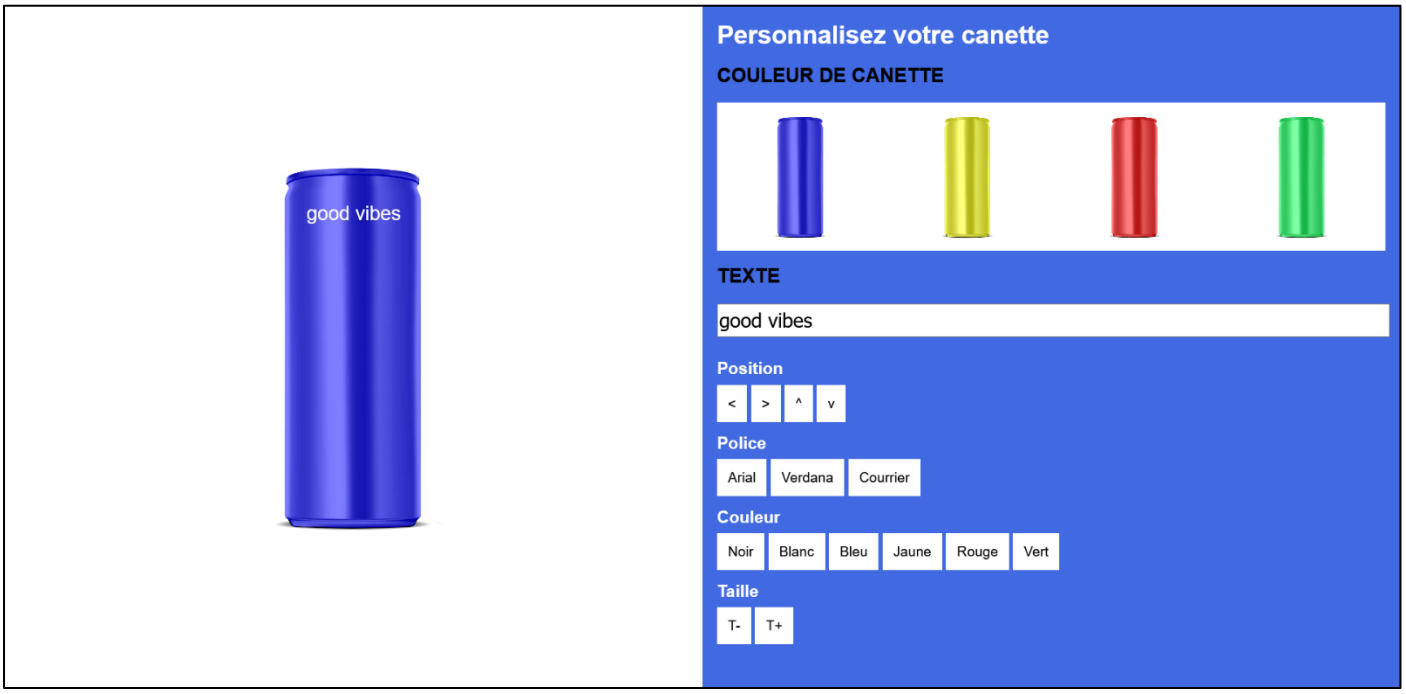
Evaluation
Cours Javascript - Alexandre Beaugrand

Module de personnalisation de canettes

L'objectif est de créer un module de personnalisation de cannettes en ligne, afin que les visiteurs du site puissent par la suite les commander. Voici l'allure finale du module :



Avant tout, veuillez télécharger les images du module

QUALITE DU RENDU ET AUTONOMIE – 1 pt

La notation prendra en compte, pour 1 point :

- l'organisation et le respect des consignes (intitulés et extensions des fichiers / intitulés des dossiers / etc.)
- l'allure du code (est ce lisible, bien indenté, toutes les balises bien fermées, etc.)

Cependant, en cas d'éventuelles aides de votre formateur (si vous le sollicitez), des points pourront être retirés.

PARTIE 1 : MISE EN PLACE – 1 pt

1) **Structure HTML**
   Créer un dossier **eval_votreNomDeFamille** contenant un fichier **index.html**.
   Y mettre la structure de base d'une page HTML5.

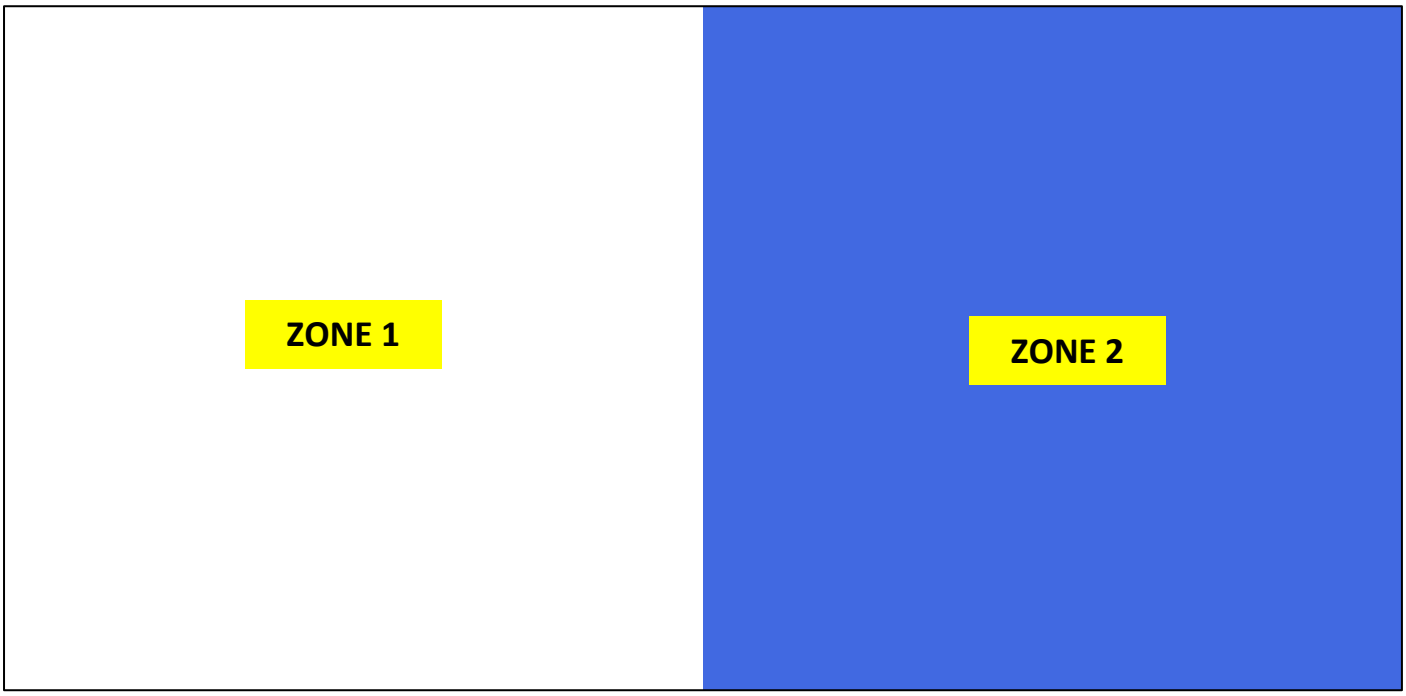
2) **Lien à la feuille de style**
   Créer un sous-dossier **/css** à l'intérieur du dossier **eval_votreNomDeFamille**. Dans ce sous-dossier **/css**, créer un fichier **styles.css**. Lier la page **index.html** au fichier **styles.css** avec la balise **link**.

