Notice d'accessibilité HTML et CSS

Date	Version	Auteur	État / commentaires
17 août 2017	3	Atalan	Mise à jour (mineure) de contenus dans les recommandations. Cette version prend en compte WCAG 2.0 et RGAA version 3 2017.
10 août 2020	4.0-v1	Atalan	Mises à jour majeures pour être conforme avec le RGAA 4 et avec les WCAG 2.1 (niveaux A et AA).
21 janvier 2020	4.0-v2	Atalan	Mise à jour de la charte graphique.
15 mars 2021	4.1-v1	Atalan	Mises à jour mineures pour être conforme avec le RGAA 4.1.
14 décembre 2021	4.1-v2	Atalan	Corrections (mineurs) des fiches 7.2 et 7.4
15 juin 2023	4.1-v3	Atalan	Optimisation de l'accessibilité du document.
22 août 2023	4.1.2-v1	Atalan	Mise à jour (mineure) des contenus.
20 août 2024	4.1.2-v2	Atalan	Mise à jour des contenus.

Sommaire

Introduction

Contexte et objectifs

Cette notice liste les règles à respecter pour que l'intégration HTML et CSS soit accessible.

Cette notice s'inscrit dans un lot de quatre notices téléchargeables sur le site www.accede-web.com :

- Notice d'accessibilité fonctionnelle et graphique.
- **Notice d'accessibilité HTML et CSS (présente notice).**
- Notice d'accessibilité des principaux composants d'interface riche.
- Notice d'accessibilité éditoriale (modèle).

À qui s'adresse ce document ? Comment l'utiliser ?

Ce document doit être transmis aux intervenants et/ou prestataires réalisant les spécifications techniques, l'intégration HTML/CSS. Il vient en complément aux spécifications techniques d'un projet. Les recommandations peuvent être complétées ou retirées selon les contextes d'utilisation, ce travail peut être notamment réalisé par la maîtrise d'ouvrage.

Les recommandations doivent être prises en compte en phase d'intégration HTML/CSS et, pour certaines d'entre elles, lors de la dynamisation des pages lorsque, par exemples, des gabarits sont affectés à un outil de gestion de contenu (CMS).

Remarque

La version en ligne de cette présente notice est agrémentée de nombreux exemples, liens vers des ressources complémentaires, etc. Celle-ci est disponible à l'adresse : https://www.accede-web.com/notices/html-et-css/

Contact

Pour toute remarque à propos de cette notice, merci de contacter Atalan, à l'initiative du projet AcceDe Web à l'adresse suivante : accede@atalan.fr.

Vous pouvez également trouver plus d'informations sur les notices méthodologiques du projet AcceDe Web sur le site www.accede-web.com.

Licence d'utilisation

1. Structure générale

1.1 Structurer la zone d'entête principale avec `<header role="banner">`

La zone d'entête principale des pages, qui peut notamment contenir le logo et le moteur de recherche, doit être balisée avec `<header role="banner">`.

⚠ Attention

Tandis que la balise `<header>` peut être utilisée plusieurs fois dans une page web, `role="banner"` ne doit lui être employé qu'une seule fois.

ℹ Remarque

Il est tout à fait possible d'imbriquer plusieurs rôles ARIA :
`<div role="search">` dans `<header role="banner">`, par exemple.

```
<header role="banner">
   <img src="mon-site-logo.png" alt="Mon site (logo)" />
   <div role="search" …>[…]</div>
   […]
</header>
```

1.2 Identifier le moteur de recherche avec `role="search"`

Le moteur de recherche du site doit être identifié avec `role="search"`.

```
<form role="search" […]>
    <input type="search" title="Recherche par mots-clés" />
    <input type="submit" value="Rechercher" />
</form>
```

1.3 Structurer la zone de contenu principal avec `<main role="main">`

La zone de contenu principal des pages doit être balisée avec `<main role="main">`.

⚠ Attention

Une page ne doit contenir qu'une seule balise `<main role="main">` **visible**.

```
<header role="banner">[…]</header>
<nav role="navigation">[…]</nav>
<main role="main">[…]</main>
<footer role="contentinfo">[…]</footer>
```

1.4 Structurer les informations relatives au site avec `<footer role="contentinfo">`

Les informations relatives au site, comme par exemple le copyright ainsi que les liens « Mentions légales » et « Crédits », doivent être structurées avec `<footer role="contentinfo">`.

⚠ Attention

Une page ne doit contenir qu'un seul `role="contentinfo"`.

```
<header role="banner">[…]</header>
<nav role="navigation">[…]</nav>
<main role="main">[…]</main>
<footer role="contentinfo">
   <ul>
      <li><a href="…">Plan du site</a></li>
      <li><a href="…">Crédits</a></li>
      <li><a href="…">Mentions légales</a></li>
   </ul>
   <p>© Mon site</p>
</footer>
```

1.5 Structurer les menus de navigation principaux et secondaires avec `<nav role="navigation">`

Les menus de navigation principaux et secondaires doivent être balisés avec `<nav role="navigation">`.

Remarque

La balise `<nav role="navigation">` est à réserver uniquement pour englober des menus contenant des **liens internes** au site.

Par exemple :

- Le menu principal.
- Le menu secondaire.
- Le fil d'Ariane.
- La pagination d'une page de résultats de recherche.

Autrement dit, elle ne doit pas être utilisée pour englober une liste de liens pointant vers des sites externes. Comme une liste de liens vers des réseaux sociaux, par exemple.

Astuce

Une bonne pratique consiste à ajouter l'attribut `aria-label` dans la balise `<nav role="navigation">` et de le renseigner en précisant le type de système de navigation dont il est question.

```
<header role="banner">[…]</header>
<nav role="navigation" aria-label="Menu principal">[…]</nav>
<nav role="navigation" aria-label="Menu secondaire">[…]</nav>
<main role="main">
   <nav role="navigation" aria-label="Vous êtes ici">[…]</nav>
   […]
</main>
<footer role="contentinfo">[…]</footer>
```

1.6 Structurer les menus de navigation avec des listes

Utiliser des listes non-ordonnées `<ul>` et `<li>` pour baliser les menus de navigation.

Dans le cas d'un menu principal à plusieurs niveaux, veiller à la bonne imbrication des listes :

```
<nav role="navigation" aria-label="Menu principal">
 <ul>
  <li><a href="…">Premier lien du menu</a></li>
  <li>
    <a href="…">Deuxième lien du menu</a>
    <ul>
      <li><a href="…">Premier lien du sous-menu</a></li>
      <li><a href="…">Deuxième lien du sous-menu</a></li>
    </ul>
  </li>
  <li><a href="…">Troisième lien du menu</a></li>
 </ul>
</nav>
```

1.6.1 Quels bénéfices ?

La structuration des listes est essentielle pour les **personnes utilisant un lecteur d'écran (personnes aveugles et malvoyantes)**. Cette structuration leur permet ainsi, à la rencontre d'une liste de :

- Naviguer de liste en liste au sein d'une page.
- Connaître dès le départ le nombre d'éléments dans celle-ci.
- Naviguer plus facilement dans la liste :
    - Passer directement à la fin de la liste si ce contenu ne les intéresse pas.
    - Revenir facilement au début de la liste.

1.7 Mettre en place une hiérarchie de titres logique et exhaustive avec les balises <h1> à <h6>

Sur chaque page, pour baliser les titres, utiliser les balises de titres allant de `<h1>` jusqu'à `<h6>`. La structure des titres doit être à la fois logique et exhaustive.

C'est-à-dire :

- Que tous les éléments qui ont valeur de titres doivent être balisés comme tels.
- Qu'il ne doit pas y avoir d'incohérences dans la hiérarchie des titres.
- Qu'idéalement, il ne doit pas y avoir de « sauts » dans la structure des titres (passage brutal d'un `<h1>` à un `<h3>` sans `<h2>` intermédiaire, par exemple).

Remarques

- Il n'est jamais gênant d'utiliser plusieurs `<h1>` dans une page si plusieurs titres de premier niveau sont présents.
- Une bonne pratique d'accessibilité consiste à ne pas ajouter de titres cachés.

Astuce

Pour mettre en place une hiérarchie de titres logique et exhaustive, il faut imaginer que les titres forment la « table des matières » de la page. Est-elle logique ? Exhaustive ?

Remarque

Le role ARIA `heading` associé à un attribut `aria-level` (renseigné avec une valeur allant de 1 à 6) permet d'affecter la valeur d'un titre à n'importe quelle balise HTML.

Concrètement, par exemple :

- `<p role="heading" aria-level="1">Titre de niveau 1</p>` est sémantiquement équivalent à `<h1>Titre de niveau 1</h1>`.
- `<div role="heading" aria-level="3">Titre de niveau 3</div>` à `<h3>Titre de niveau 3</h3>`.

Cette technique n'étant toutefois pas optimale pour l'accessibilité, elle est à utiliser en ultime recours.

1.7.1 Exemples

Ci-dessous, 3 exemples de structuration de titres pour cette page. Les deux premiers sont corrects tandis que le troisième est incorrect.

1.7.1.1 Exemple 1 (correct)

```
<h1><a href="/"><img src="lafarge.png" alt="Lafarge - Construire
des villes meilleures (aller à l'accueil)" /></a></h1>
[…]
<h2>Les sables & granulats décoratifs et esthétiques Lafarge</h2>
[…]
<h3>Contacts</h3>
[…]
<h3>À lire également</h3>
[…]
<h3>Nos réalisations Granulats</h3>
[…]
<h4>Pologne : notre béton prend la route pour rapprocher les
villes</h4>
[…]
<h4>Villes plus durables : Un nouveau stade en granulats recyclés
pour la ville de Guelph (Canada)</h4>
[…]
<h4>Trafic fluidifié et économie boostée grâce à la rocade
d'Edmonton</h4>
[…]
```

Dans cet exemple de code HTML, la structure des titres est logique et exhaustive.

1.7.1.2 Exemple 2 (correct)

```
<a href="/"><img src="lafarge.png" alt="Lafarge - Construire des
villes meilleures (aller à l'accueil)" /></a>
[…]
<h1>Les sables & granulats décoratifs et esthétiques Lafarge</h1>
[…]
<h2>Contacts</h2>
[…]
<h2>À lire également</h2>
[…]
<h2>Nos réalisations Granulats</h2>
[…]
<h3>Pologne : notre béton prend la route pour rapprocher les
villes</h3>
[…]
<h3>Villes plus durables : Un nouveau stade en granulats recyclés
pour la ville de Guelph (Canada)</h3>
[…]
<h3>Trafic fluidifié et économie boostée grâce à la rocade
d'Edmonton</h3>
[…]
```

Dans cet exemple de code HTML, la structure des titres est également logique et exhaustive.

À noter que d'autres structurations de titres sont envisageables.

1.7.1.3 Exemple 3 (incorrect)

```
<a href="/"><img src="lafarge.png" alt="Lafarge - Construire des villes meilleures (aller à l'accueil)" /></a>
[…]
<h1>Les sables & granulats décoratifs et esthétiques Lafarge</h1>
[…]
<p class="titre">Contacts</p>
[…]
<h2>À lire également</h2>
[…]
<h2>Nos réalisations Granulats</h2>
[…]
<h2>Pologne : notre béton prend la route pour rapprocher les villes</h2>
[…]
<h2>Villes plus durables : Un nouveau stade en granulats recyclés pour la ville de Guelph (Canada)</h2>
[…]
<h2>Trafic fluidifié et économie boostée grâce à la rocade d'Edmonton</h2>
[…]
```

Dans cet exemple de code HTML, la structure des titres est incorrecte car elle comporte des incohérences (les titres des réalisations sont au même niveau que le titre de section « Nos réalisations granulats »).

