage informatique le plus populaire depuis quelques années, il a surpassé Java, C/C++ avec l'essor du web et est talonné par Python.
  En 2019, presque tous les **éléments "dynamiques" d'une page web** moderne sont programmés en JavaScript.

Voici quelques éléments de comparaison entre Python et Javascript :

Instruction	**Python**	**JavaScript**
Exécuté par	le programme Python	le navigateur (généralement)
Code contenu dans	un fichier .py	une balise script d'une page html
Créer une variable	a = 2	var a = 2;
Fin d'instruction	retour à la ligne	point virugle ;
indentation	définit la structure	optionnelle
fonction	def f(x): ...	function f(x) { expr }
condition	if condition: expression	if (condition) { expr }
boucle for	for i in range(10): ...	for (var i=0; i<10; i++) { expr }

boucle while	while condition: expr	while (condition) { expr }
commentaire	# un commentaire	// un commentaire
écrire dans la console	print(expr1, expr2)	console.log(expr1, expr2);

2 Test1

Tu vas travailler avec une page html, un fichier css et un fichier js.
Ce dernier va permettre d'ajouter de l'interactivité à ta page html.

1. Crée une page html test1.html dans un dossier uaa12_web/uaa12_js/tp1_decouverte

Contenu de test1.html:

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="UTF-8">
   <title>Test 1</title>
</head>
<body>
   <h1>Voici un premier exemple de JS.</h1>
</body>
   <script>
      var prenom = prompt ('Quel est votre prénom ?');
      alert ('Bonjour ' + prenom);
   </script>
</html>
```

2. Ouvre la page dans le navigateur Firefox pour comprendre le rôle de **prompt** et **alert**.

3. Maintenant, modifie **alert** en **console.log**.

4. Tu n'as plus de fenêtre te saluant, mais tu vas découvrir la console cachée dans ton navigateur. Pour cela, ouvre les outils de développement :

menu en haut à droite -> Développement Web -> Outils de développement

ou avec le raccourci

Ctrl + Maj + i

Ensuite, sélectionne la console et observe la salutation.
Cette fonctionnalité est bien pratique pour les développeurs quand il faut débogger un code.

3 Test2

1. Dans le même dossier que précédemment, crée le fichier **test2.html**.

Contenu de test2.html:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Test 2</title>
</head>
<body>
    <h1>Voici un second exemple de JS.</h1>
</body>
    <script src="test2.js"></script>
</html>
```

Celui-ci fera appel à un fichier JavaScript **test2.js**

2. Crée le fichier test2.js

Contenu de test2.js:

```
var prenom = prompt('Quel est votre prénom ?');
var nom = prompt('Quel est votre nom ?');
var age = prompt('Quel est votre age');
var age_futur = age + 1 ;
alert('Bonjour ' + prenom + nom);
alert('Dans un an vous aurez ' + age_futur + ' ans');
```

3. A ton avis, que fais ce script ?

Ouvre test2.html dans le navigateur. Est-ce que tu t'attendais à voir cette page?

Corrigeons notre fichier. En fait, la variable age est du type "chaîne de caractère" au lieu d'être un entier. D'où le résultat précédent.

Il faut la convertir en entier en utilisant la fonction parseInt()

```
var age = prompt('Quel est votre age');
age = parseInt(age);
```

Teste le résultat, c'est mieux non ?

4 Test3 – Exercice d'application

En t'inspirant de l'exemple précédent, crée un fichier html et un fichier JavaScript.
A l'ouverture du fichier html, une fenêtre demandera la largeur d'un rectangle, puis une autre demandera la longueur d'un rectangle.
Enfin, la page affichera dans une fenêtre l'aire du rectangle.

