



FORMATION ACCESSIBILITÉ

ARIA

(ACCESSIBLE RICH INTERNET APPLICATIONS)

SOMMAIRE

1. qu'est-ce que c'est
2. Pourquoi ? le contexte
3. ce qu'il fait
4. ce qu'il ne fait pas
5. à quoi sert un rôle ?
6. exemples de rôles
7. à quoi sert une propriété ?
8. exemples de propriétés
9. à quoi sert un état .
10. exemples d'états
11. rappels sur tabindex
12. exemple du panneau d'onglets (ressources)

QU'EST-CE QUE C'EST ?

ARIA est une norme publiée par le W3C – WAI.

Elle est destinée à rendre accessible les interfaces riches :

- Pages mettant en œuvre des blocs d'interface évolués (panneau d'onglets, potentiomètres),
- Pages comportant des contenus mis à jour dynamiquement
- Page exécutant des requêtes serveur sans rechargement complet de la page (AJAX)

POURQUOI ?

L'évolution du web a nécessité de créer des éléments d'interface qui n'existaient pas dans le html (panneau d'onglets, vue en arbre, etc.)

ou de re créer des éléments existants dont on ne pouvait pas modifier l'apparence (case à cocher, bouton radio, etc.)



utilisation d'éléments existant, à d'autres usages que leur usage natif, en leur ajoutant des comportements via Java Script

Information transmise à la synthèse vocale insuffisante ou erronée



Mauvaise vocalisation

CE QU'IL FAIT

Ajout de **rôles, propriétés et états** aux éléments dont le rôle natif ou le comportement a été modifié



Transmission d'informations justes et complètes aux synthèses vocales



Vocalisation correcte (seul but d'ARIA)

CE QU'IL NE FAIT PAS

Comportement
et gestion des
interactions



Doit être fait par
JavaScript

Présentation
visuelle



Doit être géré en
CSS

Navigation
clavier



Doit être géré
avec *tabindex*

A QUOI SERT UN RÔLE

- modifie le rôle natif d'un élément
par exemple élément de liste ➡ onglet

`` ➡ `<li role="tab">`

- Donne un rôle aux éléments qui n'en ont aucun (div, span)
par exemple div ➡ bouton

`<div>Pause</div>` ➡ `<div role="button">Pause</div>`

EXEMPLES DE RÔLES (1/2)

- list → *liste d'éléments*
- listitem → *élément d'une liste*
- tablist → *liste d'onglets*
- tab → *onglet*
- tabpanel → *panneau d'onglet*
- checkbox → *case à cocher*
- presentation → *Supprime tout rôle natif*

EXEMPLES DE RÔLES (2/2)

Les rôles de type « landmark » permettent d'identifier les différentes zones de la page :

- banner → *zone d'entête*
- main → *contenu principal*
- contentinfo → *pied de page*
- navigation → *élément de navigation important*
- search → *zone du moteur de recherche*

A QUOI SERT UNE PROPRIÉTÉ

- Donne une information supplémentaire (fixe) sur un élément,
par exemple le caractère obligatoire d'un champ de saisie :

<input id="prenom" type="text" aria-required="true">

- Autre exemple donne une étiquette à un champ de saisie :
<input id="mdebut" aria-label="mois de date de début")>

EXEMPLES DE PROPRIÉTÉS

- *aria-label* → *donne un libellé caché*
- *aria-labelledby* → *donne pour libellé un texte apparent*
- *aria-describedby* → *ajoute au libellé un texte apparent*
- *aria-valuemax* → *définit une valeur maxi de saisie*
- *aria-valuemin* → *définit une valeur mini de saisie*

A QUOI SERT UN ÉTAT

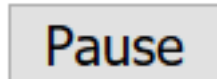
- Donne une information supplémentaire (vraie à un instant donné susceptible d'évoluer) sur un élément, par exemple le fait qu'un bouton soit enfoncé:
`<input " type="button" aria-pressed="true">`
- Autre exemple, indique que l'accordéon piloté par ce bouton est déplié :
`<input " type="button" aria-expanded="true">`
- Autre exemple, indique que la case à cocher est cochée :
`<div role="checkbox" aria-checked="true">déSSERT</div>`

EXEMPLES D'ÉTATS

- *aria-checked* ➡ *état coché (checkbox ou radio bouton)*
- *aria-expanded* ➡ *état déplié (accordéon)*
- *aria-hidden* ➡ *masqué à la synthèse vocale*
- *aria-pressed* ➡ *bouton enfoncé*
- *aria-selected* ➡ *element sélectionné*

PRÉFÉRER LES ÉLÉMENTS NATIFS (1/2)

```
<button>Pause</button>
```



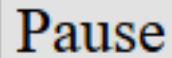
- *Vocalisé "Bouton pause"*
- *Même sans CSS on reconnaît un bouton;*
- *On peut l'atteindre avec le TAB*
- *On peut l'activer avec ENTER et ESPACE*

```
<div>Pause</div>
```

Pause

- *Vocalisé "pause"*
- *Sans CSS on ne reconnaît pas un bouton;*
- *On ne peut pas l'atteindre avec le TAB*
- *Pas activable au clavier*




PRÉFÉRER LES ÉLÉMENTS NATIFS (2/2)



```
<div class="boutonpause" role="button" tabindex="0">Pause</div>

.boutonpause {
  border:1px solid #B5B5B5;
  padding-left:4px;
  padding-right:4px;
  background-color:#E2E2E2;
}
```

Pour obtenir le même résultat qu'avec un bouton natif il a fallu :

- Ajout de styles  *CSS apparence d'un bouton*
- Ajout du rôle Bouton  *Vocalisé "Bouton pause"*
- Ajout du tabindex  *On peut l'atteindre avec le TAB et l'activer*

RAPPEL SUR TABINDEX

- `tabindex="0"` ➡ *permet d'adresser le focus clavier et par Javascript*
- `tabindex="-1"` ➡ *permet d'adresser le focus par Javascript*

EXEMPLE DU PANNEAU D'ONGLETS

```
<ul role="tablist">
  <li role="tab" tabindex="-1" aria-selected="false" aria-controls="panneau-1">Onglet 1</li>
  <li role="tab" tabindex="0" aria-selected="true" aria-controls="panneau-2">Onglet 2</li>
  <li role="tab" tabindex="-1" aria-selected="false" aria-controls="panneau-3">Onglet 3</li>
</ul>

<div role="tabpanel" id="panneau-1" aria-hidden="true">
  contenu du panneau 1 masqué par css
</div>
<div role="tabpanel" id="panneau-2" aria-hidden="false">
  contenu du panneau 2
</div>
<div role="tabpanel" id="panneau-3" aria-hidden="true">
  contenu du panneau 3 masqué par css
</div>
```

- Spécifications WAI (en anglais) : <https://www.w3.org/TR/2017/WD-wai-aria-practices-1.1-20170628/#tabpanel>
- Exemple : <https://www.w3.org/TR/2017/WD-wai-aria-practices-1.1-20170628/examples/tabs/tabs-2/tabs.html>