別添２

独立行政法人情報処理推進機構

事業者向け Web アクセシビリティ対応基準書

事業者向け　WCAG 2.0 / 2.1 /2.2 対応　Ver1.0

株式会社インフォ・クリエイツ

2025 年 10 月 8 日

改訂履歴

項号	バージョン	改定年月	改定ページ	改定内容	改定者	確認者
1	1.0 版	2025 年 10 月	全ページ	初版	IPA	

目次

はじめに

独立行政法人情報処理推進機構から委託を受け、ウェブコンテンツの制作にあたっては、必ず本ウェブアクセシビリティ対応基準書に従って制作しください。

本書に関連する文書

本対応基準書に記述されていない項目については、本対応基準書の元になっている次の文書一式を参照してください。

- JIS X 8341-3:2016 規格票
  https://webdesk.jsa.or.jp/（利用登録することで閲覧することが可能です）
- ウェブ・コンテンツ・アクセシビリティ・ガイドライン（WCAG 2.0）(日本語)
  https://waic.jp/translations/WCAG20/
- WCAG 2.0 解説書（日本語）
  https://waic.jp/translations/UNDERSTANDING-WCAG20/
- ウェブ・コンテンツ・アクセシビリティ・ガイドライン（WCAG 2.1）(日本語)
  https://waic.jp/translations/WCAG21/
- WCAG 2.1 解説書（日本語）
  https://waic.jp/translations/WCAG21/Understanding/
- ウェブ・コンテンツ・アクセシビリティ・ガイドライン（WCAG 2.2）(英語)
  https://www.w3.org/TR/WCAG22/
- ウェブ・コンテンツ・アクセシビリティ・ガイドライン（WCAG 2.2）(日本語)
  https://ipajapan.sharepoint.com/sites/portal
- WCAG 2.2 解説書（英語）
  https://www.w3.org/WAI/WCAG22/Understanding/
- WCAG 2.2 解説書（日本語）
  https://waic.jp/translations/WCAG22/Understanding/

本対応基準書の遵守

ウェブページの制作者においては、コンテンツの納品に当たって、JIS X 8341-3:2016、WCAG2.0、WCAG2.1、WCAG2.2 に基づく試験を自ら行い、適合レベル A あるいは AA に適合していることを確認した上で納品を行ってください。求めがあった場合には、アクセシビリティの確認を行ったエビデンスとして本文書に添付してある達成基準チェックリストの提供を求められる場合があります。

なお、コンテンツの内容によっては本ウェブアクセシビリティガイドラインに遵守することが難しい場合があります。そのような場合は、事前にウェブコンテンツのオーナーと協議を行い、オーナーの許可を得て対応の除外等を検討してください。ただし、その場合であっても、利用者は多様であることを忘れず、代替の方法を提供するなど、最大限のアクセシビリティを実現するよう配慮してください。

WCAG 2.2 達成基準一覧

達成基準		WCAG	適合レベル
1.1.1	非テキストコンテンツ	2.0	A
1.2.1	音声のみ及び映像だけ（収録済）	2.0	A
1.2.2	キャプション(収録済)	2.0	A
1.2.3	音声解説又はメディアに対する代替コンテンツ(収録済)	2.0	A
1.2.4	キャプション(ライブ)	2.0	AA
1.2.5	音声解説(収録済)	2.0	AA
1.3.1	情報及び関係性	2.0	A
1.3.2	意味のあるシーケンス	2.0	A
1.3.3	感覚的な特徴	2.0	A
1.3.4	表示の向き	2.1	AA
1.3.5	入力目的の特定	2.1	AA
1.4.1	色の使用	2.0	A
1.4.2	音声の制御	2.0	A
1.4.3	コントラスト(最低限レベル)	2.0	AA
1.4.4	テキストのサイズ変更	2.0	AA
1.4.5	文字画像	2.0	AA
1.4.10	リフロー	2.1	AA
1.4.11	非テキストのコントラスト	2.1	AA
1.4.12	テキストの間隔	2.1	AA
1.4.13	ホバー又はフォーカスで表示されるコンテンツ	2.1	AA
2.1.1	キーボード	2.0	A
2.1.2	キーボードトラップなし	2.0	A
2.1.4	文字キーのショートカット	2.1	A
2.2.1	タイミング調整可能	2.0	A
2.2.2	一時停止, 停止及び非表示	2.0	A
2.3.1	3 回のせん(閃)光、又はしきい(閾)値以下	2.0	A
2.4.1	ブロックスキップ	2.0	A
2.4.2	ページタイトル	2.0	A
2.4.3	フォーカス順序	2.0	A
2.4.4	リンクの目的(コンテキスト内)	2.0	A
2.4.5	複数の手段	2.0	AA
2.4.6	見出し及びラベル	2.0	AA
2.4.7	フォーカスの可視化	2.0	AA
2.4.11	隠されないフォーカス（最低限）	2.2	AA
2.5.1	ポインタのジェスチャ	2.1	A
2.5.2	ポインタのキャンセル	2.1	A
2.5.3	ラベルを含む名前（name）	2.1	A
2.5.4	動きによる起動	2.1	A
2.5.7	ドラッグ操作	2.2	AA
2.5.8	ターゲットサイズ（最低限）	2.2	AA
3.1.1	ページの言語	2.0	A
3.1.2	一部分の言語	2.0	AA
3.2.1	フォーカス時	2.0	A
3.2.2	入力時	2.0	A
3.2.3	一貫したナビゲーション	2.0	AA
3.2.4	一貫した識別性	2.0	AA
3.2.6	一貫したヘルプ	2.2	A
3.3.1	エラーの特定	2.0	A
3.3.2	ラベル又は説明	2.0	A
3.3.3	エラー修正の提案	2.0	AA
3.3.4	誤り防止(法的, 金融及びデータ)	2.0	AA
3.3.7	冗長な入力	2.2	A
3.3.8	アクセシブルな認証（最低限）	2.2	AA
4.1.1	構文解析	2.0	A
4.1.2	名前(name), 役割(role)・値(value)	2.0	A
4.1.3	ステータスメッセージ	2.1	AA

1.1.1 非テキストコンテンツ レベル A

利用者に提示されるすべての非テキストコンテンツには、同等の目的を果たすテキストによる代替が提供されている。ただし、次の場合は除く:

コントロール、入力

非テキストコンテンツが、コントロール又は利用者の入力を受け付けるものであるとき、その目的を説明する名前（name) を提供している。(コントロール及び利用者の入力を受け入れるコンテンツに関するその他の要件は、達成基準 4.1.2 を参照。)

時間依存メディア

非テキストコンテンツが、時間に依存したメディアであるとき、テキストによる代替は、少なくとも、その非テキストコンテンツを識別できる説明を提供している。(メディアに関するその他の要件は、ガイドライン 1.2 を参照。)

テスト

非テキストコンテンツが、テキストで提示されると無効になるテスト又は演習のとき、テキストによる代替は、少なくともその非テキストコンテンツを識別できる説明を提供している。

感覚的

非テキストコンテンツが、特定の感覚的体験を創り出すことを主に意図しているとき、テキストによる代替は、少なくともその非テキストコンテンツを識別できる説明を提供している。

CAPTCHA

非テキストコンテンツが、コンピュータではなく人間がコンテンツにアクセスしていることを確認する目的で用いられているとき、テキストによる代替は、その非テキストコンテンツの目的を特定し、説明している。なおかつ、他の感覚による知覚に対応して出力する CAPTCHA の代替形式を提供することで、様々な障害に対応している。

装飾、整形、非表示

非テキストコンテンツが、純粋な装飾である、見た目の整形のためだけに用いられている、又は利用者に提供されるものではないとき、支援技術が無視できるように実装されている。

画像と代替テキストについて

【ポイント】

- 画像には内容を表す適切な代替テキスト（alt 属性）を提供する。
- 装飾目的の画像の場合は、代替テキストを空にして意味を持たないことを明示する。

【解説】

img 要素によって提供される画像には alt 属性でその内容を的確に表す代替テキストを提供します。そうすることで、視覚的に画像を見ることが困難な利用者であっても、音声でその画像の内容を知ることができるようになります。

【対応方法】

- 画像の代替テキストに、画像が伝えている情報と同等の情報を alt 属性に記入する。

- 代替テキストは概ね 140 字以内を目安にする。（スクリーンリーダーは、代替テキストの読み上げにおいて、途中からの再読や微調整がしにくく、長文は負担になりやすい。また点字ディスプレイでは表示セル数が限られるため可読性が下がる。）
- 代替テキストが長くなる場合は、画像の直下や本文に説明文を配置して詳細を提供し、代替テキストには要点のみを記載する。なお、1 枚に独立した情報が複数含まれる場合は画像を分割し、それぞれに代替テキストを付ける。
- 画像周辺に同じ内容の説明が既にあるなど、画像が補助的に用いられているだけのときは、重複を避けるため 代替テキストを空にする。
- 装飾目的のイラスト画像など内容理解に関与しない画像は alt="" とする（ただし alt 属性自体は必ず付ける。省略するとファイル名を読み上げる場合がある）。
- インライン SVG の場合は、<svg> に role="img" を付け、短い説明は <title>（または aria-label のどちらか一方）で与える。説明が長いときは <desc> を併用する。
- 装飾目的の SVG は読み上げ対象から除外するため aria-hidden="true"（必要に応じて focusable="false"）を付ける。

【不適切な例】

- 代替テキストが提供されていない重要な画像。
- 代替テキストがあっても、画像の内容を説明していない（例：“画像 1”とだけ記載している場合）。

【適切な例】

- 要点を簡潔に表した代替テキストが提供されている（例：風景画像に「夕暮れの山並みと沈む太陽」）。
- 手順図：alt には「申請手順の概要（1.登録→2.確認→3.送信）」と記載し、詳細は画像直下の本文で説明している。
- 装飾目的のイラストに alt="" が指定されている。

【対応例】

- リンク先の内容が単独で分かる文言のコード例。

```
[html]
  <a href="/a">A 商品の詳細</a>
```

【補足：alt 属性以外の代替情報の提供の仕方】

☙ 隠しテキストによる情報提供

隠しテキストは、画面には出さずスクリーンリーダーにだけ補足情報を伝えたいときに用います（例：「PDF／2MB」「別ウィンドウで開く」など）。可視テキストで提示すると冗長になったりレイアウトを崩したりする情報（アイコンのみの意味付け、略語の展開、ファイル形式・容量等）を、支援技術の利用者にも確実に伝える必要がある場合に限って使用してください。なお、すべての利用者にとって重要な情報は隠さず、原則として可視テキストで提示します。

利用するには、非表示でテキスト情報を提供するクラスを用意します。以下のリンクから詳細を参照できます（参考 https://accessibility.jp/column/1631/）

そのクラスを用いて情報を提供します。例えば、リンクが PDF であることをスクリーンリーダーでもわかるようにするには以下のように行います。

202x年度の結果

```
<a href="(pdf ファイルの uri)" class="pdf_icon">202x 年度の結果
    <span class="screen-reader-text">PDF ファイル</span>
</a>
```

アクセシビリティ以外の目的で利用することはしないでください。また、display:none や visibility:hidden を用いて非表示にすることができますが、スクリーンリーダーからも読み上げられないため、ここに記載の方法で隠しテキストを提供してください。

☙ a 要素において aria-label 属性（WAI-ARIA 技術）を使用して提供

a 要素に aria-label 属性を用いてテキスト情報をスクリーンリーダーに提供することができます。詳しくは以下のリンクを参照してください（参考 https://waic.jp/docs/WCAG-TECHS/ARIA8）
例えば、次の様に aria-label によりリンク先が PDF ファイルであることをスクリーンリーダーに伝えることができます。

```
< a href="(pdf ファイルの url)" class="pdf_icon"
aria-label="202x 年度の結果 PDF ファイル">202x 年度の結果
</a>
```

a 要素に aria-label 属性を付与した場合、多くのスクリーンリーダーはリンクテキスト（a 要素内の表示文字列）ではなく、aria-label の内容のみを読み上げます。したがって、表示しているリンクテキストの内容も漏れなく aria-label に含めて記載してください。

☙ i 要素において aria-label 属性（WAI-ARIA 技術）を使用して提供

Web フォントなど i 要素で提供している場合は、次のようにしてスクリーンリーダーに読ませることができます。

```
< a href="(pdf ファイルの url)">
  202x 年度の結果<i class="pdf_icon" aria-label="PDF ファイル"></i>
</a>
```

リンク画像について

【ポイント】

- バナーなどリンクを持つ画像の場合、代替テキストにはリンク先が分かる内容を入れる。

【解説】

バナーなど画像にリンク設定をする場合、代替テキストには画像が伝えているリンク先の内容を利用者が理解できるように設定する必要があります。ただし、続くテキストがあって、それがリンクに含まれる場合は、バナー画像の代替テキストは重複しないように空にしてください。

【対応方法】

- リンク付きの画像の場合は、リンク先の内容や目的が分かるテキストを代替テキストとして記入する。
- 同じ内容のリンクテキストとリンク設定した画像（非テキスト）が隣接している場合は、不要な重複を避けるため、リンク設定した画像（非テキスト）の代替テキストは不要とする。

画像とテキストからなる一つのリンクがある場合：

図の代替テキストは空（alt=””）とします。


```
<a href="https://www.ipa.go.jp/security/otasuketai-pr/" class="icon--out" target="_blank" tabindex="0">
<div class="basic-slide__img"><figure class="basic-slide__img-box img-box">
<div class="img-box__inner"><img src="/ps6vr7000000ixim-img/ps6vr7000000ixjl.png" alt="~~サイバーセキュリティお助け隊サービス~~"></div></figure></div>
<div class="basic-slide__box">
<h3 class="basic-slide__ttl ttl">サイバーセキュリティお助け隊サービス</h3>
```

誤って画像に余計な代替テキストを設定しないように注意する。

もし、画像部分だけの場合は、代替テキストは alt=” サイバーセキュリティお助け隊サービス” とする。


```
<a href="https://www.ipa.go.jp/security/otasuketai-pr/" class="icon--out" target="_blank" tabindex="0">
<div class="basic-slide__img"><figure class="basic-slide__img-box img-box">
<div class="img-box__inner"><img src="/ps6vr7000000ixim-img/ps6vr7000000ixjl.png" alt="サイバーセキュリティお助け隊サービス"></div></figure></div>
```

長い説明が必要な画像について

【ポイント】

- 案内図など、短い代替テキストだけでは説明が難しいものは、画像の近辺に説明のテキストを配置するか説明が記載されたページへのリンクを提供する。

【解説】

簡潔な代替テキストでは内容が伝わらない画像には、長い説明文を画像の周辺に記述するか又は長い説明文が記述されたページへのリンクを設置しなければならない。

【対応方法】

次の地図の場合は、伝えたいことはページ下部にテキストで提供されている。このような場合、地図の代替テキストは最小限の alt=”本社所在地地図”あるいは”本社所在地の周辺地図” で良い。もし、この地図が重要なもので、ページ下部のテキストが無い場合は、別ページにこのテキストを用意し、そこにリンクするようにする。それほど重要なものではない場合は、要約した 100 字程度の説明を代替テキストに設定すると良い。

〒113-6591
東京都文京区本駒込二丁目28番8号
文京グリーンコートセンターオフィス（総合受付13階）
TEL:03-5978-7620
FAX:03-5978-7510

最寄駅
都営三田線「千石駅」（A1またはA3出口）徒歩4分
東京メトロ南北線「駒込駅」徒歩9分
JR山手線「駒込駅」徒歩10分
JR山手線「巣鴨駅」徒歩12分

1.2.1 音声のみ及び映像のみ（収録済）レベル A

収録済の音声しか含まないメディア及び収録済の映像しか含まないメディアは、次の事項を満たしている。ただし、その音声又は映像がメディアによるテキストの代替であって、メディアによる代替であることが明確にラベル付けされている場合は除く:

収録済の音声しか含まない場合

時間依存メディアに対する代替コンテンツによって、収録済の音声しか含まないコンテンツと同等の情報を提供している。

収録済の映像しか含まない場合

時間依存メディアに対する代替コンテンツ又は音声トラックによって、収録済の映像しか含まないコンテンツと同等の情報を提供している。

【ポイント】

- 収録済の音声のみのコンテンツでは、同等の情報をテキストで提供する。
- 収録済の映像のみ（音声なし）のコンテンツでは、同等の情報をテキスト又は音声トラック（解説音声）で提供する。

【解説】

本書でいう「同等の情報」とは、対象コンテンツの視覚的・聴覚的内容を適切な順序で記述したテキストであり、元コンテンツで可能な操作や結果（例：リンク先、手順の結果）に到達できる手がかりを含む情報を指します。

収録済の音声のみコンテンツは、その内容をテキストで提供します。収録済の映像のみコンテンツも同様に、内容をテキストで提供します。これは、再生が難しい環境・利用者でも内容を把握できるようにするためです。

一方、映像のみのコンテンツに対し、その内容を説明する音声トラック（解説音声）を提供している場合は、本達成基準における追加のテキスト代替は不要です（音声トラックが代替を満たすため）。

また、当該の音声のみ／映像のみコンテンツが、同一ページ内の直前直後の本文等で同等の情報をすでに提示している場合も、別途の書き起こしは不要です。※この場合は、①情報が同一であること、②本文へ容易に到達できること、③見出しやラベルで対応関係が明示されていること、④メディア更新時に本文も同期更新されること、を満たしてください。

【対応方法】

- 音声のみ／映像のみのコンテンツが新たな情報を伝える場合は、再生できない利用者にも等価な情報に到達できるよう、（音声のみ→テキストによる代替）、（映像のみ→テキストによる代替または音声トラック）を提供する。本文と同一内容をメディアとして提示するだけで、かつメディアが本文の代替であることを明示している場合はこの限りではない。純装飾のメディアは支援技術から隠す（例：aria-hidden="true"）。
    - 音声コンテンツには、その内容を説明する説明テキストを用意し利用者が利用できるようにする。
    - 映像コンテンツには、その内容を説明するテキストまたは音声トラック（解説音声）を提供し、利用者が内容を把握できるようにする。
    - 音声コンテンツあるいは映像コンテンツが本文の代替である場合は、その旨をキャプション等で明示し、本文と内容を同一・同期に保つ。

※　この判断が難しい場合は、代替となる説明テキストを常に提供するようにしてください。

【不適切な例】

- 商品のセットアップの仕方を映像で説明するコンテンツがある。音声は無いが、映像の指示でセットアップの仕方を説明するものである。映像コンテンツの近くには、「この映像に従ってセットアップしてください。」とあるが、それ以上の説明は何も無い。

【適切な例】

- 社長の年初メッセージを紹介するページで、音声のみのコンテンツがある。音声はすべてテキストに起こされ、テキストコンテンツとして同じページに掲載されている。

1.2.2 キャプション（収録済）レベル A

同期したメディアに含まれているすべての収録済の音声コンテンツに対して、キャプションが提供されている。ただし、その同期したメディアがメディアによるテキストの代替であって、メディアによる代替であることが明確にラベル付けされている場合は除く。

【ポイント】

- 音声を含む動画には、会話の内容、話者の識別（例：名前・役職・話者交替）、意味のある非言語音（例：効果音・環境音・拍手・BGM の意図）などを含むキャプション（字幕）を提供する。

【解説】

動画コンテンツを掲載する場合、それを聞くことが困難な利用者に対しては字幕を提供します。どんな動画でも、必ず字幕が必要な訳ではありません。この見極めを出来るようになることがとても大切です。

字幕の要否は、当該動画の音声が新たな情報を伝えているかで判断します。

必要：会話・ナレーション・意味のある効果音等で内容理解に関わる情報がある場合。

省略可：無音／BGM のみ、または音声内容が同一ページの本文等で等価に提示・容易に到達でき、更新も同期している場合。

【対応方法】

- 字幕ファイルを用意し、動画に同期させる。
- 会話・ナレーションを反映し、話者名や話者識別を明示する。
- 意味のある効果音・環境音・BGM の意図を必要に応じて記述する（例：「[ドアが閉まる]」「[緊迫した BGM]」）。
- 音声認識を利用した自動生成の字幕でも、必ず人手でレビュー・修正する。
- 公開後のフィードバックに基づき、字幕を随時更新する。

【不適切な例】

- 字幕に話者名や意味のある効果音（例：「シャッターが開く」）が含まれていない。
- 動画内テロップだけに依存し、音声内容の字幕化をしていない。
- 自動生成した字幕の誤記（固有名詞・専門用語など）を修正せずに公開している。
- 字幕が音声と同期しておらず、数秒遅れる／先行する。

【適切な例】

- 話者名が含まれている。（例：（山田）「申請は明日までです」）
- 効果音が含まれている。（例：（ガレージのシャッターが開く音））
- テロップに頼らず、音声内容が字幕化されている。
- 字幕と音声がタイミング良く同期している。

【補足】

字幕は YouTube の自動字幕生成機能を利用することをお勧めします。誤った認識もありますが修正は容易です。YouTube に音声認識させたあと「動画の詳細」から結果を編集することができます。

YouTube による字幕編集の様子

1.2.3 音声解説、またはメディアに対する代替(収録済) レベル A

同期したメディアに含まれている収録済の映像コンテンツに対して、時間依存メディアに対する代替コンテンツ又は音声解説が提供されている。ただし、その同期したメディアがメディアによるテキストの代替であって、メディアによる代替であることが明確にラベル付けされている場合は除く。

【ポイント】

- 映像中の視覚情報（動作・表情・画面上の文字・図表・強調・指示カーソル等）が理解に必要なら、音声解説または同等のテキスト代替を提供する。

【解説】

この達成基準は、映像に含まれる視覚的な情報が内容理解に不可欠かどうかを基準に判断します。音声だけでは把握できない場面転換や手振り、画面上の文字や図表の内容、強調表示などが理解に関わる場合、利用者が再生中でも等価な情報に到達できるよう、二つの方法のいずれかで補う必要があります。ひとつは映像の進行に合わせて説明を読み上げる音声解説、もうひとつは時系列に沿って要点を記したテキスト代替（台本・詳細解説）です。

一方、スピーチや朗読のように音声のみで意味が完結する映像は本項の対象外です（ただし、同時に図表・実演など不可欠な視覚情報を示す場合は対象になります）。なお、この基準はテキスト代替でも適合できる点が特徴であり、上位の 1.2.5（レベル AA）は音声解説の提供が必須となりテキスト代替だけでは満たせないことにご注意ください。

【対応方法】

- 映像の視覚情報が内容理解に不可欠かを確認し、不可欠な場合は補助手段の提供を決定する。
- 補助手段は、音声解説（同一トラックに挿入／解説用の別トラック）または時系列のテキスト代替（ページ内に併置し、再生中でも容易に参照可能）から選ぶ。
- テキスト代替は見出し・タイムコード等で映像の進行と対応付け、対応関係が分かるように示す。
- 映像が本文のテキスト代替であり、その旨を明示し、内容が本文と同一・同期している場合は、本項の追加対応は不要とする。
- 装飾のみの映像は支援技術から非表示にする（例：aria-hidden="true"）。

【不適切な例】

- 折りたたみ手順の映像で、無音かつ音声解説もテキストによる説明もなく、どこをつまみ何回折るかなどの要点が分からない。
- 図表付きの解説映像で、グラフや数値の内容が音声でも本文でも説明されず、視覚情報がないと理解できない。

【適切な例】

- 地図の見方を解説する映像で、ハイライト箇所や指示カーソルの動きを音声解説で補足し、音だけでも何を示しているか分かる。
- 研究紹介の映像で、図表や画面上の文字の内容を時系列のテキスト代替に整理して同一ページ内に併置し、セクション見出しで動画進行と対応付けて参照しやすくしている。

1.2.4 キャプション（ライブ）レベル AA

同期したメディアに含まれているすべてのライブの音声コンテンツに対してキャプションが提供されている。

【ポイント】

- 音声を含むライブ配信には、その場で内容を伝えるキャプション（字幕）を提供する。

【解説】

リアルタイムで配信される映像や音声は、聞くことが困難な利用者にとって情報が得られないリスクが高いため、可能な限りライブキャプション（リアルタイム字幕）を提供することが求められます。