TP2 - Intéraction html, css et js

Le Javascript va modifier les pages html et css en accédant aux éléments cible du DOM. (voir aussi le cours précédent).
Le **DOM** (pour **Document Object Model**) est une interface de programmation pour les documents HTML.
Une interface de programmation, qu'on appelle aussi une **API** (pour **Application Programming Interface**), est un ensemble d'outils qui permettent de faire communiquer entre eux plusieurs programmes ou, dans le cas présent, différents langages. Le terme API reviendra souvent, quel que soit le langage de programmation que tu apprendras.
Le DOM est donc une API qui s'utilise avec les documents HTML, et qui va nous permettre, via le Javascript, d'accéder au code HTML d'un document. C'est grâce au DOM que nous allons pouvoir modifier des éléments HTML (afficher ou masquer un ** ** par exemple), en ajouter, en déplacer ou même en supprimer.
Dans un cours sur le HTML, on parlera de **balises** HTML ; ici, en Javascript, on parlera **d'élément HTML**, pour la simple raison que **chaque balise est vue comme un objet**.
L'objet document représente la page Web et plus précisément la balise <html>. C'est grâce à cet élément-là que nous allons pouvoir **accéder via le DOM aux éléments HTML et les modifier.**
L'accès aux éléments html via le DOM peut se faire via 4 méthodes :

- ✓ **getElementById():** accès par **l'id de la balise**
- ✓ **getElementByName**(): accès par le **nom du formulaire**,
- ✓ **getElementByClassName():** accès par la **classe de la balise**
- ✓ **getElementByTagName() :** accès par la **classe de la balise**

4. Crée un fichier **test1.html** sous uaa12_js /tp2_interaction et saisis le code ci-dessous

Code dans test1.html

```
…
<body>
    <h1>Voici un joli titre.</h1>
    <p id='important'>Ceci est un texte qu'il faut mettre en valeur !</p>
    <button onclick="change_couleur()">Cliquez ici !</button>
</body>
    <script src="js/test1.js"></script>
</html>
```

Dans le code précédent, **<button onclick="change_couleur()">** Cliquez ici ! </button> est une balise qui va créer un bouton.
Un clic dessus va déclencher la fonction **change_couleur()**.

Cette fonction va être définie dans un fichier javascript (**test1.js**) qui va être créé par ce qui suit ...

5. Ensuite crée un fichier **test1.js** dans le sous-dossier js.

Code dans test1.js

```
function change_couleur() {
    var paragraphe = document.getElementById("important");
    console.log(paragraphe)
}
```

6. Ouvre la console des outils de développement web.

Tu dois constater que la variable **paragraphe** contient maintenant la balise paragraphe et son contenu.
Dans la variable paragraphe, l'élément d'ID (identifiant) "important" est stocké.
Ajoute le code ci-dessous au fichier js afin de modifier la couleur du paragraphe.

Code ajouté dans test1.js

```
function change_couleur() {
    var paragraphe = document.getElementById("important");
    paragraphe.style.color="red"
    console.log(paragraphe)
}
```

Appuie sur le bouton et constate le changement. Tu peux ainsi accéder à toutes les propriétés css du paragraphe.
Essayons de faire le changement plus proprement avec le css

7. Ajoute le fichier **test1.css** et insère le contenu ci-dessous

Code dans test1.css

```
h1 {
    text-align: center;
    font-size: 40px;
}
.rouge {
    color:red;
    font-size:30px;
}
```

Puis ajoute la ligne suivante dans le html pour lier le css avec votre fichier html.

```
<link rel="stylesheet" href="test1.css">
```

8. Copie test1.html et tes1.js dans les fichiers **test2.html** et test2.sj sous uaa12_js /tp2_interaction et modifie le js comme ce qui suit :

Code ajouté dans test1.js

```
function change_couleur() {
    var paragraphe = document.getElementById("important");
    console.log(paragraphe);
    paragraphe.classList.add("rouge");
}
```

Ce code aura pour effet d'ajouter la classe "rouge" à notre élément "paragraphe" Sélectionné.

9. Ajoute le bouton suivant dans le html :

Code ajouté dans test1.html

```
<button onclick="reset_couleur()"> Reset </button>
```

10. Puis ajoute la fonction suivante dans le Js :

Code ajouté dans test1.js

```
function reset_couleur() {
    var paragraphe = document.getElementById("important");
    console.log(paragraphe);
    paragraphe.classList.remove("rouge");
}
```

Teste le résultat !

11. Test3: Exercice d'application

Ajoute un bouton supplémentaire qui changera paragraphe de la façon suivante :

- ✓ couleur bleue
- ✓ taille : 70 px
- ✓ centré à droite

Pense à modifier le code du bouton reset pour qu'il fonctionne encore !

