Snow js

```
'use strict';

//Each update cycle should remove this much life from a snowflake
const LIFE_PER_TICK = 1000 / 60;
 //Number of snowflakes
const MAX_FLAKES = Math.min(75, screen.width / 1280 * 75);
//The array of snow particles to be animated. They are HTMLElements
const flakes = [];

//A variety of periodic movement functions for the x-axis to create a range of snow falling models
//The initial multiplier determines how far it moves in vw units at most, from the original
//x-axis position
const period = [
    n => 5 * (Math.sin(n)),
    n => 8 * (Math.cos(n)),
    n => 5 * (Math.sin(n) * Math.cos(2 * n)),
    n => 2 * (Math.sin(0.25 * n) - Math.cos(0.75 * n) + 1),
    n => 5 * (Math.sin(0.75 * n) + Math.cos(0.25 * n) - 1)
];

//Emojis to substitute for snowflakes, just for fun
const fun = ['⛄', '🎁', '🦌', '☃', '🍪'];

//The CSS styles for the snowflakes and container
const cssString = `.snowfall-container {
    display: block;
    height: 100vh;
    left: 0;
    margin: 0;
    padding: 0;
    -webkit-perspective-origin: top center;
            perspective-origin: top center;
    -webkit-perspective: 150px;
            perspective: 150px;
    pointer-events: none;
    position: fixed;
    top: 0;
    -webkit-transform-style: preserve-3d;
            transform-style: preserve-3d;
    width: 100%;
    z-index: 99999; }

  .snowflake {
```

```
    pointer-events: none;
    color: #ddf;
    display: block;
    font-size: 24px;
    left: -12px;
    line-height: 24px;
    position: absolute;
    top: -12px;
    -webkit-transform-origin: center;
            transform-origin: center; }`;

// Add a DOMContentLoaded listener, or fire the function immediately if that
already happened
function ready(fn) {
    if (document.attachEvent ? document.readyState === 'complete' :
document.readyState !== 'loading') {
        fn();
    }
    else {
        document.addEventListener('DOMContentLoaded', fn);
    }
}

// Reset a flake to newly randomized values
function resetFlake(flake) {
    // X-axis is in vw CSS units
    let x = flake.dataset.origX = (Math.random() * 100);
    //Y-axis is in CSS vh units
    let y = flake.dataset.origY = 0;

    //Once in awhile, have closer snowflakes
    //Z-axis is in CSS px units
    let z = flake.dataset.origZ = (Math.random() < 0.1) ?
(Math.ceil(Math.random() * 100) + 25) : 0;

    let life = flake.dataset.life = (Math.ceil(Math.random() * 4000) + 6000); //
Milliseconds
    flake.dataset.origLife = life; //Timestamps for flake creation

    flake.style.transform = `translate3d(${x}vw, ${y}vh, ${z}px)`;
    flake.style.opacity = 1.0;

    //This is the index into the period function array
    flake.dataset.periodFunction = Math.floor(Math.random() * period.length);

    if (Math.random() < 0.001) {
        //Very small chance of some fun happening
```

```
        flake.innerText = fun[Math.floor(Math.random() * fun.length)];
    }
}

// Move all the snowflakes
function updatePositions() {

    flakes.forEach((flake) => {
        // Normalize amount of time a snowfalke has been alive to the range [0,
1.0]
        let origLife = parseFloat(flake.dataset.origLife)
        let curLife = parseFloat(flake.dataset.life);
        let dt = (origLife - curLife) / origLife;

        if (dt <= 1.0) {
            // Fetch this flake's personalized periodicity for x-axis movement fromt
he array
            let p = period[parseInt(flake.dataset.periodFunction)];
            // Calculate new x-position, relative to original starting x
            let x = p(dt * 2 * Math.PI) + parseFloat(flake.dataset.origX);
            //Snowflakes fall to the bottom of the screen using a straight linear
progression over their lifespan
            let y = 100 * dt;
            // Z-depth does not vary over time, although I guess it could?
            let z = parseFloat(flake.dataset.origZ);
            // Each update, change the CSS transformation
            flake.style.transform = `translate3d(${x}vw, ${y}vh, ${z}px)`;

            if (dt >= 0.5) {
                //Start fading out flakes 1/2 down screen
                flake.style.opacity = (1.0 - ((dt - 0.5) * 2));
            }

            curLife -= LIFE_PER_TICK;
            flake.dataset.life = curLife;
        }
        else {
            //Once the lifespan is exceeded, reset the flake
            resetFlake(flake);
        }
    });

    //Using requestAnimationFrame to update the positions for a (hopefully)
smooth animation
    window.requestAnimationFrame(updatePositions);
}
```

```
function appendSnow() {
    //Append the CSS styles to the document head
    let styles = document.createElement('style');
    styles.innerText = cssString;
    document.querySelector('head').appendChild(styles);

    //Create the container for the snowflakes and add it to the document body
    let field = document.createElement('div');
    field.classList.add('snowfall-container');

    //set aria-hidden and role=presentation so that screen readers don't read the
emoji
    field.setAttribute('aria-hidden', 'true');
    field.setAttribute('role', 'presentation');
    document.body.appendChild(field);

    let i = 0;

    //Using an inner function and setTimeout to delay the initial snowfall
    //This makes it much less clumpy
    const addFlake = () => {
        let flake = document.createElement('span');
        flake.classList.add('snowflake');
        flake.setAttribute('aria-hidden', 'true');
        flake.setAttribute('role', 'presentation');
        flake.innerText = '❄';
        resetFlake(flake);
        flakes.push(flake);
        field.appendChild(flake);

        //Recursive (delayed by timeout) call to add a flake until max reached
        if (i++ <= MAX_FLAKES) {
            setTimeout(addFlake, Math.ceil(Math.random() * 300) + 100);
        }
    };
    addFlake();

    updatePositions();
}

ready(appendSnow);
```