ライブ配信では収録済の動画と異なり、事前に完璧な字幕を用意できないため、音声認識技術や速記による字幕付与などの運用上の工夫が必要です。

なお、配信の音声がなくてもページの目的が達成できる場合は、字幕は必須ではありません。

【対応方法】

- 配信の音声がないと情報が伝わらない場合は、リアルタイムで字幕を提供する。
- 音声認識による自動字幕（例：YouTube Live、Teams、Zoom などの自動キャプション機能）を活用する。もし、辞書登録のような機能がある場合は、誤認識が発生しやすい固有名詞や専門用語を事前に登録しておく。
- 会話・ナレーションを反映し、話者名や話者識別を明示する。
- 意味のある効果音・環境音・BGM の意図を必要に応じて記述する

  （例：「[ドアが閉まる]」「[緊迫した BGM]」）。
- 可能であれば人による手動編集を併用して精度を補う。

【不適切な例】

- 新製品紹介のライブ配信で発表者の説明が字幕化されていないため、聴覚に障害のある利用者が新機能を理解できなかった。
- 緊急会見のライブ配信で、音声はあるが字幕が提供されておらず、災害関連の重要情報がリアルタイムでは得られなかった。

【適切な例】

- 新製品発表イベントの自動字幕作成機能を有する動画配信システムを行ってライブ配信。情報がリアルタイムに字幕表示された。
- 行政の緊急会見で音声認識字幕と人の速記を併用し、重要な数値や地名を誤認識せずに伝えられた。

【補足】

ライブ字幕は自動生成機能を使うのが現実的です。ただし、誤変換が発生するため、事前準備（辞書登録・マイク品質の向上）や運用体制（速記者の配置、事後のアーカイブ字幕修正）を組み合わせると効果的です。

1.2.5 音声解説(収録済) レベル AA

同期したメディアに含まれているすべての収録済の映像コンテンツに対して、音声解説が提供されている。

【ポイント】

- 映像中の視覚情報（動作・表情・画面上の文字・図表・強調・指示カーソル等）が理解に必要なら、音声解説を提供する。

【解説】

この達成基準は、映像に含まれる視覚的な情報が内容理解に不可欠かどうかを基準に判断します。音声だけでは把握できない場面転換や手振り、画面上の文字や図表の内容、強調表示などが理解に関わる場合、利用者が再生中でも等価な情報に到達できるよう補うために、映像の進行に合わせて説明を読み上げる音声解説を提供する必要があります。

一方、スピーチや朗読のように音声のみで意味が完結する映像は本項の対象外です（ただし、同時に図表・実演など不可欠な視覚情報を示す場合は対象になります）。

【対応方法】

- 映像の視覚情報が内容理解に不可欠な場合は、可能な限りその視覚情報をメインの音声トラック（ナレーション）に組み込み、音声だけで理解できるようにする。これで足りる場合は追加の音声解説は不要。。
- メイン音声に十分に組み込めない／タイミングが合わない場合は、補助手段として音声解説（同一トラックに挿入／解説用の別トラック）を用いる。
- 映像が本文のテキスト代替であり、その旨を明示し、内容が本文と同一・同期している場合は、本項の追加対応は不要とする。
- 装飾のみの映像は支援技術から非表示にする（例：aria-hidden="true"）。

【不適切な例】

- 折りたたみ手順の映像で、音声解説がなく、どこをつまみ何回折るかなどの要点が音声からは分からない。
- 図表付きの解説映像で、グラフや数値の内容が音声では説明されず、視覚情報がないと理解できない。

【適切な例】

- 地図の見方を解説する映像で、ハイライト箇所や指示カーソルの動きを音声解説で補足し、音だけでも何を示しているか分かる。
- 研究紹介の映像で、図表や画面上の文字の内容を時系列のテキスト代替に整理して同一ページ内に併置し、セクション見出しで動画進行と対応付けて参照しやすくしている。

1.3.1 情報及び関係性 レベル A

何らかの形で提示されている情報、構造、及び関係性は、プログラムによる解釈が可能である、又はテキストで提供されている。

見出しについて

【ポイント】

- h1（大見出し）は「ページ全体の主題」に対して 1 つを原則とする。
- 見出しレベル（h1-h6）は、ページ内で論理構造が階層的に維持されるようにする。

【解説】

ウェブページの文書においても、紙の文章と同様に、見出し（h1～h6）を用いて内容のまとまりごとの要点を簡潔に示してください。

スクリーンリーダー利用者は「見出し」のみを選択的に読み上げさせることで、効率よくページ内を巡回できるようになります。これは紙の新聞を読む際に見出しを見て必要な情報を探すのと似たような経験です。見出し要素（h1 から h6 まで）を適切に活用することで、ページの階層構造を明確化し、スクリーンリーダー利用者を含むすべての読者にとってページの構造が分かりやすくなります

なお、HTML5 前提で<article>や<section>等のセクショニング・コンテンツを用いる場合は、各コンテンツ内の主見出しとして h1 の使用を許容することが可能です。ただし、スクリーンリーダーによっては見出しナビゲーションで混乱が生じる場合があるため、単一ページに多数の h1 を置く設計は避けることを推奨します。

【対応方法】

- 「h1」はページの主要なテーマを表すため 1 ページに原則として 1 箇所だけ使用する。
- 各内容のまとまりの先頭に適切な階層の見出し（h2～h6）を使用する。
- 見出しレベルは論理構造に沿って順序を保って使用する。
- 視覚的見出しを置けない UI では、支援技術向けに視覚非表示の見出しを検討する（CSS で可視テキストを非表示にし、読み上げは有効）。

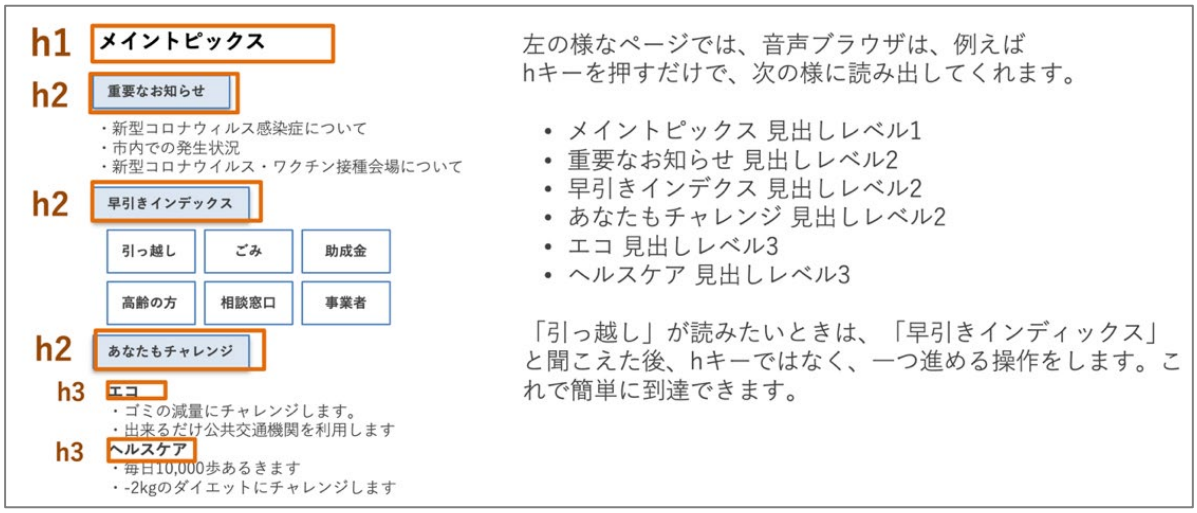
【不適切な例】

- 見出し語だと認識できるように視覚的な要素(文字サイズ、装飾、色など)が文字に付与されているが、見出し要素が設定されていない。

【適切な例】

- 見出し要素が適切に使用されているため、スクリーンリーダーで見出し要素を追いかけていくだけで、ページの構造が簡単に把握できる。

例.



リストについて

【ポイント】

- 段落番号や箇条書きは、適切なリスト要素（ol 要素、ul 要素）を用いて作成する。
- 用語とその説明をペアで示す場合は、定義リスト（dl 要素、dt 要素、dd 要素）を使用する。

【解説】

リスト要素を用いず段落番号や、箇条書きの行頭にある「・」を手入力すると、スクリーンリーダーでは段落数や箇条の個数などが把握できず、段落数や一つのリストにおける項目数などの情報をスクリーンリーダーの利用者に提供することができなくなります。また、他の数字や記号に埋もれてしまい、項目（段落番号や箇条書きの「・」）の確認がしづらくなります。

視覚的又は聴覚的な体裁によって暗に伝えられている情報や関係性を理解しやすくするためには、HTML のリスト要素を活用することが推奨されます。これにより、スクリーンリーダー利用者にも内容が伝わりやすくなります。また、視覚的にも情報が整理されていて、読む人が項目の関係性を把握しやすくなります。

用語集や名称と値の対応など「用語＋説明」の関係は、dl 要素（dt 要素が用語、dd 要素が説明）でマークアップすると、支援技術に関係性が正しく伝わります。

近年は、section 要素や div 要素に role="list"または role="listitem" を付与してリストを表現したり、nav 要素を複数用いてグループ化したりするケースがあります。これらも支援技術にはリストとして認識されますが、運用時には適切なラベル付与や区別が重要です。

【対応方法】

- 項目の順番に意味がない箇条書きリストについては ul 要素を使用する。
- 項目の順番に意味があるリストについては ol 要素を使用する。（段落番号は ol、li 要素でマークアップし、番号の形式は list-style-type など CSS で調整する）
- 箇条書きの形式ではあるが、行頭の記号や数字を出したくない場合は、該当のリストに対して行頭文字を非表示にする CSS を用意する。
- 用語集や仕様の定義などは、dl 要素を用いて「用語（dt 要素）―説明（dd 要素）」の関係を明示する。

【不適切な例】

- 中黒（・）で書き始めているところが数行ある。見た目では箇条書きに見えるが、リスト要素は用いられていない。

【適切な例】

- リスト要素を用いて箇条書きを示している。
- リスト要素を用いず、section 要素と div 要素で箇条書きが表現されている。section 要素には role="list"と役割が設定されており、それぞれの箇条部分の div 要素には role="listitem"と役割が設定されている。また、スクリーンリーダーでもリスト要素として認識されている。
- リスト要素は用いていないが、nav 要素によりグループを定義し、それぞれ aria-label 属性により区別を可能にしている。

テキストの意味付け

【ポイント】

- 意味は HTML のセマンティック要素で表し、見た目は CSS で調整する。
- 強調の意味は 、語の強勢は 、参照のハイライトは <mark> を用いる。
- 見た目だけ太字・斜体にしたい場合は b や i ではなく CSS を用いる（ 等）。
- ARIA はネイティブ要素が使えない場合の補完として最小限に用いる（置き換えではない）。

【解説】

意味は HTML、見た目は CSS を原則とします。重要性・緊急性・深刻性など意味を伴う強調には を用いて意味をマークアップし、太字化や色づけといった視覚の調整は CSS（例：font-weight や color）で行います。これにより、意味と表現が分離され、テーマ変更・デザイン更新・多言語化・自動要約や検索などの処理に対しても機械可読で一貫した構造を保てます。

強調の種類によって要素を使い分けます。内容としての重要性は 、文章中の語の強勢（アクセント）は 、検索語の参照ハイライトや文脈上の着目点は <mark> が適切です。見た目だけ太字・斜体にしたい場合は、<b> や <i> ではなく と CSS を用います（font-weight、font-style など）。<b>や<i> は意味を付与しない装飾であり、意味を伴う強調の代用には適しません。

また、過度な強調は可読性を下げます。 本当に重要な箇所に限定して を用い、見出しは見出し要素（<h1>～<h6>）で表すなど、文書構造と役割の適切な選択を徹底してください。こうした使い分けは、スタイルの変更や環境差があっても、意味の一貫性と保守性を確保するうえで有効です。

【対応方法】

- 意味を伴う強調は strong 要素、語の強勢は em 要素を用いる。
- 見た目の太字・斜体は CSS で調整する（例：… と font-weight / font-style）。
- 意味を伴わないのに b や i を使わない（装飾は CSS へ）。
- ARIA は補完的に最小限（ネイティブ要素が使える場合はそれを優先）。

【不適切な例】

- b 要素を使用してテキストを強調している。

```
<b>強調しているつもり</b>
```

- 見出しを … で代用している（構造が失われる）。

【適切な例】

- 重要な情報を strong 要素でマークし、太字等の視覚調整は CSS で行っている。

```
<strong style="font-weight: bold;">意味的にも視覚的にも強調されている</strong>
```

- 文中の語の強勢に em 要素を用いている。
- 見出しは h2 要素を用い、文中の重要語のみを strong 要素で強調し、太字の強さは **CSS** で調整している。

【補足】

- HTML5 では、strong 要素の他、em 要素や mark 要素もあります。em 要素は重要度を伴わない強調（その文章の一部にアクセントを付けたい場合）に使用します。mark 要素は、特定の語句をハイライトし、利用者が参照しやすくなる等の操作上の利点があります。
- ARIA について：role="strong" や role="emphasis" のような置き換えは存在しません。強調の意味付けは HTML 要素で行い、ARIA は名前付け（aria-label 等）や状態（aria-expanded 等）の補完に留めます。

表について

【ポイント】

- 表には見出しセル（th 要素）を適切に使用する。
- 複雑な表では、見出しセルの方向を scope 属性で明示する。

【解説】

スクリーンリーダーは表の内容を左上のセルから右へと順に読み上げていき、行が変わるごとに新しい行の左端から読み上げを始めます。もし見出しセルが適切に設定されていれば、関連するデータセルを読み上げる前にその見出しセルを読み上げます。

ただし、特定の位置（例：左上隅）では行の見出しか列の見出しか判断が難しくなる場合があります。このような状況で方向が不明確になる見出しセルは、scope 属性を使用して方向を明示します。

また、結合セルを頻繁に使用すると、スクリーンリーダー利用者に正確な意図が伝わらなくなる可能性があります。したがって、可能な限りシンプルな表を作成するよう心掛け、必要に応じて表を分割することをお勧めします。別の選択肢としては、表を避けてリスト要素等を用いる方法もあります。

【対応方法】

- 表における見出しセルは、th 要素を用いて明示する。

  見出しセルの方向が曖昧な場合、scope 属性を用いて方向を明示する。

  列方向（通常下方向）：scope="col" または グループ見出しの場合は scope="colgroup"
  行方向（通常右方向）：scope="row" または グループ見出しの場合は scope="rowgroup"

  ※ セルの方向が不明確になりがちな場所は、左上隅や複数の結合セルが使用されている場所です。

【不適切な例】

- 複雑な結合セルの使用で行と列の関連性が不明確な表。
- 行や列の見出しに適切なマークアップがない表。

【適切な例】

- 行と列の見出しに th 要素と scope 属性が適切に用いられている表。
- 内容が複雑でも適切に分割され、シンプルな構造の表。

【対応例】

- scope で見出しの方向を明示するコード例。

```
[html]
  <table><tr><th scope="col">品目</th><th scope="col">Large</th></tr><tr><th scope="row">コーヒー</th><td>320
円</td></tr></table>
```

【補足】正しいコーディングによる効果

次のような単純な表では、例えば「320 円」のセルにフォーカスを当てるとスクリーンリーダーは、『コーヒー Large 320 円』と読み上げます。

	(th) コーヒー	(th) 紅茶
(th) Large	(td) 320 円	(td) 300 円
(th) Medium	(td) 280 円	(td) 260 円

これは、Large という見出しセルは横方向に並ぶセルを対象にしていること、コーヒーという見出しセルは縦方向に並ぶセルを対象にしていることが、シンプルな表であるためスクリーンリーダーは推測可能だからです。

次の様に、scope 属性を用いて明示することもできます。scope 属性により、Large は row すなわち行方向に指定されるもの、コーヒーは col すなわち列方向に指定されるものであることをスクリーンリーダーは確実に認識できるようになります。(このようなシンプルな表の場合は、scope 属性の使用は必須としません。)

	(th scope="col") コーヒー	(th scope="col") 紅茶
(th scope="row") Large	(td) 320 円	(td) 300 円
(th scope="row") Medium	(td) 280 円	(td) 260 円

次の様に複数セルを結合した箇所がある場合は、例えば、アールグレイとある部分のセルは、もしかすると行方向の見出しかも知れません。下の表では人が見ると明らかに列方向だと判断できますが、スクリーンリーダーでは判断は難しく、次のように scope="col"と明示する必要があります。

	(th scope="col") コーヒー	(th colspan="2" scope="colgroup") 紅茶	
		(th scope="col") アールグレイ	(th scope="col") セイロン
(th scope="row") Large	(td) 320 円	(td) 310 円	(td) 300 円
(th scope="row") Medium	(td) 280 円	(td) 270 円	(td) 260 円

こうすることでスクリーンリーダーは、『紅茶 アールグレイ Large 310 円』と「アールグレイ」は 310 円に対しての見出しであると判断し読み上げるようになります。

1.3.2 意味のあるシーケンス レベル A

コンテンツが提示されている順序が意味に影響を及ぼす場合には、正しく読むシーケンスはプログラムによる解釈が可能である。

読み上げの順序について

【ポイント】

- コンテンツは論理的かつ予測可能な順序で提示されている。

【解説】

ウェブページの内容がスクリーンリーダーを用いた利用者や支援技術を利用する人々にとって論理的な流れで読み上げられることは、情報が適切に伝わりやすくするため非常に重要です。

特に、スクリーンリーダーは DOM ツリーの順番に沿ってコンテンツを読み上げるため、CSS の order プロパティなどで要素の並びを視覚的に入れ替えると、見た目と読み上げ順序がずれてしまうことがあります。

このため、視覚的な見え方と DOM 順序を一致させ、論理的な流れを保つことが必要です。

なお、WCAG 1.3.2 は特定の属性を禁止していませんが、tabindex で読み順を変えると DOM 順との不一致や保守性の低下を招くため、読み順の調整には用いないことを強く推奨します（フォーカス制御には必要に応じて使用可）。

【対応方法】

- コンテンツの論理的な流れを保持するため、視覚的な並び替えを極力避け、HTML の構造を利用して予測可能な順序を保つ。
- CSS を用いた視覚的な変更は、コンテンツの本質的な論理的順序を変更しないようにする。

【不適切な例】

- CSS の視覚的な並び替え（例：order）によって視覚順と DOM 順が不一致となり、読み上げ順と意味の流れがずれている。
- 見た目は「項目 A｜項目 B」だが、DOM は B→A の順で、スクリーンリーダーは右側（B）から読み上げる。

```
[html]
 <ul class="nav"> <li>項目 B</li> <!-- DOM 先頭 = 読み上げ先頭 --> <li>項目 A</li> </ul>
[css]
 .nav{display:flex}
 .nav li:first-child /* 視覚では後ろへ /
 .nav li:last-child / 視覚では前へ */
```

【適切な例】

- コンテンツの視覚的な並び替えは最小限であり、スクリーンリーダーでも論理的な流れで読み上げが行える。
- 視覚順と DOM 順を一致させ、読み上げも左から A→B になる。

```
[html]
 <ul class="nav"> <li>項目 A</li> <li>項目 B</li> </ul>
[css]
 .nav{display:flex} /* order 指定での入れ替えを行わない */
```

単語の読み上げについて

【ポイント】

- 単語の間に空白や改行を用いない。
- 人名など誤った読み上げ方をされる単語については読み仮名を提供する。

【解説】

見栄えなどを優先して単語の間に空白や改行を挿入すると、スクリーンリーダーでは正しく読み上げられないだけではなく、検索時にも正しく結果が表示されない可能性があります。

例えば、「日　時」のように、文字幅を空白で調整している単語は、読み上げでは「ひ　とき」のように誤って読んでしまう可能性があります。スクリーンリーダーの操作者は「ひ　とき」からは、どのような単語があるのか中々想像することができません。

文字間の調整については CSS の letter-spacing プロパティなど使用してください。

【対応方法】

- 単語の間に不必要な「空白」（半角・全角を問わず）や強制的な改行を用いない。
  （人名の場合、姓と名の間に「空白」を入力することは問題無い。）

【不適切な例】

- 「日時」「連絡先」「備考」いずれも文字間隔のバランスを取るために空白文字が間に含められている。

日　時	○○月○○日 9:00～
連 絡 先	電　話　：　03 – 1234 – 567x ＦＡＸ　：　03 – 1234 – 567y メール：　user@example.com
備　考	スリッパ持参のこと

【適切な例】

- 文字間の空白には letter-spacing プロパティが使用されている。

1.3.3 感覚的な特徴 レベル A

コンテンツを理解し操作するための説明は、形、大きさ、視覚的な位置、方向、又は音のような、構成要素が持つ感覚的な特徴だけに依存していない。

色に関する要件は、ガイドライン 1.4 を参照。

【ポイント】

- 感覚的な特徴だけに頼らず、指示や意味をテキストでも取得できるようにする（可視テキスト又はプログラムで解釈可能なラベル等）。
- 位置や形状（左右、四角や丸など）など視覚的な特徴だけに頼らない。
- 「下記」や「上記」といった位置指定を使う際は、その直後や直前に関連する情報を明示する。

【解説】

本達成基準の目的は、形・位置・色・音などの“感覚的な特徴だけ”に依存せず、指示や意味がテキストでも取得できるようにすることです。

例えば、位置や色だけで「右の赤いボタンを押す」と説明すると、スクリーンリーダー利用者や色の識別が難しい利用者には伝わりません。機能名などをテキストで示す（例：「再生ボタン（▶）」）か、可視テキストが難しい場合はプログラムで解釈可能なラベル（例：aria-label="再生"）を付与することで、誰にとっても理解・操作できる状態を確保します。

【対応方法】

- 記号（例：「◎」）が重要などの意味を持つ場合は、その記号を用いる前に「「◎」は重要を示します。」等の説明をする。
- 位置指定を行う文言（「上記」や「下記」）を用いる場合には、直前や直後に具体的な情報を示す。

【不適切な例】

- 「◎を参照」 - 記号の意味がテキストで説明されていない。
- 「右側にある情報を見てください」 - 位置指定が視覚的であり、スクリーンリーダーでのアクセスが困難。
- 「赤い丸のボタンを押してください」 — 色や形状だけの指示で、意味や機能がテキストで伝わらない。

【適切な例】

- 「「＊」は重要を示します。」 - 記号の意味がテキストで説明されている。
- 「以下のセクションに記載された情報を参照してください。」 - 位置指定が視覚的でなく、スクリーンリーダーでもアクセス可能である。
- 「『再生』ボタン（▶）を押してください」 — 機能名をテキストで明示し、記号は補助として併記している。
- 「赤い丸の『再生』ボタン（▶）を押してください」 — 機能名「再生」がテキストで示され、色・形状は補助情報として併記している。

【補足】

WCAG では「下記」「上記」という表現自体は、参照されているコンテンツが読み上げ順序の中で適切な位置にあり、文脈で内容が判断できれば禁止はされていません。ただし、正しい日本語表記法では「下記」を使用する場合、正式な記書き形式（「記」と「以上」を含む形式）にする必要があります。記書き形式ではなく、単にその下の文章を指す場合は、「以下のとおり」「次のとおり」などとするのが適切です。なお、「上記」については、記書きの必要はなく、上の文章を指します。ただし、「以下」「以上」「次」「前述」など、「下記」「上記」以外の表現を使う場合であっても、位置や文脈に関する WCAG の条件を満たす必要があります。

1.3.4 表示の向き レベル AA

コンテンツは、その表示及び操作を、縦向き（portrait）又は横向き（landscape）などの単一の向きに制限しない。ただし、その表示の向きが必要不可欠な場合は例外とする。

