

Accessible Rich Internet Applications (WAI-ARIA) 1.2

W3C Recommendation 06 June 2023



▼ More details about this document

This version:

<https://www.w3.org/TR/2023/REC-wai-aria-1.2-20230606/>

Latest published version:

<https://www.w3.org/TR/wai-aria-1.2/>

Latest editor's draft:

<https://w3c.github.io/aria/>

History:

<https://www.w3.org/standards/history/wai-aria-1.2>

[Commit history](#)

Implementation report:

<https://w3c.github.io/test-results/core-aam-1.2/>

Previous Recommendation:

<https://www.w3.org/TR/wai-aria-1.1/>

Editors:

Joanmarie Diggs ([Igalia, S.L.](#))

James Nurthen ([Adobe](#))

Michael Cooper ([W3C](#))

Carolyn MacLeod ([IBM](#))

Former editors:

Shane McCarron (Spec-Ops) (Editor until 2018)

Richard Schwerdtfeger ([Knowbility](#)) (Editor until October 2017)

James Craig ([Apple Inc.](#)) (Editor until May 2016)

Feedback:

[GitHub w3c/aria](#) ([pull requests](#), [new issue](#), [open issues](#))

Errata:

[Errata exists.](#)

See also [translations](#).

[Copyright](#) © 2013-2023 [World Wide Web Consortium](#). W3C® [liability](#), [trademark](#) and [permissive document license](#) rules apply.

Abstract

Accessibility of web content requires semantic information about widgets, structures, and behaviors, in order to allow assistive technologies to convey appropriate information to persons with disabilities. This specification provides an ontology of roles, states, and properties that define accessible user interface elements and can be used to improve the accessibility and interoperability of web content and applications. These semantics are designed to allow an author to properly convey user interface behaviors and structural information to assistive technologies in document-level markup. This version adds features new since [WAI-ARIA 1.1](#) [[wai-aria-1.1](#)] to improve interoperability with assistive technologies to form a more consistent accessibility model for [[HTML](#)] and [[SVG2](#)]. This specification complements both [[HTML](#)] and [[SVG2](#)].

This document is part of the [WAI-ARIA](#) suite described in the [WAI-ARIA Overview](#).

Status of This Document

This section describes the status of this document at the time of its publication. A list of current W3C publications and the latest revision of this technical report can be found in the [W3C technical reports index](#) at <https://www.w3.org/TR/>.

WAI-ARIA 1.2 is a W3C Recommendation. The Advisory Committee (AC) as well as the W3C Director have endorsed this specification to become a W3C Recommendation. For details about implementation experience, see the [WAI-ARIA 1.2 Implementation Report](#). A [history of changes to WAI-ARIA 1.2](#) is available in the appendix.

This document was published by the [Accessible Rich Internet Applications Working Group](#) as a Recommendation using the [Recommendation track](#).

W3C recommends the wide deployment of this specification as a standard for the Web.

A W3C Recommendation is a specification that, after extensive consensus-building, is endorsed by W3C and its Members, and has commitments from Working Group members to [royalty-free licensing](#) for implementations.

This document was produced by a group operating under the [W3C Patent Policy](#). W3C maintains a [public list of any patent disclosures](#) made in connection with the deliverables of the group; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual believes contains [Essential Claim\(s\)](#) must disclose the information in accordance with [section 6 of the W3C Patent Policy](#).

This document is governed by the [2 November 2021 W3C Process Document](#).

Table of Contents

Abstract

Status of This Document

1. Introduction

- 1.1 Rich Internet Application Accessibility
- 1.2 Target Audience
- 1.3 User Agent Support
- 1.4 Co-Evolution of WAI-ARIA and Host Languages
- 1.5 Authoring Practices
 - 1.5.1 Authoring Tools
 - 1.5.2 Testing Practices and Tools
- 1.6 Assistive Technologies

2. Important Terms

3. Conformance

- 3.1 Non-interference with the Host Language
- 3.2 All WAI-ARIA in DOM
- 3.3 Assistive Technology Notifications Communicated to Web Applications
- 3.4 Conformance Checkers
- 3.5 Deprecated Requirements

4. Using WAI-ARIA

- 4.1 WAI-ARIA Roles
- 4.2 WAI-ARIA States and Properties
- 4.3 Managing Focus and Supporting Keyboard Navigation
 - 4.3.1 Information for Authors
 - 4.3.2 Information for User Agents