3) **Lien au fichier javascript**
Créer un sous-dossier **/js** à l'intérieur du dossier **eval_votreNomDeFamille**. Dans ce sous-dossier **/js**, créer un fichier **main.js**. Lier la page **index.html** au fichier **main.js** avec la balise **script et son attribut defer**.

4) **Ajout du dossier d'images**
Ajoutez le dossier d'images téléchargées à l'intérieur du dossier **eval_votreNomDeFamille**. Vous obtenez donc un sous-dossier **/img** comprenant 4 images : can_bleue.png, can_jaune.png, can_rouge.png, et can_verte.png.

PARTIE 2 : STRUCTURE – 2 pts

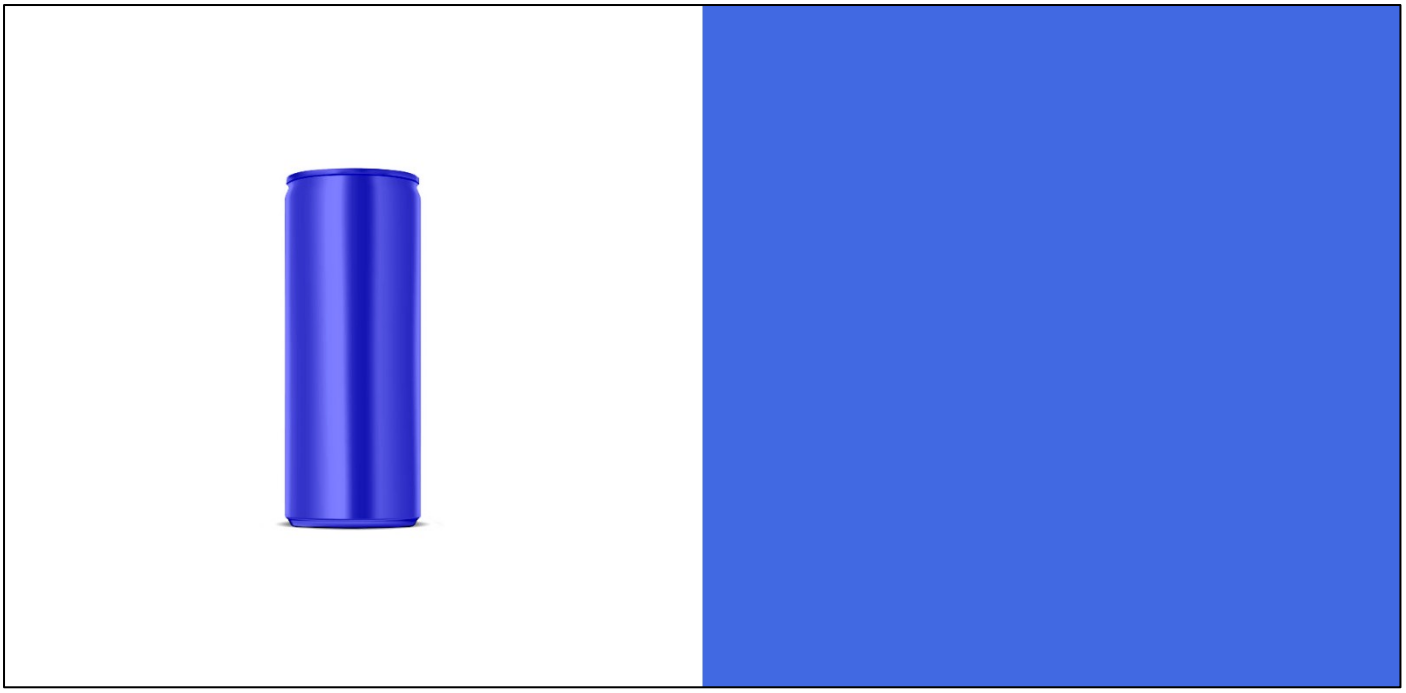
5) **Création des grandes zones de la page en HTML et en CSS**



- dans la balise **body**, créer un div d'id **zone1** : ce sera la colonne gauche du module
- dans la balise **body**, créer un div d'id **zone2 :** ce sera la colonne droite du module

6) **Mise en forme des zones**
Dans styles.css, faire en sorte d'obtenir le rendu ci-dessus :
- Tous les **divs** prennent un box-sizing :border-box ;
- **html** et **body** prennent un height de 100%, un box-sizing :border-box ; et pas de marges externes
- **body** est une flexbox, prend un font-family Arial et une taille de police de 1.2em
- La **zone1** prend une largeur de 50%, et un padding de 20px
- La **zone2** prend une largeur de 50%, un padding de 20px, et une couleur de fond royalblue

