**PROJET : YOGA-APP**

* Nous remarquons déjà que nos fichiers images sont numérotés… Nous comprendrons plus tard pourquoi c’est ainsi

Faut déjà savoir que dans ce projet en réalité la page ne change jamais… C’est juste nous qui injectons des choses dans la page (Ce sont des vues)

//On pointe d'abord le main (Vu que c'est lui qui va contenir nos cartes)

const main = document.**querySelector**('main');

//On crée une variable qui va stocker tous nos exercies

//En sachant que notre carte elle a une image, un nombre de minute qui leur soit adosser

let exerciceArray = [

]

Donc on va créer un tableau d’objets

let exerciceArray = [

  {

    pic: 1,

    min: 1

  },

  {

    pic: 2,

    min: 1

  },

  {

    pic: 3,

    min: 1

  },

  {

    pic: 4,

    min: 1

  },

  {

    pic: 5,

    min: 1

  },

  {

    pic: 6,

    min: 1

  },

  {

    pic: 7,

    min: 1

  },

  {

    pic: 8,

    min: 1

  },

  {

    pic: 9,

    min: 1

  },

  {

    pic: 10,

    min: 1

  },

]

**On stocke nos éléments dans un tableau car avec celui-ci ils seront plus facile à manipuler (Suppression, déplacement, etc…) en faite tout ce qu’on va faire sera basé sur ce tableau**

* **On se crée une classe (Qui va être le générateur d'exercices)** **C'est cette classe qui va gérer le passage d'un exercice à l'autre**

  class Exercice  {}

* **On crée un objet 'utils' dans lequel on va mettre toute nos fonction qui vont être utiles dans notre projet**

  const utils = {

  }

* **On crée un objet 'page' dans lequel on va mettre toutes nos pages (Vues)**

  const page = {

  }

* **Petit exemple d’utilisation de la fonction Lobby (Pour comprendre un peu la logique de notre code)**

 //On crée un objet 'page' dans lequel on va mettre toutes nos pages (Vues)

  const page = {

**lobby**: function(){

        document.**querySelector**('h1').innerHTML = "Paramétrage <i id='reboot' class='fas fa-undo'></i>"

      }

  }

  page.**lobby**();

* **On peut ensuite s’ajouter le bouton ‘’commencer’’**

  const page = {

**lobby**: function(){

        document.**querySelector**('h1').innerHTML = "Paramétrage <i id='reboot' class='fas fa-undo'></i>"

        main.innerHTML = "Exercices"

        document.**querySelector**(".btn-container").innerHTML = "<button id='start'>Commencer<i class='far fa-play-circle'></i></button>"

      }

  }

  page.**lobby**();

* **Si on veut éviter de se répéter tout le code précédent pour chaque vue, on peut se créer une fonction et mettre ce code-là à l’intérieur**

**NB : Ceci va nous permettre de changer les informations affichées à l’écran sans avoir besoin de réécrire à chaque fois le code mais on pourra le faire juste en appelant la fonction ‘pageContent’**

const utils = {

**pageContent**: function (title, content, btn) {

    document.**querySelector**("h1").innerHTML = title;

    main.innerHTML = content;

    document.**querySelector**(".btn-container").innerHTML = btn;

  },

};

* **Utilisation de la méthode pageContent dans les pages :**

const page = {

**lobby**: function () {

    utils.**pageContent**(

      "Paramétrage <i id='reboot' class='fas fa-undo'></i>",

      "Exercices",

      "<button id='start'>Commencer<i class='far fa-play-circle'></i></button>"

    )},

**routine**: function () {

    utils.**pageContent**(

      "Routine", "Exercice avec chrono", null

    )

  },

**finish**: function () {

    utils.**pageContent**(

      "C'est terminé !",

      "<button id='start'>Recommencer</button>",

      "<button id='reboot' class='btn-reboot'>Rénitialiser<i class='fas fa-times-cicle'></i></button>"

    )

  },

};

page.**finish**();

1. **Maintenant qu’on sait faire ça, on va se coder la logique d’une carte**
2. **Dans la fonction** lobby(), on va se créer un map(Qu’on va stocker dans une variable et qu’on va donner en paramètre à notre fonction pageContent

**lobby**: function () {

    let mapArray = exerciceArray

    .**map**(

      (exo)=>

      `

      <li>

      <div class="card-header">

        <input type="number" id="${exo.pic} min="1" max="10" value=${exo.min}>

        <span>min</span>

        </div>

      </li>

      `

    ).**join**('');

1. On passe ensuite la variable **mapArray** comme second paramètre de la fonction **pageContent** dans le **lobby()** pour signifier que le contenu de cette page là est le résultat du map **(Qui a été stocker dan le mapArray)**

const page = {

**lobby**: function () {

    let mapArray = exerciceArray

    .**map**(

      (exo)=>

      `

      <li>

      <div class="card-header">

        <input type="number" id="${exo.pic} min="1" max="10" value=${exo.min}>

        <span>min</span>

        </div>

      </li>

      `

    ).**join**('');

    utils.**pageContent**(

      "Paramétrage <i id='reboot' class='fas fa-undo'></i>",

      mapArray,

      "<button id='start'>Commencer<i class='far fa-play-circle'></i></button>"