5. The Roles Model

- 5.1 Relationships Between Concepts
 - 5.1.1 Superclass Role
 - 5.1.2 Subclass Roles
 - 5.1.3 Related Concepts
 - 5.1.4 Base Concept
- 5.2 Characteristics of Roles
 - 5.2.1 Abstract Roles
 - 5.2.2 Required States and Properties
 - 5.2.3 Supported States and Properties
 - 5.2.4 Inherited States and Properties

- 5.2.5 Prohibited States and Properties
- 5.2.6 Required Owned Elements
- 5.2.7 Required Context Role
- 5.2.8 Accessible Name Calculation
 - 5.2.8.1 Name Computation
 - 5.2.8.2 Description Computation
 - 5.2.8.3 Accessible Name and Description Computation
 - 5.2.8.4 Roles Supporting Name from Author
 - 5.2.8.5 Roles Supporting Name from Content
 - 5.2.8.6 Roles which cannot be named (Name prohibited)
- 5.2.9 Presentational Children
- 5.2.10 Implicit Value for Role
- 5.3 Categorization of Roles
 - 5.3.1 Abstract Roles
 - 5.3.2 Widget Roles
 - 5.3.3 Document Structure Roles
 - 5.3.4 Landmark Roles
 - 5.3.5 Live Region Roles
 - 5.3.6 Window Roles
- 5.4 Definition of Roles

6. Supported States and Properties

- 6.1 Clarification of States versus Properties
- 6.2 Characteristics of States and Properties
 - 6.2.1 Related Concepts
 - 6.2.2 Used in Roles
 - 6.2.3 Inherits into Roles
 - 6.2.4 Value
- 6.3 ARIA Attributes
 - 6.3.1 Multi-value Attribute Values
 - 6.3.2 IDL reflection of ARIA attributes
 - 6.3.3 Operating System Accessibility API mapping of multi-value ARIA attributes
 - 6.3.4 ARIA nullable DOMString Attributes
 - 6.3.4.1 Example Attribute Usage
- 6.4 Translatable States and Properties
- 6.5 Global States and Properties
- 6.6 Taxonomy of WAI-ARIA States and Properties
 - 6.6.1 Widget Attributes
 - 6.6.2 Live Region Attributes
 - 6.6.3 Drag-and-Drop Attributes
 - 6.6.4 Relationship Attributes

6.7 Definitions of States and Properties (all aria-* attributes)

7. Accessibility Tree

7.1 Excluding Elements from the Accessibility Tree

7.2 Including Elements in the Accessibility Tree

8. Implementation in Host Languages

8.1 Role Attribute

8.2 State and Property Attributes

8.3 Focus Navigation

8.4 Implicit WAI-ARIA Semantics

8.5 Conflicts with Host Language Semantics

8.6 State and Property Attribute Processing

8.6.1 ID Reference Error Processing

8.7 CSS Selectors

9. Handling Author Errors

9.1 Roles

9.2 States and Properties

10. IDL Interface

10.1 Interface Mixin ARIAMixin

10.2 ARIA Attribute Correspondence

10.2.1 Disambiguation Pattern

10.2.2 IDL Attribute Name Notes or Exceptions

10.3 ARIAMixin Mixed in to Element

10.4 Example IDL Attribute Usage

11. Privacy and Security Considerations

A. Mapping WAI-ARIA Value types to languages

B. Substantive changes since the WAI-ARIA 1.1 Recommendation

C. Acknowledgments

C.1 Participants active in the ARIA WG at the time of publication

C.2 Other ARIA contributors, commenters, and previously active participants

C.3 Enabling funders

D. References

D.1 Normative references

D.2 Informative references

§ Dedication

This version of the ARIA specification is dedicated to the memory of Carolyn MacLeod whose contributions are found throughout this document. She graced our work with equanimity and sagacity, and her untimely passing will long be missed by our community.

§ 1. Introduction

This section is non-normative.

The goals of this specification include:

- expanding the accessibility information that may be supplied by the author;
- requiring that supporting host languages provide full keyboard support that may be implemented in a device-independent way, for example, by telephones, handheld devices, e-book readers, and televisions;
- improving the accessibility of dynamic content generated by scripts; and
- providing for interoperability with [assistive technologies](#).