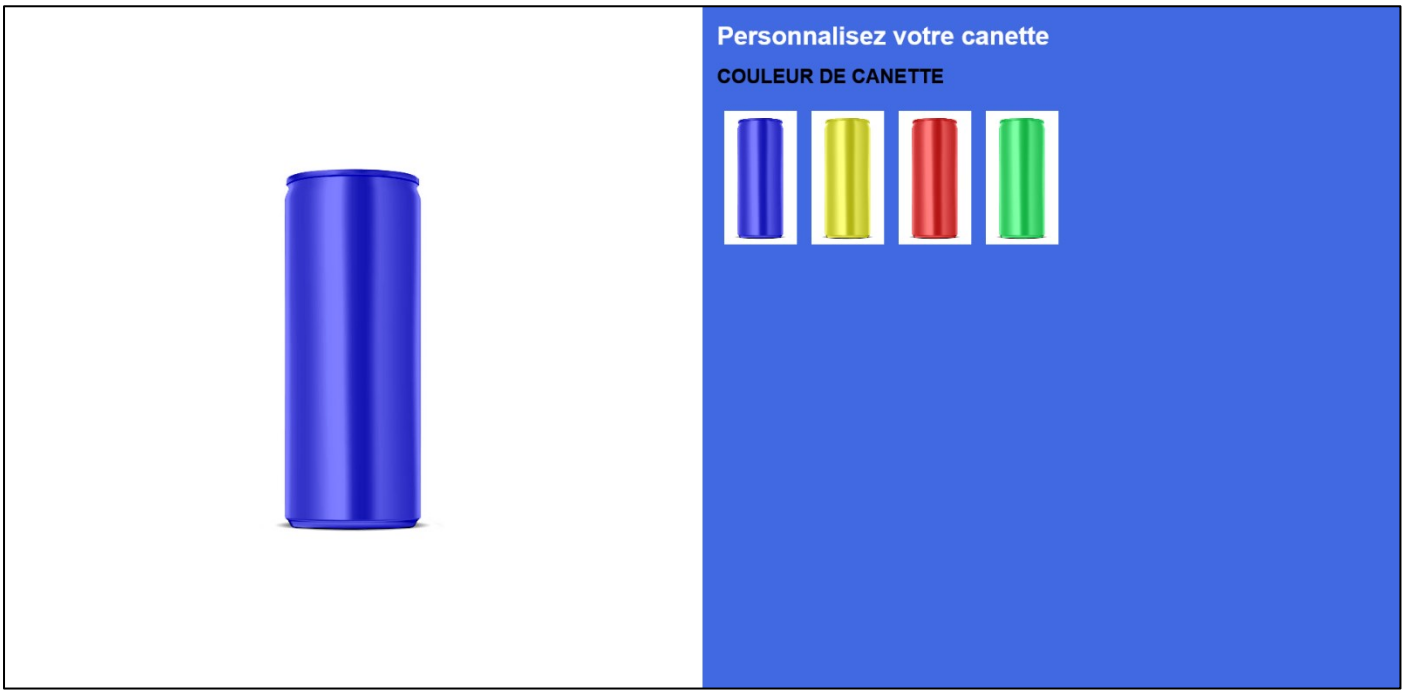
7) **Ajout d'une image de fond dans la zone 1**



- Récupérez dans une variable **zone1** l'élément d'id **zone1**, en utilisant un **querySelector()**
- Définir sur **zone1** (via sa propriété **style**) un **background** de valeur : `white url(img/can_bleue.png) no-repeat center center`

PARTIE 3 : PERSONNALISATION DU FOND – 3 pts

Nous allons maintenant faire en sorte que la canette affichée en fond soit personnalisable.



8) Partie HTML

- Ajoutez dans zone2 un **h1** « Personnalisez votre canette »
- Ajoutez dans zone2 un **h2** « Couleur de canette »
- Ajoutez dans zone2 un **div d'id choixCouleurs**

- Ajoutez dans ce div d'id choixCouleurs **4 img** ayant pour valeurs « img/can_bleue.png », « img/can_jaune.png », « img/can_rouge.png », et « img/can_verte.png ».

9) Partie CSS

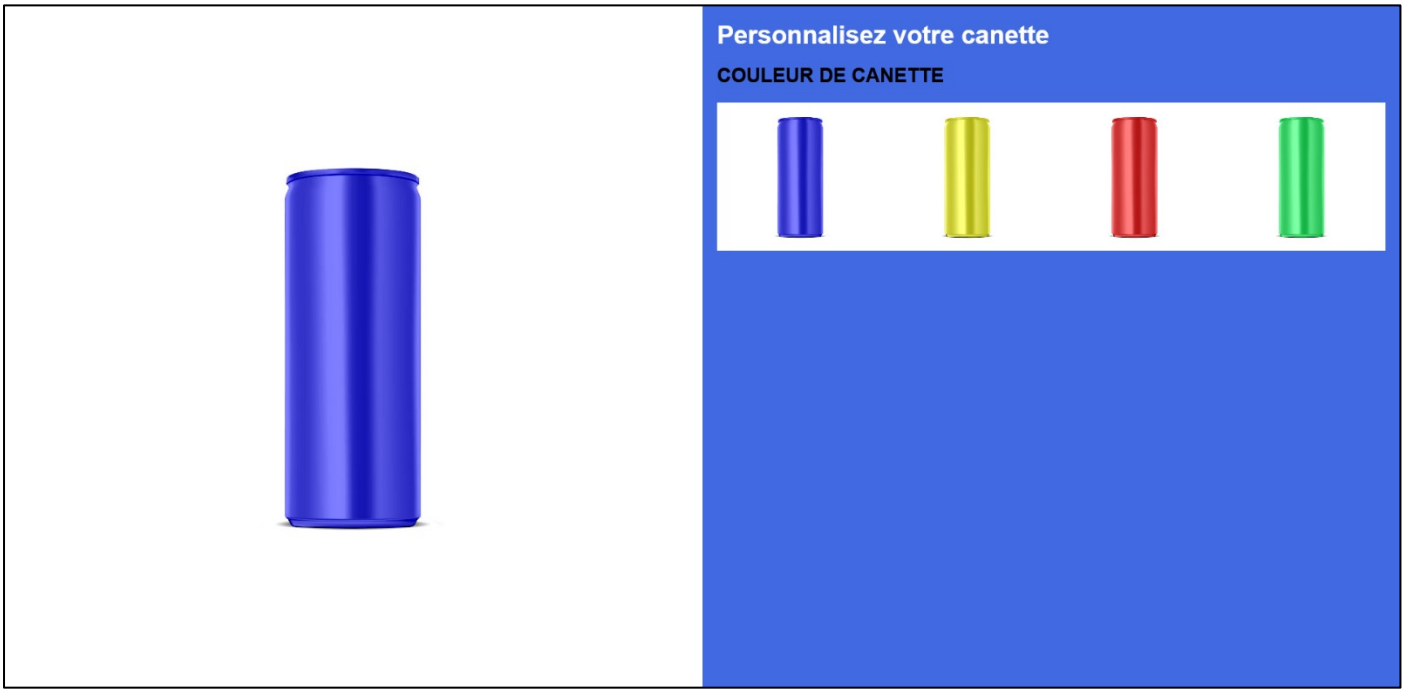
- Définir que les **img** du div d'id **choixCouleurs** ont une largeur de 100px, une marge externe de 10px, et un cursor:pointer; (pour que le pointeur de souris affiche une main).
- Les **h1** de zone2 n'ont pas de marges externes, sont blancs, et de taille 1.8em
- Les **h2** de zone 2 ont une marge externe haute de 20px, sont noirs, et ont une taille de 1.4em