    )},

1. **AJOUT DES IMAGES À NOS CARTE**

**Nous devons nous assurez que l’ajout des images soit dynamique (Donc le src va varier en fonction de l’image sur laquelle nous sommes)**

**lobby**: function () {

    let mapArray = exerciceArray

    .**map**(

      (exo)=>

      `

      <li>

      <div class="card-header">

        <input type="number" id="${exo.pic} min="1" max="10" value=${exo.min}>

        <span>min</span>

        </div>

        <img src="./img/${exo.pic}.png" />

      </li>

      `

1. **AJOUT DES DEUX BOUTONS DU BAS (Pour déplacer les cartes et le bouton pour supprimer une carte)**

**lobby**: function () {

    let mapArray = exerciceArray

    .**map**(

      (exo)=>

      `

      <li>

      <div class="card-header">

        <input type="number" id="${exo.pic} min="1" max="10" value=${exo.min}>

        <span>min</span>

        </div>

        <img src="./img/${exo.pic}.png" />

        <i class="fas fa-arrow-alt-circle-left arrow" data-pic=${exo.pic}></i>

        <i class="fas fa-times-circle deleteBtn" data-pic=${exo.pic}></i>

      </li>

      `

**Maintenant qu’on a fini de créer la logique de nos cartes, maintenant on doit pouvoir les manipuler(Ajouter des évènements)**

1. **Création de la fonction qui va nous permettre de gérer (récupérer) le nombre de minutes choisi par l’utilisateur**
2. **Création et implémentation de la fonction handleEventMinutes() dans l’objet utils{}**

**handleEventMinutes**: function(){

    document.**querySelectorAll**('input[type="number"]').

**forEach**((input)=>{

      input.**addEventListener**("input", (event)=>{

        console.**log**(event);

      })

    })

  }

**NB : Le console.log() que nous avons mis ici nous permet de visualiser (Pour le moment) le résultat de notre fonction**

* **Nous allons appeler cette fonction dans le lobby () après le pageContent() (Pour s’assurer que toutes les cartes soient chargées avant de commencer à les utiliser**
* **Nous pouvons maintenant poser une condition dans la fonction pour vérifier s’il va trouver l’élément sur lequel on agit et afficher un message dans la console**

**handleEventMinutes**: function(){

    document.**querySelectorAll**('input[type="number"]').**forEach**((input)=>{

      input.**addEventListener**("input", (e)=>{

        exerciceArray.**map**((exo)=>{

          console.**log**("Test");

          if (exo.pic == e.target.id) {

            console.**log**("Yes");

          }

        })

      })

    })

  }

**handleEventMinutes**: function(){

    document.**querySelectorAll**('input[type="number"]').**forEach**((input)=>{

      input.**addEventListener**("input", (e)=>{

        exerciceArray.**map**((exo)=>{

          if (exo.pic == e.target.id) {

            exo.min = **parseInt**(e.target.value);

            console.**log**(exerciceArray)

          }

        })

      })

    })

  }

1. **Déplacement des éléments**

* **On se crée une fonction qui va nous permettre de gérer ce déplacement**
* **On doit pouvoir identifier la position de l’élément sur lequel on a cliqué**

**handleEventArrow**: function(){

    document.**querySelectorAll**(".arrow").**forEach**((arrow)=>{

      arrow.**addEventListener**('click', (e)=>{

        let position = 0; //Variable qui va nous permettre de trouver la positon de l

        exerciceArray.**map**((exo)=>{

          if (exo.pic == e.target.dataset.pic) {

            [exerciceArray[0], exerciceArray[1]] = [exerciceArray[1], exerciceArray[0]]

             console.**log**(exerciceArray);

          } else{

            position++;

            console.**log**(position);

          }

        })

      })

    })

**On a réussi ici à intervertir les éléments aux positions 0 et 1 en utilisant la méthode :**

 [exerciceArray[0], exerciceArray[1]] = [exerciceArray[1], exerciceArray[0]]

* **Mais ce que nous voulons faire ici c’est de rendre cet échange dynamique**

**handleEventArrow**: function(){

    document.**querySelectorAll**(".arrow").**forEach**((arrow)=>{

      arrow.**addEventListener**('click', (e)=>{

        let position = 0; //Variable qui va nous permettre de trouver la positon de l

        exerciceArray.**map**((exo)=>{

          if (exo.pic == e.target.dataset.pic) {

            [exerciceArray[position], exerciceArray[position -1 ]] = [exerciceArray[position -1], exerciceArray[position]]

             console.**log**(exerciceArray);

          } else{

            position++;

            console.**log**(position);

          }

        })

      })

    })

* **Maintenant on pose une condition pour s’assurer que la position ne soit jamais inférieure à 0**

**Et on rappelle la fonction page.lobby() pour reafficher les éléments**

**handleEventArrow**: function(){

    document.**querySelectorAll**(".arrow").**forEach**((arrow)=>{

      arrow.**addEventListener**('click', (e)=>{

        let position = 0; //Variable qui va nous permettre de trouver la positon de l

        exerciceArray.**map**((exo)=>{

          if (exo.pic == e.target.dataset.pic && position !== 0) {

            [exerciceArray[position], exerciceArray[position -1 ]] = [exerciceArray[position -1], exerciceArray[position]]

             page.**lobby**();

          } else{

            position++;

          }

        })

      })

    })

  }

};

1. **SUPPRESSION DES EXERCICES**