WAI-ARIA is a technical specification that provides a framework to improve the accessibility and interoperability of web content and applications. This document is primarily for developers creating custom widgets and other web application components. Please see the [WAI-ARIA Overview](#) for links to related documents for other audiences, such as [WAI-ARIA Authoring Practices](#) [WAI-ARIA-PRACTICES-1.2] that introduces developers to the accessibility problems that WAI-ARIA is intended to solve, the fundamental concepts, and the technical approach of WAI-ARIA.

This document currently handles two aspects of [roles](#): user interface functionality and structural [relationships](#). For more information and use cases, see [WAI-ARIA Authoring Practices](#) [WAI-ARIA-PRACTICES-1.2] for the use of roles in making interactive content accessible.

Roles defined by this specification are designed to support the roles used by platform [accessibility APIs](#). Declaration of these roles on elements within dynamic web content is intended to support interoperability between the web content and assistive technologies that utilize [accessibility APIs](#).

The schema to support this standard has been designed to be extensible so that custom roles can be created by extending base roles. This allows [user agents](#) to support at least the base role, and user agents that support the custom role can provide enhanced access. Note that much of this could be formalized in [\[XMLSCHEMA11-2\]](#). However, being able to define similarities between roles, such as [baseConcepts](#) and

more descriptive definitions, would not be available in XSD.

WAI-ARIA 1.2 is a member of the [WAI-ARIA 1.2 suite](#) that defines how to expose semantics of WAI-ARIA and other web content languages to [accessibility APIs](#).

§ 1.1 Rich Internet Application Accessibility

The domain of web accessibility defines how to make web content usable by persons with disabilities.

Persons with certain types of disabilities use [assistive technologies](#) (AT) to interact with content. Assistive technologies can transform the presentation of content into a format more suitable to the user, and can allow the user to interact in different ways. For example, the user may need to, or choose to, interact with a slider widget via arrow keys, instead of dragging and dropping with a mouse. In order to accomplish this effectively, the software needs to understand the [semantics](#) of the content. Semantics is the science of meaning; in this case, used to assign roles, states, and properties that apply to user interface and content elements as a human would understand. For instance, if a paragraph is semantically identified as such, assistive technologies can interact with it as a unit separable from the rest of the content, knowing the exact boundaries of that paragraph. An adjustable range slider or collapsible list (a.k.a. a tree [widget](#)) are more complex examples, in which various parts of the widget have semantics that need to be properly identified for assistive technologies to support effective interaction.

New technologies often overlook semantics required for accessibility, and new authoring practices often misuse the intended semantics of those technologies. [Elements](#) that have one defined meaning in the language are used with a different meaning intended to be understood by the user.

For example, web application developers create collapsible tree widgets in [HTML](#) using CSS and JavaScript even though [HTML](#) has no semantic [tree](#) element. To a non-disabled user, it may look and act like a collapsible tree widget, but without appropriate semantics, the tree widget may not be [perceivable](#) to, or [operable](#) by, a person with a disability because assistive technologies may not recognize the role. Similarly, web application developers create interactive button widgets in [SVG](#) using JavaScript even though [SVG](#) has no semantic [button](#) element. To a non-disabled user, it may look and act like a button widget, but without appropriate semantics, the button widget may not be [perceivable](#) to, or [operable](#) by, a person with a disability because assistive technologies may not recognize the role.

The incorporation of [WAI-ARIA](#) is a way for an author to provide proper semantics for custom widgets to make these widgets accessible, usable, and interoperable with assistive technologies. This specification identifies the types of widgets and structures that are commonly recognized by accessibility products, by providing an [ontology](#) of corresponding [roles](#) that can be attached to content. This allows elements with a given role to be understood as a particular widget or structural type regardless of any semantics inherited from the implementing host language. Roles are a common property of platform [accessibility APIs](#) which assistive technologies use to provide the user with effective presentation and interaction.

The Roles Model includes interaction [widgets](#) and elements denoting document structure. The Roles Model describes inheritance and details the [attributes](#) each role supports. Information about mapping of roles to accessibility [APIs](#) is provided by the [Core Accessibility API Mappings](#) [CORE-AAM-1.2].

Roles are element types and will not change with time or user actions. Role information is used by assistive technologies, through interaction with the user agent, to provide normal processing of the specified element type.

