1 Animation via l'API Canvas

Veuillez adapter une application web permettant l'affichage de différents scintillements de formes selon la demande de l'utilisateur.

Le code a adapter se trouve dans le dossier /demo/canvas du repo associé au cours.

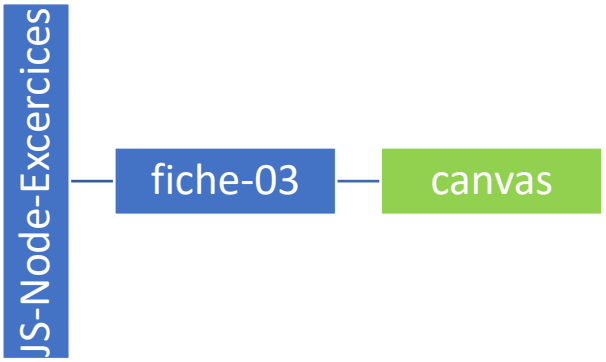
Veuillez retravailler l'animation pour que :

- elle s'arrête ou démarre lors d'un click d'un utilisateur ;
- les carrés grandissent ou rapetississent au clic sur la touche + ou la touche - ;
- lors d'un clic droit de la souris, la couleur change de manière aléatoire.

- *Vous pouvez utiliser le type d'événement « contextmenu » pour gérer les clics droits.*
  Plus d'info : https://developer.mozilla.org/en-US/docs/Web/API/Element/contextmenu_event
- *Vous pouvez utiliser le type d'événement « keydown » pour gérer l'appui sur les touches + ou -*
  Plus d'info : https://developer.mozilla.org/en-US/docs/Web/API/Document/keydown_event
- *Si vous avez besoin de générer un nombre aléatoire de 0 à 255 :*

```
Math.floor(Math.random() * 256); // [0,255]
```

Le code de votre application web doit se retrouver dans ce dossier (en vert) de votre repository local et de votre web repository (**JS-Node-Exercices**).



2 Animation via Anime.js

Vous allez réaliser une application web permettant d'afficher un nuage de mots en mouvement.

Au passage sur un mot, il doit s'agrandir. Au clic sur un mot, faites disparaître tous les autres mots et ajoutez une autre action de votre choix.

Au passage sur un mot, il doit s'agrandir. Au clic sur un mot, faites disparaître tous les autres mots et ajoutez une description associée à ce mot.

Afin de réaliser cet exercice nous vous proposons ces contraintes d'implémentation :

- Utilisez cet **Array** pour créer vos mots de manière dynamique (utilisez soit la propriété **title** ou l'**url**)

```
const animationLibraries = [
  {
    title: "Anime.js",
    url: "https://animejs.com/",
    description: `Anime.js (/ˈæn.ə.meɪ/) is a lightweight JavaScript animation
 library with a simple, yet powerful API.
It works with CSS properties, SVG, DOM attributes and JavaScript Objects.`,
  },
  {
    title: "Three.js",
    url: "https://threejs.org/",
    description:
      "Three.js is a cross-
browser JavaScript library and application programming interface used to creat
e and display animated 3D computer graphics in a web browser using WebGL.",
  },
  {
    title: "Phaser.io",
    url: "https://phaser.io/",
    description:
      "Phaser is a fast, free, and fun open source HTML5 game framework that o
ffers WebGL and Canvas rendering across desktop and mobile web browsers. ",
  },
  {
    title: "GSAP",
    url: "https://greensock.com/gsap/",
    description:
      "GSAP is a JavaScript library for building high-
performance animations that work in every major browser. Animate CSS, SVG, can
vas, React, Vue, WebGL, colors, strings, motion paths, generic objects... anyt
hing JavaScript can touch!",
  },
```

```
  {
    title: "Mo.js",
    url: "https://mojs.github.io/",
    description:
      "mo · js is a javascript motion graphics library that is a fast, retina
ready, modular and open source. In comparison to other libraries, it have a di
fferent syntax and code animation structure approach. The declarative API prov
ides you a complete control over the animation, making it customizable with ea
se.",
  },
  {
    title: "Velocity.js",
    url: "http://velocityjs.org/",
    description: `Velocity is an animation engine with the same API as jQuery'
s $.animate(). It works with and without jQuery. It's incredibly fast, and it
features color animation, transforms, loops, easings, SVG support, and scrolli
ng. It is the best of jQuery and CSS transitions combined.`,
  },
  {
    title: "AniJS",
    url: "https://anijs.github.io/",
    description: "A Library to Raise your Web Design without Coding",
  },
  {
    title: "vivus",
    url: "https://maxwellito.github.io/vivus/",
    description:
      "Vivus is a lightweight JavaScript class (with no dependencies) that all
ows you to animate SVGs, giving them the appearence of being drawn. There are
a variety of different animations available, as well as the option to create a
 custom script to draw your SVG in whatever way you like.",
  },
  {
    title: "ScrollReveal",
    url: "https://scrollrevealjs.org/",
    description:
      "ScrollReveal is a JavaScript library for easily animating elements as t
hey enter/leave the viewport. It was designed to be robust and flexible, but h
opefully you'll be surprised below at how easy it is to pick up.",
  },
  {
    title: "Typed.js",
    url: "https://mattboldt.com/demos/typed-js/",
    description:
      "Typed.js is a library that types. Enter in any string, and watch it typ
e at the speed you've set, backspace what it's typed, and begin a new sentence
 for however many strings you've set.",
  },
```

```
];
```

