We’re ARIA Live!

Mangfold i mai

Eric Eggert

Eric Eggert (he/him/his)

- 🇩🇪 Co-founder & Co-owner of *outline* Consulting (2011)
- 🇸🇪 **Web Accessibility Specialist at *Axess Lab* (2022)**
- 🇦🇹 **Lecturer at *FH Joanneum* (2015)**
- 🇺🇸 Director of Accessibility Services at *Knowbility* (’16–’22)
- 🌏 Web Accessibility Specialist at *W3C/WAI* (’13–’20)

📍 Wissen, Germany 🇩🇪

ARIA Live Regions

ARIA =
Accessible Rich
Internet Applications

Live regions

Live regions are a way to give users feedback on actions they have taken through the accessibility API.

Accessibility API

Assistive technology has to be built to work with live regions.

Currently: Only screen readers.

WCAG SC 4.1.3 Status Messages

In content implemented using markup languages, *status messages* can be programmatically determined through role or properties such that they can be presented to the user by assistive technologies without receiving focus.

status message

change in content that is *not* a change of context, and that provides *information* to the user *on the success or results of an action*, on the *waiting state* of an application, on the *progress of a process*, or on the *existence of errors*

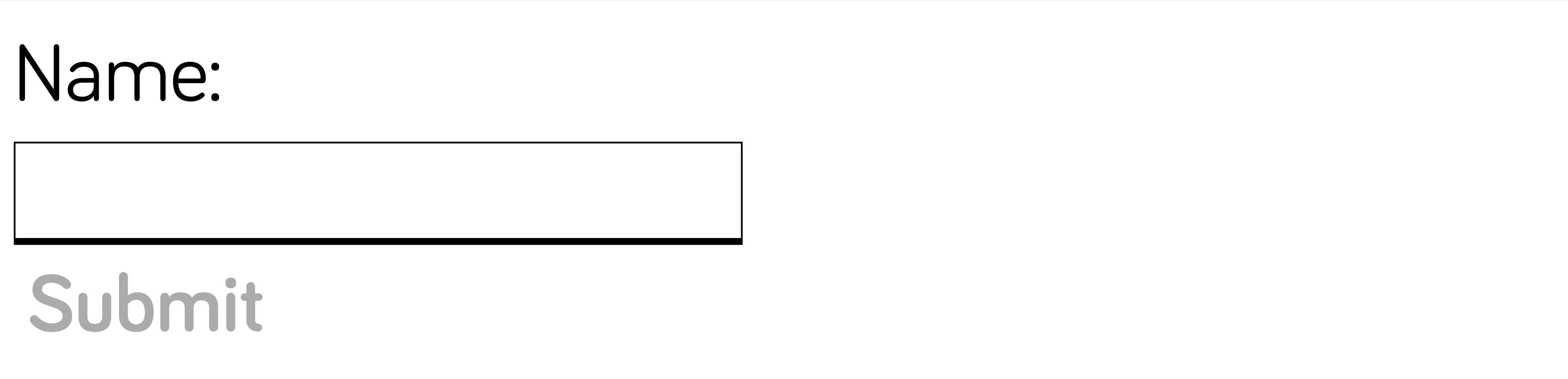
status message

- success or results of an action
- waiting state of an application
- progress of a process
- existence of errors

Other ways for user feedback

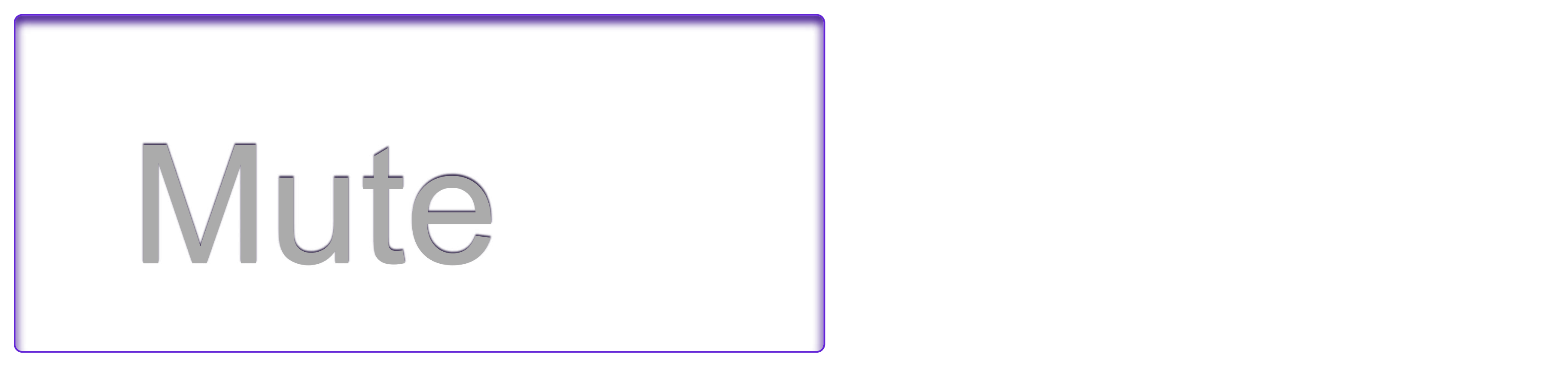
- Page Reload
- Change of ARIA properties