Aussi, le titre « Contacts » n'est pas balisé comme tel.

2. Titre de la page

2.1 Renseigner un <title> précis sur chaque page

Sur chaque page, la balise <title> doit être renseignée avec précision.

Elle doit au minimum annoncer le nom de la page courante ainsi que le nom du site.

Astuces

- Il est recommandé de faire apparaître le nom de la page courante en premier dans la balise <title>. Il est par exemple possible d'opter pour <title>[Nom de la page courante] | [Nom du site]</title>.
- Une bonne pratique d'accessibilité consiste à veiller à la cohérence de l'ordre des contenus de la balise <title> sur l'ensemble des pages du site.

Remarque

Parfois, certaines pages sont rechargées avec un contenu modifié. C'est par exemple le cas :

- Lors de l'utilisation de la pagination.
- Lorsqu'une expression est recherchée via le formulaire de recherche.

Il faut alors veiller à modifier le titre de la page en conséquence, par exemple :

```
<title>Résultats de votre recherche sur "[Expression recherchée]"
(page 3/7) | [Nom du site]</title>
```

3. Langues

3.1 Renseigner la langue principale de la page avec l'attribut `lang` sur la balise `<html>`

Afin de garantir la bonne restitution des contenus textuels, une déclaration de langue principale doit être effectuée sur chaque page.

Utiliser pour cela l'attribut `lang` sur la balise `<html>`.

Par exemple, pour une page en français :

```
<html lang="fr">
```

Astuce

Pour renseigner l'attribut `lang`, un code de langue sur deux lettres (ou, s'il n'est pas disponible, sur trois lettres) doit être utilisé.

Les principaux codes utilisés sont les suivants :

- `fr` pour le français.
- `en` pour l'anglais.
- `es` pour l'espagnol.
- `de` pour l'allemand.
- `it` pour l'italien.

Les autres sont disponibles dans cette liste exhaustive des codes de langues autorisés.

3.2 Utiliser l'attribut `lang` pour signaler les changements de langue

Si des contenus sont proposés dans une langue différente de la langue principale, alors ils doivent être signalés avec l'attribut `lang`.

Par exemple, dans le cas d'une page en français :

```
<a href="…" lang="en" hreflang="en">English version</a>
```

Astuce

Si aucune balise n'encadre directement le contenu en langue étrangère, alors utiliser la balise `<span>` ou `<div>` et renseigner son attribut `lang`.

Remarque

Signaler le changement de langue n'est pas demandé pour :

- Les noms propres.
- Les mots d'origine étrangère intégrés dans le dictionnaire de la langue principale.
- Tous les mots d'origine étrangère mais qui se prononcent et se comprennent correctement avec l'accent de la langue principale (« podcast », par exemple, si la langue principale est le français).

4. Grammaire HTML et sémantique

4.1 Écrire un code HTML valide selon les règles de grammaire du DOCTYPE utilisé

Sur chaque page, un DOCTYPE valide doit être utilisé et le code HTML **généré** doit être conforme aux règles de grammaire de ce dernier (le choix du DOCTYPE est libre).

Remarque

Ce DOCTYPE doit impérativement être intégré avant la balise `<html>`.

Concernant les règles de grammaire, il est particulièrement important de veiller à :

- La correcte imbrication des balises.
- La correcte ouverture et fermeture des balises.
- L'absence d'attributs dupliqués sur une même balise.
- L'unicité des valeurs de l'attribut id au sein d'une même page.

Attention

- Les balises et attributs HTML obsolètes et/ou destinés exclusivement à la mise en forme ne doivent pas être utilisés.
- Afin de valider les règles de grammaire, il est nécessaire de vérifier le code HTML **généré**, en le copiant par exemple dans l'inspecteur de code (via le raccourci F12 par exemple) et en le collant sur le site https://validator.w3.org/nu/#textarea.
  Récupérer le code source de la page (via le raccourci CTRL + U par exemple) ou copier-coller l'URL de sa page dans le validateur **ne permet pas** de tester complétement ce point.

4.2 Employer les balises HTML pour leur valeur sémantique

Les balises HTML doivent être utilisées pour leur valeur sémantique et non pour leur rendu visuel.

Balises	Bon usage	Mauvais usage
`<a>`	Baliser un lien hypertexte	Obtenir un bouton d'action sans arrière-plan et bordure
`<hr />`	Séparer des sujets différents dans un bloc de texte	Obtenir une ligne séparatrice
`<fieldset>`	Regrouper des éléments de formulaires de même nature	Obtenir une bordure
`<q>`	Baliser une citation courte	Obtenir un texte entre guillemets

5. Liens et boutons

5.1 Différencier les boutons des liens

Réserver les boutons pour les actions et les liens pour la navigation.

5.1.1 Boutons

Les boutons `<button>` / `<input />` permettent de déclencher des actions.

Ils sont par exemple à employer pour soumettre des informations, afficher l'élément suivant ou précédent dans un carrousel, fermer une fenêtre modale, etc.

5.1.2 Liens

Les liens `<a>` permettent de naviguer.

Ils sont à employer pour pointer sur une autre page ou sur une zone précise de la page courante par l'intermédiaire d'un système d'ancre.

5.2 Compléter les liens et les boutons non explicites avec `aria-label` ou `title`

Un lien ou un bouton considéré comme non explicite est un lien ou un bouton dont l'intitulé seul ne permet pas de comprendre sa destination ou sa fonction.

- Les liens « Lire la suite », « En savoir plus », « Plus d'informations », « Cliquer ici » sont par exemple considérés comme non explicites par nature.
- Idem pour le bouton « Ok ».

Dans ces situations l'attribut `aria-label` ou l'attribut `title` peut alors être utilisé pour préciser la destination du lien ou la fonction du bouton.

Prenons ci-dessous l'exemple d'un lien « Lire la suite » pointant vers la page du projet AcceDe Web pour présenter ces deux techniques.

5.2.1 Attribut `aria-label`

Rendre un lien ou un bouton explicite via l'attribut `aria-label` consiste à :

1. Ajouter l'attribut `aria-label` dans la balise `<a>`, `<button>` ou `<input />`.
2. Renseigner cet attribut en reprenant à l'identique la valeur de l'intitulé du lien ou du bouton lui-même puis avec les informations permettant de rendre explicite le lien ou le bouton.

```
<a href="…" aria-label="Lire la suite : Le projet AcceDe Web">
   Lire la suite
</a>
```

5.2.2 Attribut `title`

Remarque

Le support de l'attribut `title` par les navigateurs et technologies d'assistance est partiel.

Bien qu'il s'agisse d'une technique conforme, l'utilisation de l'attribut `aria-label` est donc à privilégier. Il est d'ailleurs tout à fait possible d'utiliser ces deux attributs conjointement pour un meilleur support.

Rendre un lien ou un bouton explicite via l'attribut `title` consiste à :

1. Ajouter l'attribut `title` dans la balise `<a>`, `<button>` ou `<input />`.
2. Renseigner cet attribut en reprenant à l'identique la valeur de l'intitulé du lien ou du bouton lui-même puis avec les informations permettant de rendre explicite le lien ou le bouton.

```
<a href="…" title="Lire la suite : Le projet AcceDe Web">
   Lire la suite
</a>
```

Astuce

Une bonne pratique consiste à renseigner ces attributs en commençant par la reprise de l'intitulé puis en terminant par l'information permettant de rendre explicite le lien ou le bouton.

Remarque

Une de ces deux techniques doit également être utilisée pour distinguer les liens ou boutons dont les intitulés pourraient être considérés comme explicites, mais qui pointent vers des pages ou déclenchent des actions différentes.

Par exemple, dans le cas de deux boutons « Rechercher » affichés sur la même page, et qui seraient à distinguer :

```
<input type="submit" value="Rechercher" aria-label="Rechercher une actualité" />
[…]
<input type="submit" value="Rechercher" aria-label="Rechercher une personne dans l'annuaire" />
```

Remarque

À noter qu'un lien peut être considéré comme explicite grâce à son contexte quand les éléments environnants permettent d'en comprendre le sens.

5.3 Ne pas utiliser les attributs `aria-label` et `title` sur des liens ou boutons explicites

Un lien ou un bouton considéré comme explicite est un lien ou un bouton dont l'intitulé seul et/ou le contexte permettent de comprendre sa destination ou sa fonction.

Par exemple :

- Le lien « Le projet AcceDe Web » est explicite par nature.
- Idem pour le bouton « Valider ma commande ».

Les attributs `aria-label` et `title` (qui peuvent servir à rendre accessibles des liens non explicites) ne doivent pas être employés pour ce type de liens.

Voici concrètement quelques exemples d'usages **incorrects** de l'attribut `aria-label` dans une balise `<input />` :

✖
```
<input type="submit" value="Valider ma commande" aria-label="" />
```

✖
```
<input type="submit" value="Confirmer ma commande" aria-label="Valider ma commande" />
```

Et quelques exemples d'usages **incorrects** de l'attribut `title` dans une balise `<a>` :

❌
```
<a href="…" title="">Le projet AcceDe Web</a>
```

❌
```
<a href="…" title="AcceDe Web">Le projet AcceDe Web</a>
```

6. Images et icônes

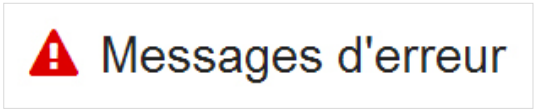
6.1 Gérer l'attribut `alt` des balises `<img />` et `<input type="image" />`

5 cas de figure :

- `<img />` décoratives/illustratives.
- `<img />` informatives « simples ».
- `<img />` informatives « complexes ».
- `<img />` liens ou boutons seules.
- `<input type="image" />`.

6.1.1 `<img />` décoratives/illustratives

Lorsqu'une balise `<img />` décorative ou illustrative est intégrée dans le code HTML, l'attribut `alt` doit être laissé vide (sans aucun espace entre les guillemets de `alt=""`).



Dans l'exemple de code HTML ci-après, la balise `<img />` accompagne simplement le titre « Messages d'erreur » (explicite par nature) :

```
<h2>
   <img src="erreur.png" alt="" />
   Messages d'erreur
</h2>
```

6.1.2 `<img />` informatives « simples »

Une image informative « simple » est une image informative qui peut être décrite par le biais d'une alternative textuelle **concise**.

Lorsqu'une balise `<img />` informative « simple » est intégrée dans le code HTML, renseigner son attribut alt avec une information égale ou équivalente à celle véhiculée par l'image.

Dans l'exemple de code HTML ci-après, la balise `<img />` indique qu'il s'agit d'une étape de marche :

```
<p>
  <img src="marche.png" alt="Marche : " />
  <span>2 min</span>
  <span>111 m</span>
</p>
```

Attention

Il est fortement déconseillé de rédiger le texte de remplacement d'une image en commençant par `alt="Image […]"`.