States and properties are used to declare important attributes of an element that affect and describe interaction. They enable the [user agent](#) and operating system to properly handle the element even when the attributes are dynamically changed by client-side scripts. For example, alternative input and output technology, such as screen readers and speech dictation software, need to be able to recognize and effectively manipulate and communicate various interaction states (e.g., disabled, checked) to the user.

While it is possible for assistive technologies to access these properties directly through the [Document Object Model](#) [DOM], the preferred mechanism is for the user agent to map the states and properties to the accessibility [API](#) of the operating system. See the [Core Accessibility API Mappings](#) [CORE-AAM-1.2] and the [Accessible Name and Description Computation](#) [ACCNAME-1.2] for details.

Figure 1.0 illustrates the relationship between user agents (e.g., browsers), accessibility [APIs](#), and assistive technologies. It describes the "contract" provided by the user agent to assistive technologies, which includes typical accessibility information found in the accessibility [API](#) for many of our accessible platforms for GUIs (role, state, selection, [event](#) notification, [relationship](#) information, and descriptions). The [DOM](#), usually [HTML](#), acts as the data model and view in a typical model-view-controller relationship, and JavaScript acts as the controller by manipulating the style and content of the displayed data. The user agent conveys relevant information to the operating system's accessibility [API](#), which can be used by any assistive technologies, such as screen readers.

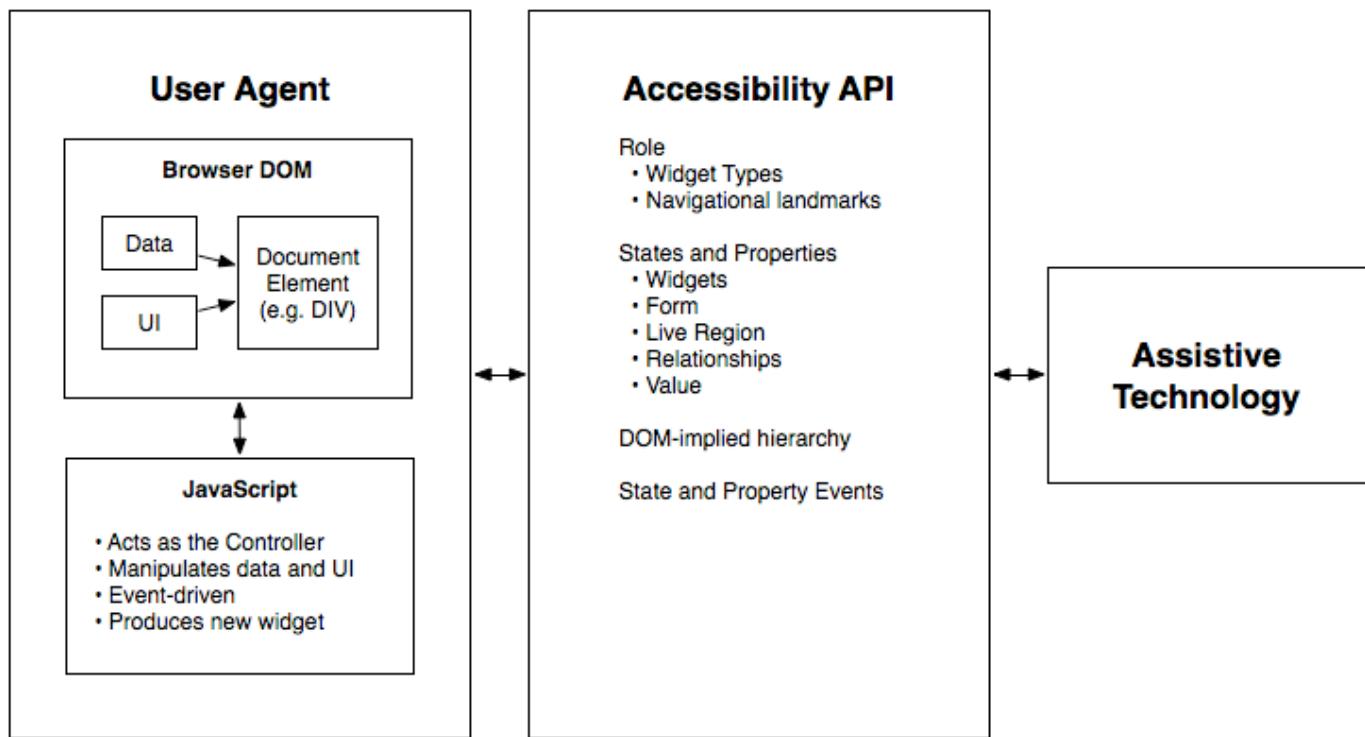


Figure 1: The contract model with accessibility APIs

For more information see [WAI-ARIA Authoring Practices](#) for the use of roles in making interactive content accessible.