Example: Page reload



✓ No need for live regions if the page reloads (as it is a *change of context*).

Example: ARIA property (1)



✓ No need for live regions as an aria property is changed.

Example: ARIA property (2)

```
<button
   id="toggle"
   aria-pressed="true">
  Mute
    <svg data-icon="unpressed" aria-hidden="true" focusable="false" data-prefix="fas" data-icon="volume"
class="svg-inline--fa fa-volume" role="img" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 576 512"><path
fill="currentColor" d="M444.6 181.9c-10.28-8.344-25.41-6.875-33.75 3.406c-8.406 10.25-6.906 25.37 3.375
33.78C425.5 228.4 432 241.8 432 256c0 14.19-6.5 27.62-17.81 36.87c-10.28 8.406-11.78 23.53-3.375 33.78c4.719
5.812 11.62 8.812 18.56 8.812c5.344 0 10.75-1.781 15.19-5.406C467.1 311.6 480 284.7 480 256S467.1 200.4 444.6
181.9zM505.1 108c-10.22-8.344-25.34-6.906-33.78 3.344c-8.406 10.25-6.906 25.37 3.344 33.78C508.6 172.9 528
213.3 528 256s-19.44 83.09-53.31 110.9c-10.25 8.406-11.75 23.53-3.344 33.78c4.75 5.781 11.62 8.781 18.56
8.781c5.375 0 10.75-1.781 15.22-5.437C550.2 367.1 576 313.1 576 256S550.2 144.9 505.1 108zM333.2 34.84c-11.5-
```

✓ No need for live regions as an aria property is changed.

Example: ARIA property (3)

```
document.querySelector('#toggle').addEventListener('click', (event) => {
  if (event.target.getAttribute('aria-pressed') == 'true') {
      event.target.setAttribute('aria-pressed', 'false');
    } else {
        event.target.setAttribute('aria-pressed', 'true');
    }
});
```

✓ No need for live regions as an aria property is changed.

But:

Screen readers do **not** announce those changes

Similar: Change of the accessible name

```
document.querySelector('#playpause').addEventListener('click', (event) => {
  if (event.target.innerText == 'Play') {
      event.target.innerText = 'Pause';
    } else {
       event.target.innerText = 'Play';
    }
});
```

✓ Is programatically determinable. (But also not read by screen readers.)

Live regions use specific roles (but not `role="region"`) & properties

Live regions are conveyed through AT

When to use Live Regions?

Live regions can be used when:

- You need to communicate a whole sentence or set of information to users.
- The message needs to be conveyed immediately.

Live region attributes

`aria-live`

- `off` (default)

`aria-live`

- **`off`** (default)
- `polite`: updates are presented at the *next graceful opportunity*: end of the current sentence or typing pauses

`aria-live`

- **`off`** (default)
- `polite`: updates are presented at the *next graceful opportunity*: end of the current sentence or typing pauses
- `assertive`: updates to the region have the *highest priority* and should be presented the user *immediately*

`aria-atomic`

- `false` (default): AT will present only the changed node(s)

`aria-atomic`

- **`false`** (default): AT will present only the changed node(s)
- `true`: AT will present the *entire changed region as a whole*, including the author-defined label if one exists.

aria-busy

- `false` (default): no expected updates for the element

`aria-busy`

- **`false`** (default): no expected updates for the element
- `true`: the element is being updated

`aria-busy`

- **`false`** (default): no expected updates for the element
- `true`: the element is being updated

When `aria-busy` is `true`, AT **may** ignore changes to content and then process all changes made during the busy period as a single, atomic update when `aria-busy` becomes `false`.