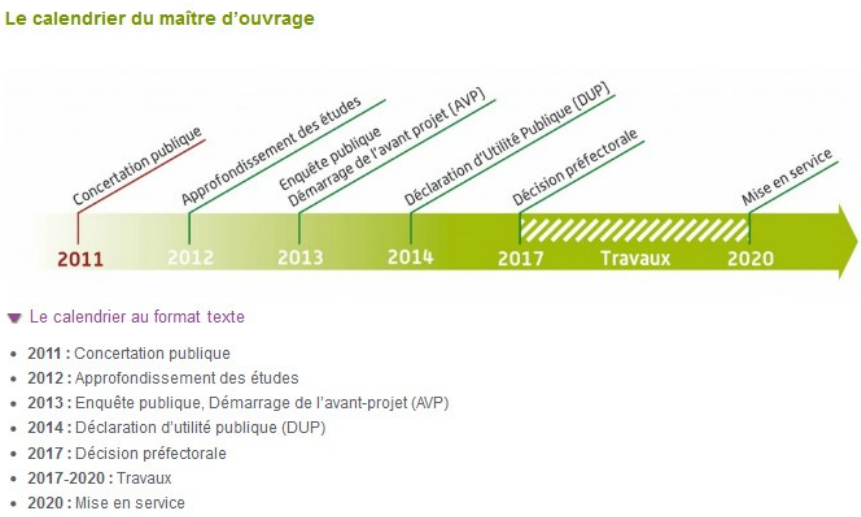
Cette information sera déjà annoncée par les technologies d'assistance lors de la lecture d'une balise `<img />`.

6.1.3 `<img />` informatives « complexes »

Une image informative « complexe » est une image informative qui nécessite une description **détaillée**.

Lorsqu'une balise `<img />` informative « complexe » est intégrée dans le code HTML :

1. Renseigner son attribut `alt` avec les informations permettant de rendre l'image explicite.
2. Proposer une description détaillée de l'image directement sous cette dernière.
3. Enfin, indiquer dans l'attribut `alt` où trouver cette description détaillée.



Dans cet exemple, un bouton plier/déplier permet d'afficher une description détaillée de l'image « complexe ».

Dans l'exemple de code HTML ci-après, la balise `<img />` indique à la fois la fonction de l'image et où trouver sa description détaillée :

```
<h2>Le calendrier du maître d'ouvrage</h2>
<img src="calendrier.png" alt="Calendrier du maître d'ouvrage et travaux prévus entre 2011 et 2020 (description détaillée ci-après)" />
<button aria-expanded="true">Le calendrier au format texte</button>
<ul>
  <li><strong>2011 :</strong> Concertation publique</li>
  [...]
</ul>
```

⚠ Attention

Il est fortement déconseillé de rédiger le texte de remplacement d'une image en commençant par `alt="Image […]"`.

Cette information sera déjà annoncée par les technologies d'assistance lors de la lecture d'une balise `<img />`.

6.1.4 `<img />` liens ou boutons seules

Lorsqu'une balise `<img />` seule (sans intitulé) servant de lien ou de bouton est intégrée dans le code HTML, intégrer les informations permettant de le rendre explicite dans son attribut `alt`.

Dans l'exemple de code HTML ci-après, la balise `<img />` pointe vers la page d'accueil d'un site :

```
<a href="/">
   <img src="accueil.png" alt="Accueil" />
</a>
```

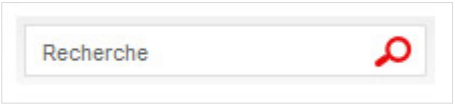
Attention

Il est fortement déconseillé de rédiger l'attribut `alt` d'une balise `<img />` servant de lien ou bouton en commençant respectivement par `alt="Lien vers [...]"` ou `alt="Bouton [...]"`.

Cette information sera déjà annoncée par les technologies d'assistance lors de la lecture d'une balise `<a>` ou `<button>`.

6.1.5 `<input type="image" />`

Lorsqu'une balise `<input type="image" />` est intégrée dans le code HTML, intégrer les informations permettant de rendre le bouton explicite dans son attribut `alt`.



Dans l'exemple de code HTML ci-après, la balise `<input type="image" />` permet de lancer une recherche dans un site :

```
<input type="image" src="loupe.png" alt="Lancer la recherche" />
```

Attention

Il est fortement déconseillé de rédiger l'attribut alt d'une balise `<input type="image" />` en commençant par `alt="Bouton [...]"`.

Cette information sera déjà annoncée par les technologies d'assistance lors de la lecture d'une balise `<input type="image" />`.

6.2 Gérer l'alternative des SVG (images vectorielles)

4 cas de figure :

- <svg> décoratifs/illustratifs.
- <svg> informatifs « simples ».
- <svg> informatifs « complexes ».
- <svg> liens ou boutons seuls.

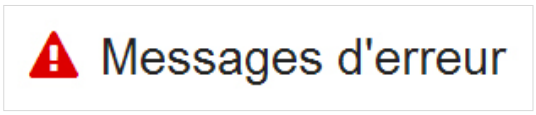
Attention

Par défaut, les balises <svg> reçoivent le focus clavier sous Internet Explorer.

Pour optimiser la navigation au clavier dans ce navigateur, l'attribut `focusable="false"` doit être ajouté sur chaque balise <svg> utilisée pour afficher une image.

6.2.1 <svg> décoratifs/illustratifs

Lorsqu'un <svg> est simplement décoratif ou illustratif, lui intégrer `aria-hidden="true"`.



Dans l'exemple de code HTML ci-après, le <svg> accompagne simplement le titre « Messages d'erreur » (explicite par nature) :

```
<h2>
   <svg aria-hidden="true" focusable="false">[…]</svg>
   Messages d'erreur
</h2>
```

Attention

Un <svg> simplement décoratif ou illustratif ne doit pas posséder d'attributs `title`, `aria-label`, `aria-labelledby` et/ou `aria-describedby`.

De la même manière, il ne doit pas posséder de balises <title> et/ou <desc>.

6.2.2 <svg> informatifs « simples »

Un <svg> informatif « simple » est une image informative qui peut être décrite par le biais d'une alternative textuelle **concise**.

Lorsqu'un <svg> informatif « simple » est intégré dans le code HTML :

1. Ajouter un attribut role="img" sur la balise <svg>.
2. Ajouter un attribut aria-label sur la balise <svg> et renseigner cet attribut avec les informations permettant de rendre l'image explicite.

Dans l'exemple de code HTML ci-après, le <svg> indique qu'il s'agit d'une étape de marche :

```
<p>
  <svg role="img" aria-label="Marche : " focusable="false">
    [...]
  </svg>
  <span>2 min</span>
  <span>111 m</span>
</p>
```

Attention

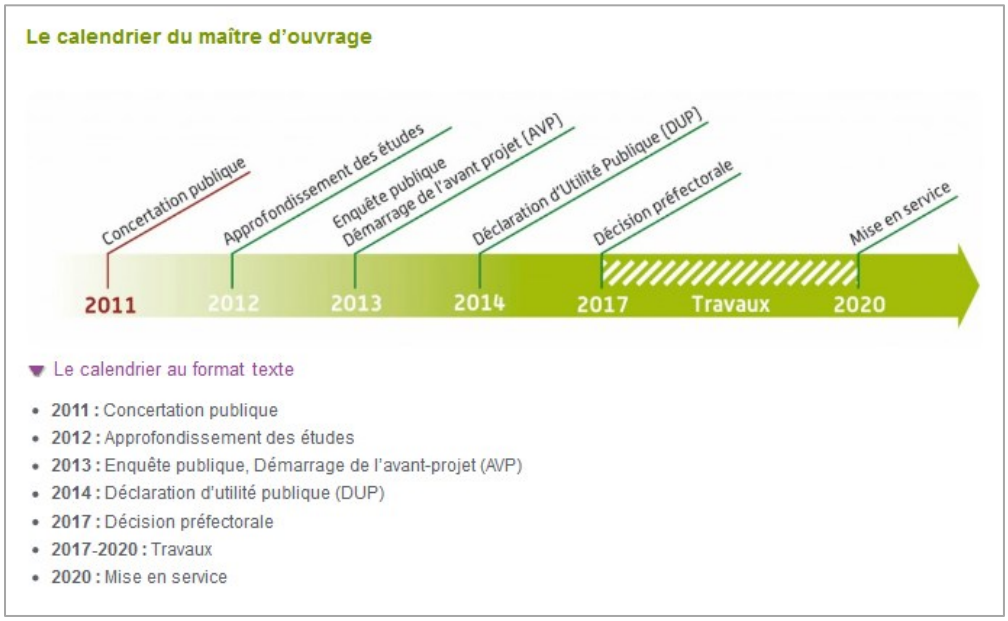
Il est fortement déconseillé de rédiger le texte de remplacement d'un <svg> en commençant par exemple par la mention « Image [...] ».

6.2.3 <svg> informatifs « complexes »

Un <svg> informatif « complexe » est une image informative qui nécessite une description **détaillée**.

Lorsqu'un <svg> informatif « complexe » est intégré dans le code HTML :

1. Ajouter un attribut role="img" sur la balise <svg>.
2. Ajouter un attribut aria-label sur la balise <svg> et renseigner cet attribut avec les informations permettant de rendre l'image explicite.
3. Proposer une description détaillée de l'image directement sous cette dernière.
4. Enfin, indiquer dans l'attribut aria-label où trouver cette description détaillée.



Dans cet exemple, un bouton plier/déplier permet d'afficher une description détaillée de l'image « complexe ».

Dans l'exemple de code HTML ci-après, le `<svg>` indique à la fois la fonction de l'image et où trouver sa description détaillée :

```
<h2>Le calendrier du maître d'ouvrage</h2>
<svg role="img" aria-label="Calendrier du maître d'ouvrage et travaux prévus entre 2011 et 2020 (description détaillée ci-après)" focusable="false">
  [...]
</svg>
<button aria-expanded="true">Le calendrier au format texte</button>
<ul>
  <li><strong>2011 :</strong> Concertation publique</li>
  [...]
</ul>
```

Attention

Il est fortement déconseillé de rédiger le texte de remplacement d'un `<svg>` en commençant par exemple par la mention « Image [...] ».

6.2.4 <svg> liens ou boutons seuls

Lorsqu'un <svg> seul (sans intitulé visible) servant de lien ou de bouton est intégré dans le code HTML :

1. Ajouter un attribut role="img" sur la balise <svg>.
2. Ajouter un attribut aria-label sur la balise <svg> et renseigner cet attribut avec les informations permettant de rendre le lien ou le bouton explicite.

Dans l'exemple de code HTML ci-après, le <svg> pointe vers la page d'accueil d'un site :

```
<a href="/">
  <svg role="img" aria-label="Accueil" focusable="false">
    [...]
   </svg>
</a>
```

Attention

Il est fortement déconseillé de rédiger le texte de remplacement d'un <svg> servant de lien ou bouton en commençant respectivement par « Lien vers [...] » ou « Bouton [...] ».

6.3 Gérer l'alternative des icônes insérées via les CSS

Remarque

Que l'icône soit décorative, informative ou qu'il s'agisse d'un lien/bouton, la balise servant à l'afficher doit, dans tous les cas, intégrer `aria-hidden="true"` afin de s'assurer que les lecteurs d'écran ne restituent aucun contenu à l'utilisateur lorsqu'elle est parcourue.

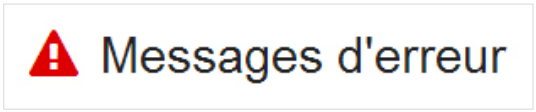
Aussi, idéalement, la balise servant à afficher l'icône doit être un `<span>` car sans valeur sémantique.

3 cas de figure :

- Icônes décoratives/illustratives.
- Icônes informatives.
- Icônes liens ou boutons seules.

6.3.1 Icônes décoratives/illustratives

Lorsqu'une icône décorative ou illustrative est intégrée dans le code HTML, rien n'est à prévoir.



