Unlike DOM events, component events don't *bubble*. If you want to listen to an event on some deeply nested component, the intermediate components must *forward* the event.

In this case, we have the same `App.svelte` and `Inner.svelte` as in the previous chapter, but there's now an `Outer.svelte` component that contains `<Inner/>`.

One way we could solve the problem is adding `createEventDispatcher` to `Outer.svelte`, listening for the `message` event, and creating a handler for it:

```
<script>
	import Inner from './Inner.svelte';
	import { createEventDispatcher } from 'svelte';

	const dispatch = createEventDispatcher();

	function forward(event) {
		dispatch('message', event.detail);
	}
</script>

<Inner on:message={forward}/>
```

But that's a lot of code to write, so Svelte gives us an equivalent shorthand — an `on:message` event directive without a value means 'forward all `message` events'.

```
<script>
	import Inner from './Inner.svelte';
</script>

<Inner on:message/>
```