TP3 - querySelector et innerHTML

1 Une autre façon de sélectionner des éléments html : les querySelector

Le principe de sélection d'un querySelector est le suivant, on donne en paramètre un seul argument : une chaîne de caractère qui doit être un sélecteur CSS.
Par exemple si dans votre fichier css tu as :

```
#menu .item p {
    color : red;
}
```

alors vos paragraphes d'identifiant menu et de classe item seront en rouge.
Il y a deux méthodes pour les querySelector .

- ✓ **querySelector**() : renvoie le premier élément trouvé correspondant au sélecteur
- ✓ **querySelectorAll**() : renvoie tous les éléments correspondant au sélecteur

Nous allons tester ces deux méthodes avec le fichier dans le drive **test1.html**
Enregistre le fichier dans le dossier uaa12_js/tp3_querySelector
Crée un fichier **test1.js** que ru vas compléter comme qui suit :

Contenu test1.js

```
var query = document.querySelector("nav ul li");
console.log("Affichage de query");
console.log(query);
```

Dans ce code :

- ✓ Une variable nommée **query** contient le premier élément de la page html dont le sélecteur css est **nav ul li**
- ✓ Ensuite un affichage est effectué dans la console du développeur

Ouvre la page HTML dans Firefox, puis ouvre la console du développeur (CTRL + MAJ + i)
Observe l'élément obtenu, tu peux dérouler le contenu de cet élément en appuyant sur le petit triangle à gauche de son nom.
Passons à querySelectorAll(), ajoute les lignes suivantes dans le fichier test1.js

Contenu test1.js

```
var queryAll_1 = document.querySelectorAll(".menu li");
console.log("Affichage de queryAll_1");
console.log(queryAll_1);
```

Cette fois dans la variable queryAll_1 , il y aura tous les éléments dont le sélecteur CSS est ".menu li", c'est à dire les items de listes dont la classe est "menu".
Ouvre le fichier HTML dans un navigateur et ouvre la console du développeur (CTRL + MAJ + i).
Tu constates qu'il y a 3 éléments, ils sont stockés dans un tableau.
Pour afficher les éléments de façon individuelle dans un tableau, cela se passe comme en python.
queryAll_1[0] affichera le 1er élément du tableau.
Ajoute les lignes suivantes dans votre fichier js.

Contenu test1.js

```
console.log("Affichage de chaque éléments de queryAll_1 par appels successifs")
console.log(queryAll_1[0]);
console.log(queryAll_1[1]);
console.log(queryAll_1[2]);
```

Les trois éléments sélectionnés seront affichés. Mais s'il y en a plus, cette méthode sera fastidieuse.
Utilisons une boucle pour afficher ces éléments un à un.
Pour ce faire, nous allons sélectionner tous les éléments "li" enfants de balise <nav>.
Un sélecteur valide est : nav li. Ajoute les lignes suivantes dans le fichier js.

Contenu test1.js

```
var queryAll_2 = document.querySelectorAll("nav ul li");
console.log("Affichage de queryAll_2");
console.log(queryAll_2);
```

Tu vas trouver 6 éléments. Affichons-les dans la console de façon individuelle avec une boucle

Contenu test1.js

```
console.log('Affichage de chaque éléments de queryAll_1 avec une boucle pour : ');
for (let index = 0; index < queryAll_2.length; index++) {
    const element = queryAll_2[index];
    console.log(element);
}
```

Ce type de boucle sera pratique pour modifier chaque élément sélectionné avec un querySelectorAll.

2 Modification du contenu des éléments HTML avec JS.

Nous sommes maintenant capables de sélectionner divers éléments HTML, mais il serait pratique de modifier leur contenu. Par exemple, modifier ou ajouter du texte dans un paragraphe.
Pour cela la méthode **innerHTML** va nous servir. Ajoute les lignes suivantes à votre fichier JS.