10) Partie JS

- Récupérez dans une variable **vignettes** les **img contenues dans le div d'id choixCouleurs**, à l'aide d'un **querySelectorAll()**
- Bouclez sur cette liste de vignettes avec une boucle **forEach()**, qui à chaque boucle récupère une **vignette** (au singuler)
- Dans cette boucle, ajoutez un **écouteur d'évènement onclick** sur la vignette, liée à une fonction anonyme
- Dans la fonction anonyme appelée en cas de clic, créez une variable **srcVignette** qui contient la valeur de l'attribut src de la vignette (que l'on peut lire en utilisant un **getAttribute("src")** sur celle-ci)
- Toujours dans cette fonction, après la récupération de cette variable, utilisez-là pour redéfinir sur **zone1** (via sa propriété **style**) un **background** de valeur : `white url(${srcVignette}) no-repeat center center`
- Testez : si vous cliquez sur une vignette le fond doit changer en conséquence.

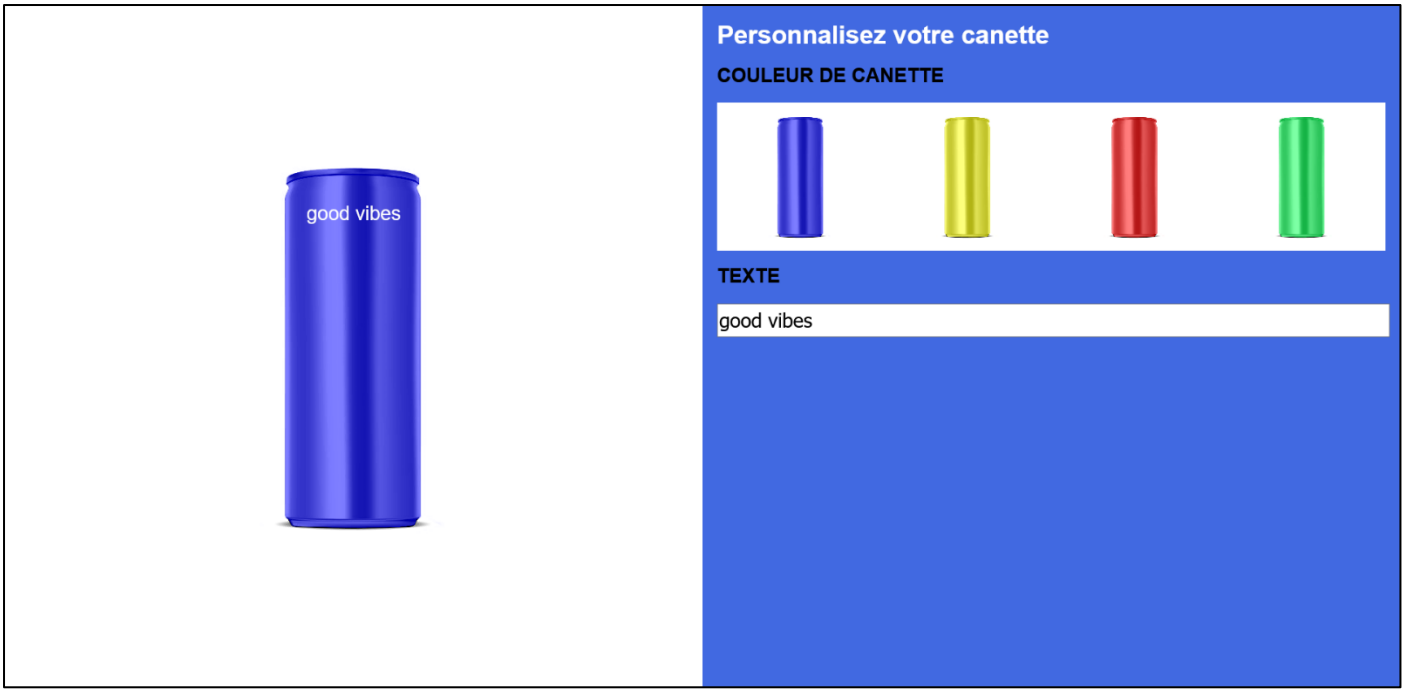
11) Modification CSS

- Améliorez l'aspect des vignettes en précisant que leur **conteneur d'id choixCouleur** à un fond blanc, et est une flexbox dont le contenu est utilise tout l'espace (justify-content:space-around)