【ポイント】

- コンテンツは、デバイスの縦向きまたは横向きのどちらの向きでも利用可能である。
- 利用者が好む表示の向きでコンテンツを閲覧できるように、レイアウトはフレキシブルである。

【解説】

モバイルデバイスを使用する利用者は、状況に応じてデバイスの向きを変更することが一般的です。視覚障害、身体障害、または特定のハードウェアを使用する利用者は、特定の表示向きでデバイスを使用することが多いため、コンテンツはどちらの向きでもアクセス可能であるべきです。ウェブページやアプリケーションが特定の表示向きに固定されていると、これらの利用者がコンテンツにアクセスし、操作することが困難になる可能性があります。

【対応方法】

- CSS メディアクエリを使用して、レスポンシブなデザインを実装し、異なる表示の向きに対応する。
- 必要な操作がすべての表示の向きで実行できることを確認し、利用者が好む表示の向きを制限しない。
- デザインテストを行い、縦向き及び横向きの表示でコンテンツが適切に表示され、機能することを保証する。

【不適切な例】

- 地図表示では横向き表示のみをサポートし、縦向きにしても画面は横向きのままである。そのため、デバイスを縦置きで固定している利用者は、見づらく、かつフリックの方向なども変わっているため操作が困難になってしまった。

【適切な例】

- ウェブサイトがレスポンシブデザインを採用し、利用者がデバイスを縦向きまたは横向きにしても、すべてのコンテンツが適切に表示され、機能が利用可能である。

1.3.5 入力目的の特定 レベル AA

利用者の情報を集める入力フィールドのそれぞれの目的は、次の場合にプログラムによる解釈が可能である

- 入力フィールドが、ユーザインタフェース コンポーネントの入力目的の節で示される目的を提供している
- フォーム入力データとして想定される意味の特定をサポートする技術を用いて、コンテンツが実装されている

【ポイント】

- フォーム入力欄では、利用者が入力する情報の種類が明確になっている。
- フォームフィールドには、プログラムが認識可能な名前や役割が関連付けられている。

【解説】

フォームにおいて、利用者が予測可能な情報を入力することは、効率的な操作とアクセシビリティの両方において重要です。特に、スクリーンリーダーを使用する利用者や自動フォーム記入ツールを使用する利用者にとって、入力欄が何の情報を求めているのかを正確に認識できることが必須です。

HTML5 のフォーム入力タイプや aria-label 属性などを用いて、入力欄ごとの目的をプログラムが理解しやすい形でマークアップすることで、よりアクセシブルなフォームが実現されます。

【対応方法】

- input 要素の type 属性を適切に使用し、入力内容がメールアドレス、電話番号などであることを示す。
- autocomplete 属性を使用して、利用者が以前に入力した値を自動的に入力欄に埋めることができるようにする。
  ※7. ユーザインタフェース コンポーネントの入力目的: https://waic.jp/translations/WCAG21/#input-purposes
- aria-label 属性や label 要素を適切に使用し、視覚的なラベルとプログラム的なラベルが一致するようにする。

【不適切な例】

- 入力フィールドが input type="text" のみで定義され、入力する情報の種類が不明確な場合。
- フォームフィールドが視覚的には明確なラベルを持っているが、これらのラベルが支援技術で読み取れる形式でマークアップされていない。その結果、スクリーンリーダーを使用する利用者に対してフィールドの目的が明確に伝わらない。

【適切な例】

- HTML5 の autocomplete 属性を用いて、フォームフィールドの目的を明確に示している。これにより、支援技術は利用者に対して各フィールドが何を期待しているかを正確に通知できる。

```
<form>
  <label for="name">Full Name:</label>
  <input type="text" id="name" name="name" autocomplete="name">
  <!-- その他のフィールド -->
</form>
```

- 各フォーム入力欄に label 要素を関連付け、視覚的な説明とプログラム用の説明が提供されている。

【対応例】

- 適切な autocomplete 属性で入力を支援しているコード例。

```
<input name="name" autocomplete="name"> <input name="postal-code" autocomplete="postal-code">
```

1.4.1 色の使用 レベル A

色が、情報を伝える、動作を示す、反応を促す、又は視覚的な要素を判別するための唯一の視覚的手段になっていない。

この達成基準は、特に色の知覚に関するものである。その他の知覚形態については、色やその他の視覚的提示のコーディングへのプログラムによるアクセスも含めて、ガイドライン 1.3 で網羅されている。

【ポイント】

- 色だけに依存せず、形状・パターン・アイコン・テキスト（ラベル）・下線／線種などの手掛かりを併用して、色の違いが分からなくても情報が伝わるようにする。

【解説】

色を情報伝達の唯一の手段にはしない。 常に色以外の視覚的手段（形状・パターン・アイコン・下線・線種 等）やテキスト（ラベル）を併用して、色覚差や表示環境の違いがあっても意味や状態が分かるようにする。

色を強調目的で使う場合は「3.4 強調」を参照すること。

【対応方法】

- 同じ機能・状態は色以外の手掛かりも付与する（例：必須項目は「必須」とテキストで示す、エラーはアイコンや枠線のパターンも付与、リンクは下線などの線種で示す）。
- グラフや図の色分けには、凡例テキストやパターン（斜線・ドット等）を併用し、色だけに頼らない。
- 色同士の区別を視認しやすくするために、グラフィカルな指示には 1.4.11 に基づき隣接色に対して 3:1 以上、テキストには 1.4.3 に基づくコントラスト比を確保する。ただし、これは 1.4.1 の「色だけに依存しない」要件を置き換えるものではない。

【不適切な例】

- 緊急情報であることを、テキストを赤色にすることでのみ表現している。
（色覚障害のある利用者は識別できないかもしれません）

【適切な例】

- 緊急情報は赤色に加えてアイコンや太字等を併用し、「緊急」等のテキストも示している。
- 重要な情報を強調する場合、色差に加えて太字や下線を使用する。
- 色で区別している情報に対し、対応するテキスト（凡例やラベル）を提供する。
- 例：

（必須）の部分は必ず入力してください。

身分証明書（任意）[　　]
連絡先（必須）　[　　]

【対応例】

色だけに頼らず、太字などの非色覚的手掛かりで重要性やリンクを示す例。

```
[html]
 <label><strong>必須</strong>  メール</label>
 <a class="link" href="/policy">利用規約</a>
[css]
 .link{font-weight:700}
```

1.4.2 音声の制御 レベル A

ウェブページ上にある音声が自動的に再生され、3 秒より長く続く場合、その音声を一時停止又は停止するメカニズム、もしくはシステム全体の音量レベルに影響を与えずに音量レベルを調整できるメカニズムが利用できる。

この達成基準を満たさないコンテンツでは、利用者がそのウェブページ全体を使用できない恐れがあるため、ウェブページ上のすべてのコンテンツは他の達成基準を満たすために用いられているか否かにかかわらず、この達成基準を満たさなければならない。適合要件 5: 非干渉を参照。

【ポイント】

- ページが読み込まれたときに音声を自動的に再生させない。
- やむを得ず自動再生する場合は、利用者が直ちに停止または一時停止できるコントロールを提供する。

【解説】

スクリーンリーダーを使用する利用者にとって、自動再生される音声は読み上げを妨害し、ページ内容の理解を妨げます。

そのため、音声はユーザー操作によって再生が開始されるようにすることが基本です。やむを得ず自動再生を行う場合には、**3 秒以内に停止できる手段**や明確な停止ボタンを提供する必要があります。

【対応方法】

- 自動再生が必要な場合、3 秒以内に自動的に停止させる。
- 自動再生を避け、ユーザーが再生ボタンを押すまで音声を開始しない。
- ユーザーが容易に停止できるように、停止ボタンをページの先頭付近に配置する。

【不適切な例】

- ページを開いた瞬間から長時間音声が流れ続け、停止手段がない。

【適切な例】

- 自動再生される短い通知音は 3 秒未満で終了する。
- ページ上に明確な停止ボタンがあり、キーボード操作でもすぐにアクセスできる。

1.4.3 コントラスト(最低限) レベル AA

テキスト及び文字画像の視覚的提示に、少なくとも 4.5:1 のコントラスト比がある。ただし、次の場合は除く

大きな文字

サイズの大きなテキスト及びサイズの大きな文字画像に、少なくとも 3:1 のコントラスト比がある。

付随的

テキスト又は文字画像において、次の場合はコントラストの要件はない

- アクティブではないユーザインタフェース コンポーネントの一部である
- 純粋な装飾である
- 誰も視覚的に確認できない
- 重要な他の視覚的なコンテンツを含む写真の一部分である

ロゴタイプ

ロゴ又はブランド名の一部である文字には、最低限のコントラストの要件はない。

【ポイント】

- テキストおよび文字画像（画像内の文字、アウトライン化した SVG 文字など）と背景色は、通常のテキストは 4.5:1 以上、大文字は 3:1 以上のコントラスト比を確保する。
- マウスホバー時やフォーカス時あるいは placeholder はコントラスト比要件の対象となることに注意する。

【解説】

視覚障害のある利用者もテキストを読めるようにするため、テキストとその背景の間のコントラストを十分に確保することが重要です。マウスホバー時やフォーカス時あるいは画像内の文字情報やプレースホルダーのテキストも含まれます。ロゴは対象外です。

【対応方法】

- 色を使用している箇所のコントラスト比をチェックしコントラスト比が 4.5:1 以上あることを確認する。
- 欧文は 18pt（24 CSS px）以上、日本語は 22pt（29 CSS px）以上なら 3:1 で可。
  太字の欧文は 14pt（18.5 CSS px）以上、太字の日本語は 18pt（24 CSS px）以上なら 3:1 で可。

	通常のテキスト コントラスト比 4.5:1	**大きなテキスト コントラスト比 3:1**
欧米語	18pt（24 CSS px）未満	18pt (24px) 以上
欧米語（太字）	14pt（18.5 CSS px）未満	14pt（18.5 CSS px）以上
日本語	22pt（29 CSS px）未満	22pt（29 CSS px）以上
日本語（太字）	18pt（24 CSS px）未満	18pt（24 CSS px）以上

注意事項

- **四捨五入しない：**コントラスト比は小数点以下も含めてそのまま評価する（例：4.483:1 は 4.5:1 を満たさない）。

- **実表示で計測する：**ブラウザに描画された状態で採色・算出する（透明度・重ね順・グラデーション・半透明オーバーレイ・フィルター等の影響を含めて評価）。
- **アンチエイリアス対策：**拡大（例：200～400%）して、文字ストロークの内側（縁から 1–2px 離す）と直下の背景の実体色を採色する。境界の中間画素は評価に含めない。
- **設計値の算出は可（テキスト）：**CSS の前景色と背景色から算出してよいが、最終確認は実表示で行う。
- **文字画像は実測：**画像内の文字・アウトライン化文字（SVG 含む）・写真上の文字は設計値が取れないため、必ず実表示で計測する。
- **評価位置の考え方：**背景が写真・グラデーション・半透明の場合は、テキストの実体部分に重なる範囲で最もコントラストが低い箇所が基準（通常 4.5:1、大きなテキスト 3:1）を満たすか確認する。
- **状態ごとに確認：**通常・ホバー・フォーカス・プレースホルダーなど、状態変化後の配色でも基準を満たすことを確認する。
- **環境差の留意：**拡大表示、OS やブラウザのハイコントラスト・ダークモード等で見え方が変わるため、代表環境で確認する。
- **カラープロファイル：**評価は sRGB 前提。画像や SVG に P3 等が埋め込まれている場合は sRGB に変換したうえで実表示を採色する。
- **単位について：**本節の pt や px は CSS の単位を指す。評価は CSS ピクセル換算で行い、デバイスの実解像度とは直結しない。
- **大きなテキストの判定：**判定は CSS px に基づく（18pt≈24 CSS px、14pt 太字≈18.5 CSS px、22pt≈29 CSS px、18pt 太字≈24 CSS px）。

【不適切な例】

- 太字ではない 26px の日本語リンクでは 4.7:1 のコントラスト比がある。しかし、マウスをテキストの上に持っていくと、コントラスト比は 3.8:1 にまで下がり、4.5:1 を満たせない。
- テキストボックスのプレースホルダーはコントラスト比が 2.5:1 しかない。

【適切な例】

- **太字**の 26px の日本語リンクでは 4.7:1 のコントラスト比がある。マウスをテキストの上に持っていくと、コントラスト比は 3.8:1 にまで下がるが、3:1 以上はある。
- Tab キーでフォーカスを移動し”New”というリンクを選択した。リンクテキストと背景の配色が反転した。反転前も反転後も、コントラスト比は 4.5:1 以上になっている。

【対応例】

- 通常・ホバー／フォーカス・プレースホルダーのすべてでコントラスト比を満たす配色にするコード例（色値は例、実値はツールで確認）。

```
[html]
  <input placeholder="郵便番号">
[css]
 :root{--text:#222;--link:#1a5fb4;--link-hover:#184e96;--ph:#6b7280}
 body{color:var(--text);background:#fff}
 a{color:var(--link)}
 a:hover,a:focus{color:var(--link-hover)} /*  ホバーでも 4.5:1 を下回らない色を選ぶ  /
 input::placeholder{color:var(--ph)} /  プレースホルダーも 4.5:1 以上にする  */
```

【確認方法】

使用色がコントラスト比を満たしているかは専用ツールを使用することで確認が可能です。ここでは、フリーソフトであるカラー・コントラスト・アナライザーの使い方を解説します。

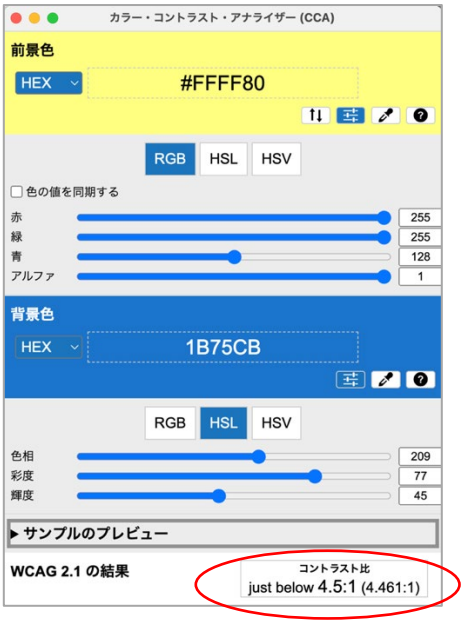
https://developer.paciellogroup.com/color-contrast-checker/

スポイトボタンを（  ）押して、表示中の画面で調べたい箇所にマウスを持っていきクリックすると色情報が取得されます。背景色と前景色（テキストの色）それぞれをスポイトで色情報を取得するとコントラスト比が表示されます。次の測定結果例では、4.5:1 をぎりぎり満たしていないことが示されています。

このツールを使って、色相を変えずに輝度のみを変えてコントラスト比を満たす色を探すことができます。

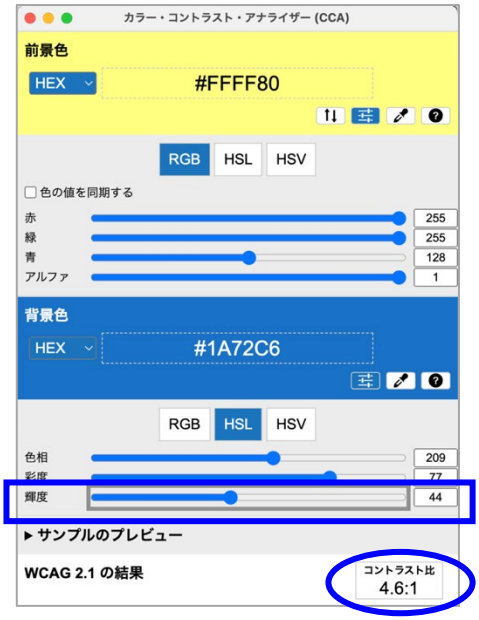
コントラスト不足

（ロゴと見なす場合は非適用）



just below では要件を満たしていない

HSL モードにして輝度を調整



輝度を調整し、色相は変えずにコントラスト比を満たす色を見つけることができる。

1.4.4 テキストのサイズ変更 レベル AA

キャプション及び文字画像を除き、テキストは、コンテンツ又は機能を損なうことなく、支援技術なしで 200%までサイズ変更できる。

【ポイント】

- ブラウザの文字サイズ変更またはページズームのどちらでも 200%まで拡大しても、ページ内のコンテンツ内容や機能を失わず、文字や画像の欠落・重なり・切り捨て等が発生しない状態を保つこと。
- 本文だけでなくフォーム・ボタン・メニュー等の操作部品も、200%拡大時に読めて操作できるか。
- 通常、ホバー、フォーカス、選択中などの状態でも、文字の欠落・重なり・切り捨てが起きないか。

※ リフローの要件は 1.4.10 で扱います（本項では義務ではありません）。

【解説】

弱視を含む多くの利用者は、テキストを自分に見やすいサイズに拡大してウェブサイトを利用します。

これに対応するため、テキストサイズは相対単位（%や em など）で指定し、ブラウザのズーム機能で 200%まで拡大してもレイアウトが崩れないように設計する必要があります。

【対応方法】

- テキストサイズは固定値ではなく、相対単位（%, em, rem）を使用する。

【不適切な例】

- テキストサイズが固定ピクセル値で指定されており、ブラウザの拡大機能で文字だけを大きくできない。

```
<p style="font-size: 14px;">サンプルテキスト</p>
```

- 拡大するとテキストがコンテナからはみ出し、横スクロールが必要になる。

【適切な例】

- テキストサイズが相対単位で指定され、ブラウザの 200%ズームでもすべての情報がビューポート内で読める。

```
<p style="font-size: 1em; line-height: 1.5em; letter-spacing: 0.1em; margin-bottom: 1em;">サンプルテキスト</p>
```

1.4.5 文字画像 レベル AA

使用している技術で意図した視覚的提示が可能である場合、文字画像ではなくテキストが情報伝達に用いられている。ただし、次に挙げる場合を除く:

カスタマイズ可能
文字画像は、利用者の要求に応じた視覚的なカスタマイズができる。

必要不可欠
テキストの特定の表現が、伝えようとする情報にとって必要不可欠である。ロゴタイプ（ロゴ又はブランド名の一部である文字）は必要不可欠なものであると考えられる。

【ポイント】

- テキストを装飾する目的で文字画像を用いない。

【解説】

テキストを装飾する目的で画像化すると、テキストの拡大や配色の変更が難しくなります。そのため、テキストを無用に画像化する行為は避けるべきです。ただし、ロゴや特定のデザイン要素としての文字が必要な場合は例外として、画像化が認められます。例えば、印刷物やホームページ以外のメディアで使用されている文字画像は、デザインの統一性を保つ目的で使用することが認められます。

【補足（定義）】

「文字画像」とは、文字の視覚的表現を画像に置き換えたものを指します（例：見出しを PNG 化）。テキストの拡大・配色変更・フォント切替の柔軟性を失うため、原則として避けます。

【例外の扱い】

ロゴや書体が本質的なデザイン要件である場合、画像化を例外的に認めます。ただし、テキスト代替（alt）を必ず付与し、リンク先や組織名が分かるようにします。利用者がフォント・配色を自分で変更できる仕組み（ユーザー設定）を提供している場合も例外に該当し得ます。

【対応方法】

- できる限り実テキスト＋CSS で表現する（影・縁取り・グラデーション等は CSS で代替）。
- 画像でしか実現できない装飾がある場合は、SVG の<text>による表現を検討する（アウトライン化は避ける）。

【不適切な例】

- ウェブサイト全体で見出しや本文を画像にして表示している。
- デザインの目的でテキストを画像として埋め込み、テキストとしての機能を失っている。

【適切な例】

- ウェブサイトのテキストが適切な HTML 要素を用いてマークアップされ、CSS でデザインされている。

1.4.10 リフロー（レベル AA）

コンテンツは、情報又は機能を損なうことなく、かつ、以下において 2 次元スクロールを必要とせずに提示できる:

- 320 CSS ピクセルに相当する幅の縦スクロールのコンテンツ。
- 256 CSS ピクセルに相当する高さの横スクロールのコンテンツ。

利用や意味の理解に 2 次元のレイアウトが必須である一部のコンテンツを除く。

320 CSS ピクセルは、1280 CSS ピクセル幅を 400 % ズームで見た場合の最初の表示幅に相当する。横スクロールになるように設計されたウェブコンテンツ (例えば、縦書きのテキスト) では、256 CSS ピクセルは、高さ 1024px を 400 %ズームで見た場合の最初の表示の高さに相当する。

2 次元のレイアウトを必要とするコンテンツの例としては、画像、マップ、図解、ビデオ、ゲーム、プレゼンテーション、データテーブル、及びコンテンツを操作している間にツールバーを表示しておく必要のあるインタフェースが挙げられる。

【ポイント】

- 320 CSS ピクセル幅で縦スクロールのみで読める（横スクロールを強いない）。
- 情報・機能・ラベル・入力欄が欠落せず、操作も維持される。
- 例外領域（地図・表・動画等）を除き、レイアウトは単一カラム等に自動再配置（リフロー）される。

【解説】

小さな表示幅や高倍率ズーム時でも、ユーザーが左右スクロールを強いられずに読み進め・操作できることが目的です。

ナビゲーション、フォーム、モーダル、オーバーレイ、ドロワーなど機能要素も再配置され、可視かつ可操作でなければなりません。画像・地図・表など二次元性が本質の要素は例外ですが、周辺 UI（説明・操作・凡例・拡大縮小ボタンなど）は縦スクロールで使える位置に配置します。

【対応方法】

- レイアウトは流動レイアウト（フルイド）＋メディアクエリでブレークポイントを設計し、320 CSS ピクセル幅で単一カラム化する。
- 固定幅・固定高さ・絶対配置の乱用を避け、テキストやカードは**折り返し（wrap）**を許可する。
- ヘッダー・フッター・サイドバーは 320 CSS ピクセル幅時にスタック（上下に積む）し、オフキャンバスにする場合は確実に開閉でき、開いた状態で主要内容を覆い尽くさないようにする。
- フォームはラベル・入力欄・エラーメッセージが 320 CSS ピクセル幅でも同一画面内で関連付けて知覚・操作できるよう、縦並びにする。
- 表は例外対象だが、可能なら列の優先度設定・折りたたみ・横スクロール（表領域に限定）で閲覧可能にし、見出しセルを貼り付けるようにして（sticky）可読性の向上を行う。
- モーダル／ダイアログは 320 CSS ピクセル幅ではみ出さず、フォーカス移動とスクロールがコンテンツ内に閉じるように実装する。
- overflow:hidden による内容の切り捨てや、ズレで操作ボタンが画面外に出る事態を防ぐ。

【不適切な例】

- 320 CSS ピクセル幅にすると本文やフォームが横スクロール必須になる（ナビと本文が横並び固定）。
- ハンバーガーメニューを開くと画面全体を覆い、本文にアクセスできない／閉じる手段が画面外に出ている。
- モーダルが画面からはみ出し、閉じるボタンや送信ボタンに到達できない。
- 重要な補助情報（ラベル、ツールチップ相当の説明）が改行や折り返しで不可視になり、操作の意味が失われる。

【適切な例】

- 320 CSS ピクセル幅では単一カラムに再配置され、本文→関連リンク→フッターの順で縦スクロールのみで読める。
- フォームはラベル→入力→エラーの順に縦並びで表示され、送信ボタンが常に画面内に到達可能。
- 地図ウィジェットは例外として横スクロールを許すが、説明文・検索欄・ズーム操作は地図の上か下に縦配置。
- データ表は表領域内にだけ横スクロールが発生し、行・列見出しを固定して意味を保つ。

【補足】

- 本項目はレイアウトの再配置に関する達成基準です。
- テキスト拡大（200%）は 1.4.4、行間・字間の調整は 1.4.12 が扱います（本項目では扱いません）。
- 実装確認は、ブラウザ幅 320 CSS ピクセル相当に縮める、または 1280px 幅端末で 400%ズームして、横スクロールの強制がないか／機能が欠落しないかを点検します。