Dans l'exemple de code HTML ci-après, l'icône accompagne simplement le titre « Messages d'erreur » (explicite par nature) :

```
<h2>
   <span class="icone erreur" aria-hidden="true"></span>
   Messages d'erreur
</h2>
```

6.3.2 Icônes informatives

Lorsqu'une icône informative est intégrée dans le code HTML :

1. Ajouter une balise (`<span>`, par exemple) juste après la balise servant à afficher l'icône.
2. Intégrer les informations permettant de rendre explicite l'icône dans cette nouvelle balise.
3. Masquer le contenu de cette balise en CSS en le sortant de l'écran (hors *viewport*) à l'aide de la propriété CSS « `position: absolute;` ».

Dans l'exemple de code HTML ci-après, l'icône indique qu'il s'agit d'une étape de marche :

```
<p>
  <span class="icone marche" aria-hidden="true"></span>
  <span class="hors-ecran">Marche : </span>
  <span>2 min</span>
  <span>111 m</span>
</p>
```

```
.hors-ecran {
   position: absolute;
   left: -99999rem;
}
```

6.3.3 Icônes liens ou boutons seules

Lorsqu'une icône seule (sans intitulé) servant de lien ou de bouton est intégrée dans le code HTML :

1. Ajouter une balise `<span>` dans la balise `<a>` ou `<button>`.
2. Intégrer les informations permettant de rendre explicite le lien ou le bouton dans ce `<span>`.
3. Masquer le contenu de cette balise en CSS en le sortant de l'écran (hors *viewport*) à l'aide de la propriété CSS « `position: absolute;` ».

Dans l'exemple de code HTML ci-après, l'icône-lien pointe vers la page d'accueil d'un site :

```
<a href="/">
   <span class="icone accueil" aria-hidden="true"></span>
   <span class="hors-ecran">Accueil</span>
</a>
```

```
.hors-ecran {
   position: absolute;
   left: -99999rem;
}
```

6.4 Baliser les images légendées avec `<figure role="figure">` et `<figcaption>`

Les images légendées doivent être balisées avec `<figure role="figure">` et `<figcaption>`.

La balise `<figure role="figure">` doit :

- Englober l'image ainsi que la légende, qui doit de son côté être balisée avec `<figcaption>`.
- Posséder un attribut aria-label dont le contenu doit reprendre celui de la balise `<figcaption>`.

Remarque

Idéalement, le texte de remplacement de l'image (l'attribut `alt` pour la balise `<img />`, par exemple) doit contenir une référence à la légende adjacente.

6.4.1 Exemple 1

Désert en Bolivie, prise de vue par Audesou.

```
<figure role="figure" aria-label="Désert en Bolivie, prise de vue par Audesou.">
   <img src="…" alt="Trois lamas dans le désert en Bolivie." />
   <figcaption>
      Désert en Bolivie, prise de vue par Audesou.
   </figcaption>
</figure>
```

6.4.2 Exemple 2

Photo 1 : prise de vue par Audesou en Bolivie.

```
<figure role="figure" aria-label="Photo 1 : prise de vue par Audesou en Bolivie.">
   <img src="…" alt="Trois lamas dans le désert. [photo 1]" />
   <figcaption>
      Photo 1 : prise de vue par Audesou en Bolivie.
   </figcaption>
</figure>
```

7. Formulaires

7.1 Utiliser la balise `<label>` ainsi que les attributs `for` et `id` pour étiqueter les champs avec intitulé visible

Pour étiqueter les champs disposant d'un intitulé visible :

1. Utiliser `<label>` pour baliser chaque intitulé.
2. Ajouter un attribut `for` dans chaque balise `<label>` ainsi qu'un attribut `id` dans chaque champ.
3. Renseigner avec une valeur unique et identique les attributs `for` et `id` de chaque couple intitulé/champ.

```
<label for="nom">Votre nom</label>
<input type="text" id="nom" name="nom" autocomplete="family-name" />
```

```
<label for="pays">Votre pays</label>
<select id="pays" name="pays" autocomplete="country-name">
   <option value="belgique">Belgique</option>
   <option value="france">France</option>
   […]
</select>
```

Remarque

Il est important de prévoir des intitulés identiques pour les champs dont la fonction est identique.

Par exemple, si plusieurs formulaires d'identification sont présents dans le site, ne pas utiliser l'intitulé « Identifiant » pour l'un et « Login » pour l'autre.

7.2 Utiliser `title` pour étiqueter les champs sans intitulé visible

Pour étiqueter les champs sans intitulé visible :

1. Intégrer un attribut `title` dans le champ.
2. Renseigner cet attribut en précisant la fonction du champ.

```
<input type="search" title="Votre recherche" name="recherche" />
<input type="submit" value="Rechercher" />
```

```
<select title="Trier les actualités" name="filtre">
   <option>Par date de publication</option>
   <option>Par thématique</option>
   […]
</select>
```

Attention

L'attribut `placeholder` ne peut pas faire office d'intitulé, notamment car il disparaît lors de la saisie.

Il peut être utilisé pour des aides à la saisie secondaires, non nécessaires à la compréhension du champ. De plus, il doit reprendre le contenu de l'attribut `title` lorsque celui-ci est présent.

7.3 Utiliser l'attribut `autocomplete` pour faciliter le remplissage automatique des champs

Pour faciliter le remplissage automatique des champs se rapportant à une information personnelle :

1. Ajouter un attribut `autocomplete` dans le champ.
2. Renseigner cet attribut `autocomplete` en fonction du type d'information attendu.

```
<label for="prenom">Votre prénom</label>
<input type="text" id="prenom" name="prenom" autocomplete="given-name" />
```

Remarque

Les valeurs possibles pour l'attribut `autocomplete` sont normées.

7.4 Intégrer les aides à la saisie directement dans les balises `<label>`

Les aides à la saisie doivent être intégrées directement dans les balises `<label>`.

C'est notamment le cas :

- Des mentions du type « Champ obligatoire ».
- Des indications qui permettent de connaître le format de saisie attendu, du type « jj/mm/aaaa » pour un format de date.
- Des indications qui permettent de connaître le type et le poids maximal autorisé pour l'envoi de fichiers.
- Etc.

```
<label for="date">
   Votre date de naissance
   <span>(au format jj/mm/aaaa)</span>
</label>
<input type="date" id="date" name="date" autocomplete="bday" />
```

```
<label for="numero">
    Votre numéro de client *
    <input type="text" id="numero" name="numero" aria-
required="true" />
    <span>Par exemple : BT-VZ</span>
</label>
```

Remarque

Dans certains cas, pour des raisons de mise en page particulière par exemple, il n'est pas possible d'intégrer l'aide à la saisie directement dans la balise `<label>`.

Dans ce cas-là :

1. Intégrer un attribut `id` dans la balise englobant l'aide à la saisie.
2. Renseigner cet attribut `id` avec une valeur unique.
3. Intégrer l'attribut `aria-describedby` dans le champ correspondant.
4. Renseigner cet attribut `aria-describedby` en reprenant la valeur de l'attribut `id` de l'aide à la saisie.

```
<label for="document">
   Ajouter un document à votre dossier
</label>
<input type="file" id="document" name="document" aria-
describedby="formats" />
<h2>Documents actuellement dans votre dossier</h2>
<ul>
   <li>CV</li>
   <li>Lettre de motivation</li>
</ul>
<p id="formats">Formats acceptés : pdf ou doc.</p>
```

À noter que l'attribut `aria-describedby` peut recevoir plusieurs `id`.

Attention

L'attribut `placeholder` ne doit pas être utilisé pour les aides à la saisie nécessaires à la compréhension des données attendues.

Il peut être utilisé pour des aides à la saisie secondaires, non nécessaires à la compréhension du champ.

7.5 Intégrer `required` ou `aria-required="true"` dans les champs obligatoires

Les champs obligatoires doivent intégrer `required` ou `aria-required="true"`.

Remarque

En complément de cet attribut, un signe distinctif doit être intégré dans la balise `<label>`.

```
<label for="email">Votre email *</label>
<input type="text" id="email" name="email" autocomplete="email"
required />
```

```
<input type="checkbox" id="conditions" aria-required="true" />
<label for="conditions">J'ai lu et j'accepte les conditions
générales de vente (obligatoire)</label>
```

7.6 Intégrer les messages d'erreurs et les suggestions de correction directement dans les balises `<label>`

Dans le cas où les messages d'erreurs et les suggestions de correction sont positionnés au niveau de chaque champ concerné, alors les intégrer directement dans les balises `<label>`.

Astuce

En complément du message d'erreur, une bonne pratique d'accessibilité consiste à ajouter `aria-invalid="true"` dans le champ.

```
<label for="nom">
   Votre nom *
   <span>Veuillez renseigner votre nom</span>
</label>
<input type="text" id="nom" name="nom" autocomplete="family-name"
aria-required="true" aria-invalid="true" />*
```

```
<label for="email">
   Votre email *
   <input type="email" id="email" name="email"
autocomplete="email" aria-required="true" aria-invalid="true" />
   <span>Veuillez respecter le format de
l'email(exemple@domaine.fr)</span>
</label>
```

Dans certains cas, pour des raisons de mise en page particulière par exemple, il n'est pas possible d'intégrer le message d'erreur directement dans la balise <label>.

Dans ce cas-là :

1. Intégrer un attribut `id` dans la balise englobant le message d'erreur.
2. Renseigner cet attribut `id` avec une valeur unique.
3. Intégrer l'attribut `aria-describedby` dans le champ correspondant.
4. Renseigner cet attribut `aria-describedby` en reprenant la valeur de l'attribut `id` du message d'erreur.

```
<label for="document">
   Ajouter un document à votre dossier
</label>
<input type="file" id="document" name="document" aria-
invalid="true" aria-describedby="formats erreur" />
```

```
<h2>Documents actuellement dans votre dossier</h2>
<ul>
   <li>CV</li>
   <li>Lettre de motivation</li>
</ul>
<p id="erreur">Format de fichier incorrect.</p>
<p id="formats">Formats acceptés : pdf ou doc.</p>
```

À noter que l'attribut `aria-describedby` peut recevoir plusieurs `id`.

Remarque

Dans le cas où la gestion des erreurs est dynamique (il n'y a pas de rechargement de la page lors de la soumission du formulaire), les recommandations suivantes sont à ajouter en complément :

- Lors de la soumission du formulaire, déplacer le focus au niveau du premier champ en erreur dans la page.
- Les messages d'erreur doivent être considérés lors des développements comme des messages d'alerte.

7.7 Regrouper et titrer les champs de même nature avec `<fieldset>` et `<legend>`

Lorsqu'un formulaire propose plusieurs groupes de champs dont certains ont des intitulés identiques, utiliser les balises `<fieldset>` et `<legend>`.

```
<fieldset>
   <legend>Participant 1</legend>
   <label for="prenom-1">Prénom</label>
   <input type="text" id="prenom-1" name="prenom-1" />
   <label for="nom-1">Nom</label>
   <input type="text" id="nom-1" name="nom-1" />
   […]
</fieldset>
<fieldset>
   <legend>Participant 2</legend>
   <label for="prenom-2">Prénom</label>
   <input type="text" id="prenom-2" name="prenom-2" />
   <label for="nom-2">Nom</label>
   <input type="text" id="nom-2" name="nom-2" />
   […]
</fieldset>
```

Attention

Les balises `<fieldset>` et `<legend>` sont systématiquement à utiliser lorsque plusieurs groupes de champs disposent d'intitulés identiques.

Par exemple :

- Une série de questions sur une même page avec comme réponses possibles « oui » ou « non ».
- Une liste de participants à un évènement avec à chaque fois « nom » et « prénom ».

