Die Web Content Accessibility Guidelines. Eine Einführung.

Jens Pelzetter

17. Oktober 2015

Inhaltsverzeichnis

1 Was ist WAI-ARIA **1**

2 Rollen **2**

3 Zustände und Eigenschaften **5**
3.1 aria-labelledby und aria-label 6
3.2 aria-describedby 7
3.3 aria-busy 7
3.4 aria-invalid 7

1 Was ist WAI-ARIA

Durch den vermehrten Einsatz von JavaScript im Web ist es für assistierende Technologien schwieriger geworden, die Elemente einer Webseite oder Webanwendung korrekt zu identifizieren und darzustellen. Mit der Kombination aus HTML, CSS und JavaScript ist es zum Beispiel möglich, eigene Widgets zu erstellen, die aber nicht als solche erkannt werden. Eine ausführliche Darstellung dieser Probleme findet sich im Dokument WAI-ARIA 1.0 Primer [3]. ARIA ist allerdings nicht auf die Verwendung mit HTML beschränkt, sondern kann theoretisch als Ergänzung beliebiger XML-basierter Sprachen eingesetzt werden, beispielsweise SVG oder EPUB. In diesem Dokument geht es nur die Verwendung mit HTML.

WAI-ARIA ist ein von der Web Accessibility Initiative (WAI) [1], einer Arbeitsgruppe des W3C, erarbeiteter Standard [1] mit dem assistierenden Technologien zusätzliche Informationen über *strukturelle* Semantik eines HTML-Dokumentes zur Verfügung gestellt werden können. Dazu definiert ARIA eine verschiedene zusätzliche Attribute, mit denen diese Informationen bereitgestellt werden können.

[1] http://www.w3.org/WAI/

ARIA ist *nicht* als Ersatz für gut strukturiertes, semantisches HTML zu gedacht. Wenn es möglich ist Beziehungen oder Rollen direkt in HTML zu beschreiben ist diese Möglichkeit grundsätzlich vorzuziehen.

Dieses Dokument soll einen ersten Überblick über ARIA bieten und die grundlegenden Konzepten erklären. Für weitere Informationen stellt das W3C verschiedene Dokumente zur Verfügung. Zielgruppe sind primär Entwickler von Webseiten als auch Webdesigner, die sich einen ersten Überblick über ARIA verschaffen wollen.

ARIA definiert zum einen eine Reihen von Rollen die Zweck eines HTML-Elements beschreiben, zum anderen eine Reihe von Attributen mit denen sich Zustände und Eigenschaften von HTML-Elemente beschrieben lassen.

2 Rollen

Die von ARIA definierten Rollen lassen sich in folgende Gruppen aufteilen:

- Abstrakte Rollen (Abstract Roles)
- Rollen für Widgets (Widget Roles)
- Rollen für zusammengesetzte Widgets (Composite Widget Roles)
- Rollen für die Struktur von Dokumenten (Document Structure)
- Rollen für Bereiche (Landmark Roles)

Eine Rolle kann Zustände und Eigenschaften definieren, die über die entsprechenden Attribute (s. 3 auf Seite 5) angegeben werden können.

Abstrakte Rollen sind nicht für die Verwendung gedacht, sondern dienen zur Definition gemeiner Zustände und Eigenschaften. Ein UML-Diagramm mit allen Rollen[2] ist als Teil des Standards verfügbar. Es ist vorgesehen, dass die im Standard [1] definierten Rollen durch weitere Standards für bestimmte Anwendungszwecke ergänzt werden können. Ein Beispiel ist das Digital Publishing WAI-ARIA Module [2].

Folgende Rollen werden durch den Standard definiert:

Abstrakte Rollen (Abstract Roles)

- `command`
- `composite`
- `input`
- `landmark`
- `range`
- `roletype`
- `section`
- `sectionhead`
- `select`
- `structure`

[2] http://www.w3.org/TR/wai-aria/rdf_model.html

- widget
- window

Widget Rollen (Widget Roles)

- alert
- alertdialog
- button
- checkbox
- dialog
- gridcell
- link
- log
- marquee
- menuitem
- menuitemcheckbox
- menuitemradio
- option
- progressbar
- radio
- scrollbar
- slider
- spinbutton
- status
- tab
- tabpanel
- textbox
- timer
- tooltip
- treeitem

