Définition

Related to Accessibilité
Cela inclut le linting code (Lint)
la gestion des préfixes CSS,
l'utilisation des outils de dev des navigateurs pour localiser les problèmes,
utiliser des polyfills pour apporter du support dans les navigateurs,
se confronter aux problèmes de responsive design et plus encore.

Introduction au test en navigateur croisé - Apprendre le développement web | MDN

Voir : Le test de navigateur croisé

Infos supplémentaires

- **Intégrer l'accessibilité clavier**
- **Tests**
- **Responsive design**
  - Faire du Mobile first
- **Performance**
- **Les textes alternatifs**
  - (alt)
- **Relations et contexte entre élément**
  - Par exemple, une liste de lien libellés "cliquez ici", "cliquez ici", etc. est vraiment mauvaise pour l'accessibilité. Il est préférable pour les textes de type lien d'avoir du sens en contexte et hors contexte.
  - Pour les formulaires, un label doit être associés à chaque champs !
  - Les tableaux doivent être sémantiquement correct : learning-area/punk-bands-complete.html at main · mdn/learning-area · GitHub
- CSS
  - Ne pas utiliser de balises ""
  - Ne pas utiliser des h1-h2-h3 pour la taille des textes
  - Vous devez vous assurez que les éléments interactifs comme les boutons et les liens ont des états focus/hover/active appropriés configuré, pour donner à l'utilisateur un indice visuel de leur fonction. Si vous supprimez les styles par défaut pour des raisons stylistiques, assurez-vous de mettre en place des styles de remplacement.
- **Couleur et contraste**
  - WebAIM: Contrast Checker
- **Rendre invisible des éléments (seulement visuellement)**
  - WebAIM: CSS in Action - Invisible Content Just for Screen Reader Users
  - voir : Screen Readers - Tailwind CSS (ils utilisent une technique pour rendre un bloc "sr-only")
- JavaScript
  - N'utilisez pas les éléments `<div>` avec du code JavaScript pour simuler une fonctionnalité si c'est possible — c'est une source d'erreur, et ça fonctionne mieux d'utiliser les fonctionnalités disponibles qu'HTML vous fournit. (***pas de on-click en Js !***)
  - WebAIM: Accessible JavaScript - Overview of Accessible JavaScript
  - WAI-ARIA pour du contenu mis à jour dynamiquement, et qu'il soit quand même lu par les lecteurs d'écrans (voir l'attribut "aria-live") & Accessible Rich Internet Applications (WAI-ARIA) 1.1
- **Checklist de tests d'accessibilité**
  - Assurez-vous que votre HTML est sémantiquement correct au possible. Le valider est un bon début, comme utiliser un outil d'Audit.
  - Vérifiez que votre contenu a du sens lorsque le CSS est désactivé.
  - Assurez-vous que votre fonctionnalité est accessible au clavier. Testez en utilisant Tab, Retour/Entrée, etc.
  - Assurez-vous que votre contenu non-textuel a un texte alternatif. Un Outil d'audit est bien pour repérer ce type de problèmes.
  - Assurez-vous que votre contraste de couleurs est acceptable, en utilisant un outil de vérification approprié.
  - Assurez-vous que le contenu caché est visible par les lecteurs d'écran.
  - Assurez-vous qu'une fonctionnalité est utilisable sans JavaScript autant que possible.
  - Utilisez ARIA pour améliorer l'accessibilité quand c'est approprié.
  - Exécutez votre site dans un Outil d'audit.
  - Testez avec un lecteur d'écran.

- Incluez une politique/déclaration d'accessibilité à un endroit que l'on peut trouver sur votre site pour dire ce que vous avez fait.

Liens et ressources liées

Gérer les problèmes courants en HTML et CSS - Apprendre le développement web | MDN
Outils :

- Classique
  - Tenon.io | Home page
  - tota11y – an accessibility visualization toolkit
- Automatisé
  - axe: Accessibility Testing Tools and Software
- IBM : Chrome Web Store - Extensions