```
import {Chronometre} from './chronometre.js'

/**
 * Gestion de l'affichage des Chronomètres (du DOM)
 * @class Document
 */
export class Document {
  /** @type {Map} */
  #chronos
  /** @type {Element} */
  #container
  /** @type {Element} */
  #boutonAjouterChrono
  /** @type {HTMLTemplateElement} */
  #chronoVide

  constructor() {
    this.#container = document.querySelector('#container');
    this.#boutonAjouterChrono = document.querySelector('#ajouter');
    this.#chronoVide = document.querySelector('#horloge-vide');
    this.#chronos = new Map();
  }

  /**
   * Configure le panneau de controle d'un Chronometre et lui
   * assigne les eventListener
   * @param {string}idChrono
   */
  #prepareControle(idChrono) {
    const ccc = document.querySelector(`#${idChrono} .Clear`)
    const sss = document.querySelector(`#${idChrono} .Start`)
    const ddd = document.querySelector(`#${idChrono} .Delete`)

    sss.textContent = 'Start'
    ccc.textContent = 'Clear'
    ddd.textContent = 'Delete'

    sss.addEventListener('click', this.#toogle.bind(this, idChrono))
    ccc.addEventListener('click', this.#chronos.get(idChrono).clearChrono.bind(this.#chronos.get(idChrono)))
    ddd.addEventListener('click', this.#supprimeChrono.bind(this, idChrono))
  }

  /**
   * Supprime un chronometre du DOM et l'éventuel timer
   * @param {string}idChrono
   */
  #supprimeChrono(idChrono) {
    this.#chronos.get(idChrono).supprimer.call(this.#chronos.get(idChrono))
    document.querySelector(`#${idChrono}`).remove()
    this.#chronos.delete(idChrono)
  }

  /**
   * Echange le comportement entre Start/Pause
   */
  #toogle(idChrono) {
    const bouton = document.querySelector(`#${idChrono} .control button:first-child`)
    let handler, classSource;
    let classDestination;

    if (bouton.classList.contains('Start')) {
      handler = 'demarrer'; // nom de la fonction
      classSource = 'Start'
      classDestination = 'Pause'
    }
    else {
      handler = 'pause';  // nom de la fonction
      classSource = 'Pause'
      classDestination = 'Start'
    }

    /* Appelle une fonction dont le nom est la chaine de caractères handler*/
    this.#chronos.get(idChrono)[handler]();

    bouton.classList.remove(`${classSource}`);
    bouton.classList.add(`${classDestination}`);
```

```
    bouton.textContent = `${classDestination}`;
  }

  /**
   * @return {Element}
   */
  get container() {
    return this.#container
  }

  /**
   * @return {Element}
   */
  get boutonAjouterChrono() {
    return this.#boutonAjouterChrono
  }

  /**
   * @return {HTMLTemplateElement}
   */
  get chronoVide() {
    return this.#chronoVide
  }

  /**
   * Effectue la mise à jour de l'affichage pour un
   * @param {string}idChrono
   */
  renderChrono(idChrono) {
    const element = document.querySelector(`#${idChrono}`);
    const [minute, seconde, dixieme] = this.#chronos.get(idChrono).horaireFromDixieme();

    element.querySelector('.minute').textContent = minute.toString().padStart(2, '0');
    element.querySelector('.seconde').textContent = seconde.toString().padStart(2, '0');
    element.querySelector('.dixieme').textContent = dixieme.toString();
  }

  /**
   * Ajoute l'affichage d'un Chronomètre dans le DOM au niveau
   * de l'élément HTML .container.
   */
  ajouterChrono() {
    const newChrono = new Chronometre(this.renderChrono.bind(this));
    const chronoElement = this.chronoVide.content.cloneNode(true);
    chronoElement.querySelector('.chronometre').id = newChrono.idChrono
    this.container.append(chronoElement);
    this.#chronos.set(newChrono.idChrono, newChrono);
    this.#prepareControle(newChrono.idChrono)
    this.renderChrono(newChrono.idChrono);
  }
}
```