Users of alternate input devices need [keyboard accessible](#) content. The new semantics, when combined with the recommended keyboard interactions provided in [WAI-ARIA Authoring Practices](#), will allow alternate input solutions to facilitate command and control via an alternate input solution.

WAI-ARIA introduces navigational [landmarks](#) through its Roles Model and the [XHTML](#) role landmarks, which can help persons with dexterity and vision impairments by providing for improved keyboard navigation. WAI-ARIA may also be used to assist persons with cognitive learning disabilities. The additional semantics allow authors to restructure and substitute alternative content as needed.

[Assistive technologies](#) need the ability to support alternative inputs by getting and setting the current value of [widget](#) states and properties. Assistive technologies also need to determine what [objects](#) are selected and manage widgets that allow multiple selections, such as list boxes and grids.

Speech-based command and control systems can benefit from WAI-ARIA semantics like the [role](#) attribute to assist in conveying audio information to the user. For example, upon encountering an element with a role of [menu](#) with child elements of role [menuitem](#) each containing text content representing a different flavor, a speech system might state to the user, "Select one of three choices: chocolate, strawberry, or vanilla."

WAI-ARIA is intended to be used as a supplement for native language semantics, not a replacement. When the host language provides a feature that provides equivalent accessibility to the WAI-ARIA feature, use the host language feature. WAI-ARIA should only be used in cases where the host language lacks the needed

role, state, and property indicators. Use a host language feature that is as similar as possible to the WAI-ARIA feature, then refine the meaning by adding WAI-ARIA. For instance, a multi-selectable grid could be implemented as a table, and then WAI-ARIA used to clarify that it is an interactive grid, not just a static data table. This allows for the best possible fallback for user agents that do not support WAI-ARIA and preserves the integrity of the host language semantics.

1.2 Target Audience

This specification defines the basic model for WAI-ARIA, including roles, states, properties, and values. It impacts several audiences:

- User agents that process content containing WAI-ARIA features;
- Assistive technologies that present content in special ways to user with disabilities;
- Authors who create content;
- Authoring tools that help authors create conforming content; and
- Conformance checkers that verify appropriate use of WAI-ARIA.

Each conformance requirement indicates the audience to which it applies.

Although this specification is applicable to the above audiences, it is not specifically targeted to, nor is it intended to be the sole source of information for, any of these audiences. The following documents provide important supporting information:

- [WAI-ARIA-PRACTICES-1.2] addresses authoring recommendations for HTML, and is also of interest to developers of authoring tools and conformance checkers.
- [CORE-AAM-1.2] addresses developers of user agents and assistive technologies.
- [ACCNAME-1.2] also addresses developers of user agents and assistive technologies.

1.3 User Agent Support

WAI-ARIA relies on user agent support for its features in two ways:

- Mainstream user agents use WAI-ARIA to alter how host language features are exposed to accessibility APIs in order to improve accessibility. The mechanism for this is defined in the Core Accessibility API Mappings.

- Assistive technologies use the enhanced information available in an accessibility API, or uses the WAI-ARIA markup directly via the DOM, to convey semantic and interaction information to the user.

Aside from using WAI-ARIA markup to improve what is exposed to accessibility APIs, user agents behave as they would natively. Assistive technologies react to the extra information in the accessibility API as they already do for the same information on non-web content. User agents that are not assistive technologies, however, need do nothing beyond providing appropriate updates to the accessibility API.

The WAI-ARIA specification neither requires nor forbids user agents from enhancing native presentation and interaction behaviors on the basis of WAI-ARIA markup. Mainstream user agents might expose WAI-ARIA navigational landmarks (for example, as a dialog box or through a keyboard command) with the intention to facilitate navigation for all users. User agents are encouraged to maximize their usefulness to users, including users without disabilities.

