Animations avancées en JavaScript

Contrôler les animations avec JavaScript et l'objet Animation (API Web Animation)

Depuis septembre 2022, il est possible de contrôler les animations avec JavaScript en utilisant l'API Web Animation. Cette API permet de créer, démarrer, arrêter, mettre en pause, reprendre et contrôler les animations CSS avec JavaScript.

Cependant, elle n'est pas prise en charge par tous les navigateurs anciens. https://developer.mozilla.org/en-US/docs/Web/API/Element/animate

La fonction animate prend deux arguments : un tableau d'objets décrivant les étapes de l'animation et un objet de configuration de l'animation.

tableau d'objets

Le tableau d'objets décrit les étapes de l'animation. Chaque objet doit contenir les propriétés CSS à animer et leur valeur. Vous pouvez définir autant d'étapes que vous le souhaitez. Cela permet de créer des animations complexes avec des étapes intermédiaires.

objet de configuration

L'objet de configuration de l'animation permet de définir les paramètres de l'animation. Vous pouvez définir la durée de l'animation, le nombre de répétitions, le sens de l'animation, etc.

https://developer.mozilla.org/en-US/docs/Web/API/KeyframeEffect/KeyframeEffect#parameters

- `id` : l'identifiant de l'animation. Sert à la retrouver plus tard avec `document.getAnimations()`
- `duration` : la durée de l'animation en millisecondes
- `iterations` : le nombre de répétitions de l'animation
- `direction` : le sens de l'animation (normal, reverse, alternate, alternate-reverse)
- `fill` : le comportement de l'animation à la fin (none, forwards, backwards, both)
- `easing` : la fonction d'interpolation de l'animation (linear, ease, ease-in, ease-out, ease-in-out, cubic-bezier)
- `delay` : le délai avant le démarrage de l'animation en millisecondes
- `endDelay` : le délai après la fin de l'animation en millisecondes

```
let elementHTML = document.querySelector("section");

let animation = elementHTML.animate([{ opacity: 1 }, { opacity: 0 }], {
    duration: 1000,
    fill: "forwards",
    id: "disparition",
```

```
});

animation.addEventListener("finish", function () {
    console.log("L'animation est terminée !");
});
```

Exécuter une fonction à la fin d'une animation

Pour exécuter une fonction à la fin d'une animation, vous pouvez utiliser l'événement `finish`. Cela permet de déclencher une fonction une fois que l'animation est terminée.

```
animation.addEventListener("finish", function () {
    console.log("L'animation est terminée !");
});
```

Redémarrer une animation

Pour redémarrer une animation, vous pouvez utiliser la méthode `animation.play()`. Cela permet de relancer l'animation depuis le début.

```
animation.addEventListener("finish", function () {
    animation.play();
});
```

Mettre en pause une animation

Pour mettre en pause une animation, vous pouvez utiliser la méthode `animation.pause()`. Cela permet de suspendre l'animation à un moment donné.

```
animation.addEventListener("click", function () {
    animation.pause();
});
```

États d'une animation

Une animation peut être dans plusieurs états à un moment donné. Vous pouvez vérifier l'état d'une animation en utilisant les propriétés suivantes :

- `animation.playState` : l'état de l'animation (idle, pending, running, paused, finished)
- `animation.startTime` : le moment où l'animation a démarré
- `animation.currentTime` : le temps écoulé depuis le début de l'animation

- `animation.effect` : l'objet KeyframeEffect associé à l'animation
- `animation.id` : l'identifiant de l'animation

Créer un groupe d'animation et une séquence d'animation

Présentement, les groupes et les séquences ne sont pas supportés par tous les navigateurs. Cependant, ils le seront bientôt. Je vous recommande de consulter les documentations pour plus d'informations.

La documentation de l'API Web Animation

http://waapi.guide/

Un tutoriel expliquant comment utiliser l'API Web Animation

https://danielcwilson.com/blog/2015/07/animations-intro/ - Certaines fonctionnalités ne sont pas encore supportées par tous les navigateurs.

RequestAnimationFrame

Une version plus flexible mais plus complexe à utiliser que l'API Web Animation est `requestAnimationFrame`. Cette fonction permet d'exécuter une fonction à chaque rafraîchissement de l'écran, ce qui permet de créer des animations fluides. Ceci correspond à environ 40 à 60 fois par seconde. Il est essentiel de rappeler la fonction à la fin de celle-ci pour qu'elle soit exécutée en boucle.

Pour créer une durée, il est possible d'utiliser `performance.now()` pour obtenir le temps actuel et le comparer à un temps de départ. `Performance.now()` renvoie une précision de microsecondes plus grande que `Date.now()` mais les deux peuvent être utilisés pour mesurer le temps.

```
let tempsDepart = performance.now();

function bougerElement() {
    let tempsActuel = performance.now();
    // let tempsActuel = Date.now(); // Alternative

    // Calculer le temps écoulé depuis le début de l'animation. Permet de créer une
animation en fonction du temps.
    let tempsEcoule = tempsActuel - tempsDepart;

    // Placer ici la condition pour arrêter l'animation.
    // Peut être un temps écoulé ou une position finale.

    if (tempsEcoule < 5000) {
        //Ou titre.style.left < window.innerWidth
        titre.style.left = tempsEcoule / 10 + "px";
        // On rappelle la fonction pour boucler l'animation tant que la condition
n'est pas remplie.
        requestAnimationFrame(bougerElement);
```

```
    }
}

// Lancer l'animation
requestAnimationFrame(bougerElement);
```

Librairies d'animation

La majorité des projets utilisent des librairies d'animation pour simplifier la création d'animations complexes. Voici quelques exemples de librairies populaires :

- GSAP - Puissant mais payant pour certaines fonctionnalités
- Anime.js - Simple et léger, facile à prendre en main
- Three.js - Permet d'intégrer des animations 3D dans une page web