1.4.11 非テキストのコントラスト（レベル AA）

以下の視覚的提示には、隣接した色との間で少なくとも 3:1 のコントラスト比がある。

ユーザインタフェース コンポーネント

ユーザインタフェース コンポーネント及び状態（state）を特定するのに必要な視覚的な情報。ただし、アクティブではないユーザインタフェース コンポーネントや、そのコンポーネントの見た目がユーザエージェントによって提示されていてコンテンツ制作者が変更していない場合は除く。

グラフィカルオブジェクト

コンテンツを理解するのに必要なグラフィック部分。ただし、そのグラフィック特有の提示が、情報を伝えるうえで必要不可欠な場合は除く。

【ポイント】

- グラフィック要素や利用者インターフェースコンポーネントは、隣接する色に対して十分なコントラストを持つ。
- 情報を伝えるための視覚的要素は、利用者にとって識別可能でなくてはならない。

【解説】

ウェブサイト上でのボタン、アイコン、画像、インジケータなどの非テキスト要素は、色覚障害や視覚障害のある利用者にとっても識別しやすい必要があります。これらの要素が低コントラストであると、必要な情報が伝わらず、ユーザーエクスペリエンスが低下します。そのため、これらの要素は少なくとも 3:1 のコントラスト比を持つことが求められます。

【対応方法】

- グラフィック要素やインタラクティブなコントロールは、周囲の色に対して 3:1 以上のコントラスト比を確保する。色相だけに頼らず、明度差や境界線・パターン等で識別性を補強する。
- ユーザインターフェースのコンポーネントは、その機能と状態を識別するのに十分な視覚情報を提供する。

【不適切な例】

- ラジオボタンが背景色と似た色でデザインされており、特に屋外や明るい場所でスクリーンを見る際にボタンの輪郭が見えづらくなっている。コントラスト比を測定すると 2.6:1 しかない。これでは、ボタンは無いのと同じである。

- データを示す円グラフが複数の似た色を用いて作成されており、色覚障害のある利用者には各セグメントを区別することが非常に困難になっている。グラフ内のテキストラベルも小さく、コントラストが不十分であるため、情報の理解を妨げている。これでは、ただの円が表示されているのに等しい。
  青 ： 緑 ＝#646EFB：#01CC96 = 2.0：1
  緑 ： 赤 ＝#01CC96：#EF553B = 1.7：1
  赤 ： 青 ＝#EF553B：#646EFB = 1.2：1

【適切な例】

- 通話を開始するアイコンは薄い青色の背景に濃い青色の電話のシルエットを用いており、3:1 以上のコントラスト比を確保し、色覚障害のある利用者でも明瞭に識別できるようになっている。(#0370C0, #DCE6F2, 4.1:1)

- インタラクティブな要素、例えばフォームの入力フィールドの枠線は、周囲の背景とはっきり区別できる 3:1 以上のコントラストがあり、どこが入力エリアなのかを利用者に明確に伝えている。(#7F7F7F, #FFFFFF, 4:1)

1.4.12 テキストの間隔（レベル AA）

以下のテキストスタイルプロパティをサポートするマークアップ言語を用いて実装されているコンテンツにおいては、以下をすべて設定し、かつ他のスタイルプロパティを変更しないことによって、コンテンツ又は機能の損失が生じない:

- 行の間隔（行送り）をフォントサイズの少なくとも 1.5 倍に設定する
- 段落に続く間隔をフォントサイズの少なくとも 2 倍に設定する
- 文字の間隔（字送り）をフォントサイズの少なくとも 0.12 倍に設定する
- 単語の間隔をフォントサイズの少なくとも 0.16 倍に設定する

例外: テキスト表記においてこれらのテキストスタイルプロパティの一つ以上を使用しない自然言語及び文字体系では、その言語と文字体系の組み合わせに存在するプロパティだけを用いて、この達成基準に適合することができる。

【ポイント】

- ユーザーが行間や字間を大きくしても、テキストが重なったり欠落したりしないこと。
- CSS で指定されたスタイルをユーザーが上書きしても、情報の関係性が保たれること。
- 主要なテキスト要素（本文・見出し・ボタンラベル・フォームなど）は、最低限以下の値まで拡張しても正しく表示されること。

【解説】

弱視・ディスレクシア・読字障害などの利用者は、標準の行間や字間では読みにくいため、ブラウザ拡張や独自 CSS で間隔を広げて可読性を向上させることがあります。この際、テキストが重なったりはみ出したり、UI 部品のラベルが隠れるなどの問題が起きると、コンテンツが利用できなくなります。1.4.12 は「ユーザーのカスタム間隔を許容する」ことを求めており、1.4.4（サイズ変更）や 1.4.10（リフロー）とは別の視点です。

【対応方法】

- 行間・字間・段落間を固定ピクセル値ではなく、em・rem・%など相対単位で指定する。
- フォームのラベル・入力欄・ボタン・リンクなど、テキストを含む UI 要素の上下左右に十分な余白（padding / margin）を確保する。
- テキストコンテナの高さを固定せず、高さは内容に応じて拡張する。
- テキストがボックスからはみ出さないか、背景画像と重ならないかを確認する。
- ツールチップやポップアップなど、制約の強い UI でも間隔を広げても崩れないことを確認する。

【不適切な例】

- 行間を 1.5 倍にすると、テキストが隣のコンテンツに重なり読めなくなる。
- 段落間を広げると、テキストがコンテナからはみ出し、下のボタンが隠れる。
- ボタンラベルの字間を広げるとテキストがボタン枠からはみ出す。

【適切な例】

- 行間・段落間を広げても、テキストやボタン・フォームの UI が正しく拡張され、レイアウトが崩れない。

- ボタンのテキストが字間を広げても枠内に収まり、クリック領域も保持されている。

【補足】

1.4.12 はテキスト間隔の調整を目的とし、文字サイズ変更（1.4.4）やリフロー（1.4.10）は別基準です。

テスト方法としては、ブラウザ拡張（例：Stylus）や開発者ツールを使って上記の基準値を適用し、表示崩れ・重なり・情報欠落がないかを確認します。

1.4.13 コンテンツ上にホバーやフォーカスで表示される情報 レベル AA

ポインタホバー又はキーボードフォーカスを受け取ってから外すことで、追加コンテンツを表示させてから非表示にさせる場合は、以下の要件を全て満たす:

非表示にすることができる

ポインタホバー又はキーボードフォーカスを動かさずに追加コンテンツを非表示にするメカニズムが存在する。ただし、追加コンテンツが入力エラーを伝える場合や、他のコンテンツを不明瞭にしたり置き換えたりしない場合は除く。

ホバーすることができる

ポインタホバーによって追加コンテンツを表示させることができる場合、その追加コンテンツを消すことなく、ポインタを追加コンテンツ上で動かすことができる。

表示が継続される

ホバーやフォーカスが解除される、利用者が非表示にする、又はその情報が有効でなくなるまでは、追加コンテンツが表示され続ける。

例外: 追加コンテンツの視覚的提示がユーザエージェントによって制御されていて、かつコンテンツ制作者が変更していない場合は例外とする。

ユーザエージェントによって制御されている追加コンテンツの例としては、HTML の title 属性を用いて作られているブラウザのツールチップが挙げられる。

ホバー時やフォーカス時に表示されるカスタムツールチップ、サブメニュー、他の非モーダルポップアップは、この達成基準の適用対象となる「追加コンテンツ」の例である。

【ポイント】

- 追加コンテンツは、すべての利用者が到達でき、表示・保持・閉じる等の操作が可能で、内容を知覚・理解できること（= 利用可能）。
- ホバーやフォーカスにより表示される追加コンテンツは、利用者の操作により閉じられるまで持続する。

【解説】

ウェブサイト上でツールチップやポップアップを使用して情報を提供することは一般的ですが、これらが突然消えたり、拡大時に全内容が見えなくなったりすると、特に視覚障害のある利用者にとってアクセスの障壁となります。そのため、追加される情報は利用者がコントロール可能であり、操作が終了するまで持続するように設計することが重要です。

【対応方法】

- ホバーやフォーカスにより表示される情報は、利用者が明示的に閉じる操作をするまで持続する。
- 情報がスクリーンリーダー等の支援技術で読み取り可能であり、ポインタ（マウス・タッチ）およびキーボードの双方で、表示・保持・閉じる操作が可能であることを確認する。
- 弱視の利用者が拡大して読むことを考慮し、ツールチップやポップアップが拡大表示に耐えうる設計となっていること。
- 表示された追加コンテンツ上で操作できるようにする。 ポインタ移動・クリックおよびキーボード操作（フォーカス移動、Enter／Space 等）で操作でき、意図せず閉じないようにする。
- ホバーに依存せず、同一情報に別経路でも到達できるようにする。 例：クリック操作やキーボードショートカット。

【不適切な例】

- ボタンのホバーでツールチップを表示しているが、ボタンからツールチップへポインタを移動した時点でツールチップが消えるため、ツールチップ上で読む／操作することができない。
- 情報が拡大表示時に画面外に切れ、利用者が全内容を閲覧できない。

【適切な例】

- 利用者がツールチップを閉じるための「閉じる」ボタンをクリックするまで、ツールチップが表示され続ける。
- キーボードナビゲーションを利用して、ツールチップ内の全ての情報を拡大して読むことができる。
- マウスをツールチップの上に持っていったときも、ツールチップの表示は継続される。

【対応例】

- ホバーまたはフォーカスで表示を持続させる簡易ツールチップコード例。

[html]

```
<button aria-describedby="tip1">用語</button><span id="tip1" class="tip" role="tooltip">説明テキスト</span>
```

[css]

```
.tip{position:absolute;display:none}.tip:focus-withinbutton:hover+.tip,button:focus+.tip{display:block}
```

2.1.1 キーボード レベル A

コンテンツのすべての機能は、個々のキーストロークに特定のタイミングを要することなく、キーボードインタフェースを通じて操作可能である。ただし、その根本的な機能が利用者の動作による終点だけではない軌跡に依存する入力を必要とする場合は除く。

上記の例外は、根本的な機能に関するものであり、入力手法に関するものではない。例えば、テキスト入力に手書き入力を用いるのであれば、その入力手法（手書き）は利用者の動作による軌跡に依存した入力を必要とするが、その根本的な機能（テキスト入力）は利用者の動作による軌跡に依存した入力を必要とするものではない。

これは、キーボード操作に加えて、マウス入力、又はその他の入力手段を提供することを禁ずるものでも妨げるものでもない。

【ポイント】

- 展開式のナビゲーションや入力フォーム、動画プレイヤーなど利用者の操作が必要なコンテンツを使用する場合、マウスを使用せずにキーボードのみで操作できるようにする。

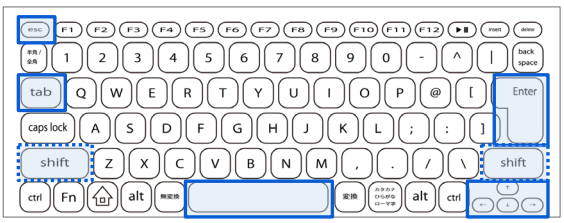
【解説】

マウスポインタを見ることができない、手に震えがあって細かな操作は難しいなど、マウスを使用できない利用者は、キーボードを駆使してウェブページを閲覧している場合があります。そのため、キーボード操作だけでウェブページを閲覧できるようにウェブページを制作しておく必要があります。

【対応方法】

- 代表的な実装として、リンクは a 要素、ボタンは button 要素、フォームは標準のフォームコントロール（input／select／textarea）を用いる（独自 UI が必要な場合も原則ネイティブ優先）。参考：HTML のフォームコントロール及びリンクを使用する（H91）
- https://waic.jp/translations/WCAG21/Techniques/html/H91
- マウスカーソルのホバーやクリックのみで動作するコンテンツを制作しない。
- キーボードによる操作でも、マウス操作と同様の動作になるようにコンテンツを実装する。
- キーボードで操作できない機能を使用する場合は、別の方法で同様の情報を提供する手段を用意する。

良く使用されるキー



tab	フォーカスの移動
shift-tab	逆方向への移動
Enter	項目選択 リンク等
Space	項目選択 ボタン等
矢印キー	項目移動 リスト等
Esc	ダイアログクローズ等

【不適切な例】

- ハンバーガーメニューはマウスクリックで展開したり閉じたりさせることができるが、キーボードで選択することができない。
- 展開ボタンをキーボードで選択はできるが、選択して Enter キーを押しても何も反応しない。
- キーボード操作でフォーカスを移動させることができないポップアップウィンドウやモーダルダイアログがある。

【適切な例】

- 動画プレイヤーでは、tab キーで再生ボタンを選択することができ、Enter キーを押すと動画が再生される。

2.1.2 キーボードトラップ無し レベル A

キーボードインタフェースを用いてキーボードフォーカスをそのウェブページのあるコンポーネントに移動できる場合、キーボードインタフェースだけを用いてそのコンポーネントからフォーカスを外すことが可能である。さらに、修飾キーを伴わない矢印キー、Tab キー、又はフォーカスを外すその他の標準的な方法でフォーカスを外せない場合は、フォーカスを外す方法が利用者に通知される。

この達成基準を満たさないコンテンツでは、利用者がそのウェブページ全体を使用できない恐れがあるため、ウェブページ上のすべてのコンテンツは他の達成基準を満たすために用いられているか否かにかかわらず、この達成基準を満たさなければならない。適合要件 5: 非干渉を参照。

【ポイント】

- キーボード操作によってフォーカスが閉じ込められるトラップを作成しない。

【解説】

利用者がキーボード操作を使用してウェブコンテンツを移動する際、キーボードトラップとなる現象を避ける必要があります。これは、利用者が特定のコンテンツやエレメントにフォーカスを移動させた際に、キーボードだけではフォーカスをそのエリアから移動させることができない状況を指します。これにより、ウェブサイトのアクセスが制限され、利用者が必要な情報や機能にアクセスできなくなる可能性があります。以下のような状況は避けるべきです：

- 特定の項目やコンテンツにフォーカスが固定され、他のエリアへの移動ができない。
- キーボードの操作で前進または後退ができない項目がある。

【対応方法】

- ウェブコンテンツはキーボードのみで操作可能であり、利用者がフォーカスを閉じ込められるトラップに陥ることなく、任意の場所に移動できるようにする。

【不適切な例】

- キーボードでのナビゲーションが制限され、特定のセクションから抜け出ることができない。

【適切な例】

- すべてのコンテンツと機能がキーボード操作で簡単にアクセスでき、フォーカスは自由に移動できる。
- 利用者がキーボードのみを使用しても、任意のエリアやコンテンツに簡単にアクセスできる。

【対応例】

- モーダルから確実に退出できる経路を備えているコード例。
  モーダル内でも Tab/Shift+Tab だけで「閉じる」に到達し、Esc キー または閉じるにより背景へ戻れるようにしている。閉じた後は起動ボタンへフォーカスを戻し、tab を JavaScript で無効化したりフォーカスを固定したりしていない。

[html]

```
<div role="dialog" aria-modal="true" id="dlg" hidden> <button id="closeBtn">閉じる</button> </div> <button id="openBtn">開く</button>
```

[js]

```
openBtn.onclick=()=>{dlg.hidden=false;closeBtn.focus()};closeBtn.onclick=()=>{dlg.hidden=true;openBtn.focus()}
```

2.1.4 文字キーのショートカット レベル A

文字 (大文字と小文字を含む)、句読点、数字、又は記号のみを使用したキーボードショートカットがコンテンツに実装されている場合、少なくとも次のいずれかを満たしている:

解除

ショートカットを解除するメカニズムが利用できる

再割り当て

一つ以上の非印字文字 (例えば Ctrl や Alt など) を使用するようにショートカットを再割り当てするメカニズムが利用できる

フォーカス中にのみ有効化

ユーザインタフェース コンポーネントのキーボードショートカットは、そのコンポーネントがフォーカスをもっているときのみ有効になる。

【ポイント】

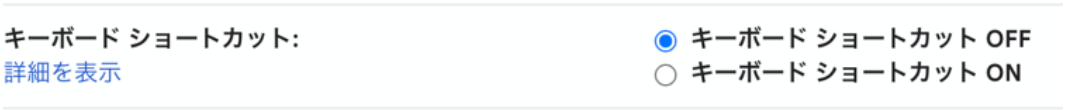
- ショートカットキーは、キーボード操作に便利さをもたらすが、意図しない操作を引き起こさないよう配慮する。
- 文字キーを使用するショートカットは、利用者が無効化、変更、または使用しないように設定できる。

【解説】

文字キー（例：単一の文字、数字、句読点）によるキーボードショートカットは、多くのウェブアプリケーションで操作の効率化を図るために提供されています。しかしこれらは、スクリーンリーダーなどの支援技術を使用している利用者や、特定のキーが他のコマンドと競合する場合に問題を引き起こす可能性があります。このような不便を避けるために、利用者がショートカットを簡単に制御できるようにすることが重要です。

特に音声認識ソフトを使用して、音声でアプリケーションをコントロールしている利用者の場合、単純なショートカットキーでは、発声の誤認識や、テレビや他社の発話を誤って認識することによって、誤動作してしまうことがあります。そのため、ショートカットキーは組合せキーを用いることが多く、また、すでに多くのショートカットキーをアサインしている可能性もあり、この要件が求められています。

例えば Gmail では、キーボードショートカットを無効化することができます。



【対応方法】

- ショートカットキーを完全に無効化するオプションを提供する。
- 利用者が個々のショートカットキーをカスタマイズできる設定を実装する。
- ショートカットキーがアクティブになるのは、利用者がその機能を使用する時だけに限定する（例：テキスト入力フィールドにフォーカスがあるときなど）。

【不適切な例】

- 文字キー「S」を押すだけで、ウェブサイト上で無意識のうちに保存操作が行われてしまった。
- 誰かが「どう?」と話しかけてきたので、音声認識ソフトは do と認識し、それをウェブアプリケーションに送信した。ウェブアプリケーションは d とタイプされたと認識し、それまで作成していたメールを消してしまった。

【適切な例】

- 利用者が設定メニューからキーボードショートカットをカスタマイズできる。
- ショートカットは単独の文字キーでは反応せず、Ctrl／Alt／Cmd などの修飾キーとの組み合わせに再割り当てされている（例：Ctrl+S、Alt+N など）。

2.2.1 タイミング調整可能 レベル A

コンテンツに制限時間を設定する場合は、次に挙げる事項のうち、少なくとも一つを満たしている

解除

制限時間があるコンテンツを利用する前に、利用者がその制限時間を解除することができる。又は、

調整

制限時間があるコンテンツを利用する前に、利用者が少なくともデフォルト設定の 10 倍を超える、大幅な制限時間の調整をすることができる。又は、

延長

時間切れになる前に利用者に警告し、かつ少なくとも 20 秒間の猶予をもって、例えば「スペースキーを押す」などの簡単な操作により、利用者が制限時間を少なくとも 10 倍以上延長することができる。又は、

リアルタイムの例外

リアルタイムのイベント（例えば、オークション）において制限時間が必須の要素で、その制限時間に代わる手段が存在しない。又は、

必要不可欠な例外

制限時間が必要不可欠なもので、制限時間を延長することがコンテンツの動作を無効にすることになる。又は、

20 時間の例外

制限時間が 20 時間よりも長い。

この達成基準は、制限時間の結果として、コンテンツ又は状況の予期せぬ変化を引き起こさないように利用者がタスクを完了できるようにするためのものである。この達成基準は、利用者の動作の結果としてのコンテンツ又はコンテキストの変化を制限する達成基準 3.2.1 と併せて考慮すること。

【ポイント】

- 制限時間があるコンテンツが存在する場合、利用者に制限時間を解除するオプションを提供する。

【解説】

数秒後に自動的にページ転送を行う処理や、制限時間内に項目の選択や操作が必要なコンテンツがある場合は、利用者が操作に時間がかかることを考慮して、制限時間の延長や解除が可能なオプションを提供してください。ただし、制限時間のない即時リダイレクト処理はこの達成基準の適用外です。（数秒待ってからのリダイレクト処理はこの達成基準に該当します。）

セキュリティ上の懸念に対応するために設けた制限時間（例：セッションタイムアウト、確認コードの有効期限）も、本達成基準における「コンテンツが設定した制限時間」に含まれます。 原則として、解除・調整・延長のいずれかを提供することが求められます。どうしても延長ができない設計が機能上必要不可欠な場合（例：延長すると安全性が損なわれるワンタイムコード等）は「必要不可欠の例外」に該当し得ますが、その場合でも、期限切れの明示、再送・再発行の簡便な手段、入力内容の保持（再ログイン後に復元できる等）など、実質的な救済策を併せて提供してください。

リアルタイムイベント（オークション等）は別途の例外に該当します。

【対応方法 下記について少なくとも 1 つを満たす必要がある】

- 制限時間があるコンテンツには制限時間解除の機能を提供する。
- 制限時間の設定について、利用者が制限の 10 倍以上延長することができる。
- 時間切れの前に利用者に警告し、少なくとも 20 秒の猶予をもって簡単な操作（スペースキーを押すなど）で 10 倍以上の延長を可能にする。
- 制限時間が 20 時間より長い。
- リアルタイムのイベントで制限時間が必須の場合（オークションなど）は例外とする。
- 制限時間が必須で制限時間の延長や解除がコンテンツの動作を無効にする場合は例外とする。

【不適切な例】

- ウェブサイトのフォームで制限時間が設けられているが、利用者がその時間を延長したり解除したりできるオプションが提供されていない。

【適切な例】

- ウェブサイトのフォームには制限時間があるが、時間切れ前に入力継続の確認ポップアップを表示し、画面全体をオーバーレイで遮蔽する。ポップアップ表示中はタイマーを一時停止し、Enter／Space／「はい」の簡単操作で必要な回数だけ延長できる。フォーカスはモーダルに移動し、キーボード操作だけで完結する。

2.2.2 一時停止、停止、非表示 レベル A

動きのある、点滅している、スクロールする、又は自動更新する情報は、次のすべての事項を満たしている

動き、点滅、スクロール

動きのある、点滅している、又はスクロールしている情報が、(1) 自動的に開始し、(2) 5 秒よりも長く継続し、かつ、(3) その他のコンテンツと並行して提示される場合、利用者がそれらを一時停止、停止、又は非表示にすることのできるメカニズムがある。ただし、その動き、点滅、又はスクロールが必要不可欠な動作の一部である場合は除く。

自動更新

自動更新する情報が、(1) 自動的に開始し、 (2) その他のコンテンツと並行して提示される場合、利用者がそれを一時停止、停止、もしくは非表示にする、又はその更新頻度を調整することのできるメカニズムがある。ただし、その自動更新が必要不可欠な動作の一部である場合は除く。

画面がちらつく、又は閃光を放つコンテンツに関する要件は、ガイドライン 2.3 を参照。

この達成基準を満たさないコンテンツでは、利用者がそのウェブページ全体を使用できない恐れがあるため、ウェブページ上のすべてのコンテンツは他の達成基準を満たすために用いられているか否かにかかわらず、この達成基準を満たさなければならない。適合要件 5: 非干渉を参照。

定期的にソフトウェアによって自動的に更新されるコンテンツ、又はユーザエージェントにストリーム配信されるコンテンツでは、コンテンツ再生の一時停止と再開の操作の間に生成、又は受信される情報を保持、又は提示する必要はない。これは技術的に不可能であることが考えられ、多くの状況において利用者の混乱を招くことにつながる可能性があるためである。

コンテンツの読み込み中やそれに類似した状況の一部として表示されるアニメーションについては、この段階ですべての利用者に対していかなるインタラクションも発生する可能性がなく、かつコンテンツ読み込みの進行状況を表示しないことが利用者の混乱を招いたり、コンテンツが動作を停止した、又はコンテンツが破損しているという誤解を生じたりする可能性がある場合には、必要不可欠なものと考えることができる。