Si le formulaire est long mais qu'aucun groupe de champs ne dispose d'intitulés identiques, utiliser les balises `<h1>` à `<h6>` pour titrer les groupes de champs.

Remarque

L'utilisation des balises `<fieldset>` et `<legend>` est notamment nécessaire lors de l'intégration de listes de boutons radio ou de cases à cocher dans la page.

Par exemple :

```
<fieldset>
  <legend>Sports pratiqués</legend>
  <ul>
    <li>
      <input type="checkbox" id="basket" />
      <label for="basket">Basket</label>
    </li>
    <li>
      <input type="checkbox" id="tennis" />
      <label for="tennis">Tennis</label>
    </li>
    […]
  </ul>
</fieldset>
```

8. Listes

8.1 Baliser les listes non ordonnées avec <ul> et

Utiliser les balises <ul> et <li> pour baliser les listes d'éléments pour lesquelles l'ordre n'a pas d'importance (menus, onglets, boutons de partage, plan du site, etc.).

Le cas échéant, veiller à la bonne imbrication des listes :

```
<ul>
   <li>Santé</li>
   <li>
      Droits et démarches
      <ul>
         <li>Handicap</li>
         <li>Famille</li>
      </ul>
   </li>
   <li>Actualités</li>
</ul>
```

8.1.1 Quels bénéfices ?

La structuration des listes est essentielle pour **les personnes utilisant un lecteur d'écran (personnes aveugles et malvoyantes)**. Cette structuration leur permet ainsi, à la rencontre d'une liste de :

- Naviguer de liste en liste au sein d'une page.
- Connaître dès le départ le nombre d'éléments dans celle-ci.
- Naviguer plus facilement dans la liste :
    - Passer directement à la fin de la liste si ce contenu ne les intéresse pas.
    - Revenir facilement au début de la liste.

8.2 Baliser les listes ordonnées avec <ol> et

Utiliser les balises <ol> et <li> pour baliser les listes d'éléments pour lesquelles l'ordre a de l'importance (liste d'étapes dans un processus, classement quelconque, etc.). C'est-à-dire lorsque l'information ne serait plus comprise si les éléments étaient placés dans un ordre différent.

Le cas échéant, veiller à la bonne imbrication des listes :

```
<ol>
   <li>Étape 1 : Connexion</li>
   <li>
      Étape 2 : Remplir le formulaire de demande
      <ul>
         <li>Informations personnelles</li>
         <li>Informations professionnelles</li>
      </ul>
   </li>
   <li>Étape 3 : Récapitulatif de la demande</li>
</ol>
```

8.2.1 Quels bénéfices ?

La structuration des listes est essentielle pour **les personnes utilisant un lecteur d'écran (personnes aveugles et malvoyantes)**. Cette structuration leur permet ainsi, à la rencontre d'une liste de :

- Naviguer de liste en liste au sein d'une page.
- Connaître dès le départ le nombre d'éléments dans celle-ci.
- Naviguer plus facilement dans la liste :
    - Passer directement à la fin de la liste si ce contenu ne les intéresse pas.
    - Revenir facilement au début de la liste.

8.3 Baliser les listes de descriptions avec `<dl>`, `<dt>` et `<dd>`

Utiliser les balises `<dl>`, `<dt>` et `<dd>` pour baliser les listes de descriptions.

Une liste de descriptions correspond à un groupement de clés/valeurs que l'on peut rencontrer par exemple sur une fiche produit ou un glossaire.

Remarque

Une clé (balise `<dt>`) peut avoir plusieurs valeurs (balise `<dd>`).

Exemple avec une description d'un événement :

```
<h2>Conférence sur l'accessibilité web</h2>
<dl>
   <dt>Lieu</dt>
   <dd>Paris</dd>
   <dt>Dates</dt>
   <dd>Samedi 7 septembre</dd>
   <dd>Mercredi 14 octobre</dd>
   <dt>Heure</dt>
   <dd>À partir de 10 heures</dd>
</dl>
```

Exemple avec un glossaire :

```
<dl>
   <dt><dfn>ARIA</dfn></dt>
   <dd lang="en">Accessible Rich Internet Application</dd>
   […]
   <dt><dfn>RGAA</dfn></dt>
   <dd>Référentiel Général d'Amélioration de l'Accessibilité</dd>
   […]
</dl>
```

8.3.1 Quels bénéfices ?

La structuration des listes est essentielle pour **les personnes utilisant un lecteur d'écran (personnes aveugles et malvoyantes)**. Cette structuration leur permet ainsi, à la rencontre d'une liste de :

- Naviguer de liste en liste au sein d'une page.
- Connaître dès le départ le nombre d'éléments dans celle-ci.
- Naviguer plus facilement dans la liste :
    - Passer directement à la fin de la liste si ce contenu ne les intéresse pas.
    - Revenir facilement au début de la liste.

9. Tableaux

9.1 Baliser le titre des tableaux de données avec la balise <caption>

Lorsqu'un tableau de données est introduit par un titre, celui-ci doit être balisé avec <caption>.

Remarque

Ce titre doit être précis et concis.

```
<table>
   <caption>Températures moyennes mensuelles des 3 plus grandes
villes de France.</caption>
   […]
</table>
```

9.2 Baliser chaque cellule d'entête de ligne et de colonne avec <th>

Dans chaque tableau de données, baliser chaque cellule d'entête de ligne et de colonne avec la balise <th>.

C'est-à-dire que chaque fois qu'une cellule est nécessaire à la compréhension des données proposées dans le tableau, celle-ci doit être balisée avec <th>.

```
<table>
   <caption>Températures moyennes mensuelles des 3 plus grandes
villes de France.</caption>
   <tr>
      <td> </td>
      <th>Paris</th>
      <th>Marseille</th>
      <th>Lyon</th>
   </tr>
   <tr>
      <th>Juin</th>
      <td>22°C</td>
      <td>28°C</td>
      <td>26°C</td>
   </tr>
   <tr>
      <th>Juillet</th>
      <td>24°C</td>
      <td>30°C</td>
      <td>28°C</td>
   </tr>
</table>
```

Températures moyennes mensuelles des 3 plus grandes villes de France

	Paris	Marseille	Lyon
Juin	22°C	28°C	26°C
Juillet	24°C	30°C	28°C

9.3 Utiliser l'attribut `scope` pour associer les cellules aux entêtes des tableaux de données simples

Un tableau de données simple est un tableau dans lequel les cellules d'entêtes s'appliquent systématiquement à la totalité des cellules de données d'une ligne ou d'une colonne.

Pour associer les entêtes à leurs données dans ce type de tableaux, utiliser l'attribut `scope` sur les balises `<th>`. La valeur de cet attribut changera selon que la cellule d'entête concerne :

- La totalité d'une colonne : `scope="col"`.
- La totalité d'une ligne : `scope="row"`.

```
<table>
   <caption>Températures moyennes mensuelles des 3 plus grandes
villes de France.</caption>
   <tr>
      <td> </td>
      <th scope="col">Paris</th>
      <th scope="col">Marseille</th>
      <th scope="col">Lyon</th>
   </tr>
   <tr>
      <th scope="row">Juin</th>
      <td>22°C</td>
      <td>28°C</td>
      <td>26°C</td>
   </tr>
   <tr>
      <th scope="row">Juillet</th>
      <td>24°C</td>
      <td>30°C</td>
      <td>28°C</td>
   </tr>
</table>
```

Températures moyennes mensuelles des 3 plus grandes villes de France

	Paris	Marseille	Lyon
Juin	22°C	28°C	26°C
Juillet	24°C	30°C	28°C

9.4 Intégrer `role="presentation"` dans chaque balise `<table>` de mise en forme

Signaler les tableaux de mise en forme en intégrant `role="presentation"` dans la balise `<table>`.

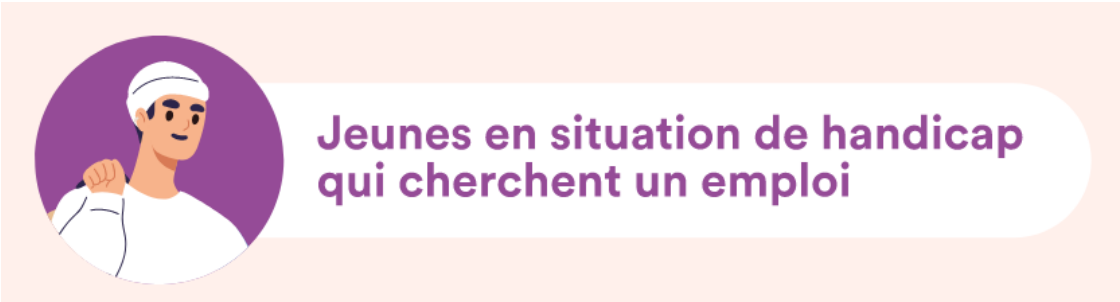
```
<table role="presentation">[…]</table>
```

10. Usage des CSS

10.1 Utiliser CSS pour mettre en forme les textes

Les textes ne doivent pas être intégrés sous forme d'images, mais sous forme de textes au format HTML stylés en CSS.

L'objectif est de permettre aux utilisateurs malvoyants de personnaliser l'ensemble des textes pour les lire, en modifiant par exemple la police, la taille ou la couleur des textes.



Exemple d'un texte à intégrer en HTML plutôt que sous forme d'image.

10.2 Garantir la lisibilité des contenus lorsque les images ne sont pas affichées

Lorsque les images ne sont pas affichées, les contenus de la page doivent rester visibles et lisibles. Aucune information ne doit être perdue et le contraste entre la couleur du texte et la couleur d'arrière-plan doit être suffisamment élevé.

C'est-à-dire que tous les contenus doivent pouvoir être lus :

- Même lorsque les images intégrées en CSS ne sont pas chargées dans la page.
- Même lorsque les images intégrées en HTML sont remplacées par le contenu de leur attribut `alt`.

Remarque

Chaque fois qu'un texte est superposé à une image d'arrière-plan, mettre en place une couleur de remplacement qui garantira la lisibilité du texte en l'absence de cette dernière.

```
background: black url(../images/fond-sombre.png) repeat-x;
```

Cette couleur de remplacement peut être héritée d'un élément parent.

Attention

Une attention particulière doit être portée sur la lisibilité des alternatives textuelles des images intégrées dans le code HTML lorsque les images ne sont pas affichées.

10.3 Garantir la compréhension de l'information même lorsque CSS est désactivé

Veiller à ce que l'information reste disponible et compréhensible même lorsque CSS est désactivé, notamment lorsque des couleurs, des tailles, des formes ou encore des positions sont vectrices d'informations.

Veiller également à ne pas générer de contenus informatifs en passant seulement par CSS.

10.3.1 Exemples

10.3.1.1 Icônes liens ou boutons seules (exemple 1)

Lorsqu'une icône seule (sans intitulé) servant de lien ou de bouton est intégrée via les CSS, un texte de substitution devra être intégré en HTML dans le contenu de la balise lien ou bouton.

Un exemple de mise en œuvre est disponible dans le chapitre sur la gestion de l'alternative des icônes insérées via les CSS.

Attention

L'ajout d'un attribut `aria-label` ou `title` comme alternative textuelle ne sera pas valide ici, car le lien ou bouton ne possèdera aucun contenu HTML et ne sera alors plus disponible lorsque CSS est désactivé.

10.3.1.2 Position courante dans les menus (exemple 2)

Dans cet exemple, pour le lien « Organigramme », une couleur de fond différente ainsi qu'une mise en gras et une encoche indiquent qu'il s'agit de la page courante.

