Accessibility (shortened to a11y) isn't always easy to get right, but Svelte will help by warning you at compile time if you write inaccessible markup. However, keep in mind that many accessibility issues can only be identified at runtime using other automated tools and by manually testing your application.

Here is a list of accessibility checks Svelte will do for you.

`a11y-accesskey`

Enforce no `accesskey` on element. Access keys are HTML attributes that allow web developers to assign keyboard shortcuts to elements. Inconsistencies between keyboard shortcuts and keyboard commands used by screen reader and keyboard-only users create accessibility complications. To avoid complications, access keys should not be used.

```
<!-- A11y: Avoid using accesskey -->
<div accessKey='z'></div>
```

`a11y-aria-attributes`

Certain reserved DOM elements do not support ARIA roles, states and properties. This is often because they are not visible, for example `meta`, `html`, `script`, `style`. This rule enforces that these DOM elements do not contain the `aria-*` props.

```
<!-- A11y: <meta> should not have aria-* attributes -->
<meta aria-hidden="false">
```

`a11y-autofocus`

Enforce that `autofocus` is not used on elements. Autofocusing elements can cause usability issues for sighted and non-sighted users alike.

```
<!-- A11y: Avoid using autofocus -->
<input autofocus>
```

`a11y-distracting-elements`

Enforces that no distracting elements are used. Elements that can be visually distracting can cause accessibility issues with visually impaired users. Such elements are most likely deprecated, and should be avoided.

The following elements are visually distracting: `<marquee>` and `<blink>`.

```
<!-- A11y: Avoid <marquee> elements -->
<marquee />
```

`a11y-hidden`

Certain DOM elements are useful for screen reader navigation and should not be hidden.

```
<!-- A11y: <h2> element should not be hidden -->
<h2 aria-hidden="true">invisible header</h2>
```

a11y-img-redundant-alt

Enforce img alt attribute does not contain the word image, picture, or photo. Screen readers already announce `img` elements as an image. There is no need to use words such as *image*, *photo*, and/or *picture*.

```
<img src="foo" alt="Foo eating a sandwich." />

<!-- aria-hidden, won't be announced by screen reader -->
<img src="bar" aria-hidden="true" alt="Picture of me taking a photo of an image" />

<!-- A11y: Screen readers already announce <img> elements as an image. -->
<img src="foo" alt="Photo of foo being weird." />

<!-- A11y: Screen readers already announce <img> elements as an image. -->
<img src="bar" alt="Image of me at a bar!" />

<!-- A11y: Screen readers already announce <img> elements as an image. -->
<img src="foo" alt="Picture of baz fixing a bug." />
```

a11y-invalid-attribute

Enforce that attributes important for accessibility have a valid value. For example, `href` should not be empty, `'#'`, or `javascript:`.

```
<!-- A11y: '' is not a valid href attribute -->
<a href=''>invalid</a>
```

a11y-label-has-associated-control

Enforce that a label tag has a text label and an associated control.

There are two supported ways to associate a label with a control:

- Wrapping a control in a label tag.
- Adding `for` to a label and assigning it the ID of an input on the page.

```
<label for="id">B</label>

<label>C <input type="text" /></label>

<!-- A11y: A form label must be associated with a control. -->
<label>A</label>
```

a11y-media-has-caption

Providing captions for media is essential for deaf users to follow along. Captions should be a transcription or translation of the dialogue, sound effects, relevant musical cues, and other relevant audio information. Not only is this important for accessibility, but can also be useful for all users in the case that the media is unavailable (similar to `alt` text on an image when an image is unable to load).

The captions should contain all important and relevant information to understand the corresponding media. This may mean that the captions are not a 1:1 mapping of the dialogue in the media content. However, captions are not necessary for video components with the `muted` attribute.

```
<video><track kind="captions"/></video>

<audio muted></audio>

<!-- A11y: Media elements must have a <track kind=\"captions\"> -->
<video></video>

<!-- A11y: Media elements must have a <track kind=\"captions\"> -->
<video><track /></video>
```

`a11y-misplaced-role`

Certain reserved DOM elements do not support ARIA roles, states and properties. This is often because they are not visible, for example `meta`, `html`, `script`, `style`. This rule enforces that these DOM elements do not contain the `role` props.

```
<!-- A11y: <meta> should not have role attribute -->
<meta role="tooltip">
```

`a11y-misplaced-scope`

The scope attribute should only be used on `<th>` elements.

```
<!-- A11y: The scope attribute should only be used with <th> elements -->
<div scope="row" />
```

`a11y-missing-attribute`

Enforce that attributes required for accessibility are present on an element. This includes the following checks:

- `<a>` should have an href (unless it's a [fragment-defining tag](#))
- `<area>` should have alt, aria-label, or aria-labelledby
- `<html>` should have lang
- `<iframe>` should have title
- `<img>` should have alt
- `<object>` should have title, aria-label, or aria-labelledby
- `<input type="image">` should have alt, aria-label, or aria-labelledby

```
<!-- A11y: <input type=\"image\"> element should have an alt, aria-label or aria-
labelledby attribute -->
<input type="image">

<!-- A11y: <html> element should have a lang attribute -->
<html></html>

<!-- A11y: <a> element should have an href attribute -->
<a>text</a>
```

a11y-missing-content

Enforce that heading elements (h1 , h2 , etc.) and anchors have content and that the content is accessible to screen readers

```
<!-- A11y: <a> element should have child content -->
<a href='/foo'></a>

<!-- A11y: <h1> element should have child content -->
<h1></h1>
```

a11y-mouse-events-have-key-events

Enforce that on:mouseover and on:mouseout are accompanied by on:focus and on:blur , respectively. This helps to ensure that any functionality triggered by these mouse events is also accessible to keyboard users.

```
<!-- A11y: on:mouseover must be accompanied by on:focus -->
<div on:mouseover={handleMouseover} />

<!-- A11y: on:mouseout must be accompanied by on:blur -->
<div on:mouseout={handleMouseout} />
```

a11y-no-redundant-roles

Some HTML elements have default ARIA roles. Giving these elements an ARIA role that is already set by the browser has no effect and is redundant.

```
<!-- A11y: Redundant role 'button' -->
<button role="button" />

<!-- A11y: Redundant role 'img' -->
<img role="img" src="foo.jpg" />
```

a11y-positive-tabindex

Avoid positive tabindex property values. This will move elements out of the expected tab order, creating a confusing experience for keyboard users.

```
<!-- A11y: avoid tabindex values above zero -->
<div tabindex='1'/>
```

a11y-structure

Enforce that certain DOM elements have the correct structure.

```
<!-- A11y: <figcaption> must be an immediate child of <figure> -->
<div>
    <figcaption>Image caption</figcaption>
</div>
```

a11y-unknown-aria-attribute

Enforce that only known ARIA attributes are used. This is based on the WAI-ARIA States and Properties spec.

```
<!-- A11y: Unknown aria attribute 'aria-labeledby' (did you mean 'labelledby'?) -->
<input type="image" aria-labeledby="foo">
```

a11y-unknown-role

Elements with ARIA roles must use a valid, non-abstract ARIA role. A reference to role definitions can be found at WAI-ARIA site.

```
<!-- A11y: Unknown role 'toooltip' (did you mean 'tooltip'?) -->
<div role="toooltip"></div>
```