【ポイント】

- 自動で動く、スクロールする、または点滅するコンテンツが 5 秒以上続く場合、ユーザーが一時停止・停止・再開できる手段を提供する。
- 重要な情報を表示するコンテンツは、自動的に消えたり切り替わったりしないよう配慮する。
- 動きを止められるコントロールは常に見つけやすく、キーボードでも操作可能であること。

【解説】

ニュースティッカーやカルーセル、広告バナーなど、自動的に動き続けるコンテンツは視覚的な注意を奪い、情報の理解を妨げる可能性があります。

特に注意欠陥障害（ADHD）のある利用者や読み書きに時間がかかる人にとっては、自動で変化するコンテンツは理解の妨げや混乱の原因となります。

そのため、動きや変化をユーザーがコントロールできるようにすることが重要です。

また、キーボード操作やスクリーンリーダーでも確実に一時停止や停止ができることが求められます。

【対応方法】

- 自動で動くコンテンツ（例：カルーセル、ニュースティッカー、スライドショーなど）が 5 秒以上続く場合は、一時停止・再開・停止ボタンを必ず設置する。
- コントロールボタンは、見つけやすく、視覚的に目立ち、キーボードでフォーカス可能であることを確認する。

- 自動で消えるコンテンツ（例：エラーメッセージや通知など）は、利用者が読む前に消えないよう十分な表示時間を確保するか、ユーザーが閉じるタイミングを選べるようにする。
- 点滅するコンテンツは、点滅を最小限にし、点滅を停止できる設定を提供する。

【不適切な例】

- カルーセルが 3 秒ごとに次のスライドへ切り替わるが、一時停止ボタンが存在しない。
- スクロールするニュースティッカーが常に流れ続け、マウス操作なしでは止められない。
- エラーメッセージが 5 秒で自動的に消えてしまい、読み終わる前に情報が失われる。

【適切な例】

- カルーセルに「◀」「▶」「一時停止」ボタンが常時表示され、キーボード操作でも利用できる。
- 通知バナーは自動的に消えず、「閉じる」ボタンをユーザーが押すまで表示される。
- 点滅する広告バナーに「動きを停止する」ボタンがある。

【補足】

- この基準は音声ではなく視覚的な動きや変化が対象です。音声に関しては 1.4.2「音声の制御」を参照します。
- 動きを制御するコントロールは、見た目のデザインよりもアクセスのしやすさと一貫した動作を優先します。
- 自動再生のカルーセルは利用者に不意の混乱を与えることが多いため、初期状態では自動再生を無効にするのが望ましいです。

2.3.1 3 回の閃光(せんこう)、又は閾値(いきち)以下 レベル A

ウェブページには、どの 1 秒間においても 3 回を超える閃光を放つものがない、又は閃光が一般閃光閾値及び赤色閃光閾値を下回っている。

この達成基準を満たさないコンテンツでは、利用者がそのウェブページ全体を使用できない恐れがあるため、ウェブページ上のすべてのコンテンツは他の達成基準を満たすために用いられているか否かにかかわらず、この達成基準を満たさなければならない。適合要件 5: 非干渉を参照。

【ポイント】

- 閃光や点滅を起こすコンテンツについては極力使用しない。
- 使用する場合は基準値を下回っているか確実に検証を行ってから使用する。

【解説】

閃光を放つコンテンツは、けいれんなどの発作を引き起こす可能性があります。（光過敏性てんかん）そのため必要がない限り、閃光を放つコンテンツを含めないようにしてください。例えば、記者会見の場などにおいて、多数のフラッシュがたかれるようなシーンなどには配慮が必要です。

閃光を放つコンテンツを制作する場合は、1 秒間に 3 回を超えないように制作してください。または、一般閃光閾値及び赤色閃光閾値を超えないように制作します。一般閃光閾値及び赤色閃光閾値を超えないように制作するには、下記の 2 つの方法のいずれかを用います。

【対応方法】

- 閃光を放つエリアを十分に小さく（視野 10 度の 25%未満）する。
  一般的なディスプレイ解像度 1024×768 ピクセルの場合、視野 10 度はおおよそ 341×256 ピクセルである。視野 10 度の 25%である、21,824 平方ピクセルに収まるように制作する。視野 10 度（341×256 ピクセル）以内で同時に閃光を放っているコンテンツが複数ある場合、それらのコンテンツの領域を合計して計算する。
- 下記にあるツールを用いて、一般閃光閾値及び赤色閃光閾値を確認し、調整する。
  Photosensitive Epilepsy Analysis Tool http://trace.umd.edu/peat/

閃光を放つコンテンツは動画だけではなく、アニメ GIF も対象となる。

【不適切な例】

- 速い間隔で点滅する背景光を含む動画。
- カメラフラッシュが頻繁に起きる記者会見のシーンを含む動画。
- テキストや画像が急速に点滅するバナー広告。

【適切な例】

- 点滅や閃光を含まない平滑なトランジションを使用した動画。
- 低コントラストとソフトトランジションを利用した背景アニメーション。
- 静止画像を使用したバナー広告。

2.4.1 ブロックスキップ レベル A

複数のウェブページ上で繰り返されているコンテンツのブロックをスキップするメカニズムが利用できる。

【ポイント】

- 各ページで繰り返し表示されるブロック部分には、スキップできるリンクを設置してアクセスを容易にする。
- ランドマークの付与（main 要素, nav 要素, aside 要素, header 要素, footer 要素等）により、支援技術の領域移動を容易にする。

【解説】

視覚的に見ながら操作することが困難なスクリーンリーダー利用者や四肢麻痺等によりマウス使用が困難でキーボードで操作する利用者に配慮し、サイト共通のナビゲーションや反復されるコンテンツブロックを素早くスキップできる機能を提供します。利用者はキーボードなどの操作を大きく減らすことができ、精神的・肉体的負担を減らすことができます。

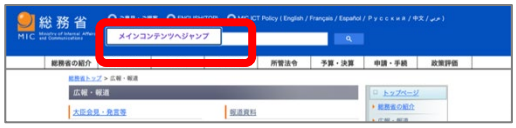
【対応方法】

- ページ最上部に「本文へスキップ」機能を持つリンクを設置し、共通コンテンツのブロックをスキップできるようにする。
- リンクは通常は非表示だが、フォーカス時には視覚的に表示する。
- 本文セクションの開始部分に見出し要素を利用し、支援技術を用いてスキップを容易に実行できるよう配慮する。
- スキップ先（例：<main id="main">）にはフォーカス可能化（tabindex="-1" など）を施し、リンク選択時に実フォーカスが移動するようにする。
- ランドマークを併用し、主要ブロックを示す（<main>, <nav aria-label="主要ナビゲーション"> など）。
- 必要に応じて JavaScript を用いてフォーカス移動を補強する
（例：document.querySelector('#main').focus() を実行）。
- 視覚的表示は CSS で実装し、フォーカス時に :focus-visible で可視化する。

ブロックスキップの例（総務省サイト）

- 初期状態では特に何も表示されない。

- Tab キーを数回押すと、メインコンテンツにスキップするためのリンクが現れる。



- このリンクを選択すると、フォーカスはすぐに本文部分へと移動する。

【対応例】

主内容へ即移動できる手段を設けているコード例。

```
[html]
 <a class="skip" href="#main">本文へスキップ</a>
 <main id="main" tabindex="-1">…</main>
[css]
 .skip{position:absolute;left:-9999px}.skip:focus{left:0;top:0;padding:.5em}
```

2.4.2 ページタイトル レベル A

ウェブページには、主題又は目的を説明したタイトルがある。

【ポイント】

- ページタイトルは、そのページを明確に特定できるようにする。
- 他のページと重複させない。

【解説】

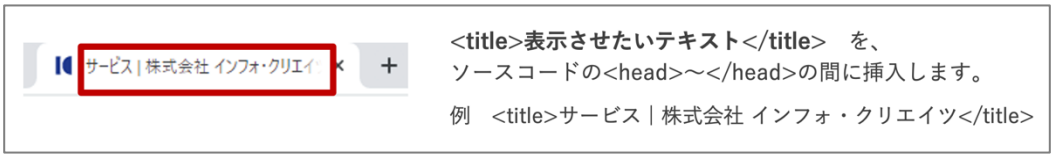
タイトルは title 要素に設定されます。スクリーンリーダーの利用者がウェブページにアクセスした際には、ページタイトルが最初に読み上げられる情報です。これにより、利用者はページの内容を閲覧するかどうかを決定します。また、多くの検索エンジンでは「ページタイトル」のテキスト情報が高い優先度を持ち、検索結果の一覧に表示されます。

したがって、ページタイトルはウェブページの内容を的確に伝える重要な要素となります。そのため、理解しやすく、明確な内容のタイトルを設定することが必須です。また、利用者に混乱をもたらす可能性があるため、複数のページに同じタイトルを設定することは避けてください。

【対応方法】

- 「タイトル」は、利用者がページ内容を予測できるようにその内容を適切に要約する。
- 定期的に更新されるコンテンツ（例: 月間記事）は、その発行年月をタイトルに含める。

例.



【不適切な例】

- 月ごとに発行される報告書が掲載されるページで、全てのページのタイトルが「報告書」となっている。

【適切な例】

- 月ごとの発行に合わせて、「報告書 - 2020 年 8 月」のように各ページに異なるタイトルが設定されている。

【参考情報】

- SEO の観点からも、各ページの内容を端的に示し、サイト内で重複しない固有のページタイトルを設定することで、検索意図との一致度が高まり、検索結果での掲載順位の向上が期待できます。
- タイトルは全角 25～30 字を目安に、重要語を先頭に置き、冒頭 10～15 字で内容が伝わるようにします。長すぎるタイトルは検索結果で省略され、理解の低下や SEO 上の不利につながるため避けます。

2.4.3 フォーカス順序 レベル A

ウェブページにおいて、フォーカスの移動順序が意味又は操作に影響を及ぼす場合、フォーカス可能なコンポーネントは、意味及び操作性を損なわない順序でフォーカスを受け取る。

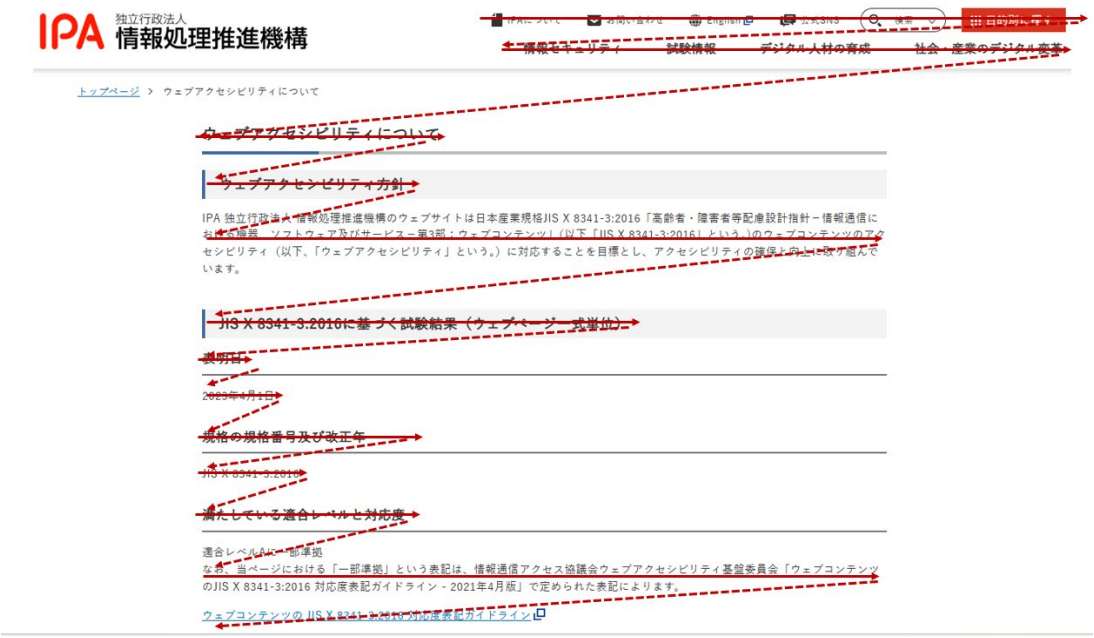
【ポイント】

- フォーカスの順序が視覚的なコンテンツの配列と一致していることを確認する。

【解説】

キーボード操作時、フォーカスの順序はページの視覚的な配列と一致していることが重要です。この一致が取れていない場合、キーボードを使用している利用者はフォーカスを見失う恐れや混乱を感じる可能性があります。視覚的な配列と異なるフォーカス順序は理解しづらく、利用者が目的のコンテンツにアクセスし操作するのを困難にします。

一般に、キーボードのフォーカス移動は DOM（ソース）順に従います。CSS だけで視覚順を入れ替える（例：flex の order／grid の並べ替え／position による視覚移動）と、視覚順と DOM 順がずれて混乱を招くため、可能な限り視覚順と DOM 順を一致させてください（tabindex の正の値等で順序を操作するのは原則避ける）。



日本語のコンテンツでは、左から右、または上から下への順序でフォーカスが移動するのが一般的とされています。キーボード操作を使用してウェブページ内を移動する際、その移動順序がウェブページの視覚的なレイアウトと一致していることを確認してください。

【対応方法】

- CSS を使用してコンテンツのレイアウトを指定する際、視覚的な配列とフォーカス順序が一致するように配慮する。
- tabindex は 0 または省略を基本とし、正の値は原則使用しない。

【不適切な例】

- キーボードでのタブ移動がランダムな順序で行われ、視覚的なレイアウトと明らかな不一致がある。
- ページ右下の「トップへ戻る」ボタンを Enter キーで実行すると先頭までスムーズスクロールするが、フォーカスは元の要素に残ったままになっている。続けて Tab キーを押すと、表示中の先頭ではなく画面外（スクロール前の位置）の要素へフォーカスが移動し、表示位置とフォーカス位置がずれる。

【適切な例】

- ウェブページのコンテンツが視覚的に上から下へと流れる順序で整理され、キーボードのタブ移動もこの流れに沿った一貫した順序で行える。
- ページ右下の「トップへ戻る」ボタンを Enter キーで実行すると先頭までスクロールし、スクリプトで先頭アンカー（例：#top の見出しや <main>）へフォーカスを移動した。続けて Tab キーを押してもページ先頭から順に移動でき、表示位置とフォーカス位置が一致している。

2.4.4 リンクの目的(コンテキスト内) レベル A

それぞれのリンクの目的が、リンクのテキスト単独で、又はリンクのテキストとプログラムによる解釈が可能なリンクのコンテキストから判断できる。ただし、リンクの目的がほとんどの利用者にとって曖昧な場合は除く。

リンクテキスト

【ポイント】

- リンクテキストは、リンクテキストだけでリンク先やリンクの目的がわかるようにする。

【解説】

リンクテキストは、利用者がリンクを選択するかどうかを判断する際に役立つテキストであるべきです。スクリーンリーダーではリンクテキスト部分のみを拾い読みする機能があるため、リンク先を適切に表現したリンクテキストを設定することが大切です。リンク先がリンクテキストと異なる内容だと、利用者は混乱し、戻るしか選択肢がありません。また、「詳細はこちら」という同じリンクテキストが並ぶ場合、利用者はそれぞれのリンクが何の詳細かを判別するのが難しくなります。

なお、「リンクの目的がほとんどの利用者にとって曖昧な場合は除く」という例外規定は、「曖昧なテキストを使用してよい」ということではなく、障害の有無に関わらず、どんな利用者にも、そのリンクを実際にクリックして動作させるまで、リンクの目的がわからない状態である場合は対象外にするというもので、きわめて限定的なケースです。例えば、オンラインゲームサイトやエンターテイメントサイトなどのように驚きの要素がコンテンツの本質的な価値となっている場合や、ランダム選択機能（今日のおすすめレシピ、ランダム記事表示等）を持ち、リンク先の内容が意図的に隠されることがコンテンツの体験の一部となっているウェブサイトなどの場合に、この例外規定が適用されることがあります。

公的機関のウェブサイトにおいて、その目的はサービスや情報を正確かつ公平に伝えることであり、すべての利用者が平等にアクセスできることが求められています。そのため、リンクの目的が「ほとんどの人にとって曖昧」という状況は、情報の透明性や公平性の観点から、通常あってはならないため、この例外規定を適用できるケースは実質的に存在しないと考えるべきです。

【対応方法】

- リンクテキストはリンク先ページのタイトルやリンク先の内容を想像できる内容を含める。
- 「詳細はこちら」のようなリンクテキストを使用する場合は、リンクの直前に何の詳細なのかがわかるよう記述する。
  ただし、当対応基準の方針として、以下の理由により、リンクの目的が文脈によって推測できるかどうかに関わらず、リンクテキスト単体で内容（またはそこで行われる操作）が明確にわからないものは不適合とします。
  必ずリンクテキストのみで、遷移先やそこで行われる操作がわかる記述にしてください。
  - 「文脈によって推測できるか」の判断基準が制作者によって異なる（個人差がある）ため、品質統制が困難になる
  - スクリーンリーダー利用者にとって、文脈をたどりながらリンクの意味を判断する行為は負担が大きい
  - スクリーンリーダーには「リンク一覧」など、ページ内のリンクだけを抜き出して読み上げる機能（文脈は読み上げない）があり、リンクテキスト単体で目的がわからない場合に内容を把握できない

【不適切な例】

- 何をダウンロードするのかわからない。

```
<p>〇〇資料（PDF 形式 2MB）<a href="/download/xxxx.pdf">資料ダウンロード</a></p>
どこへ行くのかわからないお問い合わせは専用フォームからお願いします。<a href="/form.html">こちら</a>
```

【適切な例】

- リンクだけ読み上げていると「詳細」「詳細」「詳細」としか聞こえない箇所がある。その一つで、直前のテキストを確認した。「特選商品についての」としか聞こえない。何の商品の詳細かは分からない。
- 「ニュースランキング」というリンクテキストをクリックしたところ、そのリンク先ページのタイトルが読み上げられ、リンクテキストと内容が一致していることがすぐに分かった。
- 対象名＋動作で単独でも意味が分かるリンク。

```
<a href="/products/a">A 商品の詳細</a>
```

- アイコンのみのリンクに目的のラベルを付与。

```
<a href="/cart" aria-label="カートを見る">
  <img src="cart.svg" alt="">
</a>
```

- 記事一覧で「続きを読む」を使う場合、記事名を含めて識別可能にします。

```
<article>
  <h3><a href="/news/123">A 社が新製品を発表</a></h3>
  <p>…</p>
  <a href="/news/123" aria-label="A 社が新製品を発表 を読む">続きを読む</a>
</article>
```

PDF、Word、Excel ファイル、別ドメインのページへのリンク

【ポイント】

- リンク先が PDF、Word、Excel ファイルや別ドメインのページの場合、リンクテキストの末尾にそれぞれの種類を示すシンボルやファイル種別を明示する。
- 追加されたシンボルやテキストはスクリーンリーダーで読み上げることが可能である。

【解説】

リンク先が PDF、Word、Excel ファイルや別ウィンドウで開くページの場合、利用者へその情報を明示してください。そうしないと、利用者は意図せず開かれるアプリケーションや増加するブラウザタブにより、元のページへのスムーズな復帰が困難になることがあります。

【対応方法】

- リンク先が PDF などのファイルや別サイトである場合、リンクテキストには別のアプリケーションやブラウザを開くことを示す追加情報を提供して伝える。
- target="_blank"で別ウィンドウを開く際には、その情報がスクリーンリーダーによって読み取られるようにする。

【不適切な例】

- CSS により PDF であることを示すアイコンが表示されているが、スクリーンリーダー利用者への配慮がなされていない。スクリーンリーダーを利用しても PDF であることを識別することができなかった。

【適切な例】

- PDF であることを示すアイコンが CSS によって表示されている。そのファイルを開くテキストリンクには"（PDF:42KB）"と PDF ファイルであることが追記されている。
- target="_blank"で別ウィンドウを開くリンクがあり、リンクテキストの後ろには、別ウィンドウを開くことを示す小さなアイコン画像が配置されている。また、そのアイコンの画像の代替テキストには alt="別ウィンドウで開く" とある。

- 別ウィンドウで開くことは CSS の疑似画像を表示して知らせている。かつ、aria-label を用いて別ウィンドウで開くことが伝えている。
- PDF で開くことは Web フォントを利用して表示している。スクリーンリーダーの利用者にもその事がわかるように配慮されている。（この方法での実装は必ずスクリーンリーダーでの読み上げ確認テストが必要）

```
<a href="polarstar.html" target="_blank" class="newwindow" aria-label="
北極星の説明 別ウィンドウで開く">北極星の説明</a>
```

```
<a href="polarstar-book.html" target="_blank">北極星パンフ
<i class="pdf-file-icon" aria-label="PDF"></i></a>
```

2.4.5 複数の手段 レベル AA

ウェブページ一式の中で、あるウェブページを見つける複数の手段が利用できる。ただし、ウェブページが一連のプロセスの中の 1 ステップ又は結果である場合は除く。

【ポイント】

- 利用者がページを探しやすいように、2 つ以上の探索手段を提供する。

【解説】

この達成基準は、同一のウェブページ一式において、各ページへ到達するための少なくとも 2 種類の探索手段を提供することを求めます（ただし、そのページが一連のプロセスの途中または結果である場合は対象外とします）。評価の観点は「使いやすさの主観」ではなく、手段が実際に 2 種類以上提供されているかという客観条件です。探索手段の例としては、サイト内検索、サイトマップ、目次、関連ページへのナビゲーション、全ページ一覧、トップページから全ページへのリンクなどが挙げられます。利用者は支援技術の利用状況や認知特性、操作環境が多様であるため、複数の経路を用意することで到達可能性と予測可能性を高めます。

【対応方法】

- サイト探索手段として、次のうち 2 つ以上を採用する: （プロセス中のページは除外）
  - サイトマップを提供する場合は、全てのページからリンクを設置して利用可能にする。
  - 関連するページへ移動できる導線（リンクやボタン）を提供する。
  - 検索機能を提供する。
  - すべてのページへのリンク一覧を提供する。
  - トップページからサイト上すべてのページへのリンクを提供する。
  - 目次を提供する。

※ 「すべてのページへのリンク一覧を提供する」は WCAG Techniques G126 に相当します。

※ 「トップページからサイト上すべてのページへのリンクを提供する」は G185 に相当し、小規模サイトなどではサイトマップに代わる探索手段として機能します。

※ 「目次」とは、ページまたは文書内の見出しや章タイトルを一覧し、各項目から対応する箇所へ移動できるリンクが付与された一覧を指します。

【不適切な例】

- サイト内にトップページナビゲーションしかなく、検索やサイトマップがないため、深い階層のページにたどり着けない。
- サイトマップはあるが、最新ページや一部の下層ページへのリンクが欠落しており、結果として探索手段が 1 つしか機能していない。
- 申込み完了画面などのプロセス中でない一般ページにも、探索手段が 1 つしか提供されていない。

【適切な例】

- トップページにグローバルナビゲーションがあり、さらにサイトマップも用意されているため、どちらの手段でも目的のページへ到達できる。
- 各ページのヘッダーに検索ボックスが常時配置され、併せてサイトフッターに全ページリンク一覧が用意されている。
- 製品一覧ページに目次リンクとサイト内検索の両方があり、ユーザーが好みに応じていずれの方法でも目的の製品ページを開ける。

2.4.6 見出し及びラベル レベル AA

見出し及びラベルは、主題又は目的を説明している。

【ポイント】

- 見出しは、そのセクションの主題や内容をわかりやすく示す。
- ラベルは、ボタン・リンク・フォームなどの目的を正確に伝える。

【解説】

この達成基準は、ページの各セクションやフォーム、リンクなどに付けられた**見出しやラベルの文言が、その先の内容や目的を明確に伝えること**を求めています。例えば「お問い合わせフォーム」や「検索結果一覧」のように、見出しを見ただけでその内容がわかることが重要です。曖昧な言葉（例：「こちら」「その他」「詳しくはこちら」）ではなく、**利用者が目的を予測できる具体的な表現**を用いることが必要です。また、同じ機能を提供するラベルはサイト全体で一貫した名前を用いることで、利用者が混乱せずに操作できます。