WAI-ARIA is intended to provide missing semantics so that the intent of the author may be conveyed to assistive technologies. Generally, authors using WAI-ARIA will provide the appropriate presentation and interaction features. Over time, host languages may add WAI-ARIA equivalents, such as new form controls, that are implemented as standard accessible user interface controls by the user agent. This allows authors to use them instead of custom WAI-ARIA enabled user interface components. In this case the user agent would support the native host language feature. Developers of host languages that implement WAI-ARIA are advised to continue supporting WAI-ARIA semantics when they do not adversely conflict with implicit host language semantics, as WAI-ARIA semantics more clearly reflect the intent of the author if the host language features are inadequate to meet the author's needs.

1.4 Co-Evolution of WAI-ARIA and Host Languages

WAI-ARIA is intended to augment semantics in supporting languages like [HTML] and [SVG2], or to be used as an accessibility enhancement technology in other markup-based languages that do not explicitly include support for ARIA. It clarifies semantics to assistive technologies when authors create new types of objects, via style and script, that are not yet directly supported by the language of the page, because the invention of new types of objects is faster than standardized support for them appears in web languages.

It is not appropriate to create objects with style and script when the host language provides a semantic element for that type of object. While WAI-ARIA can improve the accessibility of these objects, accessibility is best provided by allowing the user agent to handle the object natively. For example, it's better to use an h1 element in HTML than to use the heading role on a div element.

It is expected that, over time, host languages will evolve to provide semantics for objects that currently can only be declared with WAI-ARIA. This is natural and desirable, as one goal of WAI-ARIA is to help stimulate the emergence of more semantic and accessible markup. When native semantics for a given feature become available, it is appropriate for authors to use the native feature and stop using WAI-ARIA for that

feature. Legacy content may continue to use WAI-ARIA, however, so the need for user agents to support WAI-ARIA remains.

While specific features of WAI-ARIA may lose importance over time, the general possibility of WAI-ARIA to add semantics to web pages is expected to be a persistent need. Host languages may not implement all the semantics WAI-ARIA provides, and various host languages may implement different subsets of the features. New types of objects are continually being developed, and one goal of WAI-ARIA is to provide a way to make such objects accessible, because web authoring practices often advance faster than host language standards. In this way, WAI-ARIA and host languages both evolve together but at different rates.

Some host languages exist to create semantics for features other than the user interface. For example, SVG expresses the semantics behind production of graphical objects, not of user interface components that those objects may represent. Host languages might, by design, not provide native semantics that map to WAI-ARIA features. In these cases, WAI-ARIA could be adopted as a long-term approach to add semantic information to user interface components.

1.5 Authoring Practices

1.5.1 Authoring Tools

Many of the requirements in the definitions of WAI-ARIA [roles](#), [states](#), and [properties](#) can be checked automatically during the development process, similar to other quality control processes used for validating code. To assist authors who are creating custom widgets, authoring tools may compare widget roles, states, and properties to those supported in WAI-ARIA as well as those supported in related and cross-referenced roles, states, and properties. Authoring tools may notify authors of errors in widget design patterns, and may also prompt developers for information that cannot be determined from context alone. For example, a scripting library can determine the labels for the tree items in a tree view, but would need to prompt the author to label the entire tree. To help authors visualize a logical accessibility structure, an authoring environment might provide an outline view of a web resource based on the WAI-ARIA markup.

In both HTML and SVG, `tabindex` is an important way browsers support keyboard [focus navigation](#) for implementations of WAI-ARIA; authoring and debugging tools may check to make sure `tabindex` values are properly set. For example, error conditions may include cases where more than one treeitem in a tree has a `tabindex` value greater than or equal to 0, where `tabindex` is not set on any treeitem, or where [aria-activedescendant](#) is not defined when the element with the role tree has a `tabindex` value of greater than or equal to 0.

§ 1.5.2 Testing Practices and Tools

The accessibility of interactive content cannot be confirmed by static checks alone. Developers of interactive content should test for device-independent access to [widgets](#) and applications, and should verify accessibility API access to all content and changes during user interaction.

§ 1.6 Assistive Technologies

Programmatic access to accessibility semantics is essential for assistive technologies. Most assistive technologies interact with user agents, like other applications, through a recognized accessibility [API](#). Perceivable objects in the user interface are exposed to assistive technologies as accessible objects, defined by the accessibility [API](#) interfaces. To do this properly, accessibility information – role, states, properties as well as contextual information – needs to be accurately conveyed to the assistive technologies through the accessibility [API](#). When a state change occurs, the user agent provides the appropriate event notification to the accessibility [API](#). Contextual information, in many host languages like [HTML](#), can be determined from the [DOM](#) itself as it provides a contextual tree hierarchy.

