The `svelte/transition` module has a handful of built-in transitions, but it's very easy to create your own. By way of example, this is the source of the `fade` transition:

```
function fade(node, {
    delay = 0,
    duration = 400
}) {
    const o = +getComputedStyle(node).opacity;

    return {
        delay,
        duration,
        css: t => `opacity: ${t * o}`
    };
}
```

The function takes two arguments — the node to which the transition is applied, and any parameters that were passed in — and returns a transition object which can have the following properties:

- `delay` — milliseconds before the transition begins
- `duration` — length of the transition in milliseconds
- `easing` — a `p => t` easing function (see the chapter on tweening)
- `css` — a `(t, u) => css` function, where `u === 1 - t`
- `tick` — a `(t, u) => {...}` function that has some effect on the node

The `t` value is `0` at the beginning of an intro or the end of an outro, and `1` at the end of an intro or beginning of an outro.

Most of the time you should return the `css` property and *not* the `tick` property, as CSS animations run off the main thread to prevent jank where possible. Svelte 'simulates' the transition and constructs a CSS animation, then lets it run.

For example, the `fade` transition generates a CSS animation somewhat like this:

```
0% { opacity: 0 }
10% { opacity: 0.1 }
20% { opacity: 0.2 }
/* ... */
100% { opacity: 1 }
```

We can get a lot more creative though. Let's make something truly gratuitous:

```
<script>
    import { fade } from 'svelte/transition';
    import { elasticOut } from 'svelte/easing';

    let visible = true;

    function spin(node, { duration }) {
        return {
            duration,
            css: t => {
                const eased = elasticOut(t);

                return `
                    transform: scale(${eased}) rotate(${eased * 1080}deg);
                    color: hsl(
                        ${Math.trunc(t * 360)},
                        ${Math.min(100, 1000 - 1000 * t)}%,
                        ${Math.min(50, 500 - 500 * t)}%
                    );`
            }
        };
    }
</script>
```

Remember: with great power comes great responsibility.