Element: ariaNotify() method

- Syntax
  - Parameters
  - Return value
- Description
  - Announcement priorities
  - Language selection
  - Permissions policy integration
- Examples
  - Basic ariaNotify() usage
    - Result
- Specifications
- Browser compatibility
- See also

The `ariaNotify()` method of the Element interface queues a string of text to be announced by a screen reader.

Syntax

```
ariaNotify(announcement)
ariaNotify(announcement, options)
```

Parameters

- `announcement`
  - : A string specifying the text to be announced.
- `options`
  - : An options object containing the following properties:

- `priority`
  - : An enumerated value specifying the priority of the announcement. Possible values are:
    - `normal`
      - : The announcement has normal priority. It will be spoken after any announcement that a screen reader is currently making. This is the default value.
    - `high`
      - : The announcement has high priority. It will be spoken immediately, interrupting any announcement that a screen reader is currently making.

Return value

None (undefined).

Description

The **`ariaNotify()`** method can be used to programmatically trigger a screen reader announcement.

This method provides similar functionality to ARIA live regions, with some advantages:

- Live regions can only make announcements following changes to the DOM, whereas an `ariaNotify()` announcement can be made at any time.
- Live region announcements involve reading out the updated content of the changed DOM node, whereas `ariaNotify()` announcement content can be defined independently of DOM content.

Developers often work around the limitations of live regions using hidden DOM nodes with live regions set on them, which have their contents updated with the content to be announced. This is inefficient and error-prone, and `ariaNotify()` provides a way to avoid such issues.

Some screen readers will read out multiple `ariaNotify()` announcements in order, but this cannot be guaranteed across all screen readers and platforms. Normally, only the most recent announcement is spoken. It is more reliable to combine multiple announcements into one.

For example, the following calls:

```
elemRef.ariaNotify("Hello there.");
elemRef.ariaNotify("The time is now 8 o'clock.");
```

would be better combined:

```
elemRef.ariaNotify("Hello there. The time is now 8 o'clock.");
```

An `ariaNotify()` call can be fired on any element in the DOM, except for ones that the browser does not consider "interesting" for accessibility, and ignores when building the accessibility tree. Exactly which elements are ignored varies by browser, but the list generally includes container elements of little to no semantic value, such as the html and body elements.

`ariaNotify()` announcements do not require transient activation; you should take care not to spam screen reader users with too many notifications, as this could create a bad user experience.

Announcement priorities

An `ariaNotify()` announcement with `priority: high` set is announced before an `ariaNotify()` announcement with `priority: normal` set.

`ariaNotify()` announcements are roughly equivalent to ARIA live region announcements as follows:

- `ariaNotify()` `priority: high`: `aria-live="assertive"`.
- `ariaNotify()` `priority: normal`: `aria-live="polite"`.

However, `aria-live` announcements will take priority over `ariaNotify()` announcements.

Language selection

Screen readers choose an appropriate voice with which to read `ariaNotify()` announcements (in terms of accent, pronunciation, etc.) based on the language specified in the element's `lang` attribute or, if the element does not have a specified `lang` attribute, the `lang` attribute set on its nearest ancestor. If there is no `lang` attribute specified in the HTML, the user agent's default language is used.

Permissions policy integration

Usage of `ariaNotify()` in a document or iframe can be controlled by an aria-notify Permission Policy.

Specifically, where a defined policy blocks usage, any announcements created using `ariaNotify()` silently fail (they will not be sent).

Examples

For a more substantial example, see the Accessible shopping list example on the Document.ariaNotify() page. The example would work just the same if you called `ariaNotify()` on an element reference rather than the `Document` object.

Basic `ariaNotify()` usage

This example includes a button that fires a screen reader announcement on itself when clicked.

```
<button>Press</button>
```

```
html,
body {
  height: 100%;
}

body {
  display: flex;
  justify-content: center;
  align-items: center;
}
```

```
document.querySelector("button").addEventListener("click", () => {
  document.querySelector("button").ariaNotify("You ain't seen me, right?");
});
```

Result

The output is as follows:

Try activating a screen reader and then pressing the button. You should hear "You ain't seen me, right?" spoken by the screen reader.

Specifications

Browser compatibility

See also

- Document.ariaNotify()
- ARIA live regions

□□□ #3
□ □□□ □□□ 2026-10-31 03:27:53 UTC
□ cocobook-sync □□□ 2026-11-04 17:34:13 UTC