The `<svelte:options>` element allows you to specify compiler options.

We'll use the `immutable` option as an example. In this app, the `<Todo>` component flashes whenever it receives new data. Clicking on one of the items toggles its `done` state by creating an updated `todos` array. This causes the *other* `<Todo>` items to flash, even though they don't end up making any changes to the DOM.

We can optimise this by telling the `<Todo>` component to expect *immutable* data. This means that we're promising never to *mutate* the `todo` prop, but will instead create new todo objects whenever things change.

Add this to the top of the `Todo.svelte` file:

```
<svelte:options immutable={true}/>
```

> *You can shorten this to `<svelte:options immutable/>` if you prefer.*

Now, when you toggle todos by clicking on them, only the updated component flashes.

The options that can be set here are:

- `immutable={true}` — you never use mutable data, so the compiler can do simple referential equality checks to determine if values have changed
- `immutable={false}` — the default. Svelte will be more conservative about whether or not mutable objects have changed
- `accessors={true}` — adds getters and setters for the component's props
- `accessors={false}` — the default
- `namespace="..."` — the namespace where this component will be used, most commonly `"svg"`
- `tag="..."` — the name to use when compiling this component as a custom element

Consult the API reference for more information on these options.