While some assistive technologies interact with these accessibility [APIs](#), others may access the content directly from the [DOM](#). These technologies can restructure, simplify, style, or reflow the content to help a different set of users. Common use cases for these types of adaptations may be the aging population, persons with cognitive impairments, or persons in environments that interfere with use of their tools. For example, the availability of regional navigational landmarks may allow for a mobile device adaptation that shows only portions of the content at any one time based on its semantics. This could reduce the amount of information the user needs to process at any one time. In other situations it may be appropriate to replace a custom user interface control with something that is easier to navigate with a keyboard, or touch screen device.

§ 2. Important Terms

This section is non-normative.

While some terms are defined in place, the following definitions are used throughout this document.

Accessibility API

Operating systems and other platforms provide a set of interfaces that expose information about [objects](#) and [events](#) to [assistive technologies](#). Assistive technologies use these interfaces to get information about and interact with those [widgets](#). Examples of accessibility [APIs](#) are [Microsoft Active Accessibility](#) [MSAA], [Microsoft User Interface Automation](#) [UI-AUTOMATION], MSAA with [UIA Express](#) [UIA-EXPRESS], the [Mac OS X Accessibility Protocol](#) [AXAPI], the [Linux/Unix Accessibility Toolkit](#) [ATK]

and [Assistive Technology Service Provider Interface](#) [AT-SPI], and [IAccessible2](#) [IAccessible2].

Accessibility Subtree

An [accessible object](#) in the [accessibility tree](#) and its descendants in that tree. It does not include objects which have relationships other than parent-child in that tree. For example, it does not include objects linked via [aria-flowto](#) unless those objects are also descendants in the [accessibility tree](#).

Accessibility Tree

Tree of [accessible objects](#) that represents the structure of the user interface (UI). Each node in the accessibility tree represents an element in the UI as exposed through the [accessibility API](#); for example, a push button, a check box, or container.

Accessible Description

An accessible description provides additional information, related to an interface element, that complements the [accessible name](#). The accessible description might or might not be visually perceivable.

Accessible Name

The accessible name is the name of a user interface element. Each platform [accessibility API](#) provides the accessible name property. The value of the accessible name may be derived from a visible (e.g., the visible text on a button) or invisible (e.g., the text alternative that describes an icon) property of the user interface element. See related [accessible description](#).

A simple use for the accessible name property may be illustrated by an "OK" button. The text "OK" is the accessible name. When the button receives focus, assistive technologies may concatenate the platform's role description with the accessible name. For example, a screen reader may speak "push-button OK" or "OK button". The order of concatenation and specifics of the role description (e.g., "button", "push-button", "clickable button") are determined by platform [accessibility APIs](#) or [assistive technologies](#).

Accessible object

A [node](#) in the [accessibility tree](#) of a platform [accessibility API](#). Accessible objects expose various [states](#), [properties](#), and [events](#) for use by [assistive technologies](#). In the context of markup languages (e.g., HTML and SVG) in general, and of WAI-ARIA in particular, markup [elements](#) and their [attributes](#) are represented as accessible objects.

Activation behavior

The action taken when an [event](#), typically initiated by users through an input device, causes an element to fulfill a defined role. The role may be defined for that element by the host language, or by author-defined variables, or both. The role for any given element may be a generic action, or may be unique to that element. For example, the activation behavior of an HTML or SVG [`<a>`](#) element shall be to cause the user agent to traverse the link specified in the [`href`](#) attribute, with the further optional parameter of specifying the browsing context for the traversal (such as the current window or tab, a named window, or a new window); the activation behavior of an HTML [`<input>`](#) element with the [`type`](#) attribute value [`submit`](#) shall be to send the values of the form elements to an author-defined IRI by the author-defined

HTTP method.

Assistive Technologies

Hardware and/or software that:

- relies on services provided by a user agent to retrieve and render Web content
- works with a user agent or web content itself through the use of APIs, and
- provides services beyond those offered by the user agent to facilitate user interaction with web content by people with disabilities

This definition may differ from that used in other documents.