【対応方法】

- 見出しやラベルは、その先の内容・機能・目的が一目でわかる語句を用いる。
- 曖昧な表現（「こちら」「その他」「詳しくはこちら」など）を避け、**利用者が予測できる具体的な表現**にする。
- サイト内で同じ機能や目的を持つコンポーネントには、**同じラベル・用語を一貫して使用する。**
- アイコンや画像ボタンには、機能を示す代替テキスト（aria-label 等）を付与する。
- フォーム入力欄には、**何を入力すべきかが明確にわかるラベル**をつける。

【不適切な例】

- サイトのあるページでは検索ボタンが「検索」、別のページでは「さがす」と表示されており、同じ機能なのに表記が統一されていない。
- 「その他」「詳しくはこちら」など、内容が予測できない見出しやリンクテキストを使用している。
- お問い合わせフォームの入力欄にラベルがなく、何を入力すべきかわからない。
- 「ゴミ箱」アイコンに代替テキストがなく、支援技術利用者には削除機能だと伝わらない。

【適切な例】

- 検索ボタンは全ページで「検索」に統一されている。
- 見出しに「お知らせ一覧」「利用規約」「製品仕様」など、内容を明確に表す語句を用いている。
- 問い合わせフォームの各入力欄に「氏名」「メールアドレス」「お問い合わせ内容」など明確なラベルが付いている。
- 「ゴミ箱」アイコンには aria-label="削除"が設定されており、支援技術利用者にも機能が伝わる。

2.4.7 フォーカスの可視化 レベル AA

キーボード操作が可能なあらゆるユーザインタフェースには、フォーカスインジケータが見える操作モードがある。

【ポイント】

- キーボード操作時にどこにフォーカスがあるか視覚的に明確にする。

【解説】

マウスで操作する場合は、利用者が直接目で見て操作したいコンポーネントを選択できます。しかし、キーボードで操作する場合は、視覚的な表示がないとフォーカスがどこに移動したかを特定することが困難となります。そのため、フォーカスがどこにあるかを視覚的に示す工夫が必要です。

【対応方法】

CSS を利用して以下のような視覚的変化を起こさせてください。

- コンポーネントをボーダーで囲む。

- 入力域をハイライトする。
- 選択箇所のボタン色を反転させる。

【不適切な例】

- フォーカスが移動しているにもかかわらず、何の視覚的変化もなく、利用者がどこにフォーカスがあるかを識別できない。
- CSS にて:focus{outline:0} のように設定して、フォーカスを非表示にしている。

【適切な例】

- キーボードでタブを移動するたびに、現在のフォーカス位置が明確なボーダー或いは背景色の変更で明示され、利用者が容易に識別できる。
- 2 ドットのフォーカスが表示される。

【対応例】

- キーボードフォーカスを十分な太さとコントラストで明示しているコード例。

```
[css]
  :focus-visible{outline:3px solid;outline-offset:2px}
```

2.4.11 隠されないフォーカス（最低限）レベル AA

ユーザインタフェース コンポーネントがキーボードフォーカスを受け取るとき、コンテンツ制作者が作成したコンテンツによって、そのコンポーネントの全体が隠されるようなことがない。

設定可能なインタフェースの中にあるコンテンツが利用者によって再配置可能な場合、この達成基準のテスト及び適合性の対象として検討されるのは、利用者によって移動可能なコンテンツの初期位置だけである。
利用者が開いたコンテンツによって、フォーカスを受け取るコンポーネントが隠される場合がある。利用者がキーボードフォーカスを移動せずにフォーカスを持つコンポーネントを表示できる場合、そのフォーカスを持つコンポーネントは、コンテンツ制作者が作成したコンテンツによって隠されたとはみなされない。

【ポイント】

- 利用者がキーボードでフォーカスを移動しているときも、フォーカス中の項目が常に見分けられる（少なくともフォーカス指標の一部が画面に残る）。
- ポップアップ／固定ヘッダー／バナーなど他のコンテンツが重なったときも、フォーカス中の項目が隠れないで見分けられる（少なくともフォーカス指標の一部が画面に残る）。

【解説】

キーボード利用者や視覚障害のある利用者はフォーカスの位置を視覚的に把握してサイトを操作します。しかし、ページ上のポップアップやモーダルウィンドウ、例えばクッキーの同意を求めるバナーが表示されることで、本来のフォーカスが覆われたり、背景の項目にフォーカスが移ってしまったりすると、利用者は次に進むべき場所を見失ってしまいます。そのため、フォーカスが常に一部でもよいので見えるていることが求められます。

【対応方法】

- フォーカスされた項目には、明瞭な視覚的インジケータ（例: 枠線や色の変更）を提供する。
- ページ上のどの項目にフォーカスがあっても、常にインジケータが見えるようにする。
- クッキー同意バナーなどの項目が表示されている間でも、フォーカスが隠れないよう設計する。
- JavaScript を使用してフォーカス管理を行い、ポップアップが開いている間は背景の項目にフォーカスを設定しない。
- （参考）フォーカスインジケーターなどのグラフィカルな指示は、1.4.11 に基づき 隣接色に対して少なくとも 3:1 のコントラストを確保する（WCAG 2.2 を適用する場合は 2.4.13 の要件も満たす）。

【不適切な例】

- 利用者がサイトを訪れた際に表示されるクッキー同意ポップアップが、キーボードフォーカスを取得し、それを閉じるまで他の項目にフォーカスを移せない。
- フォーカスがある項目が、クッキーの同意ポップアップによって視覚的に隠され、利用者が次のアクションを取るのが難しい。
- ページ上部の固定ヘッダーが高さを持っており、フォーカス移動時に見出しやフォームの入力欄がヘッダーの下に隠れてしまい、利用者がフォーカス位置を確認できない。

【適切な例】

- クッキーの同意を求めるポップアップが画面下部に表示されていても、フォーカス中の項目が明瞭にハイライト表示され、少なくとも指標の一部が見えてどの項目か見分けられる。
- ポップアップが開いている時は、キーボードフォーカスがポップアップ内の項目に限定され、背景のコンテンツはフォーカスできないようになっている。

- 長いフォームで入力欄へタブ移動すると、スクロールが自動で調整され、フォーカス中の入力欄が常に画面内に収まり、固定ヘッダーに隠れないように表示される。

【対応例】

固定ヘッダーに隠れないよう、フォーカス移動時のスクロール位置を余白込みで確保するコード例。

[html]

```
<header class="hdr">…</header> <main> <a href="#s1">見出しへ</a> <h2 id="s1" tabindex="-1">見出し</h2>
</main>
```

[css]

```
.hdr{position:sticky;top:0;height:4rem}
html{scroll-padding-top:4rem}
:focus-visible{outline:2px solid;outline-offset:2px}
```

2.5.1 ポインタのジェスチャ レベル A

マルチポイント又は軌跡ベースのジェスチャを使って操作する機能はすべて、軌跡ベースのジェスチャなしのシングルポインタで操作することができる。ただし、マルチポイント又は軌跡ベースのジェスチャが必要不可欠である場合は例外とする。

この要件は、ポインタの動作を解釈するウェブコンテンツに適用される（ユーザエージェントや支援技術の操作に必要なアクションには適用されない）。

【ポイント】

- すべての操作は、複雑なジェスチャではなく単純なタップやクリックで行える。
- 複数の指を使うジェスチャではなく、一本指でのタップやロングプレスで操作可能である。

【解説】

モバイルデバイス上のインタラクティブな要素は、単一のタップやクリックで利用が可能でなければなりません。複雑なジェスチャ（例えば、ピンチズームやスワイプ）は一部の利用者にとって使いづらい場合があります。特に、運動障害がある利用者や高齢の利用者、片手しか使えない利用者にとっては、単純なジェスチャの方が操作は容易です。また、単純なジェスチャは、利用者が意図しない操作を引き起こす可能性を減らします。

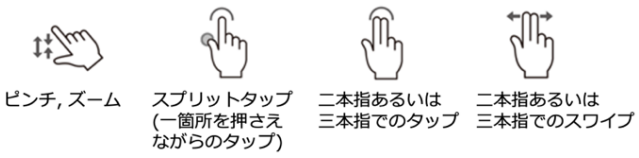
【対応方法】

- ウェブサイトやアプリケーションの全ての機能は、単純なタップやクリックだけで操作できるようにする。
- 複雑なジェスチャが必要な場合、それと同等の機能を単純な操作で提供するオプションを用意する。
- インタラクティブな要素は、明確なフィードバックを提供することで、操作が成功したことを利用者に示す。

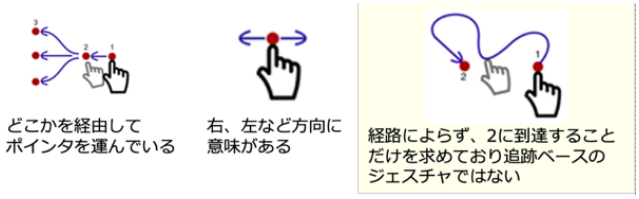
【不適切な例】

- スライドショーのスライドを選択するのにスワイプ操作しか選べない。
- スクロールバーはスワイプでは操作できるが、任意のポイントをクリックしての移動はサポートされていない。

（マルチポイントの例）



- 追跡ベースのジェスチャの例（最後の経路に依らない場合は問題無い）



【適切な例】

- 画像を拡大するためにピンチズームを使う代わりに、ダブルタップや専用のズームボタンを提供している。
- 音量操作パネルにはスライダーだけでなく[+]と[-]のボタンがあり、そのボタンの操作でも音量は調整できる。
- シングルポイントの例　（マウスでは、シングルクリック、クリック&ホールド、ダブルクリックなどが相当する）

2.5.2 ポインタのキャンセル レベル A

シングルポインタを使って操作できる機能は、以下の要件の少なくとも 1 つを満たす。

ダウンイベントがない

機能を実行する目的でポインタのダウンイベントを使用していない。

中止又は元に戻すことができる

機能の完了にはアップイベントを使用し、かつ機能の完了前に中止する、又は機能の完了後に元に戻すためのメカニズムが利用できる。

アップイベントで反転

アップイベントによって、先のダウンイベントのすべての結果が反転する。

必要不可欠

ダウンイベントによって機能を完了させることが必要不可欠である。

キーボード又はテンキーパッドのキープレスをエミュレートする機能は必要不可欠とみなされる。
この要件は、ポインタの動作を解釈するウェブコンテンツに適用される（ユーザエージェントや支援技術の操作に必要なアクションには適用されない）。

【ポイント】

- 利用者が誤って操作した場合、容易にキャンセルできるメカニズムを提供する。
- 操作が意図的であることを確認するために、追加の確認手順を設ける。

【解説】

モバイルデバイスやタッチスクリーンデバイスでは、利用者が誤ってタッチやクリックをすることがあります。このような誤操作が重要な機能を引き起こしてしまうと、利用者にストレスを与えることになります。誤操作を容易にキャンセルできるようにすることで、利用者が安心して操作できるようになります。また、意図しない操作による誤った結果を防ぐために、確認手順を設けることが重要です。

【対応方法】

- タップやクリックの操作が終了するまで機能が起動しないようにする（例：「タッチの開始」ではなく「タッチの終了」でイベントをトリガーする）。
- 操作をキャンセルするオプションを提供し、利用者が誤操作を修正できるようにする。
- 重要な操作（例：注文の確定、アカウントの削除）には確認ダイアログを表示し、利用者が操作を確認できるようにする。

【不適切な例】

- 「削除」ボタンを誤ってタップした瞬間に削除処理が確定し、指を離す前にキャンセルする手段がない。
- 画像の「いいね」ボタンをタップしただけで即時投稿が確定し、指をスライドしてボタン外に逃がして（キャンセルして）も取り消せない。
- pointerdown イベントでメニュー遷移が発生し、誤って押した場合でも戻る／キャンセルができない。

【適切な例】

- 商品をカートに入れた後、誤って「追加」をタップしても、小さい「×」マークをタップするとすぐに取り消せる。

- ボタンのタッチ操作においては、利用者がボタンをタップ（押し込む動作）しても、直ちにアクションが実行されるのではなく、タップしたまま指をスライドさせてボタンから外に移動し、その状態でリリース（指を離す動作）することでアクションをキャンセルすることができる。

【対応例】

- 押下ではなく「離したとき」に実行し、途中キャンセルも許可するコード例。

```
[html]
  <button id="buy">購入</button>
[js]
  let down=false;buy.onpointerdown=()=>down=true;addEventListener('pointerup',e=>{if(down&&e.target===buy)/*
実行 */;down=false});addEventListener('pointercancel',()=>down=false)
```

※標準のボタンやフォームコントロールは、ブラウザが既に「押下した瞬間ではなく、離したときに動作する」「Esc やキャンセルを認識する」といった仕組みを持っているため、特別な対応は不要です。一方で、JavaScript による独自 UI やカスタムコンポーネントを作成する場合は注意が必要です。pointerdown と pointerup を分離し、押した瞬間に確定してしまわないようにし、ユーザーが途中で操作をキャンセルできるように実装することが求められます。これは、その例になります。

2.5.3 ラベルを含む名前 レベル A

達成基準 2.5.3 ラベルを含む名前（name）(レベル A): ユーザインタフェース コンポーネントがテキスト又は文字画像を含むラベルを持つ場合、視覚的に提示されたテキストが名前（name）に含まれている。

ベストプラクティスは、ラベルのテキストを名前の最初に使用することである。

【ポイント】

- インタラクティブな要素には、その機能と目的を示す明確なラベルが提供される。
- ラベルは、支援技術によって識別可能であり、理解しやすい。

【解説】

利用者がインタラクティブな要素（ボタン、リンク、フォームフィールドなど）の目的を正確に理解できるように、それぞれの要素には明確なラベルまたは説明が必要です。特に、スクリーンリーダーを使用する視覚障害のある利用者や音声コマンドシステムを使用する利用者にとって、これらのラベルは要素の操作を可能にするための重要な手がかりとなります。ラベルは、要素の機能を正確に伝えるために、視覚的なテキストだけでなく、アクセシビリティ API を通じてプログラム的にも提供されるべきです。

【対応方法】

- aria-label、aria-labelledby または label 要素を使用して、インタラクティブ要素にラベルを関連付ける。
- ラベルテキストは、要素の機能や目的を明確に伝えるものである。
- ラベルは視覚的にもプログラム的にも利用可能であり、すべての利用者にとって意味が理解しやすい。

【不適切な例】

- 「送信」ボタンが視覚的には「送信」と表示されているが、スクリーンリーダーには「ボタン 1」とだけ報告される。

【適切な例】

- ショッピンフォームの各入力フィールドに label 要素が関連付けられており、スクリーンリーダーがフィールドの目的を明確に伝える。

```
<label for="email">メールアドレス:</label>
<input type="email" id="email" name="email">
```

- アイコンボタンに aria-label 属性が使用されており、ボタンの機能がプログラム的にも明確に識別される。

【対応例】

- 可視ラベル「削除」をそのままアクセシブルネームに含めるコード例。

```
[html]
  <button aria-label="削除 アイテム A">削除</button>
```

【補足】

2.5.3 はボタンなど**操作要素全般のラベル**を扱い、3.3.2 は**フォーム入力欄のラベルや説明**に焦点を当てています。
両者ともラベルを明確にし、利用者が目的を理解しやすくする点で共通しています。

2.5.4 動きによる起動 レベル A

デバイスの動き又は利用者の動きで操作できる機能は、ユーザインタフェース コンポーネントでも操作でき、かつ偶発的な起動を防ぐために動きへの反応を無効化することができる。ただし、以下の場合は例外とする。

サポートされたインタフェース

アクセシビリティ サポーテッドなインタフェースを通じて機能を操作するために動きが用いられる。

必要不可欠

その機能にとって動きが必要不可欠であり、この達成基準に従うと動作を無効化してしまう。

【ポイント】

- デバイスの動きや利用者のジェスチャによって機能が起動する場合、同じ機能は画面上のボタンやメニューなど、タップやクリック可能な UI 要素を通じても利用可能である。
- 利用者はデバイスの動きによる操作を無効にすることができる。

【解説】

多くのモバイルアプリケーションやウェブサイトでは、デバイスを振る、傾ける、回転させるなどの動きによって特定の機能を起動することができます。これらの動きによるインタラクションは便利で直感的ですが、身体的な制約や環境的な理由から動きを利用できない利用者もいます。そのため、同等の機能がインターフェースのコントロールを通じても利用できるようにすることが重要です。また、不意の動きによる誤操作を防ぐため、利用者がこれらの機能を制御できるようにする必要もあります。

【対応方法】

- デバイスの動きによって起動する機能は、画面上のボタンやスイッチといった明確な UI コントロールによっても操作できるようにする。
- 利用者が設定メニューから動きによる操作を無効にできるようにする。
- 動きによる操作が誤って起動しないように、利用者からの確認を求める。

【不適切な例】

- 撮った写真をキャンセルするには、デバイスをシェイクするしかない。
- スマートフォンの音楽アプリでは、曲を切り替えるためにデバイスを左右にシェイクする操作が必要だったが、これ以外に曲を切り替える方法がなく、手の不自由な利用者は曲を切り替えることができない。

【適切な例】

- 利用者がナビゲーションメニューを開くためにデバイスを傾ける代わりに、画面上のメニューボタンをタップすることで同じ機能にアクセスできる地図 Web アプリ。
- 利用者が設定で振る動作による機能起動をオフにできるフィットネス系 Web アプリ。

2.5.7 ドラッグ操作 レベル AA

操作にドラッグ動作を用いる全ての機能は、ドラッグなしのシングルポインタで完遂できる。ただし、ドラッグが必要不可欠である、又はその機能がユーザエージェントによって決定され、かつコンテンツ制作者によって変更されない場合は除く。

この要件は、ポインタの動作を解釈するウェブコンテンツに適用される（すなわち、これはユーザエージェント又は支援技術の操作に必要なアクションには適用されない)。

【ポイント】

- ドラッグアンドドロップ操作に依存する機能は、キーボード操作や他の代替手段でも利用できる。

【解説】

ドラッグアンドドロップは、ファイルの整理、項目の順序付け、設定の調整など、多くのアプリケーションでよく使われる操作です。しかし、運動障害を持つ利用者や、スクリーンリーダーを利用するキーボード操作の利用者にとって、ドラッグ操作は特に困難です。このため、ドラッグ操作に依存する機能は、ドラッグの代わりにクリックやタップなどの単一操作でも実行できる手段を提供することが重要です。このような代替手段を提供することで、すべての利用者が等しくサービスを利用できるようになります。

【対応方法】

- ドラッグ操作を必要とする機能に対して、キーボードショートカットやボタン操作といった代替の操作方法を提供する。
- 支援技術を使用している利用者がドラッグ操作を行えるよう、適切な ARIA 属性や他のアクセシビリティ機能を実装する。
- ドラッグ操作に関連する要素は大きく、クリックやタップが容易であり、視覚的なフィードバックを提供する。（これだけで適合にできるわけではない）

【不適切な例】

- あるプロジェクト管理ツールでは、タスクを並べ替える機能がドラッグアンドドロップのみに対応しており、キーボード操作では項目の順序を変更できない。

【適切な例】

- ドラッグアンドドロップによる画像のアップロードをサポートしながらも、「ファイルを選択」ボタンを通じてキーボードやスクリーンリーダーで操作が可能なファイルアップロードインターフェース。
- ドラッグ操作で項目を並べ替えることができるリストに加え、並べ替えのための上下矢印ボタンが提供され、キーボード操作でも同様の操作が可能なアプリケーション。

【対応例】

- ドラッグ以外の方法（ボタン）で並べ替えさせる場合のコード例。

  HTML は各項目に↑/↓ボタンを置き、ドラッグせずクリックで並べ替えを行う入口を用意している。
  JS は押されたボタンの data-move 値に応じて当該 li を前後へ DOM 移動し、リスト順を変更する。

[html]

```
<ul id="reorder"> <li>項目  <button data-move="-1">↑</button><button data-move="1">↓</button></li> </ul>
```

[js]

```
addEventListener('click',e=>{const b=e.target.closest('button[data-move]');if(!b)return;const
li=b.closest('li'),d=+b.dataset.move;d<0&&li.previousElementSibling&&li.before(li);d>0&&li.nextElementSibling&&li.aft
er(li.nextElementSibling);});
```

2.5.8 ターゲットサイズ（最低限）（レベル AA）

ポインタ入力のターゲットのサイズは、少なくとも 24 × 24 CSS ピクセルである。ただし、次の場合は除く:

- **間隔:** 複数の小さなターゲット（24 × 24 CSS ピクセル未満のもの）が配置されており、それぞれの境界ボックスの中心に直径 24 CSS ピクセルの円があるとした場合に、その円が別のターゲットと重なる、又は別の小さなターゲットの円と重なることがない。
- **同等:** その機能は、この達成基準を満たす同一ページ上の別のコントロールを通じて達成できる。
- **インライン:** ターゲットが文中に存在する、又は、そのサイズがターゲット以外のテキストに対する行の高さによって制約されている。
- **ユーザエージェントのコントロール:** ターゲットのサイズがユーザエージェントによって決定され、かつコンテンツ制作者によって変更されていない。
- **必要不可欠:** そのターゲットを特定の方法で 提示することが、必要不可欠である、又は伝達される情報に対して法的に要求されている。

ターゲット内の位置に基づいて空間的に値を選択できるターゲットは、この達成基準の目的上、一つのターゲットとみなされる。例としては、スライダー、色のグラデーションを表示するカラーピッカー、カーソルを置いた位置を編集できる領域などがある。

インラインのターゲットの場合、行の高さはテキストのフローに対して垂直方向として解釈されるべきである。例えば、縦書きで表示される言語では、行の高さは水平方向となる。

【ポイント】

- タッチターゲット（ボタン、リンクなどの操作対象）は十分な大きさを持ち、タッチ操作がしやすい。
- ターゲットの間隔は十分に確保されており、誤タップのリスクを減らす。

【解説】

タッチスクリーンデバイスでは、利用者がタッチターゲットを正確にタップするために、それぞれのターゲットは十分な大きさと適切な間隔を持つ必要があります。小さすぎるターゲットやターゲット同士が非常に近い配置にあると、視覚障害のある利用者や手の不自由な利用者、高齢者などが正確に操作することが困難になります。十分なサイズのターゲットを提供することで、すべての利用者が操作を容易に行えるようにし、誤操作のリスクを減らします。

【対応方法】

- タッチターゲットの最小サイズを確保する（例：WCAG 2.5.8 では原則 24×24 CSS ピクセル以上（あるいは間隔で満たす））。
- ターゲット同士の間隔を十分に確保し、利用者が意図しないターゲットを誤ってタップするリスクを減らす。
- ターゲットサイズと間隔については、デバイスの解像度や利用者の手の大きさなどを考慮して、適切な設計を行う。

【不適切な例】

- リンクやボタンが非常に小さく、視覚障害や運動障害のある利用者が正確にタップすることが困難である。

【適切な例】

- タッチターゲットが十分に大きく、間隔も適切に確保されており、利用者が容易にタップできるモバイルアプリケーション。
- ターゲット間に適切な空間を持ち、誤タップのリスクが低減されたインターフェースデザイン。

【対応例】

ターゲットを 24px 相当以上、または十分な間隔で確保しているコード例。

[css]

```
button,a[role="button"]{min-width:24px;min-height:24px;padding:.5rem}
.navbar a{margin:0 .75rem}
```

3.1.1 ページの言語 レベル A

それぞれのウェブページのデフォルトの自然言語がどの言語であるか、プログラムによる解釈が可能である。

【ポイント】

- ページ全体の言語を <html lang="言語コード"> で正確に指定する。

【解説】

ページ全体の言語を指定することは、スクリーンリーダーがページを正しく解析するうえで必須です。適切に指定されていない場合、意図しない言語で読み上げられるなど、利用者に混乱を与える原因になります。