aria-relevant

`aria-relevant`

- `additions`: Announce added nodes

`aria-relevant`

- `additions`: Announce added nodes
- `removals`: Announce removed nodes

`aria-relevant`

- `additions`: Announce added nodes
- `removals`: Announce removed nodes
- `text`: Announce text nodes

`aria-relevant`

- `additions`: Announce added nodes
- `removals`: Announce removed nodes
- `text`: Announce text nodes
- `all`: equivalent to `additions removals text`

`aria-relevant`

- `additions`: Announce added nodes
- `removals`: Announce removed nodes
- `text`: Announce text nodes
- `all`: equivalent to `additions removals text`

Default: `additions text`

Live region roles

`alert`, `status`, `marquee`, `timer`, `log`

Basically pre-configured aria live region property configurations.

`role="status"`

Advisory information for the user that is not important enough to justify an alert; often but not necessarily presented as a status bar.

Defaults:

- `aria-live="polite"`
- `aria-atomic="true"`

`role="alert"`

Messages that may be immediately important to users.

Defaults:

- `aria-live="assertive"`
- `aria-atomic="true"`

`role="marquee"`

Non-essential information that changes frequently. Needs an accessible name.

Defaults:

- `aria-live="off"`

`role="log"`

New information is added in meaningful order and old information may disappear.

Defaults:

- `aria-live="polite"`

`role="timer"`

Contains a number which indicates an amount of elapsed time from a start point, or the time remaining until an end point.

Defaults:

- **`aria-live="off"`**

Change of the accessible name with live region

```
<button
   id="playpause" aria-live="polite">
  Play
</button>
```

✓ Is programatically determinable and also read by screen readers.

Use an existing element if…

- the status is also visible to users.
- the conveyed text is equivalent to the visible text.

Example: Chat

Use a separate element for status updates if…

- you want to give users more context.
- the output text has additional information that is not conductive to being announced.

Example: Carousel page confirmation

Some pitfalls!

There is no guarantee that your live region is read.

AT might decide that your message is outdated & discard it.

Screen readers are unaware of recently-added live regions

⊗ Do not add and update a live region immediately.

Example: Immediate Live Regions

Beware of `innerHTML` shortcuts

⊗ Replacing all of the content might look the same visually but can be a different experience in a screen reader.

Example: Chat

Hidden live regions are not announced

☒ Avoid:

- `display: none`
- `visibility: hidden`
- `hidden` attribute

Use "visually hidden" instead

✓ Use:

```css
.visually-hidden {
  border: 0;
  clip: rect(0, 0, 0, 0);
  height: 1px;
  margin: -1px;
  overflow: hidden;
  padding: 0;
  position: absolute;
  white-space: nowrap;
  width: 1px;
}
```

Note: This also means that you want to remove the text from the live region after a short delay to avoid users being able to navigate to it.

Live regions have no structure and cannot be navigated

ⓧ Do not:

- Wrap whole pages in a live region
- Put long text into live regions (especially if unavailable otherwise)

Some screen readers might read live regions in hidden windows or tabs

```
const notifications = document.getElementById('notifications');

document.addEventListener('visibilitychange', () => {
  let setting = document.hidden ? ['none', 'off'] : ['status', 'polite'];

  notification.setAttribute('role', setting[0]);
  notification.setAttribute('aria-live', setting[1]);
});
```

Source: Inclusive Components: Notifications

Best Practices

Is the message is **important enough** to be announced at all?

- Critical information
- Expected updates

Best Practices

Would a user want to **interact with the announced element**? If so, setting the focus can be the right thing to do.

Example: Carousel direct page choice

Best Practices

You can initialize general purpose region(s) on page load:

```
var demo = document.createElement("div");
demo.classList.add('visually-hidden');
demo.innerHTML = '<div id="status" role="status"></div>' +
                 '<div id="alert" role="alert"></div>';
document.querySelector('body').appendChild(demo);
```

Best Practices

Sometimes you want to use an audio cue (“earcon”) instead of a screen reader announcement.

Best Practices

Make sure equivalent information is available visually.

Example: WCAG Quick Reference

Wrap Up

- `role="alert"`/`role="status"` are your friends (and sensible defaults)
- Do not overuse live regions
- Ensure visual information as well
- Have live regions in the DOM early
- Use other methods where appropriate

Thank you!

Eric Eggert

- Website: yatil.net
- YouTube: youtube.com/yatil
- Mastodon: @yatil@toot.cafe
- Twitter: @yatil
- Email: mail@yatil.net