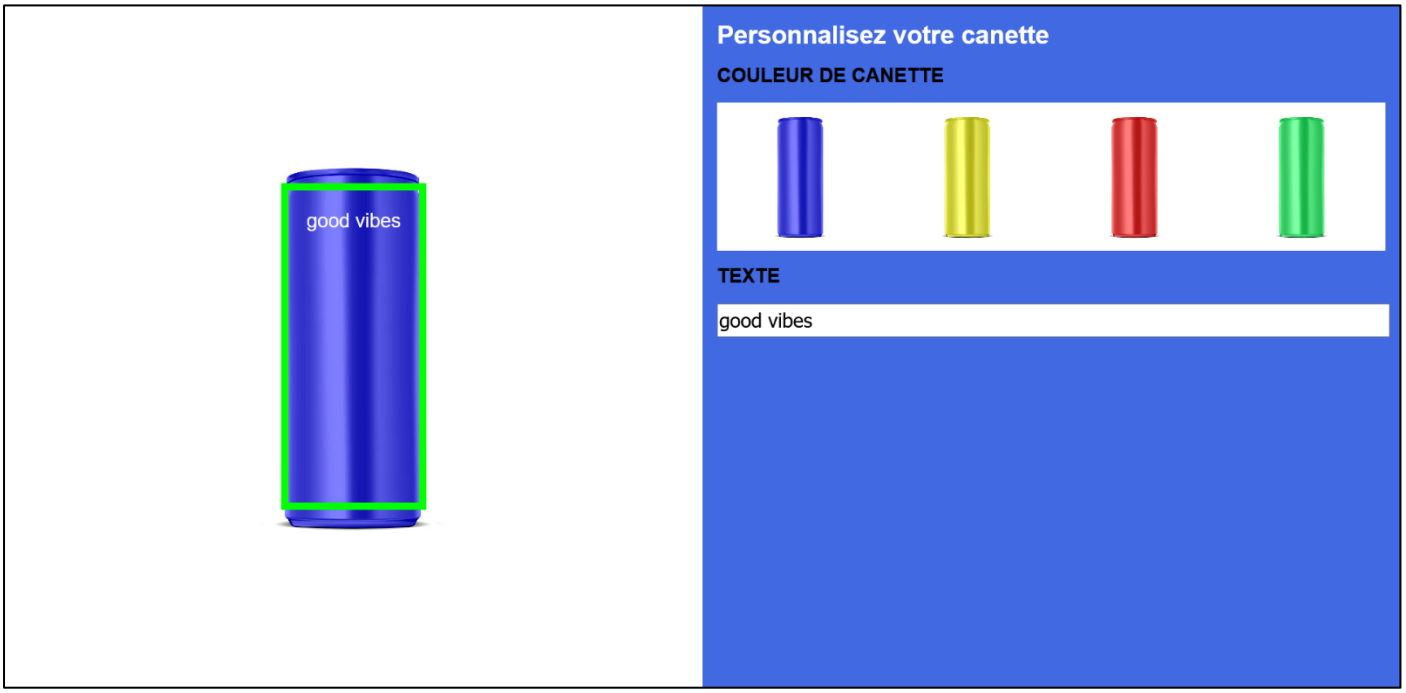


PARTIE 4 : PERSONNALISATION DU TEXTE – 3 pts

Nous allons maintenant faire en sorte qu'un texte définit dans la zone 2 s'affiche au-dessus de la canette dans la zone 1, telle que le montre l'image ci-dessous.



Même si elle est cachée en temps normal, on pourra provisoirement afficher une bordure autour de la zone de texte superposée à la canette, car c'est pratique pour programmer :



12) Partie HTML

- Ajouter en zone1 un div d'id **zoneTexte**, contenant le texte « good vibes »
- Ajouter en zone2 un h2 « TEXTE »
- Ajouter en zone2 un input **type text** d'id **ligne** avec un placeholder « votre texte » et une valeur par défaut « good vibes » (identique au contenu de zoneTexte)

13) Partie CSS

- Définir que l'élément d'id **ligne** à une largeur de 100%, une hauteur de 40px, et une taille de police de 1.4em
- Définir que la **zone1** est une flexbox dont le contenu est centré horizontalement (avec justify-content:center;) et verticalement (avec align-items:center;)

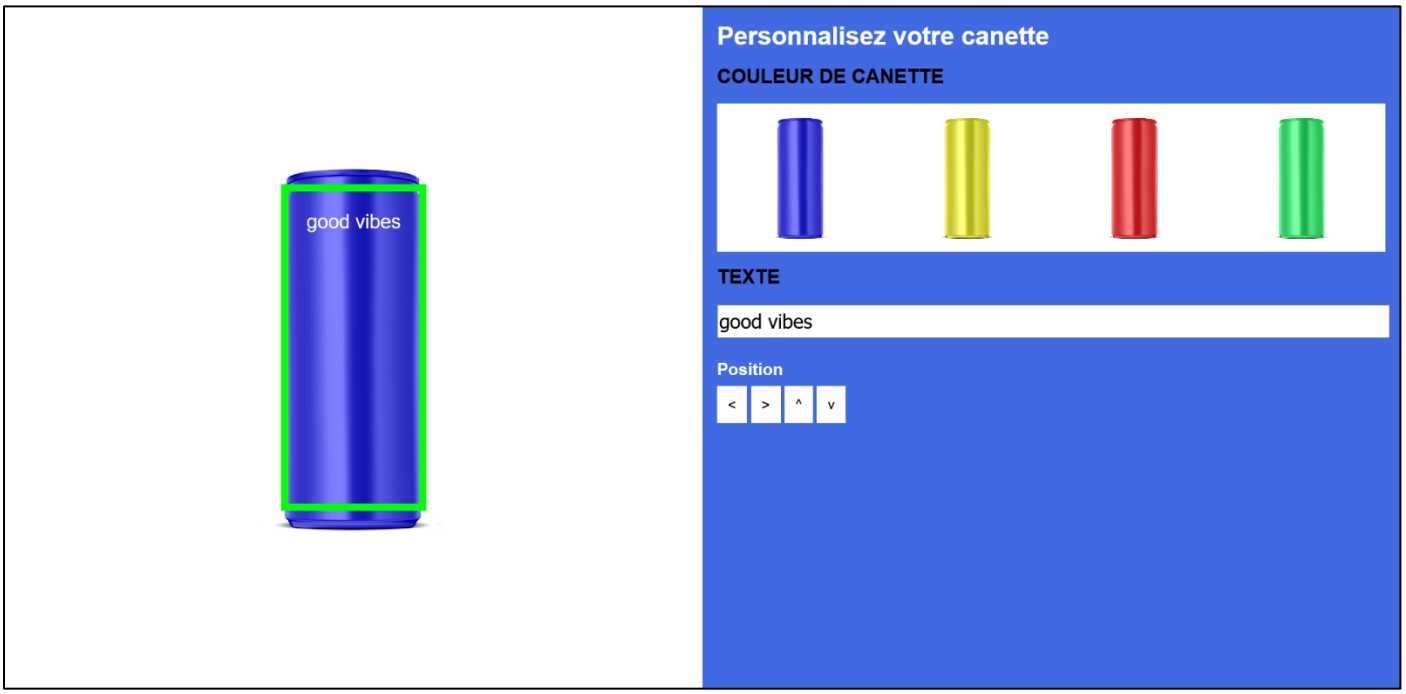
- Définir que l'élément d'id **zoneTexte** à une bordure (provisoire) de 10px solid lime, une largeur de 200px, une hauteur de 450px, une couleur de texte blanche, un alignement de texte au centre, un box-sizing :border-box; et un position:relative; et une taille de texte de 1.4em