特に支援技術は html 要素の lang 属性を利用するため、正しい言語コードを設定することが重要です。

【対応方法】

- HTML の lang 属性を使用し、ページ全体の言語を指定する。
- 言語コードは ISO 規格に準拠したものを使用する（例：日本語は lang="ja"、英語は lang="en"）

【不適切な例】

- html 要素に lang 属性を設定していない。ページの既定言語が不明となり、スクリーンリーダーが不適切な言語の音声エンジンで読み上げる。

【適切な例】

- 日本語のページでは <html lang="ja"> を指定し、支援技術が正しい読み上げを行える。

3.1.2 一部分の言語 レベル AA

コンテンツの一節、又は語句それぞれの自然言語がどの言語であるか、プログラムによる解釈が可能である。ただし、固有名詞、技術用語、言語が不明な語句、及びすぐ前後にあるテキストの言語の一部になっている単語又は語句は除く。

【ポイント】

- ページの指定言語と異なる言語を用いた箇所では、その言語を lang 属性で明示する。

【解説】

ページの主要言語と異なる言語の単語や文章が含まれる場合、その箇所を正しく読み上げたり翻訳したりするためには、該当箇所に適切な lang 属性を付けることが必要です。

ただし、固有名詞や技術用語、または日本語として広く使われている外来語（例：NEW、TOP、NEWS など）は指定不要です。

【対応方法】

- ページの主要言語と異なる部分には、その言語を lang 属性で指定する。

```
<p>フランス語で「こんにちは」は「<span lang=”fr”>Bonjour!</span>」になります。<p>
```

- 固有名詞や日本語として定着した外来語には、原則として言語属性を付与しない（省略する）。

【不適切な例】

- 日本語ページ中のフランス語「Bonjour」に言語属性が付与されておらず、スクリーンリーダーが日本語読みしてしまう。

【適切な例】

- 文章中のフランス語部分に lang="fr" が付与され、スクリーンリーダーがその箇所をフランス語として正しく読み上げる。

3.2.1 フォーカス時 レベル A

いずれのユーザインタフェース コンポーネントも、フォーカスを受け取ったときにコンテキストの変化を引き起こさない。

【ポイント】

- フォーカスを受け取った際に、コンテキストの変更を引き起こさない。

【解説】

単にマウスを合わせたり、キーボードでフォーカスを当てただけで何らかの動作を起こしたりする仕掛けは、利用者が予期しない行動を引き起こす可能性があり、それが予測可能性を損ねるため、避けるべきです。以下のような動作が起きるものは禁止されています：

- フォームが自動的に送信される。
- 新しいページが自動で開かれる。
- 予期せず項目が選択される。
- フォーカスが意図しない箇所へ移動する。
- レイアウトが変更される。

【対応方法】

- 実行ボタンを設け、利用者の意思を確認してから実行する。
- コンテキストの変化を引き起こす処理には、focus ではなく activate（Enter / Space / Click 等）を使用する。

【不適切な例】

- マウスオーバーやキーボードフォーカスだけで突如として新しいウィンドウが開く。
- フォーカスを当てただけで自動的にフォーム送信が行われる。

【適切な例】

- 利用者が明示的に「送信」ボタンをクリックしたり、「Enter」キーを押したりすることでのみ、フォーム送信が行われる。
- マウスオーバーやキーボードフォーカスによる予期せぬ動作が起きず、利用者の意志で行動がコントロールできる。

【対応例】

- フォーカス時は見た目だけ変え、動作は発火しないコード例。

```
[html]
  <button class="menu">メニュー</button>
[css]
  .menu:focus-visible{outline:2px solid;outline-offset:2px} /* click/Enter でのみ開く実装にする */
```

3.2.2 入力時 レベル A

ユーザインタフェース コンポーネントの設定を変更することが、コンテキストの変化を自動的に引き起こさない。ただし、利用者が使用する前にその挙動を知らせてある場合を除く。

【ポイント】

- フォームフィールドやリンクがフォーカスを受けても、自動的にページ遷移や送信をしない。
- フォーカス時のコンテキスト変化は、利用者が明示的な操作（Enter やクリックなど）を行った後にのみ実行されるようにする。

【解説】

キーボード利用者やスクリーンリーダー利用者は、Tab キーで順にフォーカスを移動させながら操作します。もしフォーカスを当てただけでページが切り替わったりフォームが送信されたりすると、利用者は意図しない操作をされ、入力の途中でデータが失われたり混乱が生じたりします。特に次のような UI では注意が必要です：

- ドロップダウンリストで項目にフォーカスしただけでページが遷移する。
- フォームのラジオボタンやチェックボックスにフォーカスした瞬間に送信される。

この達成基準は、フォーカス時のコンテキスト変化を抑制し、予測可能で安全な操作を保証することを目的としています。

【対応方法】

- フォーカスを受け取っただけでは、ページの遷移や送信などを実行しない。
- onfocus イベントでのページ遷移や送信処理は避け、クリックや Enter などの明示的なアクション時に実行する。
- ドロップダウンリストやナビゲーションメニューでは、選択肢が決定されたタイミング（例：Enter キーやクリック）でのみ遷移を行うようにする。
- JavaScript のイベントハンドラーは、onfocus ではなく onchange や onclick を使用する。

【不適切な例】

- ドロップダウンメニューにフォーカスを当てただけで、自動的に他のページに遷移する。
- ラジオボタンを Tab キーで選択した瞬間にフォームが送信される。

【適切な例】

- フォーカスを当てた段階では何も起こらず、Enter キーを押して初めてページが切り替わる。
- ドロップダウンリストで項目を選択し、決定ボタンをクリックした後にのみページ遷移が発生する。

【対応例】

フォーカスではなく、ボタン押下という明示的な操作でコンテキストが変化するようにした例。

```
<!-- ✕ 不適切：onfocus でページ遷移 -->
<select onfocus="location.href=this.value">
  <option value="page1.html">Page 1</option>
  <option value="page2.html">Page 2</option>
</select>

<!-- ✅ 適切：ユーザーの決定アクションでのみ遷移 -->
<select id="menu">
```

```
  <option value="">選択してください</option>
  <option value="page1.html">Page 1</option>
  <option value="page2.html">Page 2</option>
</select>
<button onclick="navigate()">移動</button>
<script>
function navigate() {
  const menu = document.getElementById('menu');
  if(menu.value) location.href = menu.value;
}
</script>
```

3.2.3 一貫したナビゲーション レベル AA

ウェブページ一式の中にある複数のウェブページ上で繰り返されているナビゲーションのメカニズムは、繰り返されるたびに相対的に同じ順序で出現する。ただし、利用者が変更した場合は除く。

【ポイント】

- サイト内の共通ナビゲーションは、ページごとに順序や位置を変えない。
- 利用者が予測できる位置に常に配置し、探し直さずに移動できるようにする。

【解説】

ナビゲーションとは、サイト内で利用者がページ間や主要機能へ移動するために設けられたリンク群やメニューを指します。代表例として、グローバルナビゲーション、ローカルナビゲーション、フッターメニュー、パンくずリストなどがあります。本ガイドラインでは、特に複数ページで共通して繰り返し提供されるナビゲーションを対象とします。

利用者はサイト内を移動する際、このナビゲーションの位置や項目の相対的な順序が一定であることを手がかりに操作します。ページによってメニューの位置や順番が変わると、毎回探し直さなければならず、特に認知障害や視覚障害のある利用者にとっては大きな負担となります。そのため、共通のナビゲーションはサイト全体で一貫した順序・位置を保つことが重要です。

※ ローカルナビゲーションはカテゴリごとに項目や順序が異なる場合がありますが、この達成基準で一貫性を求めるのは、各ページで共通して繰り返し提供されるナビゲーション部分です。なお、利用者が並び順をカスタマイズした場合はこの限りではありません。

【対応方法】

- グローバルナビゲーションやフッターメニューなど、全ページ共通のナビゲーションの順序を固定する。
- ナビゲーションの表示位置（例：上部ヘッダー・左側メニューなど）をページごとに変えない。
- レイアウト変更（レスポンシブデザインなど）による表示位置の変化は、機能や順序を損なわないようにする。
- 多言語サイトや特別なセクションでも、共通部分は同じ順序と配置を維持する。

【不適切な例】

- トップページでは「ホーム→製品→サポート→会社情報」の順だが、下層ページでは「会社情報→ホーム→サポート→製品」のように順序が入れ替わっている。
- モバイル表示ではグローバルナビゲーションがフッターに移動し、主要メニュー項目が一部欠落している。

【適切な例】

- すべてのページで、ヘッダー上部に「ホーム→製品→サポート→会社情報→お問い合わせ」の順序で一貫して表示されている。
- レスポンシブ対応でハンバーガーメニューに変わっても、展開した際のメニュー順序はデスクトップ版と同じ。

3.2.4 一貫した識別性 レベル AA

ウェブページ一式の中で同じ機能を有するコンポーネントは、一貫して識別できる。

【ポイント】

- 同じ機能・同じ目的を持つ要素には、常に同じラベルやテキストを用いる。
- アイコンや代替テキストも、同じ役割を示す場合は一貫した表現を使用する。

【解説】

利用者は、ボタンやリンクなどのインタラクティブ要素のテキストやアイコンの表現を手がかりに機能を理解します。同じ機能を持つボタンがページによって異なるラベルで表示されると、混乱を招き操作ミスの原因となります。たとえば、検索機能のボタンがあるページでは「検索」、別のページでは「探す」と表示されていると、同じ機能だと気付かない可能性があります。一貫したラベルやアイコンを用いることで、利用者はどこにいても同じ体験を予測でき、迷わず操作できます。

また、対象はボタンやリンクだけではありません。タブ・アコーディオン・モーダルの「閉じる」・ページネーション・パンくず・ハンバーガーメニュー・並び替え（ソート）／絞り込み（フィルタ）・カルーセルの「前へ／次へ」・地図/スライダーの「拡大／縮小」・メディアの「再生／一時停止」など、同じ役割の UI は名称（ラベル／aria-label）・アイコン・代替テキストを全ページで統一してください。

【対応方法】

- 同じ機能・役割・目的の UI は、視覚ラベル／読み上げ名（aria-label・alt）／ツールチップ／ヘルプ文を同一の表現で統一する。
- アイコンは同じ意味なら同じ絵柄・スタイルを用い、代替テキスト（aria-label・alt）も同一語に統一する（テキスト併記時は重複読上げを避ける）。
- 状態・アクション語は対で統一する（例：「開く／閉じる」「表示／非表示」「開始／一時停止」「追加／削除」）。表示ラベルと aria-* の状態変化は連動させる。
- 配置や文脈が変わっても（ヘッダー／本文／モバイル UI 等）、同一機能には同じ用語と順序を用いる。
- サイト全体で用語ガイド（文言表）を管理し、新規作成・改修時はそれに必ず準拠する。変更が必要な場合は、全ページ一括で整合させる。

【不適切な例】

- 検索ボタンがトップページでは「検索」、別のページでは「探す」と表示されている。
- 登録データ削除ボタンのアイコンがページ毎に異なり、代替テキスト「削除」「ゴミ箱」「消去」と統一されていない。
- アイコンは同じ機能だがデザインや alt が不一致である。

```
<img src="trash-bin.svg" alt="ゴミ箱">
<img src="delete.svg" alt="消去">
<img src="remove-icon.svg" alt="削除する">
```

【適切な例】

- サイト内のすべてのページで、検索機能のボタンは「検索」と統一されている。
- 「削除」ボタンは全ページで同じゴミ箱アイコンを使用し、代替テキストも「削除」で統一されている。
- 製品詳細ページ群のタブは全ページで「概要／仕様／レビュー」の並びと文言を統一し、モバイルのドロップダウン表示でも同じ順序・同じラベル（aria-label 含む）を維持している。

3.2.6 一貫したヘルプ レベル A

ウェブページが次のヘルプのメカニズムのいずれかを含み、かつ、それらのメカニズムがウェブページ一式の中にある複数のウェブページで繰り返されている場合、他のページコンテンツに対して相対的に同じ順序で出現する。ただし、変更が利用者によって行われた場合は除く。

- 人間への連絡先
- 人間への連絡メカニズム
- 自己解決のためのオプション
- 完全に自動化された連絡メカニズム

ヘルプのメカニズムは、ページ上で直接提供されることもあれば、その情報を含む別のページへの直接リンクを経由して提供されることもある。
この達成基準における「他のページコンテンツに対して相対的に同じ順序」は、ページが線形化されたときのコンテンツの並べ方と考えることができる。ヘルプのメカニズムの視覚的な位置は、ページの同じバリエーション（例えば CSS のブレークポイント）に対してページ間で一貫している可能性が高い。利用者は、ページのズーム、ページの向きを変えるなどの変更を発生させることができ、これはページのバリエーションが変化するきっかけになる可能性がある。この達成基準で扱うのは、同じバリエーション（例えば、同じズームレベル、向き）で表示されるページ同士の間における、相対的な順序である。

【ポイント】

- ヘルプやサポートへのリンクは、全ページで同じ位置・順序に配置する。
- 問い合わせ窓口・FAQ・チャットサポートなどは、予測可能で一貫した方法で提供する。

【解説】

利用者は困ったときにすぐにヘルプを見つけられることが重要です。ページごとにヘルプの場所やリンク名が異なると、利用者は探し直さなければならず、特に認知や学習の負担が増します。そのため、サイト全体でヘルプ情報への導線（位置・ラベル・手順）を統一しておくことが推奨されます。一貫性が保たれていれば、ユーザーはどのページにいても迷わずサポートへアクセスできます。

【対応方法】

- ヘルプ・FAQ・問い合わせフォームなどのリンクを、ヘッダー・フッター・サイドバーなど一定の位置に固定して配置する。
- サポートリンクのラベル名は統一する（例：「サポート」か「お問い合わせ」のいずれかに統一）。
- チャットボットなどの支援ツールも、全ページで同じ場所・同じアイコンを用いる。
- モバイル版でもヘルプリンクの位置をわかりやすく確保する。

【不適切な例】

- ヘルプへのリンクがトップページではフッター、製品ページではヘッダーにあり、ページごとに探し直す必要がある。
- 問い合わせ窓口のラベルがページによって「サポート」「問い合わせ」「ヘルプ」と異なり、利用者が混乱する。
- チャットサポートのアイコンが一部のページで非表示になり、ヘルプにたどり着けない。

【適切な例】

- 全ページのフッター右側に常に「お問い合わせ」リンクを配置し、どこからでも同じ導線でアクセスできる。

- サポートへの案内を「ヘルプ」と統一したラベルで、ヘッダーとフッターの両方に同じ位置で表示している。
- チャットボットのアイコンを全ページ右下に固定表示し、利用者が迷わず起動できる。
- ヘルプリンクをフッターに一貫して配置している。

```
<!-- ヘルプリンクをフッターに一貫して配置 -->
<footer>
  <nav aria-label="補助ナビゲーション">
    <a href="/help" aria-label="ヘルプ">ヘルプ</a>
    <a href="/contact" aria-label="お問い合わせ">お問い合わせ</a>
    <a href="/faq" aria-label="よくある質問">FAQ</a>
  </nav>
</footer>
<!-- チャットサポートのアイコンを全ページ右下に固定 -->
<style>
.chatbot-button {
  position: fixed;
  bottom: 1rem;
  right: 1rem;
}
</style>
<button class="chatbot-button" aria-label="チャットサポート">💬</button>
```

ヘルプ関連リンクを常にフッター内に配置し、チャットボタンも同じ位置に固定してユーザーが迷わないようにする。

3.3.1 エラーの特定 レベル A

入力エラーが自動的に検出された場合は、エラーとなっている箇所が特定され、そのエラーが利用者にテキストで説明される。

【ポイント】

- 入力項目にエラーがあった場合、そのエラー箇所と内容を明確に特定できるようにする。

【解説】

フォームなどで必須入力を未入力のまま送信したり、形式の誤ったデータを入力したりした場合、利用者にどの項目がエラーなのか、何が間違っているのかを明確に伝える必要があります。単に「エラーがあります」とまとめて表示するだけでは、ユーザーは次にどこを確認すべきか迷い、操作が滞ります。エラー箇所が特定できるメッセージを表示することで、利用者は問題点を迅速に見つけ、対応できるようになります。

【対応方法】

- エラーが発生した場合は、エラー箇所を特定できるメッセージを個別に表示する。
- エラー発生箇所に aria-invalid="true" を付与し、支援技術がエラー項目を特定できるようにする。
- エラーメッセージは関連するフォーム要素と関連付ける（例：aria-describedby 属性）。
- エラー表示は視覚的にもわかるよう、色やアイコン、テキストなどで強調する（ただし色だけに頼らない）。
- エラーがリアルタイムに発生した場合は、リアルタイムに伝えるために ARIA「role=alert」を利用する。（表示したタイミングでスクリーンリーダーが読み上げてくれる）

【不適切な例】

- 「入力項目にエラーがあります。」としか提示されず、どの項目にエラーがあるのかわからない。

【適切な例】

- 「郵便番号は 7 桁で入力してください。」と表示され、エラー箇所が郵便番号入力欄であることがわかる。

【対応例】

- エラー発生時に、何をどう直すかを具体的に示し、支援技術へ通知しているコード例。

```
[html]
  <input id="zip" aria-describedby="zip-err" aria-invalid="true">
  <span id="zip-err" role="alert">郵便番号は 7 桁で入力してください。</span>
```

【例外】

- ログインエラーについて、ID の間違えであれば「ID に誤りがあります。」、パスワードに誤りがあれば「パスワードに誤りがあります。」など、原則個別のエラーを表示する必要があります。
  しかし、セキュリティの問題であえて ID とパスワードを曖昧にしたエラーを表示する場合は、下記の例のようにエラー内容の確認を促す文章を表示してください。

ユーザーID またはパスワードに誤りがあります。 入力したユーザーID またはパスワードに誤りがないか、ご確認ください。

3.3.2 ラベル又は説明 レベル A

コンテンツが利用者の入力を要求する場合は、ラベル又は説明文が提供されている。

【ポイント】

- 入力欄には、どのような情報を入力すべきかの手がかりを、ラベルや説明文により提供する。
- 入力内容に制限（例：文字数、使用できる文字、書式、必須／任意など）がある場合は、placeholder だけに頼らず、説明文として常時可視で示す。

【解説】

スクリーンリーダーの利用者は、操作効率を向上させるため、テキスト入力エリアのみを拾い読みする場合があります。このような時に、フォームコントロールに label 要素による明確なラベルや title 属性によるタイトルが設定されていれば、何を入力すべきかが速やかに理解でき、大幅な操作効率の向上が期待できます。

あるいは、スクリーンリーダーでのサポート状況を確認して、慎重に利用する必要がありますが、aria-label や aria-labelledby 属性が使われることも増えています。

【対応方法】

- 入力欄や選択項目に、どのような内容を入力すべきかを示す明瞭なラベルを配置する。
  - **label 要素による方法：**

このサイトを検索

```
<form id="search" role="search">
 <label for="search-label">このサイトを検索</label>
 <input type="search" id="search-label" name="search">
 <input value="検索する" type="submit">
</form>
```

  - **title 属性による方法：**

なまえ

```
<p>なまえ</p>
<input type="text" name="secondname" title="せい">
<input type="text" name="firstname" title="めい">
```

  - **aria-label 属性による方法：**

```
<input type="text" name=”secondname” aria-label="せい" />
<input type="text" name=”firstname” aria-label="めい" />
```

- 入力データに条件がある場合、その条件や入力例を明記する。
  - 文字数、文字種（ひらがな/カタカナ/半角/全角等）、書式（例：YYYY-MM-DD）、
    入力例（例：tanaka@example.com）などを常時可視で示し、対象の入力欄に関連付ける。

（注：placeholder 属性は入力開始と同時に消えてしまうため、これだけでの説明では不十分です。）

必須入力項目は、フォームの先頭や入力項目前でその旨の説明を行う。

名前 (必須)

【不適切な例】

- フォームに何を入れるべきか placeholder による表示でしか説明していない。
- 文字数・書式などのルールを placeholder だけで案内しており、入力開始とともに説明が消えてしまう。

【適切な例】

- label 要素あるいは title 属性をしっかりと使用し説明している。
- 入力ルール（例：文字数／文字種／書式／入力例）を説明文として常時可視で表示している。

【対応例】

- プレースホルダー依存を避け、明示的ラベルを関連付けているコード例。

```
[html]
  <label for="email">メール</label><input id="email" type="email" autocomplete="email">
```

```
[html]
  <label for="kana">ふりがな</label>
  <p id="kana-hint">ひらがなで入力(全角 20 文字以内)／例:たなか はなこ</p>
  <input id="kana" name="kana" type="text" aria-describedby="kana-hint">
```

3.3.3 エラー修正の提案 レベル AA

入力エラーが自動的に検出され、修正方法を提案できる場合、その提案が利用者に提示される。ただし、セキュリティ又はコンテンツの目的を損なう場合は除く。

【ポイント】

- 入力エラーが発生した場合、何をどのように直せばよいかを明確に提示する。

【解説】

エラー内容だけを表示するのではなく、修正のための手がかりを具体的に提供することで、利用者は迷わず次の行動に移れます。例えば、「郵便番号が間違っています」だけでは不足です。「郵便番号は 7 桁の数字で入力してください」といった具体的な修正方法が提示される必要があります。特に高齢者やデジタルに不慣れなユーザーにとって、エラー修正の手がかりがあるかどうかは利用体験を大きく左右します。

【対応方法】

- エラー箇所の説明に加え、修正方法をガイドする文章を表示する。
- 可能な場合は、入力例やフォーマット例（例：YYYY/MM/DD 形式など）を提示する。
- 修正ガイドはエラー発生箇所の近くに表示し、支援技術（スクリーンリーダー）にも認識されるようにする。
- セキュリティ上の理由で詳細を表示できない場合も、できる範囲で次に取るべき行動を示す。
  （例：パスワードが間違っている場合には、「再入力してください」「入力内容を表示するアイコンで確認できます」など、利用者が次に何をすべきかを伝える。）

【不適切な例】

- 「入力エラーがあります」とだけ表示され、どこをどのように直せばよいかがわからない。
- 「メールアドレスが無効です」としか書かれておらず、どの形式が正しいのか示されない。

【適切な例】

- 「郵便番号は 7 桁の数字で入力してください」とフォーマットを具体的に示す。
- 「パスワードは 8 文字以上の英数字を含めてください」と条件を明確に伝える。
- 「日付は YYYY/MM/DD 形式で入力してください」と入力例を添える。エラー箇所と修正方法が提案されている。

【対応例】

- エラー発生時に、何をどう直すかを具体的に示し、支援技術へ通知しているコード例。

```
<form>
  <label for="zip">郵便番号:</label>
  <input id="zip" aria-describedby="zip-err" aria-invalid="true">
  <span id="zip-err" role="alert">
    郵便番号は 7 桁の数字で入力してください（例：1234567）。
  </span>
</form>
```

エラー時に修正方法を含めたメッセージを表示、aria-describedby で関連付け読み上げられるようにする。

3.3.4 誤り防止（法的、金融、データ）レベル AA

利用者にとって法律行為もしくは金融取引が生じる、利用者が制御可能なデータストレージシステム上のデータを変更もしくは削除する、又は利用者が試験の解答を送信するウェブページでは、次に挙げる事項のうち、少なくとも一つを満たしている:

取消

送信を取り消すことができる。

チェック

利用者が入力したデータの入力エラーがチェックされ、利用者には修正する機会が提供される。

確認

送信を完了する前に、利用者が情報の見直し、確認及び修正をするメカニズムが利用できる。

【ポイント】

- 法的な内容や金融取引、その他利用者の利益に関わるデータに関しては、利用者が誤操作から簡単に回復できる手段を用意する必要がある。

【解説】

法的義務や金融取引が生じるコンテンツは利用者にとって非常に重要なため、一層の配慮が求められます。【対応方法】のいずれかの手段を取り入れて、利用者が安心して取引やデータ入力ができるようにします。

