Créons des classes

Nous avons vu comment créer des classes en JS.

Vous allez maintenant devoir créer les votres. Chaque exercice est indépendant (sauf indication contraire).

Ils peuvent donc avoir chacun leur propre HTML et/ou CSS. Mais chacune des classes doit être dans un fichier à part importé par module.

Balles rebondissantes

Créons un canvas qui au clique fera apparaître une balle rebondissant sur les bords du canvas, de taille, de couleur, de vitesse et à une position aléatoire.

Chaque balles est créer à partir du classe, qui contiendra les propriétés susmentionné.

Elles auront aussi une méthode permettant de gérer la leur déplacement et une autre gérant leur dessin dans le canvas.

le script général ne s'occupera que de la création des balles et l'appelle de leur méthode dans une animation.

Calculatrice

Créer une calculatrice permettant les opérations les plus simple.

L'objet doit générer une calculatrice comportant les boutons pour les chiffres et les 4 opérations basiques (addition, soustraction, division et multiplication). Elle doit aussi contenir une touche "égale", une touche "virgule", une touche "delete" et une touche "clear"

La zone d'affichage doit être sur deux lignes, celle du bas affichera le nombre actuel, celle du haut affichera le nombre précédent et l'opération choisi.

Pour des raisons de lisibilité, il serait bon que la calculatrice affiche un espace entre chaque suite de 3 chiffres.

Slider

Nous avons vu que notre Slider construit en objet fonctionnait bien jusqu'à ce qu'on lui demande d'en créer un second. Ils entrent alors en conflit.

Avec une classe nous allons pouvoir arranger cela et avoir autant de Slider que voulu.

Transformez le slider en classe et créez deux slider différents.

Easy Dom

Créons une classe "EasyDom" qui aura pour rôle de faciliter notre manipulation du DOM.

Elle contiendra les méthodes suivantes :

- "**tag**" Cette méthode facilitera la création d'élément HTML, elle prendra en argument :
    - Un string contenant le nom d'une balise HTML

    - Optionnellement un objet pouvant contenir les attributs ou texte que l'on souhaite ajouter.
- "**select**" Cette méthode facilitera le querySelector en effectuant à la fois le **querySelector** et le **querySelectorAll**, elle prendra en argument :
    - Un string contenant le selecteur css souhaité.
    - Optionnellement un parent autre que le document.
- "**event**" Cette méthode facilitera l'ajout d'évènement en pouvant accepter une collection d'élément html comme argument, elle prendra en argument :
    - l'élément HTML (ou la collection) sur lequel on souhaite ajouter un évènement.
    - l'évènement sous forme de string.
    - la fonction à lancer lors de l'évènement.

Exemple d'utilisation :

```
const ed = new EasyDom();
// Je crée une div avec 3 classes, un id et du html.
const div = ed.tag("div", {
    class: "truc bidule machin",
    id: "chaussette",
    html: "test"
    });
// Je récupère un élément html
const span = ed.select("span#specialSpan");
// Je récupère une collection d'élément html.
const spans = ed.select("span");
// J'ajoute un évènement sur une collection d'élément html
ed.event(spans, "click", (e)=>console.log("collection"));
// J'ajoute un évènement sur un seul élément html.
ed.event(span, "click", (e)=>console.log("unique"));
```

Bonus

Faites une seconde version du slider qui hérite de EasyDom et se sert de ses nouvelles méthodes.