Rollen für zugesetzten Widgets (Composite Widget Roles)

- combobox
- grid
- listbox
- menu
- menubar
- radiogroup
- tablist
- tree
- treegrid

Rollen für Dokumentstruktur (Document Structure)

- article
- columnheader
- definition
- directory
- document
- group
- heading
- img

- `list`
- `listitem`
- `math`
- `note`
- `presentation`
- `region`
- `row`
- `rowgroup`
- `rowheader`
- `separator`
- `toolbar`

Landmark Roles

- `application`
- `banner`
- `complementary`
- `contentinfo`
- `form`
- `main`
- `navigation`
- `search`

Im folgenden sollen einige der Rollen vorgestellt werden.

Die Rolle `alert` wird dazu verwendet wichtige Informationen auszuzeichnen. Die von `alert` abgeleitete Rolle `alertdialog` bezeichnet einen modalen Dialog mit wichtigen (kritischen) Informationen.

Desktopanwendungen enthalten oftmals Fortschrittsanzeigen in Form von Balken. In Webanwendungen werden diese in der Regel als leere `div`-Container implementiert, und der Zustand wird über CSS grafisch dargestellt.

```
<div id="percent-loaded" role="progressbar" aria-valuenow="75"
aria-valuemin="0" aria-valuemax="100" />
```

Listing 2.1: Fortschrittsanzeige mit ARIA. Quelle: MDN

Listing 2.1[3] zeigt, wie eine solche Fortschrittsanzeige mit ARIA barrierefrei gestaltet werden kann. Das `div`-Element erhält die Rolle `progressbar` um den `div`-Container als Fortschrittsanzeigen zu kennzeichnen. Der aktuelle Wert wird über das Attribut `aria-valuenow` bereitgestellt. Die Attribute `aria-valuemin` und `aria-valuemax` definieren den minimalen und maximalen Wert.

Weitere Widget-Rollen sind zum Beispiel `tab` zur Kennzeichnung eines Tabs, `menu` zur Auszeichnung eines Menüs, `menuitem` zur Auszeichnungen eines Eintrags in einem Menü oder `tree` zur Auszeichnenung einer Baumkompomente.

[3] Quelle: https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA/Web_applications_and_ARIA_FAQ#Can_you_show_me_an_example_of_ARIA_in_action.3F

Für die Dokumentstruktur definiert ARIA verschiedene Rollen um beispielsweise Toolbars (`role="toolbar"`) oder Separatoren (`role="separator"`) auszuzeichnen. Überschriften können mit der Rolle `heading` ausgezeichnet werden. `heading` ist keine Ersatz für die HTML-Elemente `<h1>` bis `<h6>`. Allerdings kann die Rolle `heading` in Verbindung mit dem Attribut `aria-level` verwendet werden, um Überschriften ab der siebten Ebene auszuzeichnen.

Landmark Roles dienen dazu, ganze Bereiche auszuzeichnen. Mit der Rolle `application` kann ein Bereich als Webapplikation (Bereich mti interaktiven Elementen) auszeichnet werden. Der primäre Inhalt einer Webseite kann mit der Rolle `main` kenntlich gemacht werden. Hier ist allerdings das `main`-Element aus HTML5 vorzuziehen. Im Sinne der Abwärtskompatibilität wird derzeit allerdings empfohlen zusätzlich die Rolle `main` anzugeben (`<main role="main">`).

3 Zustände und Eigenschaften

Mit den Attributen für Zustände und Eigenschaften können Zustände und Eigenschaften von Widgets und Bereichen beschrieben werden. Folgende Attribute stehen zur Verfügung:

Attribute für Widgets (Widget Attributes)

- `aria-autocomplete`
- `aria-checked` (state)
- `aria-disabled` (state)
- `aria-expanded` (state)
- `aria-haspopup`
- `aria-hidden` (state)
- `aria-invalid` (state)
- `aria-label`
- `aria-level`
- `aria-multiline`
- `aria-multiselectable`
- `aria-orientation`
- `aria-pressed` (state)
- `aria-readonly`
- `aria-required`
- `aria-selected` (state)
- `aria-sort`
- `aria-valuemax`
- `aria-valuemin`
- `aria-valuenow`
- `aria-valuetext`