Ici, comme il s'agit de la page courante, une solution efficace pour véhiculer cette information sans CSS consiste à ne pas englober cet intitulé d'une balise `<a>`.

```
<ul>
   <li><a href="…">Missions</a></li>
   <li><a href="…">Budget</a></li>
   <li aria-current="page">Organigramme</li>
   <li><a href="…">Instances</a></li>
   <li><a href="…">Dates clés</a></li>
</ul>
```

10.3.1.3 Contenus générés en CSS (exemple 3)

Les contenus informatifs, nécessaires à la compréhension, ne doivent pas être générés en CSS.

```
a[href*=".pdf"]::after{
   content: ' (PDF)';
}
```

Dans cet exemple, l'indication du format des fichiers PDF en téléchargement est ajoutée en CSS. Ce qui est incorrect.

```
label.obligatoire::after {
   content: ' *';
}
```

```
input[aria-required=true]::before {
   content: '* ';
}
```

Dans ces exemples, l'astérisque marquant les champs obligatoires d'un formulaire est ajoutée en CSS. Ce qui est également incorrect.

11. Zoom et taille des textes

11.1 Utiliser uniquement des tailles relatives (`rem`, `em`, `%`, etc.) pour les polices de caractères

Pour définir les tailles des polices de caractères, utiliser uniquement des unités relatives pour la propriété CSS `font-size` comme `rem`, `em`, `%`, ou les mots-clés (`x-small`, `small`, etc.).

Ne pas utiliser d'unités absolues comme `pt`, `cm`, etc.

Attention

Bien que conforme, l'utilisation du pixel (`px`) pour définir les tailles des polices de caractères peut empêcher l'agrandissement de ces derniers dans certaines situations.

Nous en déconseillons donc très fortement l'utilisation.

Remarque

Les CSS d'impression (type de média `print`) ne sont pas concernés par cette règle.

11.2 Garantir la lisibilité des contenus même lorsque la taille du texte est doublée

Garantir la lisibilité des contenus jusqu'au point où la taille du texte est doublée par rapport à la taille par défaut.

Sur l'ensemble de cette plage d'agrandissement de la taille des textes, les contenus et fonctionnalités ne doivent pas se chevaucher ou disparaître.

Pour garantir le bon respect de cette règle :

- Utiliser uniquement des unités relatives (`rem`, `em`, `%`, etc.) pour la gestion des espacements (propriétés CSS `margin` et `padding`).
- Il n'est pas recommandé d'utiliser des unités (`px`, `pt`, `em`, `%`, etc.) avec la propriété CSS `line-height`.
- Il est recommandé de ne pas définir de hauteur fixe (propriété CSS `height`) sur les éléments susceptibles d'accueillir du contenu textuel, notamment les champs de formulaires.
- Attention à l'utilisation du positionnement en absolu (déclaration CSS `position: absolute;`). Bien que ce positionnement soit compatible avec l'accessibilité, dans certains cas de figure, il peut entraîner des superpositions de contenu.

12. Navigation au clavier

12.1 Garantir la visibilité de la prise de focus au clavier

La visibilité du focus au clavier est définie par la propriété CSS `outline`. Elle ne doit pas être dégradée par rapport au style de focus natif du navigateur.

En particulier, toutes les propriétés qui suppriment ou réduisent la visibilité du focus (par exemple les propriétés `outline: none;` et `outline: 0;` appliquées en CSS sur tous les éléments interactifs) sont à supprimer.

Chaque élément interactif (liens, boutons, champs de formulaires, etc.) doit être visuellement mis en avant lors de la prise de focus, afin de permettre aux personnes qui naviguent au clavier de se situer dans la page.

Remarque

Veiller à ce que cette mise en avant visuelle soit suffisamment contrastée.

Nous recommandons fortement de ne pas personnaliser la visibilité du focus au clavier, afin de laisser le style par défaut du navigateur qui est suffisamment contrasté et visible sauf cas particulier. De plus, ce style est réalisé avec la classe CSS `:focus-visible`, limitant l'affichage du focus uniquement aux utilisateurs clavier.

L'utilisation de la propriété CSS `box-shadow` pour personnaliser la visibilité du focus au clavier est fortement déconseillée, car elle n'est pas visible en mode de contraste élevé, contrairement à la propriété `outline`.

Astuce

Une très bonne pratique d'accessibilité consiste à doubler systématiquement chaque règle `:hover` par une règle `:focus` dans la CSS.

Comme par exemple :

```
main a:hover,
main a:focus {
   text-decoration: none;
}
```

12.2 Garantir le fonctionnement de l'interface à la souris et au clavier

Toutes les interactions réalisables à la souris doivent également être possibles au clavier (et inversement).

En d'autres termes, chaque fois qu'il est possible d'interagir avec un composant à la souris, celui-ci doit également être atteignable et fonctionnel au clavier.

C'est par exemple le cas lors de l'affichage/masquage :

- D'un sous-menu.
- D'une fenêtre modale.
- D'une infobulle personnalisée.

Attention

Que la navigation clavier s'effectue en avant ou en arrière, la page ne doit contenir aucun piège au clavier.

C'est-à-dire que le focus clavier ne doit jamais :

- Être bloqué sur un élément sans possibilité d'en sortir.
- Boucler dans une zone de la page sans possibilité d'en sortir.

Remarque

Veiller à ce que les interactions à la souris et au clavier fonctionnent également au toucher (et inversement).

12.3 Veiller à ce que l'ordre de tabulation suive la logique de l'ordre de lecture

L'ordre de tabulation doit suivre la logique de l'ordre de lecture visuel de la page.

C'est-à-dire que lorsqu'un élément interactif en précède immédiatement un autre lors du parcours visuel de la page, le focus doit continuer sur le second élément immédiatement après avoir quitté le premier.

Attention

Ne pas utiliser l'attribut `tabindex` avec une valeur supérieure à 0 au risque de perturber la logique de l'ordre de tabulation dans la page.

Remarque

Dans certains composants d'interface riche, comme par exemple les systèmes d'onglets, la navigation d'un élément interactif à un autre se fait avec les flèches directionnelles plutôt qu'avec la touche Tab.

12.4 Mettre en place un lien d'évitement

Un lien d'évitement du type « Aller au contenu » doit systématiquement être présent sur chaque page afin de faciliter la navigation au clavier.

Ce lien doit être le premier élément interactif dans le code HTML.

Il s'agit d'un lien interne qui doit permettre un accès direct au contenu principal de la page.

```
<body>
 <a class="evitement" href="#contenu">Aller au contenu</a>
 […]
 <main role="main" id="contenu">
   […]
 </main>
 […]
</body>
```

Remarque

Le lien d'évitement peut être masqué par défaut. En revanche, il doit dans tous les cas être rendu visible à la prise de focus au clavier.

Par conséquent, le lien d'évitement ne doit jamais être masqué à l'aide des propriétés CSS `display: none;` et/ou `visibility: hidden;` sous peine de le rendre totalement inatteignable au clavier.

Privilégier une autre solution, par exemple l'utilisation des codes suivants :

```
a.evitement {
   display: inline-block ;
```

```
   color: #555 ;
   background: #fff ;
   padding: .5em ;
   position: absolute ;
   left: -99999rem ;
   z-index: 100 ;
}

a.evitement:focus {
   left: 0;
}
```

Astuce

Dans certaines situations, de nombreuses tabulations sont nécessaires pour accéder aux menus principal/secondaire et/ou au moteur de recherche depuis le sommet de la page.

Dans ce cas-là, mettre en place une liste de plusieurs liens d'évitement.

Comme par exemple :

```
<ul id="evitement">
   <li>
      <a href="#contenu">Aller au contenu</a>
   </li>
   <li>
      <a href="#menu">Aller au menu</a>
   </li>
   <li>
      <a href="#recherche">Aller à la recherche</a>
   </li>
</ul>
```

13. Iframes

13.1 Renseigner l'attribut `title` sur chaque `<iframe>`

Chaque fois qu'une balise `<iframe>` est intégrée dans la page, ajouter un attribut `title` sur cette dernière.

La valeur de cet attribut doit introduire le contenu de la balise `<iframe>` de manière claire et concise.

Par exemple, dans le cas d'une `<iframe>` qui charge un lecteur vidéo dans la page :

```
<iframe src="https://www.youtube.com/embed/y525BrxyvhA"
title="Vidéo YouTube : L'accessibilité numérique à toutes les
étapes d'un projet">
   […]
</iframe>
```

14. Règles supplémentaires pour la conformité

Certains critères présents dans les référentiels d'accessibilité n'ont pas été conservés dans le corps de cette notice car considérés comme rarement applicables.

Le respect de ces règles supplémentaires est toutefois nécessaire pour garantir la conformité RGAA 4.1.2 et WCAG 2.1.

Elles sont listées ci-dessous :

- Assurer la compréhension de l'information même en l'absence de couleurs
- Garantir la lisibilité des contenus même lorsque les propriétés d'espacement du texte sont personnalisées
- Intégrer le résumé de chaque tableau de données complexe dans la balise `<caption>`
- Fournir une alternative textuelle à chaque balise `<canvas>`, `<embed>` et `<object>` informative
- Fournir une alternative textuelle à chaque balise `<canvas>`, `<embed>` et `<object>` servant de lien ou de bouton
- Masquer chaque balise `<canvas>`, `<embed>` et `<object>` décoratives ou illustratives aux technologies d'assistance
- Ne pas lancer d'action tant que l'élément déclencheur est dans l'état « pressé »
- Permettre l'anticipation du comportement de l'interface
- Regrouper les options de même nature avec `<optgroup>` dans les `<select>`
- Renseigner l'attribut `title` sur chaque <frame>
- Utiliser l'attribut `dir` pour signaler les changements de sens de lecture
- Veiller à utiliser correctement les techniques de masquage de contenu

Baliser les blocs de citations avec `<blockquote>`

Chaque fois qu'un bloc de citation est à intégrer dans la page HTML, entourer ce dernier d'une balise `<blockquote>`.

Remarque

Toute citation isolée, qui n'est pas incorporée directement en ligne dans le flux du texte environnant, est considérée comme un bloc de citation.

Baliser les citations en ligne avec `<q>`

À chaque fois qu'une citation est intégrée à l'intérieur d'un texte, entourer cette dernière avec la balise `<q>`.

Écrire le code HTML en suivant la logique de l'ordre de lecture

L'ordre d'écriture des balises dans le flux HTML doit suivre la logique de l'ordre de lecture de la page.

C'est-à-dire que lorsqu'une balise précède immédiatement une autre balise lors du parcours visuel de la page, la première doit également précéder immédiatement la seconde dans le flux HTML.

Astuce

Pour tester cette règle, il suffit de désactiver les CSS est de s'assurer que le résultat obtenu correspond bien à l'ordre de lecture de la page lorsque les CSS sont activées.

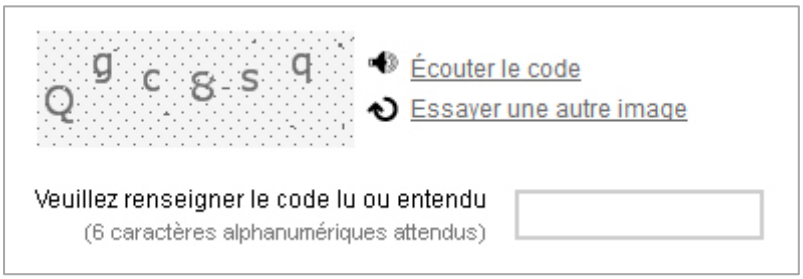
Attention

Si des contenus sont masqués par défaut, veiller à leur bon positionnement dans le flux HTML lorsque les styles sont désactivés.