Contenu test1.js

```
var important = document.querySelector("#important");
/* Affichage de l'élément sélectionné ayant l'ID important. */
alert(important);
/* Affichage du contenu du paragraphe ayant l'ID important. */
alert(important.innerHTML);
```

La variable important contient l'élément HTML dont l'ID est "important". Pour accéder à son contenu textuel, il faut utiliser la méthode **innerHTML** .
La syntaxe est la suivante : objet.méthode donc ici important.innerHTML est le contenu textuel de l' "objet " important.
Il est possible de changer ce texte

```
/* Modification du paragraphe */
important.innerHTML = " Voici une citation d'Alan Turing : <q> Les tentatives de création de machines pensantes nous seront d'une grande aide pour découvrir comment nous pensons nous-mêmes. </q>";
alert('Observe le 2eme paragraphe du deuxième article. Il va encore se modifier quand tu cliqueras sur OK. ');
important.innerHTML += "<br> Cette citation date de 1951";
```

Avec la première ligne de code, la variable **important.innerHTML** reçoit un nouveau texte entre guillemets. Il est possible d'y placer des balises html.
Dans la dernière ligne de code, comme en python, += permet d'ajouter du contenu à la variable. (cela reviendrais à écrire :
important.innerHTML = **important.innerHTML** + "
 Cette citation date de 1951";

Exercice d'application

Ajoute du code dans le fichier js pour que :

- ✓ l'utilisateur clique sur un bouton puis
- ✓ une fenêtre de type alert s'affiche. Elle contient le texte :
  " Attention les titres vont changer"
- ✓ Ensuite, après un clic sur ok, il faudra ajouter " super interessant !" à la fin de chaque titre.
  Par exemple: le premier titre deviendra : Un premier article super intéressant !

TP4 – Les évènements

Interactions avec l'utilisateur :

Les événements permettent de déclencher une fonction selon qu'une action s'est produite ou non. Par exemple, faire apparaître une fenêtre alert() lorsque l'utilisateur survole une zone d'une page web, ou ajouter un élément lors d'un clic de bouton de la souris (ou du clavier).

Liste des évènements :

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
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.)
input	Taper un caractère dans un champ de texte (son support n'est pas complet sur tous les navigateurs)
select	Sélectionner le contenu d'un champ de texte (input,textarea, etc.)

Test1 – survol avec la souris [David Roche A faire vous meme 10]

Il existe d'autres événements que "onclick" vu précédemment, par exemple, il est possible de détecter le "survol" par le curseur de la souris d'un élément HTML.

12. Crée un fichier **test1.html** sous uaa12/tp4_evenements et saisis le code ci-dessous

13. Crée un fichier test1.js et écris les 2 fonctions **foncEntre()** et **foncQuitte()** qui vont modifier la classe bouton en fonction du déplacement de la souris: quand la souris survole le bouton, ce dernier devient rouge ; le reste du temps il est blanc.

Contenu de test1.html

```
<!doctype html>
<html lang="fr">
<head>
    <meta charset="utf-8">
    <title> Survol souris </title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <h1>Le trio : HTML, CSS et JavaScript</h1>
    <div onmouseover="foncEntre()" onmouseout="foncQuitte()" id="maDiv">
    <p>Survolez-moi</p></div>
</body>
<script src="script.js"></script>
</html>
```

"**onemouseover**" correspond au survol par le curseur de la souris d'un élément HTML. L'événement "**onemouseout**" est lui déclenché quand le curseur de la souris quitte un élément HTML donné.

Contenu de style.css

```
h1 {
    text-align: center;
}
p {
    text-align : center;
}
#maDiv {
    width : 200px;
    height : 100px;
    margin : 0 auto;
    border : 2px solid black;
}
.rouge {
    background-color:red;
}
.blanc {
    background-color : white;
}
```

TP5 - Exercices complémentaires

Planification des tests à faire en classe, à la maison, en remédiation

Test	A faire en classe (C)	A faire à la maison (M)
Test1 – aire disque		M
Test2 - IMC	C	
Test3 - querySelector	C	
Test4 - random		M
Test5 – querySelectorAll	C	
Test6 – querySelector.classList.add/remove		M
Test7 – Carrés de couleur	C	
Test8 – Texte cache		M

1 Les Bases : Variables et alert

1.1 Test 1 – Aire d'un disque

[math du yeti]

Crée un script JavaScript qui lors de l'ouverture du fichier html :

- demande le rayon d'un disque à l'utilisateur
- affiche dans une boîte alert l'aire de ce disque.