14) Partie JS

- Récupérez dans une variable **zoneTexte** l'élément d'id **zoneTexte**, à l'aide d'un **querySelector()**
- Récupérez dans une variable **ligne** l'élément d'id **ligne**, à l'aide d'un **querySelector()**
- Placez un écouteur d'évènement **onkeyup** qui, en cas d'appui sur une touche du clavier, appelle automatiquement une fonction **majTexte()**
- Créer la fonction **majTexte()**, qui définit que le contenu (**innerHTML**) de **zoneTexte** correspond à la valeur (propriété **value**) de **ligne** (éventuellement encadrée de balises)
- **Appelez cette fonction** en fin de script, pour qu'elle soit executée une première fois dès que la page se charge, sans attendre que le visiteur appuie sur une touche clavier
- Dans le **HTML** vous pouvez ainsi **supprimer le contenu de l'élément d'id zoneTexte**. En principe il sera placé automatiquement par javascript et correspondra à la valeur de l'input de la zone 2

PARTIE 5 : PERSONNALISATION DE LA POSITION DU TEXTE – 3 pts

Nous allons maintenant faire en sorte qu'un texte soit déplaçable :



15) Partie HTML

- Ajoutez en zone2 un **h3** contenant le mot « Position »
- Ajoutez en zone2 4 ancres (**<a>**) d'ids **posGauche**, **posDroite**, **posHaut**, **posBas**. Elles ont toutes une classe **bt** et affiche des flèches (que vous pouvez simplement faire avec **>** / **<** / **^** / **v**)

16) Partie CSS

- Les **h3** de zone2 ont une marge haute de 30px et une taille de 1.2em
- Les éléments de class **bt** ne sont pas soulignés, prennent un fond blanc, une couleur de texte noire et une marge interne de 15px. Si on les **survole** leur fond devient orange