Indiquer dans l'alternative des CAPTCHA graphiques où trouver la version non-graphique du CAPTCHA

Chaque fois qu'un CAPTCHA graphique est utilisé, indiquer dans son alternative textuelle les renseignements permettant :

- D'identifier qu'il s'agit d'un système de sécurité.
- De trouver la version non graphique du CAPTCHA.



Souvent, il s'agit d'une version audio, opter alors par exemple pour :

```
<img src="captcha.png" alt="Système de protection antispam : si vous n'y avez pas accès, utilisez la version sonore disponible ci-après." />
```

Remarque

Pour renseigner correctement l'alternative, selon la technique d'intégration du CAPTCHA, se référer aux fiches de la thématique Images et icônes.

Ne pas utiliser de balises ou d'attributs propres aux tableaux de données dans les tableaux de mise en forme

Un tableau de mise en forme ne doit pas posséder de balises ou d'attributs propres aux tableaux de données.

C'est-à-dire :

- Que les balises `<caption>`, `<colgroup>`, `<tfoot>`, `<th>` et `<thead>` ne doivent pas être utilisées dans les tableaux de mise en forme.
- Que les attributs `axis`, `headers` et `scope` ne doivent pas être utilisés dans les tableaux de mise en forme.

Prévoir une alternative à chaque contenu multimédia (<video>, <audio>, <object>, <embed>, etc.)

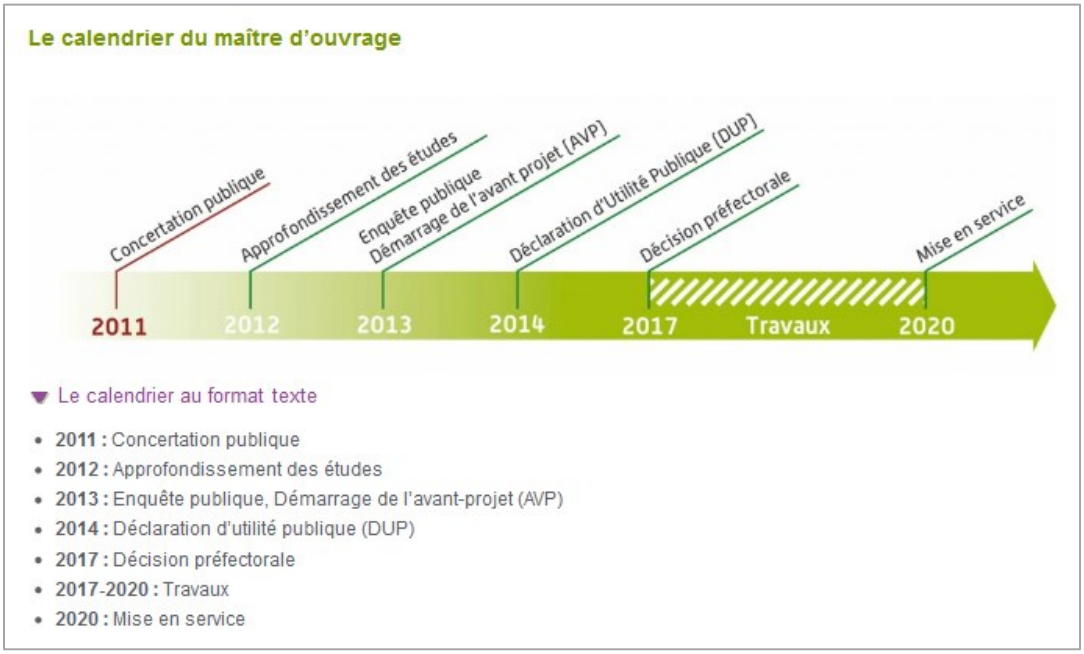
Prévoir une alternative à chaque contenu multimédia embarqué dans la page avec l'aide par exemple des balises <video>, <audio>, <object>, <embed>, etc.

Cette alternative permettra de fournir le même niveau d'informations aux personnes ne pouvant pas accéder au contenu multimédia (absence de plugin, navigateur vieillissant, etc.).

Elle peut par exemple prendre la forme d'un contenu HTML.

Astuce

Lorsqu'une vidéo est affichée sur une page, sa transcription peut être considérée comme une alternative.



Dans cet exemple, le calendrier est une vidéo porteuse d'informations, un lien « Le calendrier au format texte » permet d'afficher une alternative au format HTML.

Renseigner l'attribut `alt` de chaque image mappée et de ses balises `<area />`

Lorsqu'une image mappée est intégrée dans le code HTML :

- Un attribut `alt` doit être rajouté sur la balise `<img />` ainsi que sur chaque balise `<area />`.
- L'attribut `alt` de la balise `<img />` doit annoncer la fonction de l'image mappée.
- L'attribut `alt` de chaque balise `<area />` doit exprimer la destination du lien.

Par exemple, dans le cas d'une carte de France où chaque département est cliquable et mène vers une fiche associée au département :

```
<img src="…" alt="Carte de France" usemap="#carte-france" />
<map name="carte-france">
  <area shape="poly" coords="…" href="…" alt="Ain" />
  <area shape="poly" coords="…" href="…" alt="Aisne" />
  <area shape="poly" coords="…" href="…" alt="Allier" />
  […]
</map>
```

Utiliser les attributs `headers` et `id` pour associer les cellules aux entêtes des tableaux de données complexes

Un tableau de données complexe est un tableau dans lequel les cellules d'entêtes ne s'appliquent pas systématiquement à la totalité des cellules de données d'une ligne ou d'une colonne.

Pour associer les entêtes aux données dans ce type de tableaux, utiliser les attributs `id` (identifiants) sur les cellules `<th>` et `headers` sur les cellules `<td>`.

Il s'agit ensuite de renseigner l'attribut `headers` avec les identifiants des cellules d'entêtes associées. Si plusieurs entêtes sont associés à une cellule de données, les identifiants doivent être séparés par des espaces dans `headers`.

```
<table>
   <caption>Comparatif du chiffre d'affaires des entreprises
Dupond et Dupont en France et dans le monde</caption>
   <tr>
       <th id="entete-1">En millions d'euros</th>
       <th id="entete-2">En France</th>
       <th id="entete-3">Dans le monde</th>
   </tr>
```

```
    <tr>
        <th id="entete-4">Dupond</th>
        <td headers="entete-4 entete-2 entete-1">
            50,7
        </td>
        <td headers="entete-4 entete-3 entete-1">
            139,3
        </td>
    </tr>
    <tr>
        <th id="entete-5">Dupont</th>
        <td headers="entete-5 entete-2 entete-1">
            27,1
        </td>
        <td headers="entete-5 entete-3 entete-1">
            476,0
        </td>
    </tr>
</table>
```

Comparatif du chiffre d'affaires des entreprises Dupond et Dupont en France et dans le monde

En millions d'euros	En France	Dans le monde
Dupond	50,7	139,3
Dupont	27,1	476,0

Attention

L'attribut `headers` ne doit pas être utilisé en combinaison de l'attribut `scope`.

Remarque

Une bonne pratique d'accessibilité consiste à respecter un ordre logique lorsque les valeurs des attributs `id` des cellules d'entêtes associées à une cellule de données sont intégrées dans l'attribut `headers` de cette dernière.

En effet, un lecteur d'écran (synthèse vocale et/ou plage braille) annoncera les entêtes dans cet ordre.

Veiller à la cohérence de l'ordre du flux HTML d'une page à l'autre

L'ordre d'intégration des principaux blocs dans le flux HTML doit rester cohérent d'une page à l'autre du site.

Par exemple, si le menu secondaire est placé après le contenu principal, il est important de conserver cet ordre sur l'ensemble des pages du site.

Veiller à l'ordre de lecture des tableaux de mise en forme

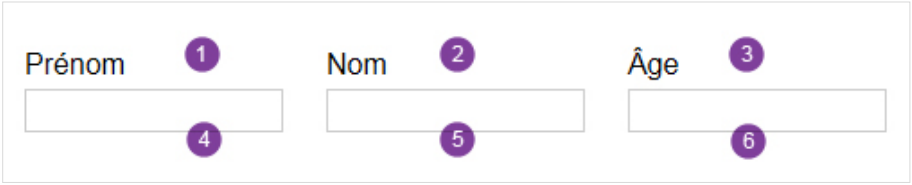
Lorsqu'une personne utilise un lecteur d'écran (synthèse vocale et/ou plage braille) pour accéder à un tableau de mise en forme, son contenu est restitué de manière linéaire. C'est-à-dire que le contenu du tableau est lu cellule après cellule, de gauche à droite et ligne après ligne.

Il faut donc veiller à ce que l'ordre de lecture reste cohérent dans chaque tableau de mise en forme.

Par exemple, si le code source qui suit est utilisé, l'ordre de lecture est :

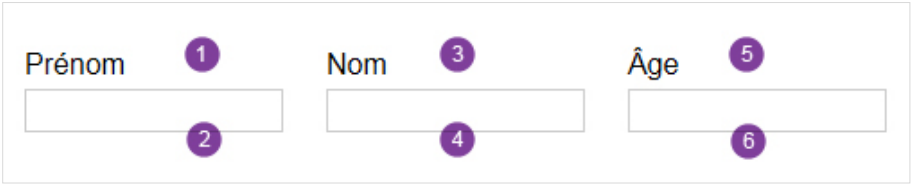
1. Prénom.
2. Nom.
3. Âge.
4. Champ « Prénom ».
5. Champ « Nom ».
6. Champ « Âge ».

```
<table role="presentation">
 <tr>
  <td><label for="prenom">Prénom</label></td>
  <td><label for="nom">Nom</label></td>
  <td><label for="age">Âge</label></td>
 </tr>
 <tr>
  <td><input type="text" name="prenom" id="prenom" /></td>
  <td><input type="text" name="nom" id="nom" /></td>
  <td><input type="text" name="age" id="age" /></td>
 </tr>
</table>
```



Pour obtenir un ordre de lecture cohérent tout en conservant le tableau de mise en forme, opter pour :

```
<table role="presentation">
 <tr>
  <td>
   <label for="prenom">Prénom</label>
   <input type="text" name="prenom" id="prenom" />
  </td>
  <td>
   <label for="nom">Nom</label>
   <input type="text" name="nom" id="nom" />
  </td>
  <td>
   <label for="age">Âge</label>
   <input type="text" name="age" id="age" />
  </td>
 </tr>
</table>
```



Attention

Afin de faciliter la lecture par les aides techniques et de garantir l'ordre de lecture, il est fortement recommandé de limiter l'imbrication des tableaux de mise en forme.

Bonnes pratiques

En complément des règles obligatoires détaillées dans les sections précédentes de cette notice, la prise en compte des bonnes pratiques listées ci-dessous a un impact bénéfique pour les personnes en situation de handicap.

Par conséquent, nous vous recommandons fortement de les appliquer.

Baliser les sigles avec `<abbr>`

Fusionner les liens adjacents pointant vers la même page

Lorsque plusieurs liens pointant vers la même page sont accolés, il est recommandé de les fusionner dans une seule et même balise `<a>`.