TT s'obtient avec Math.PI Elle est décrite sur w3scool :

https://www.w3schools.com/jsref/jsref_pi.asp

La fonction parseFloat() permet de transformer une chaîne de caractères en un réel. Elle est décrite sur w3scool :

https://www.w3schools.com/jsref/jsref_parsefloat.asp

1.2 Test2 - IMC

Crée un script JavaScript qui lors de l'ouverture du fichier html :

- demande la taille en m et le poids en kg,
- affiche dans une boîte alert l'IMC (Indice de Masse Corporelle) de cette personne.
- suivant la valeur de l'IMC, une deuxième fenêtre s'affichera pour donner l'interprétation de l'IMC (dénutrition ou anorexie, maigreur, poids idéal, ...)

Voir ici pour les informations sur l'IMC:

https://fr.wikipedia.org/wiki/Indice_de_masse_corporelle

2 JavaScript et HTML/CSS

2.1 Test3 - querySelector

Crée la page html suivante :

Code de test3.html:

```
<h1 title=" Quand on clique , modif du paragraphe"> Grand titre </h1>

<p id=premier" title="Quand on clique , centrage du titre"> Premier paragraphe original. </p>
```

Note en passant l'utilisation des attributs **title** pour informer sur les éléments de titre et paragraphe (pour les tester, laisse la souris au-dessus de l'élément).

1. Crée une fonction **modifParag()**, appelée en cliquant sur le **titre**.

   Cette fonction change le pragraphe par une autre phrase écrite en italique.

2. Crée une fonction **centreH1()**, appelée en cliquant sur le **paragraphe**.

   Cette fonction centre le titre1.
   L'attribut css pour centrer du texte et sa description dans le DOM sont décrits sur w3scool :
   https://www.w3schools.com/cssref/pr_text_text-align.ASP
   https://www.w3schools.com/jsref/prop_style_textalign.asp

2.2 Test4 - Random

Ecris une page web avec un bouton sur lequel sera affiché "changer la couleur".
Lorsque l'on clique sur ce bouton, une fonction JavaScript **changeCouleurFond** se lance et change la couleur du fond de page de façon aléatoire au format RGB.

Indications :

- Une couleur RGB s'écrit rgb(rouge, vert, bleu) dans lesquels rouge, vert et bleu sont trois entiers compris entre0 et 255.
- On peut tirer un nombre aléatoire entre 0 et max avec la fonction getRandomInt(max) dont voici le code :

```
function getRandomInt(max) {
  return Math.floor(Math.random () * Math.floor(max));
}
```

3 QuerySelector

3.1 Test5 - querySelectorAll

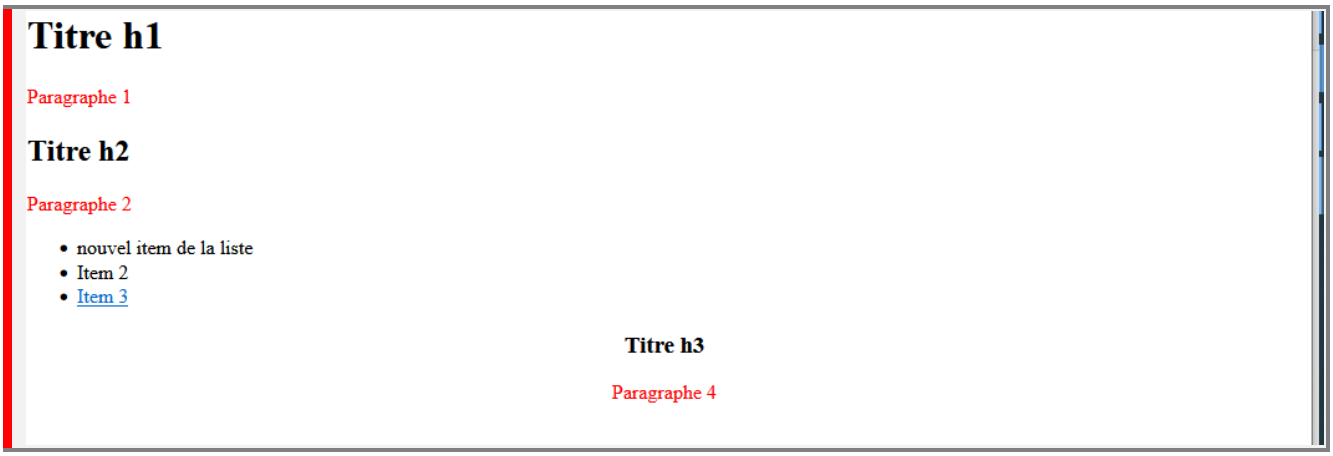
Créez la page html suivante :