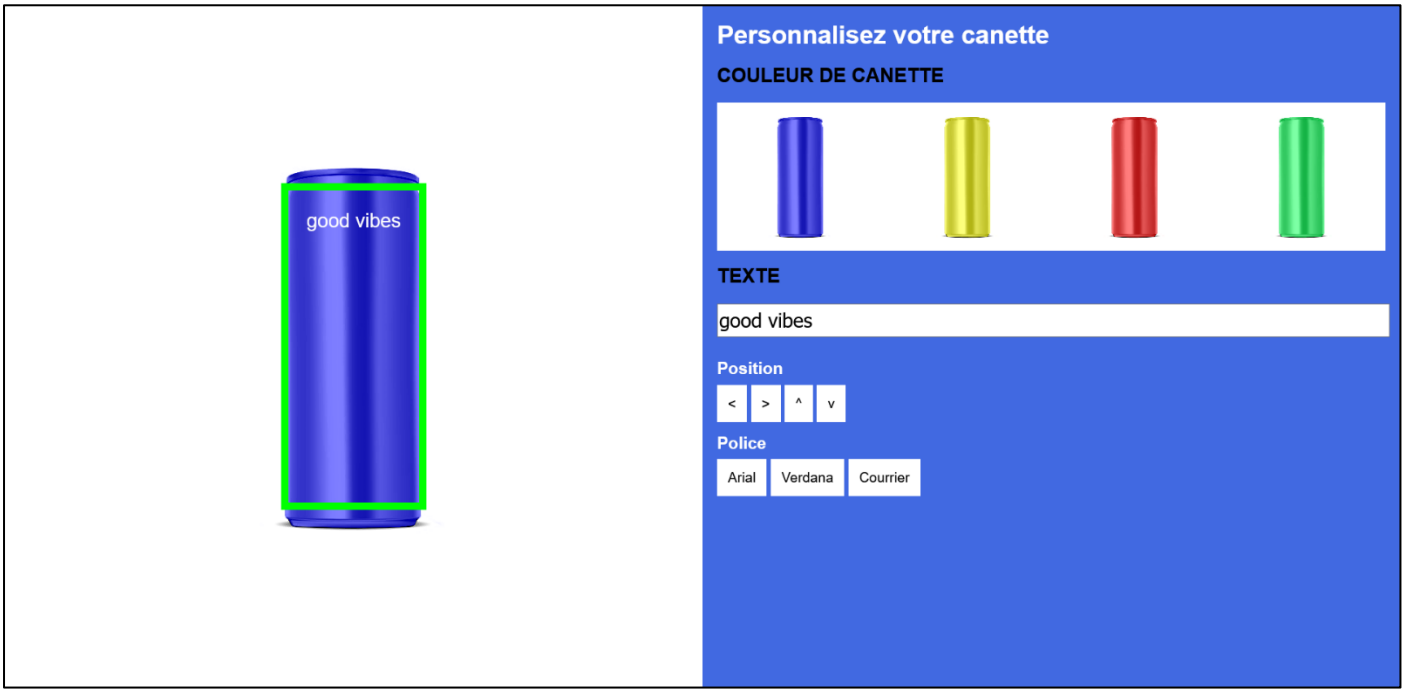
17) Partie JS

- Définir en haut de programme 2 variables **posX** et **posY** égales à **0**

- Définir en haut de programme une 3e variable **pas** égale à **10**
- Dans la fonction **majTexte()**, ajouter une ligne définissant que la propriété CSS **left** de **zoneTexte** (accessible par **zoneTexte.style.left**) est égale à **posX** suivie de **"px"**
- Dans la fonction **majTexte()**, ajouter une ligne définissant que la propriété CSS **top** de **zoneTexte** (accessible par **zoneTexte.style.top**) est égale à **posY** suivie de **"px"**
- En dehors de la fonction majTexte(), récupérez dans une variable **posHaut** l'élément d'id **posHaut**, à l'aide d'un **querySelector()**
- Placez sur **posHaut** un écouteur **onclick**, appelant une **fonction anonyme** en cas de clic
- Dans cette fonction anonyme, précisez que **posY** diminue du **pas** (posY-=pas;) puis appelez **majTexte()**.
- **Sur ce principe, activez les 3 autres boutons pour qu'ils augmentent ou diminuent posX ou posY avant d'appeler majTexte()**

PARTIE 6 : PERSONNALISATION DES POLICES – 3 pts

Nous allons maintenant faire en sorte que la police du texte soit personnalisable :



18) Partie HTML

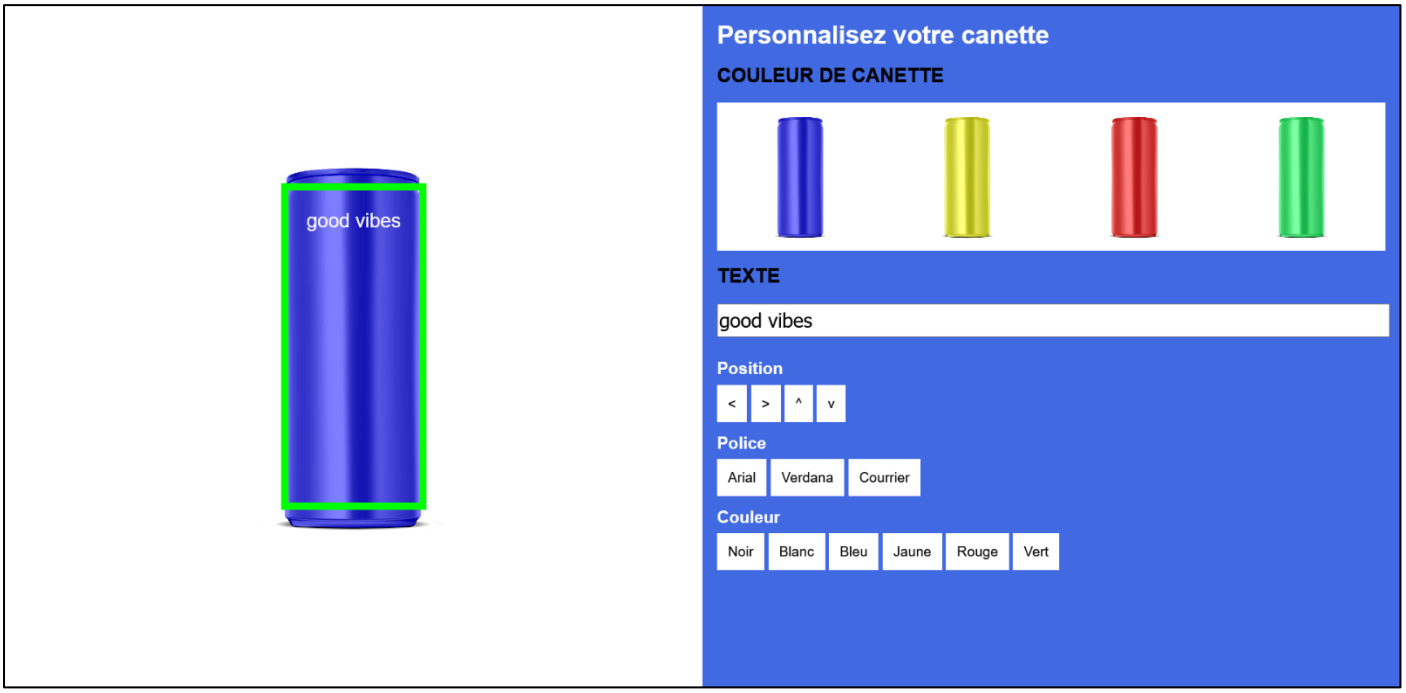
- Ajoutez en zone2 un **h3** contenant le mot « Police »
- Ajoutez en zone2 3 ancres (**<a>**) **avec pour chacune un attribut data-police="Arial", data-police="Verdana", ou data-police="Courrier"**. Elles ont toutes une classe **bt**

19) Partie JS

- Récupérez dans une variable **btsPolices** la liste des éléments ayant l'attribut **data-police**, à l'aide d'un **querySelectorAll(["data-police"])**
- Bouclez sur cette liste de **btspolices** avec une boucle **forEach()**, qui à chaque boucle récupère un **btpolice** (au singuler)
- Dans cette boucle, placez sur **btPolice** un écouteur **onclick**, appelant une **fonction anonyme** en cas de clic
- Dans cette fonction anonyme, créez une variable **police** qui contient la valeur de l'attribut **data-police** de **btPolice** (que l'on peut lire en utilisant un **getAttribute("data-police")** sur celui-ci)
- Toujours dans cette fonction anonyme, utilisez cette variable **police** pour redéfinir la police de zoneTexte (par un **zoneTexte.style.fontFamily=police;**).