Voici par exemple une intégration **non recommandée** d'une image, d'un titre, d'une date et d'un texte introductif adjacents servant de lien vers le détail d'une actualité :

```
<a href="…">
   <img src="boulonnais.jpg" alt="Un jeudi dans le Boulonnais" />
</a>
<h2><a href="…">Un jeudi dans le Boulonnais</a></h2>
<p class="date">
   <a href="…">Publié le 13 mai 2016</a>
</p>
<p class="intro">
   <a href="…">Michel Serin est allé à la rencontre des habitants
et des acteurs locaux du Boulonnais. […]</a>
</p>
```

Une intégration **davantage accessible** serait la suivante :

```
<a href="…">
   <img src="boulonnais.png" alt="" />
   <h2>Un jeudi dans le Boulonnais</h2>
   <p class="date">Publié le 13 mai 2016</p>
   <p class="intro">Michel Serin est allé à la rencontre des
habitants et des acteurs locaux du Boulonnais. […]</p>
</a>
```

Remarque

Cette recommandation est également à appliquer sur les `<button>`.

Voici par exemple une intégration **non recommandée** d'une icône et d'un texte adjacents servant de boutons d'ajout aux favoris :

```
<button>
   <img src="favoris.png" alt="Ajouter à mes favoris" />
</button>
<button>Ajouter à mes favoris</button>
```

Une intégration **davantage accessible** serait la suivante :

```
<button>
   <img src="favoris.png" alt="" />
   Ajouter à mes favoris
</button>
```

Astuce

Il est aussi possible de placer le lien sur un seul élément, puis d'étendre la zone de clic via les CSS afin de permettre à la souris d'activer le lien en cliquant n'importe où sur l'actualité.

Exemple :

```
<div class="block">
   <img src="boulonnais.png" alt="" />
   <h2><a href="...">Un jeudi dans le Boulonnais</a></h2>
   <p class="date">Publié le 13 mai 2016</p>
   <p class="intro">Michel Serin est allé à la rencontre des
habitants et des acteurs locaux du Boulonnais. […]</p>
</div>
```

Code CSS associé :

```
.block a[href]::after {
   content: "";
   display: block;
   position: absolute;
   top: 0;
   bottom: 0;
   left: 0;
   right: 0;
}
```

Lors de l'utilisation de cette technique, s'assurer que l'élément qui porte le lien est suffisamment explicite, sinon le compléter avec l'utilisation de l'attribut aria-label ou title.

Identifier l'étape courante des formulaires à étapes multiples avec `aria-current="step"`

Dans le menu de navigation d'un formulaire à étapes multiples, il est recommandé d'identifier l'étape courante avec `aria-current="step"`.

```
<nav aria-label="Étapes de votre commande">
   <ol>
      <li><a href="…">Informations personnelles</a></li>
      <li aria-current="step"><em>Paiement</em></li>
      <li>Prévisualisation</li>
   </ol>
</nav>
```

Remarque

À noter qu'il est également recommandé de permettre de revenir en arrière sur les formulaires à étapes multiples, à l'aide d'un lien par exemple.

Identifier la position courante dans les systèmes de navigation avec `aria-current`

Dans les systèmes de navigation (menus, fil d'Ariane, pagination, etc.), il est recommandé d'identifier la position courante avec `aria-current="page"`.

De même, uniquement dans les menus (et non dans le fil d'Ariane), il est également recommandé d'identifier les éventuelles rubriques et sous-rubriques parentes avec `aria-current="true"`.

Exemple sur un menu

Dans l'exemple de code ci-après d'un menu de navigation, la rubrique parente « Agenda » est identifiée via `aria-current="true"` intégré dans sa balise `<a>` et la page courante « Éducation » est identifiée via `aria-current="page"` intégré dans sa balise `<li>`.

```
<nav role="navigation" aria-label="Menu principal">
   <ul>
      <li><a href="…">Accueil</a></li>
      <li><a href="…" aria-current="true">Agenda</a>
         <ul>
            <li><a href="…">Culture</a></li>
            <li aria-current="page">Éducation</li>
            <li><a href="…">Sports</a></li>
         </ul>
      </li>
      <li><a href="…">Actualités</a></li>
      <li><a href="…">Contact</a></li>
   </ul>
</nav>
```

Exemple sur une pagination

Dans l'exemple de code ci-après d'une pagination, la page courante « 2 » est identifiée via `aria-current="page"` intégré dans sa balise `<li>`.

```
<nav role="navigation" aria-label="Pagination">
   <ul>
      […]
      <li><a href="…" aria-label="Page 1">1</a></li>
      <li aria-current="page">2</li>
      <li><a href="…" aria-label="Page 3">3</a></li>
      […]
   </ul>
</nav>
```

Exemple sur un fil d'Ariane

Dans l'exemple de code ci-après d'un fil d'Ariane, la page courante « Notice éditoriale » est identifiée via `aria-current="page"` intégré dans sa balise `<li>`.

```
<nav role="navigation" aria-labelledby="intro-fil-ariane">
   <p id="intro-fil-ariane">Vous êtes ici :</p>
   <ul>
      <li><a href="…">Accueil</a></li>
      <li><a href="…">Les notices AcceDe Web</a></li>
      <li aria-current="page">Notice éditoriale</li>
   </ul>
</nav>
```

Intégrer le format et le poids des fichiers dans chaque lien et bouton permettant de les télécharger

Chaque fois qu'un lien ou un bouton pointe sur un fichier à télécharger, il est recommandé d'intégrer les informations suivantes dans l'intitulé :

- Le titre du fichier.
- Le format du fichier.
- Le poids du fichier.

```
<a href="plan-reseau-bus-lyon.pdf">
Télécharger le plan du réseau de bus de Lyon (PDF - 2 Mo)
</a>
```

Remarque

Dans le cas où le format du fichier à télécharger est indiqué via un pictogramme, se référer à la thématique « Images et icônes » pour l'intégrer de manière accessible.

Astuce

Dans le cas où le poids du fichier à télécharger ne peut être connu à l'avance (génération à la volée, par exemple), une bonne pratique d'accessibilité consiste à indiquer un poids approximatif ou maximal.

Pour obtenir par exemple :

```
<a href="facture.pdf">
Télécharger votre facture du mois de mars (PDF - environ 1,5 Mo)
</a>
```

Mettre à jour le `<title>` de la page quand celle-ci renvoie une erreur ou un message de confirmation

Chaque fois qu'un formulaire renvoie une erreur ou un message de confirmation , il est recommandé de mettre à jour la balise `<title>`.

Par exemple, dans le cas d'une réussite :

```
<title>Confirmation - Contact | [Nom du site]</title>
```

Et dans le cas d'une erreur :

```
<title>Erreur - Contact | [Nom du site]</title>
```

Astuce

Il est aussi recommandé de faire apparaître cette mention en premier dans la balise `<title>`.

Remarque

Dans certaines situations, il n'est pas nécessaire d'ajouter une mention car le titre de la page affichée après la soumission rend le résultat de l'action évident.

Par exemple :

- Formulaire de connexion qui envoie vers une page de profil.
- Bouton « Passer à l'étape suivante » qui envoie à l'étape suivante dans un formulaire à étapes multiples.
- Formulaire de contact qui envoie une page de prévisualisation.

Ne pas brider le zoom avec la propriété `user-scalable`

Il est important de laisser la possibilité de zoomer dans la page.

Pour cela, les déclarations suivantes ne doivent notamment pas être utilisées :

```
<meta content="width=device-width; initial-scale=1.0 ; maximum-scale=1.0 ; user-scalable=1 ;" name="viewport" />
```

```
<meta name="viewport" content="user-scalable=no" />
```

Ne pas utiliser CSS pour afficher les images porteuses d'informations

Il est recommandé de ne pas utiliser CSS pour afficher les images porteuses d'informations.

En d'autres termes, chaque fois qu'une image apporte de l'information, il est recommandé de l'intégrer en dur dans le code HTML (via les balises `<img />` ou `svg` par exemple).

Remarque

L'usage de sprites (images chargées en arrière-plan via CSS) est donc fortement déconseillé pour les images porteuses d'informations.

Astuce

Sont considérées comme porteuses d'informations toutes les images qui entraîneraient une perte de contenus ou de fonctionnalités si elles n'étaient pas disponibles :

- Logos.
- Images-textes.
- Images servant de liens ou de boutons.
- Etc.

Attention

En particulier, la technique qui consiste à sortir un texte de l'écran pour le remplacer par une image d'arrière-plan est fortement déconseillée.

Lorsque les images ne sont pas chargées, l'information peut être perdue.

```
<a href="/" id="logo">[Le texte de mon logo]</a>

#logo {
   display: block;
   width: 300px;
   height: 100px;
   text-indent: -99999rem;
   background: url('images/logo.png');
}
```

Il est fortement conseillé de remplacer cette technique par l'utilisation de HTML pur.

Ordonner les options de manière logique dans les listes déroulantes

Lorsque des listes déroulantes sont utilisées, il est recommandé d'ordonner les options qu'elles contiennent de manière logique.

L'ordre logique dépend du contexte. Il peut s'agir notamment :

- D'un ordre alphabétique (une liste de langues, par exemple).
- D'un ordre numérique (une liste de départements, par exemple).
- D'un ordre pratique (« France » en premier dans un formulaire d'inscription à un service français).

Astuce

Une autre bonne pratique d'accessibilité consiste à ne pas préfixer le contenu de la balise `<option>` avec des caractères décoratifs (tirets, étoiles, espaces, etc.).

Ceci afin de permettre un accès direct à une valeur ou à un groupe de valeurs souhaité par simple pression d'une touche au clavier (pression sur la touche E pour atteindre le pays « Espagne », par exemple).

```
   <select>
      <option>--Allemagne</option>
      <option>--Belgique</option>
      <option>--Espagne</option>
      <option>--France</option>
      <option>--Italie</option>
   </select>
```

Sera remplacé avantageusement par :

```
  <select>
    <option>Allemagne</option>
    <option>Belgique</option>
    <option>Espagne</option>
    <option>France</option>
    <option>Italie</option>
  </select>
```

Signaler l'ouverture des nouvelles fenêtres

Chaque fois qu'un lien ou un bouton déclenche l'ouverture d'une nouvelle fenêtre dans le navigateur, il est recommandé d'ajouter une mention du type « (nouvelle fenêtre) ».

Soit dans l'attribut `aria-label` :

```
<a href="cgv.html" target="_blank" aria-label="Conditions
Générales de Vente (nouvelle fenêtre)">
   Conditions Générales de Vente
</a>
```

Soit via le texte de remplacement d'une icône/image, sous la forme :

- D'une icône insérée via les CSS.
- D'un svg (image vectorielle).
- D'une balise <img />.

Remarque

À noter qu'il est également possible de passer par l'attribut `title` :

```
<a href="cgv.html" target="_blank" title="Conditions Générales de
Vente (nouvelle fenêtre)">
   Conditions Générales de Vente
</a>
```

Utiliser uniquement des tailles relatives (`rem`, `em`, `%`, etc.) pour les points de rupture des media queries

Pour définir les points de rupture des media queries, il est recommandé d'utiliser uniquement des unités relatives (comme `rem`, `em` ou `%`) pour la règle CSS `@media`.

```
@media (max-width: 25rem) {
  nav li.tile {
    width: 33%;
  }
}
```

Plutôt que des unités absolues comme `px`, `pt`, `cm`, etc.

Remarque

Suivre également cette recommandation pour les media queries intégrées directement dans la balise `<link />` :

```
<link rel="stylesheet" href="/css/extra-small.css" media="all and (max-width: 25rem)" />
```

Veiller à traduire les contenus masqués

Pour les sites multilingues ou lors de l'utilisation de scripts développés dans une langue étrangère, veiller notamment à traduire dans la langue courante du site tous les contenus masqués dans les :

- Attributs `alt` des images.
- Attributs `aria-label`.
- Attributs `title`.
- Balises non-visibles à l'écran (classe CSS « `.sr-only` », par exemple).