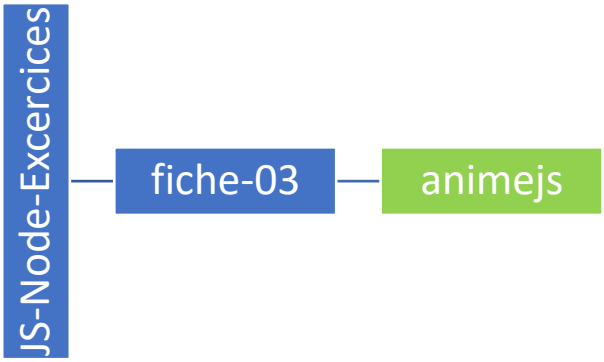
- Vous pouvez bien sûr repartir de la démonstration /demo/animejs. Néanmoins, afin d'apprendre à intégrer une librairie externe, nous vous conseillons de partir d'une page blanche et de suivre les instructions qui suivent.

- Apprenez à intégrer une librairie JS externe. Pour ce faire, utilisez la dernière version de la librairie anime.js sans CDN mais localement :
  - Télécharger la dernière version : https://github.com/juliangarnier/anime/archive/master.zip
    La librairie se trouve dans le répertoire anime-master/lib
    Vous pouvez utiliser la version de production : anime.min.js.
  - Intégrer cette librairie à votre application :

```
<script src="anime.min.js"></script>
```

- *Pour comprendre comment utiliser la librairie anime.js, inspirez-vous de la démo* /demo/animejs ainsi que de https://animejs.com/documentation/
- *Pour faire bouger des mots, il est utile d'utiliser des éléments html dont le comportement d'affichage est celui attendu.*
  On pourrait utiliser des spans (comportement d'affichage de type « inline »), des paragraphes (comportement d'affichage de type « block »), des div…
  Dans la démo, on a utilisé des div avec « inline-block » comme comportement d'affichage, via l'utilisation de la classe Bootstrap ***d-inline-block****. Ca nous permet d'avoir des blocs qui prennent juste la taille de leur contenu.*
  NB : les différents types d'affichage :
  https://www.w3schools.com/cssref/pr_class_display.asp

Le code de votre application web doit se retrouver dans ce dossier (en vert) de votre repository local et de votre web repository (**JS-Node-Exercices**).



3 Exercice optionnel

Vous pourriez compléter l'exercice fait au §1 pour afficher une nouvelle forme (un rond, un triangle…) lors d'un clic droit.