PARTIE 7 : PERSONNALISATION DE LA COULEUR DU TEXTE – 2 pts

Nous allons maintenant faire en sorte que la couleur du texte soit personnalisable :



20) Partie HTML

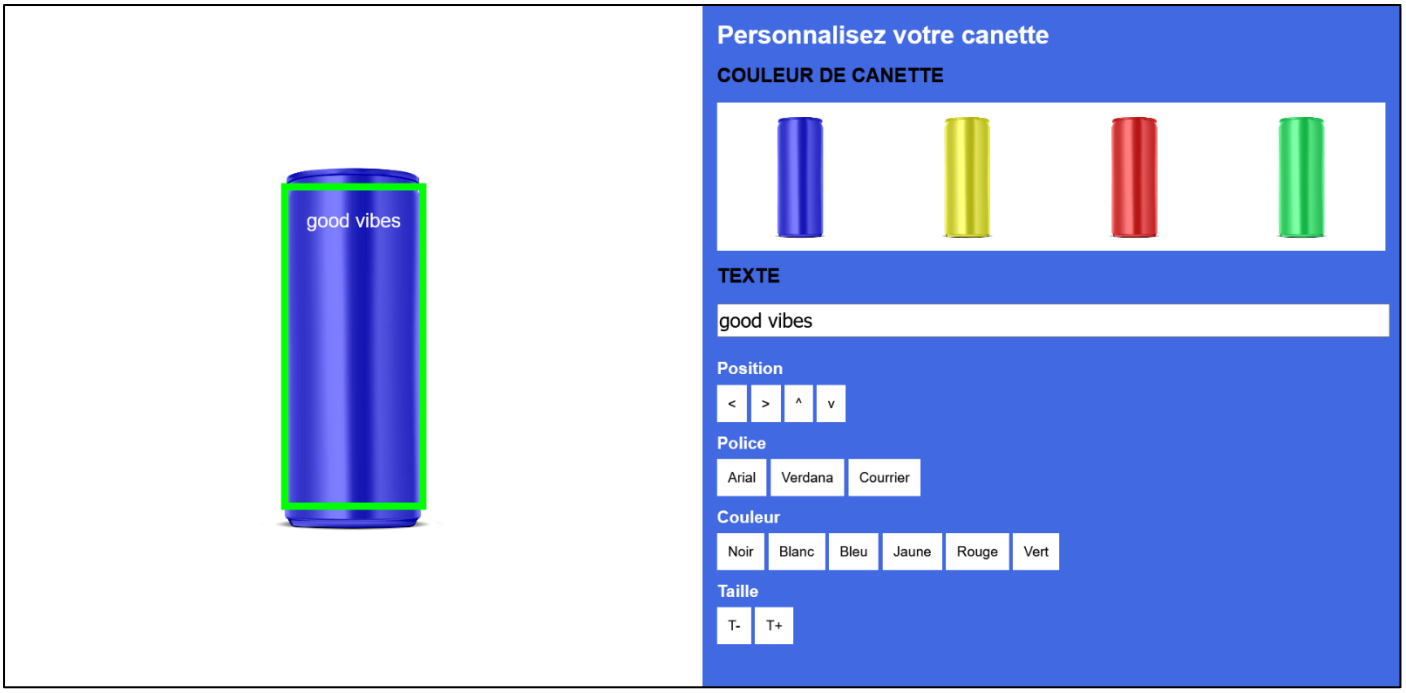
- Ajoutez en zone2 un **h3** contenant le mot « Couleur »
- Ajoutez en zone2 6 ancres (**<a>**) **avec pour chacune un attribut data-couleur="black", data-couleur="white", data-couleur="blue", data-couleur="yellow", data-couleur="red", ou data-couleur="green"**. Elles ont toutes une classe **bt**

21) Partie JS

- Récupérez dans une variable **btsCouleurs** la liste des éléments ayant l'attribut **data-couleur**, à l'aide d'un **querySelectorAll(["data-couleur"])**
- Bouclez sur cette liste de **btsCouleurs** avec une boucle **forEach()**, qui à chaque boucle récupère un **btCouleur** (au singuler)
- Dans cette boucle, placez sur **btCouleur** un écouteur **onclick**, appelant une **fonction anonyme** en cas de clic
- Dans cette fonction anonyme, créez une variable **couleur** qui contient la valeur de l'attribut **data-couleur** de **btCouleur** (que l'on peut lire en utilisant un **getAttribute("data-couleur")** sur celui-ci)
- Toujours dans cette fonction anonyme, utilisez cette variable **couleur** pour redéfinir la couleur de zoneTexte (par un **zoneTexte.style.color=couleur;**).

PARTIE 8 : PERSONNALISATION DE LA TAILLE DU TEXTE – 2 pts

Nous allons maintenant faire en sorte que la taille du texte soit personnalisable :



22) Partie HTML

- Ajoutez en zone2 un **h3** contenant le mot « Taille »
- Ajoutez en zone2 2 ancres (**<a>**) ayant les id **taillePetit** et **tailleGrand**. Elles ont toutes une classe **bt**

23) Partie JS

- Définir en haut de programme une variable **tailleTexte** égale à **1.4**
- Dans la fonction **majTexte()**, ajouter une ligne définissant que la propriété CSS **fontSize** de **zoneTexte** (accessible par **zoneTexte.style.fontSize**) est égale à **tailleTexte** suivie de **"em"**
- En dehors de la fonction majTexte(), récupérez dans une variable **taillePetit** l'élément d'id **taillePetit**, en utilisant un **querySelector()**
- Placez un écouteur **onclick** sur **taillePetit**, appelant une **fonction anonyme** en cas de clic
- Dans cette fonction anonyme, précisez que **tailleTexte** diminue de 0.2 (tailleTexte-=0.2;) puis appelez majTexte() ;
- Sur ce principe, activez le bouton **tailleGrand** qui augmente **tailleTexte** de **0.2** avant d'appeler **majTexte()**

24) Partie CSS

- Supprimez les **bordures** provisoire de l'élément d'id **zoneTexte**

FINALISATION

25) Finalisation

Sauvegardez tout et fermez votre éditeur. Puis, zippez le dossier eval_votreNomDeFamille , contenant tous vos éléments (sous windows : clic droit / envoyer vers / dossier compressé) et envoyez-moi ce fichier zip s'il vous plait. **Vérifiez svp avec moi que le livrable à été réceptionné et conservez bien vos fichiers.**

Bon courage !