The use directive

Actions are essentially element-level lifecycle functions. They're useful for things like:

- interfacing with third-party libraries
- lazy-loaded images
- tooltips
- adding custom event handlers

In this app, we want to make the orange modal close when the user clicks outside it. It has an event handler for the `outclick` event, but it isn't a native DOM event. We have to dispatch it ourselves. First, import the `clickOutside` function…

```
import { clickOutside } from "./click_outside.js";
```

…then use it with the element:

```
<div class="box" use:clickOutside on:outclick="{() => (showModal = false)}">
    Click outside me!
</div>
```

Open the `click_outside.js` file. Like transition functions, an action function receives a `node` (which is the element that the action is applied to) and some optional parameters, and returns an action object. That object can have a `destroy` function, which is called when the element is unmounted.

We want to fire the `outclick` event when the user clicks outside the orange box. One possible implementation looks like this:

```
export function clickOutside(node) {
    const handleClick = (event) => {
        if (!node.contains(event.target)) {
            node.dispatchEvent(new CustomEvent("outclick"));
        }
    };

    document.addEventListener("click", handleClick, true);

    return {
        destroy() {
            document.removeEventListener("click", handleClick, true);
        },
    };
}
```

Update the `clickOutside` function, click the button to show the modal and then click outside it to close it.