Code de test5.html

```
<h1>Titre h1</h1>
<p class="par"> Paragraphe 1</p>
<h2> Titre h2 </h2>
<p class="par"> Paragraphe 2</p>
<ul>
    <li> Item 1</li>
    <li>Item 2</li>
    <li> <a href="http ://www.google.fr"> Item 3 </a> </li>
</ul>
<div id="bloc">
    <h3> Titre h3 </h3>
    <p class="par"> Paragraphe 4 </p>
</div>
```

3. Centre le texte sélectionné par
4. Change le texte "item 1" par "le nouvel item de la liste"
5. Mets tous les paragraphes en rouge

Page test5.html dans le navigateur



3.2 Test6 - classList.add/remove

[David Roche A faire vous meme 8]

Ecris le code permettant d'obtenir la page suivante.

Page test6.html dans le navigateur quand on clique sur le bouton Rouge

Le trio : HTML, CSS et Javascript

Voici une page qui ne fait pas grand-chose

Rouge Vert

Page test6.html dans le navigateur quand on clique sur le bouton Vert

Le trio : HTML, CSS et Javascript

Voici une page qui ne fait VRAIMENT pas grand-chose !!!

Rouge Vert

Propriétés dans test6.css

« Le trio : HTML, CSS et Javascript »	1er niveau de titre, centré
« Voici une page qui ne fait pas grand-chose »	Paragraphe, rouge, de taille 20px
« Voici une page qui ne fait VRAIMENT pas grand-chose !!! »	Paragraphe, vert, de taille 30px

4 Les évènements en JS.

4.1 Test7 – Carrés de couleur

[Diderot test6]

6. Crée la page avec les propriétés css ci-dessous.

   Le carré est au départ rouge et doit devenir bleu puis jaune à chaque fois que l'on clique dessus.

Code css

```
<style>
    body{
        width: 300px;
        height:300px;
        border: 1px solid black;
     }
     #carreRouge {
        margin:0;
        width: 50px;
        height: 50px;
        background-color: red;
     }
</style>
```

7. Sous le carré, rajoute le message « Clique sur la carré rouge ». Le message changera à chaque clique avec la couleur du carré en « Clique sur la carré bleu / Jaune ».

8. Change aussi la taille du carré bleu en 100px et du carré jaune en 200px. Pour cela utilise la fonction ClassList.remove, classList.add

4.2 Test8 – Texte caché

[Diderot test7bis]

9. Crée la page avec le contenu et le css suivants :

Code html

```
<section>
   <h2 id="titre">Une identité remarquable</h2>

   <article id="texte">  Pour tous réels a et b, on a a<sup>2</sup> - b<sup>2</sup> = (a-b)(a+b).  </article>
   <div id="demonstration">  Il suffit de développer le membre de droite!  </div>
</section>
```

Attributs css

```
<style>
   section{
      border: 1px solid red;
      margin:1em;
      padding:1em;
   }
   article{
      border: 2px dotted black;
      margin: 1em;
      text-align: center;
   }
   #texte {
      display:none;
    }
   #demonstration{
      display:none;
    }
</style>
```

10. Ecris le code de façon à ce que :

   ✓ Un clic sur le titre du théorème montre le texte du théorème (mais pas la démonstration) si ce texte est caché et cache le texte du théorème (et la démonstration) si le texte est visible.

- ✓ Si le texte est visible, un clic sur le texte cache ou montre la démonstration suivant qu'elle est visible ou non.

Remarque

Le texte n'est pas visible si l'attribut **display** a comme valeur « **none** », et sera au contraire visible avec la valeur « **block** ».