Attribute für Ausgabebereiche (Live Region Attributes)

- `aria-atomic`
- `aria-busy` (state)
- `aria-live`
- `aria-relevant`

Attribute für Drag-and-Drop (Drag-and-Drop-Attributes)

- `aria-dropeffect`
- `aria-grabbed` (state)

Attribute für Beziehungen (Relationship Attributes)

- `aria-activedescendant`
- `aria-controls`
- `aria-describedby`
- `aria-flowto`
- `aria-labelledby`
- `aria-owns`
- `aria-posinset`
- `aria-setsize`

Universalattribute

Die folgenden Zustände und Eigenschaften können für alle Rollen (und unabhängig von einer Rolle) verwendet werden:

- `aria-atomic`
- `aria-busy` (state)
- `aria-controls`
- `aria-describedby`
- `aria-disabled` (state)
- `aria-dropeffect`
- `varia-flowto`
- `aria-grabbed` (state)
- `aria-haspopup`
- `aria-hidden` (state)
- `aria-invalid` (state)
- `aria-label`
- `aria-labelledby`
- `aria-live`
- `aria-owns`
- `aria-relevant`

Alle anderen Attribute dürfen nur in Verbindung mit einer bestimmten Rolle verwendet werden. Im folgenden sollen einige der Universalattribute vorgestellt werden.

3.1 aria-labelledby und aria-label

Das Attribute `aria-labelledby` wird dazu verwendet, auf das Label für ein Elemente zu verweisen.

In Listing 3.1 wird `aria-labelledby` dazu verwendet auf den Titel des Bereiches zu verweisen. `aria-labelledby` kann auch mehrere IDs enthalten. Diese sind mit einem Leerzeichen zu trennen.

`aria-labelledby` ist *kein* Ersatz für die Verwendung von `<label for="...">` in Formularen.

`aria-label` dient ebenfalls dazu ein Label festzulegen. Das Label wird allerdings direkt als Wert des Attributes angegeben. `aria-label` ist immer dann zu verwenden, wenn das

```
<div role="main" aria-labelledby="page_title">
    <h1 id="page_title">Top News Stories</h1>
    ... main page contents here ...
</div>
```

Listing 3.1: Verwendung von `aria-labelledby`

Label für sehende Benutzer nicht sichtbar sein soll oder kann (zum Beispiel aus Design-Gründen).

3.2 aria-describedby

`aria-describedby` erfüllt ähnliche Aufgaben wie `aria-labelledby`. Der Wert ist ebenfalls eine Liste von IDs. `aria-describedby` soll dazu verwendet werden auf eine ausführliche Beschreibung eines Elementes (Widgets) zu verweisen (Kontextsensitive Hilfe).

3.3 aria-busy

Über JavaScript werden in vielen Webanwendungen einzelne Bereiche aktualisiert. Zum Teil können diese Aktualisierungen einige Zeit in Anspruch nehmen. In diesem Fall wird üblicherweise ein Grafik (Throbber, Sanduhr) als Hinweis angezeigt. Für blinde Nutzer ist nicht erkennbar etwas passiert. Über das Attribut `aria-busy="true"` kann angezeigt werden dass ein Bereich gerade aktualisiert wird.

3.4 aria-invalid

Dieses Attribut kann dazu verwendet werden, Elemente mit fehlerhaften Eingaben zu markieren. Auf diese Weise wird die Information das beispielsweise in Textfeld falsch ausgefüllt würde assistierender Technologie zugänglich gemacht. Diese kann die Information dann in geeigneter Weise dem Benutzer zugänglich machen.

Literatur

[1] Craig, James und Michael Cooper: *Accessible Rich Internet Applications (WAI-ARIA) 1.0*, April 2014. http://www.w3.org/TR/wai-aria/complete.

[2] Garrish, Matt, Tzviya Siegman, Markus Gylling und Shane McCarron: *Digital publishing wai-aria module 1.0. w3c first public working draft 07 july 2015.*, Juli 2015. http://www.w3.org/TR/dpub-aria-1.0/.

[3] Pappas, Lisa, Richard Schwerdtfeger und Michael Cooper: *Wai-aria 1.0 primer - an introduction to rich internet application accessibility challenges and solutions*, 2010. http://www.w3.org/TR/wai-aria-primer/.