Français ▾

Check-list pour l'accessibilité mobile

Ce document fournit une liste concise des points à vérifier par les développeurs pour garantir l'accessibilité d'une application mobile. Ce document est amené à évoluer pour tenir compte de nouvelles bonnes pratiques.

Couleur

- Le contraste des couleurs **DOIT** respecter le niveau AA de WCAG 2.0 :
  - Un contraste dont le ratio est de 4.5:1 pour les textes normaux (dont la fonte est inférieure à 18 points ou 14 points en gras) ;
  - Un contraste dont le ratio est de 3:1 pour les grands textes (18 points minimum ou 14 points en gras).
- L'information véhiculée par la couleur **DOIT** toujours être disponible via un autre moyen (ex : les liens peuvent être bleus mais aussi soulignés, etc.).

> **Note** :
>
> - Sur Yoyo Design vous trouverez l'outil nColor de sélection des couleurs. Il vous permettra de simuler les différents types de visions daltoniennes ;
> - Jon Snook a écrit un outil intéressant permettant de vérifier le contraste des couleurs : Colour Contrast Checker (en) ;
> - Alternativement, le *Tanaguru Contrast-Finder* fait un travail similaire, mais il propose en plus une palette de couleurs similaires mais offrant un meilleur contraste.

La visibilité

- Les techniques de masquage du contenu comme une opacité nulle, l'ordre des z-index et la position hors-écran **NE DOIVENT PAS** être utilisées aux seules fins de visibilité ;
- Tout ce qui n'apparaît pas dans la partie visible de l'écran **DOIT** réellement être invisible (en particulier pour les applications d'une page comportant plusieurs « cartes ») :
  - Il **FAUT** utiliser l'attribut `hidden` ou les propriétés de style `visibility` ou `display` ;

    - À moins d'être absolument indispensable, l'attribut `aria-hidden` **NE DOIT PAS** être utilisé.

Le focus

- Tous les éléments activables **DOIVENT** pouvoir porter le focus :
    - Les contrôles standards tels que les liens, les boutons et les champs de formulaire peuvent, par défaut, porter le focus ;
    - Les contrôles non standards **DOIVENT** être assignés à un rôle ARIA, comme `button`, `link` ou `checkbox`.
- Le focus doit être géré de façon logique et cohérente.

Les équivalents textuels

- Les équivalents textuels **DOIVENT** être fournis pour chacun des éléments de l'application qui n'est pas strictement lié à la mise en forme :
    - Les attributs `alt` et `title` doivent être utilisés aux endroits appropriés (*lire l'article de* Steve Faulkner sur L'utilisation de l'attribut HTML `title` (en)) ;
    - Si les attributs ci-dessus ne sont pas appliquables, on utilise les propriétés ARIAappropriées comme `aria-label`, `aria-labelledby` et `aria-describedby`.
- Les images avec du texte **DOIVENT** être évitées ;
- Tous les contrôles de formulaire **DOIVENT** posséder des éléments `<label>` pour permettre aux lecteurs d'écran de les utiliser.

La gestion des états

- Les contrôles standards comme les boutons radio, les cases à cocher, sont gérés par le système d'exploitation. En revanche, pour les contrôles spécifiques, les changements d'états doivent être fournis via les états ARIA tels que `aria-checked`, `aria-disabled`, `aria-selected`, `aria-expanded` et `aria-pressed`.

Principales recommandations

- Un titre **DOIT** être fourni pour désigner l'application ;
- Les titres **DOIVENT** respecter une structure hiérarchique :

```
<h1>Titre de premier niveau</h1>
  <h2>Titre de deuxième niveau</h2>
  <h2>Un autre titre de niveau 2</h2>
    <h3>Un titre inférieur</h3>
```

- Les points de repères ARIA **DOIVENT** être utilisés pour décrire une application ou la structure d'un document
comme `banner`, `complementary`, `contentinfo`, `main`, `navigation`, `search` ;
- Les gestionnaires d'événements tactiles **NE DOIVENT PAS** être déclenchés avant l'événement `touchend` ;
- Les points d'interaction tactiles **DOIVENT** être suffisamment grands pour garantir une bonne interaction (voir les recommandations de la BBC sur l'accessibilité mobile (en) sur ce sujet).

Note : Le service de test automatique d'accessibilité de Tanaguru fournit un moyen pratique de corriger les erreurs d'accessibilité pouvant se glisser dans une page Web, ou d'une application Web locale (par exemple Firefox OS). Vous trouverez plus d'informations concernant l'implémentation technique de Tanaguru, et comment contribuer au projet, sur tanaguru.org.

Note : la version originale, anglaise, de ce document a été écrite par Yura Zenevich.