【対応方法】

- 送信した内容のキャンセルが可能である。
  送信内容の確認ページを介し、必要に応じて送信をキャンセルしたり、送信後に一定期間内に内容をキャンセルしたりできる機能を提供する。
- 確認ページを用意する。
  送信前に、入力項目を再確認できるページを設け、必要に応じて修正できる機能を提供する。
- 送信確認のチェックボックスを用意する。
  送信ボタンの直前に、「入力項目を確認しました」というチェックボックスを設置し、このチェックボックスが未選択の場合は送信が行えないようにする。

【不適切な例】

- 送信ボタンを押した瞬間に取引が完了し、キャンセルや修正ができない。
- エラー時に具体的なエラーメッセージが表示されず、利用者がどの項目を修正すればよいか分からない。

【適切な例】

- 入力内容に誤りがあった場合、どの項目に誤りがあるのかを明示されるので、簡単に修正できる。
- オンラインショッピングでの購入前に「注文内容の最終確認」ページが表示され、注文を確定する前に内容の確認と修正が行える。

3.3.7 冗長な入力 レベル A

以前に利用者によって入力された、又は利用者に対して提供された情報であって、同一のプロセスにおいて再入力する必要がある情報は、次のいずれかである。

- 自動入力される。又は、
- 利用者が選択可能である。

ただし、次の場合は除く:

- 情報の再入力が必要不可欠である。
- その情報がコンテンツのセキュリティを確保するために必要である。又は、
- 以前に入力された情報が無効になっている。

【ポイント】

- 利用者が以前に提供した情報は再利用し、入力の手間を減らす。
- 同一のウェブページ内で繰り返し入力が求められる場合、自動的にフィールドを埋める機能を提供する。

【解説】

ウェブサイトやアプリケーションにおけるフォームの利用は頻繁に行われますが、利用者にとって同じ情報を何度も入力させることは煩わしいだけでなく、特に認知障害や身体障害を持つ利用者にとっては大きな負担となります。利用者が以前に入力した情報を保存し、再利用することで、入力の効率を高めることができます。

【対応方法】

- 利用者が一度入力したデータは、セッションをまたいで保存し再利用する。
- ブラウザの autocomplete 機能を有効にし、利用者がフォームを速やかに完成させられるようにする。
- 複数のフォームフィールドにわたって同じ情報（例えば、利用者名や住所）を使用する場合は、最初の入力後、自動で他の関連フィールドに情報をコピーする。

【不適切な例】

- 利用者がアカウント情報を更新するたびに、同じ個人情報を再度入力しなければならない。

【適切な例】

- 利用者が住所を一度入力すると、次回の購入時にその住所が自動的に入力フィールドに表示される。

【補足】

Autocomplete は、以前に入力した値をフィールドへ自動補完するブラウザの機能です。同じ情報の再入力を減らすのに役立ちます。HTML の input 要素に autocomplete 属性を指定することで、この機能を利用できます。

```
<!-- 名前の入力フィールド -->
<input type="text" name="name" autocomplete="name">
<!-- メールアドレスの入力フィールド -->
<input type="email" name="email" autocomplete="email">
<!-- 新しいパスワードの入力フィールド -->
<input type="password" name="new-password" autocomplete="new-password">
```

autocomplete 属性に指定された値（例：name、email、new-password）は、ブラウザがどの情報をフィールドに自

動入力するかの判断に役立ちます。安全性のため、特定の情報（とくにセキュリティ関連）では自動完了を無効化することも重要です。さらに、プライバシーを十分尊重し、許可なく個人データを自動入力しないようにします。

【対応例】

- 既入力情報を再利用できる設計のコード例。

```
[html]
  <input name="name" autocomplete="name"> <input name="billing_name" autocomplete="name">
```

3.3.8 アクセシブルな認証（最低限）レベル AA

認知機能テスト (パスワードを記憶する、パズルを解く、など) は、認証プロセスのどのステップにおいても要求されない。ただし、そのステップが次の少なくとも一つを提供する場合は除く。

代替手段

認知機能テストに依存しない別の認証方法がある。

メカニズム

認知機能テストを利用者が完遂できるように支援するメカニズムが利用できる。

物体の認識

認知機能テストは、物体を認識させるものである。

個人特有のコンテンツ

認知機能テストは、その利用者本人がウェブサイトに提供した非テキストコンテンツを識別させるものである。

「物体の認識」及び「個人特有のコンテンツ」は、画像、映像、又は音声によって表現される場合がある。

この達成基準を満たすメカニズムの例としては、次が挙げられる。

1. パスワードマネージャーによるパスワード入力の支援によって、記憶の必要性を軽減する。
2. コピー＆ペーストによって、再度のタイピングに伴う認知負荷を軽減する。

【ポイント】

- すべての利用者が利用しやすい認証方法を提供する。
- 認知能力に依存しない認証オプションを提供し、より幅広い利用者のニーズに対応する。

【解説】

認証プロセスは、安全であると同時に、すべての利用者がアクセスしやすいものでなければなりません。記憶に依存する従来の認証方法は、認知障害を持つ利用者にとって障壁となり得るため、代替の手段が必要です。記憶依存や知覚依存の課題を必須にせず、代替手段を用意します。これには、SMS コード（携帯にコードを送信する 2 段階認証の１つ）、生体認証、セキュリティトークンなどが含まれます。

【対応方法】

- パスワード以外の認証方法を提供し、利用者が選択できるようにする。
- 認証情報を記憶することなく、利用者がアカウントにアクセスできる方法を提供する。
- 認証試行に失敗した場合、利用者が再試行しやすい支援を提供する。

【不適切な例】

- パスワードのコピー＆ペーストやパスワードマネージャの利用を禁止し、毎回手入力を強制している。このような実装は、記憶依存を高め利用者の認知的負荷を不必要に増大させるため、認証障壁となる。
- 利用者がログインするたびに、複雑なパスワードとセキュリティ質問の答えを入力する必要がある。これは特に、記憶力に依存する認知障害を持つ利用者や高齢者にとって大きな障壁となる。
- コミュニティフォーラムへのログインでは、利用者が登録時に選んだ複数の写真から特定のものを選ぶ必要がある。この方法は視覚障害のある利用者や記憶が困難な利用者にとって非常に困難である。

- ウェブサービスのログインにおいて、利用者はパズルを解くよう求められる。パズルではピースをドラッグして画像を完成させる。この認証方法は、手の不自由な利用者や認知障害のある利用者に困難で、ログインの障害となる。スクリーンリーダー利用者にとっては、このパズルは不可能に近い。

【適切な例】

- ログイン時に、パスワードのコピー＆ペーストやパスワードマネージャの自動入力を許可しており、記憶や再入力を強いられることなく認証を完了できる。
- ログインでは、パスワードに加え、スマートフォンに送信される一時コードを用いる多要素認証を導入する。パスワード、受信した一時コードともにコピー＆ペーストでを入力して認証を完了できる。
- 会員専用ウェブサイトでは、ログイン時に生体認証（指紋認証や顔認証）を使用するオプションを提供している。利用者はパスワードを覚える必要なく、簡単にアクセスできるようになっている。
- e コマースサイトは、ログイン時にメールへリンクを送り、そのリンクのクリックで認証する「パスワードレスログイン」を採用している。利用者はパスワードを覚える必要がなく、メールにアクセスできれば簡単・迅速にログインできる。

【対応例】

- 記憶依存を避ける補助（表示切替・貼り付け許可）のコード例。

```
[html]
   <input id="pw" type="password" autocomplete="current-password">
   <button type="button" aria-controls="pw" aria-pressed="false" onclick="const
s=pw.type==='password';pw.type=s?'text':'password';this.setAttribute('aria-pressed',s)">表示/非表示</button>
```

4.1.1 構文解析 レベル A（WCAG 2.2 では「廃止・削除」）

マークアップ言語を用いて実装されているコンテンツにおいては、要素には完全な開始タグ及び終了タグがあり、要素は仕様に準じて入れ子になっていて、要素には重複した属性がなく、どの ID も一意的である。ただし、仕様で認められているものを除く。

HTML 又は XML を使用するすべてのコンテンツでは、この達成基準は常に満たされているとみなすべきである。

この基準が作成されて以降、HTML Living Standard は、ユーザエージェントが不完全なタグ、不正な要素の入れ子、重複した属性、及び一意でない ID をどのように処理しなければならないかを規定する明確な要件を採用した。
HTML Standard では、これらのケースのいくつかをコンテンツ制作者に対する不適合として扱うが、仕様ではユーザエージェントがこれらのケースの一貫した処理をサポートすることを要求しているため、この達成基準の説明にある「仕様で認められているもの」とみなせる。実際には、この基準自体は、もはや障害のある人々に何の利益ももたらさない。
不適切に入れ子にされた要素による役割の欠落、ID の重複による不正な状態や名前などの問題は、別の達成基準で扱われており、4.1.1 の問題としてではなく、それらの基準に基づいて報告されるべきである。

【ポイント】

- 作成するページが正確な構文で記述されているか、文法チェックツールで確認する。
- テキストの意味や役割に適した要素や属性が使用されているか確認する。

【解説】

WCAG 2.2 では、達成基準 4.1.1「構文解析」は削除されました。

この変更は、HTML 標準仕様の改善によりユーザーエージェント（ブラウザなど）が文法上のエラーを自動補正し、一貫した処理を行うことが規定されたためです。

しかし、文法的に正しい HTML を記述することは依然として重要です。

構文エラーが多いと、ブラウザや支援技術による解釈に差異が生じ、アクセシビリティの低下や予期しない表示崩れが起こる可能性があります。

したがって、WCAG 2.2 では直接評価対象ではないものの、開発者は引き続き正しい構文を保つようにしてください。

【対応方法】

- W3C Markup Validation Service などの文法チェックツールで、エラーがないことを確認する。
- 開始タグ・終了タグの対応、属性の重複や未閉じの要素をなくす。
- id 属性は一意であることを確認する。

【不適切な例】

- ブラウザでは表示されるが、文法エラーが多く、スクリーンリーダーがコンテンツを正しく認識できない。
- 同じページ内で複数の要素に同じ id が設定されている。
- 見た目を整えるために や だけを多用し、意味のある要素を使っていない。

【適切な例】

- バリデータでエラーがなく、HTML が適切な要素と ARIA ロールで記述されている。

【補足：WCAG 2.2 における 4.1.1「構文解析」の扱いについて】

WCAG 2.2 では達成基準 **4.1.1「構文解析」は廃止・削除され**、HTML 標準に従ってユーザーエージェント（ブラウザなど）は HTML 文書の文法的な不備に対して一貫した処理を行うことが求められるようになりました。これにより、HTML 文書の不完全な要素や非一意な ID などの問題がユーザーエージェントによってうまく処理されるため、4.1.1 は直接的なアクセシビリティの利益を提供しなくなりました。それでも、正しい HTML 構文を使用し、関連する問題は他の適切な達成基準に基づいて評価することが重要です。

4.1.2 名前（name）・役割（role）・値（value）レベル A

すべてのユーザインタフェース コンポーネント（フォームを構成する要素、リンク、スクリプトが生成するコンポーネントなど）では、名前（name）及び役割（role）は、プログラムによる解釈が可能である。又、状態、プロパティ、利用者が設定可能な値はプログラムによる設定が可能である。そして、支援技術を含むユーザエージェントが、これらの項目に対する変更通知を利用できる。

この達成基準は、主に独自のユーザインタフェース コンポーネントを開発、又はスクリプトで実装するコンテンツ制作者に向けたものである。例えば、標準的な HTML のコントロールを仕様に沿って使用していれば、既にこの達成基準を満たしていることになる。

【ポイント】

- すべての UI コンポーネントにプログラムから認識可能な名前(name)と役割(role) を設定する。
- コンポーネントの状態・値の変化は支援技術に通知されるようにする。
- 標準 HTML 要素を仕様どおりに使用し、必要に応じて ARIA 属性で補完する。

【解説】

この達成基準は、スクリーンリーダーや音声入力などの支援技術が UI コンポーネントの**名前・役割・状態・値を正しく認識し、ユーザーに伝えられること**を保証するためのものです。

標準的な HTML 要素（例：<button>や<input>など）は仕様どおりに使用すれば多くの場合達成基準を満たしますが、スクリプトで独自 UI を作成する場合は role 属性や ARIA プロパティを適切に指定する必要があります。

【対応方法】

- 標準 HTML 要素を正しく使用する（例：ボタンには<button>を用いる）。
- カスタム UI コンポーネントには、role 属性で役割を明示する（例：カスタムボタンに role="button"）。
- 動的な開閉や切り替えなど状態変化がある要素には aria-expanded や aria-checked などの ARIA 属性を適切に設定し、変化時に値を更新する。
- 状態や値が変化した際に、支援技術がその変化を認識できるよう JavaScript で値を更新する。

【不適切な例】

- で作られたボタンに role="button"がなく、スクリーンリーダーがボタンと認識しない。
- アコーディオンメニューが開閉しても aria-expanded が更新されない。

【適切な例】

- button 要素を使用し、開閉に伴い aria-expanded の値を true/false で更新している（スクリーンリーダーに開閉状況が通知される）。

【対応例】

- 開閉ボタンの role/状態を明示するコード例。

```
<!-- 開閉ボタンの role と状態を明示した例 -->
<button aria-expanded="false" aria-controls="sec" id="t">詳細</button>
<section id="sec" hidden>…</section>
```

```
<script>
t.onclick = () => {
   const expanded = t.getAttribute('aria-expanded') === 'true';
   t.setAttribute('aria-expanded', !expanded);
   sec.hidden = expanded;
};
</script>
```

※ ボタンに aria-expanded を設定し、状態変化を支援技術にも伝えている。

4.1.3 ステータスメッセージ レベル AA

マークアップ言語を使って実装されたコンテンツでは、ステータスメッセージは、役割（role）又はプロパティを通してプログラムによる解釈が可能であり、フォーカスを受けとらなくても支援技術によって利用者に提示することができる。

【ポイント】

- 利用者の操作結果によって発生するステータスメッセージは、スクリーンリーダーなどの支援技術で識別しやすくする。
- ステータスメッセージは、ページのコンテンツを読み直さずに、利用者に重要な情報を提供する。

【解説】

この達成基準は、利用者がページの別の部分に移動したり、再読み上げをしたりしなくても操作結果や進行状況を即座に把握できるようにすることを目的としています。

典型的なステータスメッセージの例：

- フォーム送信後の「保存しました」「送信に失敗しました」
- ショッピングカートへの「商品が追加されました」
- ページ内で進行状況が変化した際の「50%完了しました」や「検索結果が更新されました」

メッセージはユーザーの操作を中断させる必要がないため、モーダルウィンドウやフォーカス移動ではなく、スクリーンリーダーが自動的に読み上げる仕組み（ARIA Live リージョン）を活用します。

【対応方法】

- 役割に応じた ARIA ロールを使用する
  - 情報更新や進行状況の通知には role="status" または aria-live="polite" を使用
  - エラーや警告など即時通知が必要なものには role="alert" または aria-live="assertive" を使用
- ステータスメッセージのテキストを JavaScript で更新する際は、要素を再描画するのではなく既存の要素のテキストを変更して通知を発火させる。
- 視覚的な変更がなくても、支援技術が認識できるように aria-live を常に有効化した要素をページに配置しておく。

【不適切な例】

- フォーム送信後に「保存しました」というメッセージがページ上部に表示されるが、role や aria-live がないためスクリーンリーダーが読み上げない。
- 非同期で検索結果が更新されたが、利用者には変更があったことが伝わらず、結果を見直さないと気付けない。
- エラーが発生したが、画面下部に赤字で表示されただけで、スクリーンリーダーでは検出されない。

【適切な例】

- カート追加時に、画面右上の非表示エリアにある が「商品がカートに追加されました」と更新され、スクリーンリーダーが直ちに読み上げる。
- フォーム送信エラーを role="alert" に設定し、「郵便番号を 7 桁で入力してください」と即座に伝える。フォーカス移動は行わず、ユーザーは作業を続けられる。
- ローディング中は aria-live="polite" のリージョンで「検索中です…」と伝え、完了後に「検索が完了しました。10 件の結果があります」と更新する。

【対応例】

- 非モーダルの結果を支援技術へ通知しているコード例。

```
<!-- ページ常駐のステータスメッセージ領域 -->
<div id="msg" role="status" aria-live="polite"></div>

<script>
  // 保存成功時に通知
  function saveData() {
    // 保存処理...
    msg.textContent = '保存しました'; // テキストを更新すると支援技術が読み上げる
  }

  // エラー時にはより強い通知を使用
  function showError(message) {
    msg.setAttribute('role','alert');
    msg.textContent = message;   // 例: "郵便番号を 7 桁で入力してください"
  }
</script>
```

その他役に立つ情報

1. Colour Contrast Analyzer
   https://www.tpgi.com/color-contrast-checker/
   スポイトツールで画面上の 2 点のコントラスト比を計るツール

2. miChecker（エムアイチェッカー）
   http://www.soumu.go.jp/main_sosiki/joho_tsusin/b_free/michecker.html
   WCAG 2.0 の達成基準の内、機械的に判定可能な項目をチェックするツール

3. Web Developer
   https://addons.mozilla.org/ja/firefox/addon/web-developer/
   ブラウザに組み込んで使用するアクセシビリティチェッカー
   （上記のリンクは Firefox 版。Google Chrome 版も提供されています）

4. NVDA 日本語版
   https://www.nvda.jp/
   無料（オープンソース）のスクリーンリーダー（音声読み上げソフト）

達成基準チェックリスト (参考 : WCAG 2.2 用)

WCAG 2.2					
達成基準		**適合レベル**	**適用**	**結果**	**注記**
1.1.1	非テキストコンテンツ	A			
1.2.1	音声のみ及び映像のみ (収録済)	A			
1.2.2	キャプション (収録済)	A			
1.2.3	音声解説、又はメディアに対する代替 (収録済)	A			
1.2.4	キャプション (ライブ)	AA			
1.2.5	音声解説 (収録済)	AA			
1.3.1	情報及び関係性	A			
1.3.2	意味のあるシーケンス	A			
1.3.3	感覚的な特徴	A			
1.3.4	表示の向き	AA			
1.3.5	入力目的の特定	AA			
1.4.1	色の使用	A			
1.4.2	音声の制御	A			
1.4.3	コントラスト (最低限)	AA			
1.4.4	テキストのサイズ変更	AA			
1.4.5	文字画像	AA			
1.4.10	リフロー	AA			
1.4.11	非テキストのコントラスト	AA			
1.4.12	テキストの間隔	AA			
1.4.13	ホバー又はフォーカスで表示されるコンテンツ	AA			
2.1.1	キーボード	A			
2.1.2	キーボードトラップなし	A			
2.1.4	文字キーのショートカット	A			
2.2.1	タイミング調整可能	A			
2.2.2	一時停止、停止、非表示	A			
2.3.1	3 回の閃光、又は閾値以下	A			
2.4.1	ブロックスキップ	A			
2.4.2	ページタイトル	A			
2.4.3	フォーカス順序	A			
2.4.4	リンクの目的 (コンテキスト内)	A			
2.4.5	複数の手段	AA			
2.4.6	見出し及びラベル	AA			

2.4.7	フォーカスの可視化	AA			
2.4.11	隠されないフォーカス（最低限）	AA			
2.5.1	ポインタのジェスチャ	A			
2.5.2	ポインタのキャンセル	A			
2.5.3	ラベルを含む名前（name）	A			
2.5.4	動きによる起動	A			
2.5.7	ドラッグ操作	AA			
2.5.8	ターゲットのサイズ（最低限）	AA			
3.1.1	ページの言語	A			
3.1.2	一部分の言語	AA			
3.2.1	フォーカス時	A			
3.2.2	入力時	A			
3.2.3	一貫したナビゲーション	AA			
3.2.4	一貫した識別性	AA			
3.2.6	一貫したヘルプ	A			
3.3.1	エラーの特定	A			
3.3.2	ラベル又は説明	A			
3.3.3	エラー修正の提案	AA			
3.3.4	誤り防止（法的、金融、データ）	AA			
3.3.7	冗長な入力項目	A			
3.3.8	アクセシブルな認証（最低限）	AA			
4.1.1	構文解析 *	A			
4.1.2	名前（name）・役割（role）・値（value）	A			
4.1.3	ステータスメッセージ	AA			

※ 達成基準 4.1.1 は JIS X 8341-3:2016 あるいは WCAG 2.0/2.1 を遵守する場合に必要。
WCAG 2.2 に遵守する場合は除外し適用欄は「－（非適用）」とする。

付録

付録 A スクリーンリーダー動作確認の手順

CMS 上で作成・更新したページを、**音声読み上げ環境で最低限確認**するための手順を示します。開発者向けの詳細検証ではなく、公開前チェックを想定しています。

前提環境

- Windows 11 標準搭載の ナレーター を使用
- ブラウザは Microsoft Edge（最新版）

操作手順

1. **ナレーターを起動**
   - キー：Windows + Ctrl + Enter
2. **対象ページを開く**
   - Edge でプレビューを表示し、Alt + Tab で切り替え
   - 読み上げが長い場合は Ctrl で一時停止
3. **確認内容別アクション**（必要なものだけ実施）
   - 見出し構造の確認 → H / Shift + H
   - リンクテキストの確認 → Tab / Shift + Tab
   - ランドマークの確認 → D / Shift + D
   - フォームコントロールの確認 → F / Shift + F
   - 各箇所の確認 → 上下矢印キー
4. **ナレーターを終了**
   - キー：Windows + Ctrl + Enter

付録 B リンク実装ガイド

画像＋テキストを 1リンクにまとめる

同じ URL を指す画像とテキストは **1つの a 要素**にまとめ、画像 alt を空にして冗長読み上げを防ぎます（達成方法 H30＋H2）。

```
<a href="lp_fl.html">花の一覧<img src="fl.gif" alt=""></a>
```

リンク目的を補足する 3 つの方法

以下は表示テキストだけで目的が伝わらない場合に使用します。

1. aria-label 属性（ARIA8）

```
<a href="sample" aria-label="拡大地図を表示">地図</a>      <!—“拡大地図を表示”のみを読み上げる  -->
```

2. visually-hidden クラス（C7）

```
<a href="sample">地図<span class="visually-hidden">（拡大表示）</span></a>

.visually-hidden{ position:absolute; width:1px; height:1px; margin:-1px; border:0; padding:0; clip:rect(0 0 0 0);     clip-path:inset(50%); overflow:hidden; white-space:nowrap; }
```

3. aria-labelledby で外部テキスト参照（ARIA1）

```
<span id="moreNewsTxt" class="visually-hidden">最新ニュースをもっと見る</span>

<a href="news.html" aria-labelledby="moreNewsTxt">▶</a>
```

※こちらの例では、visually-hidden で隠しテキストを設け、aria-labelledby で隠しテキストを参照してリンクのラベルとしています。通常のテキストなど隠しテキストでなくとも参照先とすることは可能です。

アイコンのみリンク／ボタン

```
<button aria-label="検索">🔍</button>

<a href="file.pdf" aria-label="PDF をダウンロード">📄</a>
```

外部リンク・新規タブを知らせる

・ aria-label 属性を使用する場合

```
<a href="guide.pdf" target="_blank" rel="noopener" aria-label="PDF を別ウィンドウで開く">

PDF をダウンロード</span>

</a>
```

・ visually-hidden を使用する場合

```
<a href="guide.pdf" target="_blank" rel="noopener" >

PDF をダウンロード<span class="visually-hidden">新しいウィンドウで開く</span>

</a>
```

NG 例（音声読み上げ不可）

```
<span style="display:none">隠しテキスト</span>     <!-- × display:none -->
```

display:none / visibility:hidden はスクリーンリーダーも無視するため、補足情報には **使用不可** です。