Examples of assistive technologies that are important in the context of this document include the following:

- screen magnifiers, which are used to enlarge and improve the visual readability of rendered text and images;
- screen readers, which are most-often used to convey information through synthesized speech or a refreshable Braille display;
- text-to-speech software, which is used to convert text into synthetic speech;
- speech recognition software, which is used to allow spoken control and dictation;
- alternate input technologies (including head pointers, on-screen keyboards, single switches, and sip/puff devices), which are used to simulate the keyboard;
- alternate pointing devices, which are used to simulate mouse pointing and clicking.

Attribute

In this specification, attribute is used as it is in markup languages. Attributes are structural features added to *elements* to provide information about the *states* and *properties* of the *object* represented by the element.

Class

A set of instance *objects* that share similar characteristics.

Deprecated

A deprecated *role*, *state*, or *property* is one which has been outdated by newer constructs or changed circumstances, and which may be removed in future versions of the WAI-ARIA specification. User agents are encouraged to continue to support items identified as deprecated for backward compatibility. For more information, see Deprecated Requirements in the Conformance section.

Desktop focus event

Event from/to the host operating system via the accessibility API, notifying of a change of input focus.

DOMString

Sequence of 16-bit unsigned integers, typically interpreted as UTF-16 code units. This corresponds to the JavaScript primitive String type.

Element

In this specification, element is used as it is in markup languages. Elements are the structural elements in markup language that contains the data profile for *objects*.

Event

A programmatic message used to communicate discrete changes in the *state* of an *object* to other objects in a computational system. User input to a web page is commonly mediated through abstract events that describe the interaction and can provide notice of changes to the state of a document object. In some programming languages, events are more commonly known as notifications.

Expose

Translated to platform-specific *accessibility APIs* as defined in the [Core Accessibility API Mappings](#).

Graphical Document

A document containing graphic representations with user-navigable parts. Charts, maps, diagrams, blueprints, and dashboards are examples of graphical documents. A graphical document is composed using any combination of symbols, images, text, and graphic primitives (shapes such as circles, points, lines, paths, rectangles, etc).

Hidden

Indicates that the *element* is not visible, *perceivable*, or interactive to *any* user. An element is considered *hidden* if it or any one of its ancestor elements is not rendered or is explicitly hidden.

Informative

Content provided for information purposes and not required for conformance. Content required for conformance is referred to as *normative*.

Keyboard Accessible

Accessible to the user using a keyboard or *assistive technologies* that mimic keyboard input, such as a sip and puff tube. References in this document relate to [WCAG 2.1 Guideline 2.1: Make all functionality available from a keyboard](#) [WCAG21].

Landmark

A type of region on a page to which the user may want quick access. Content in such a region is different from that of other regions on the page and relevant to a specific user purpose, such as navigating, searching, perusing the primary content, etc.

Live Region

Live regions are perceivable regions of a web page that are typically updated as a result of an external event when user focus may be elsewhere. These regions are not always updated as a result of a user interaction. Examples of live regions include a chat log, stock ticker, or a sport scoring section that updates periodically to reflect game statistics. Since these asynchronous areas are expected to update

outside the user's area of focus, assistive technologies such as screen readers have either been unaware of their existence or unable to process them for the user. WAI-ARIA has provided a collection of properties that allow the author to identify these live regions and process them: aria-live, aria-relevant, aria-atomic, and aria-busy.

Primary Content Element

An implementing host language's primary content element, such as the `body` element in [HTML](#).

Managed State

[Accessibility API state](#) that is controlled by the user agent, such as focus and selection. These are contrasted with "unmanaged states" that are typically controlled by the author. Nevertheless, authors can override some managed states, such as `aria-posinset` and `aria-setsize`. Many managed states have corresponding [CSS](#) pseudo-classes, such as `:focus`, and pseudo-elements, such as `::selection`, that are also updated by the user agent.

Nemeth Braille

The Nemeth Braille Code for Mathematics is a braille code for encoding mathematical and scientific notation. See [Nemeth Braille on Wikipedia](#).

Node

Basic type of [object](#) in the [DOM](#) tree or [accessibility tree](#). [DOM](#) nodes are further specified as [Element](#) or [Text nodes](#), among other types. The nodes of an [accessibility tree](#) are [accessible objects](#).

Normative

Required for conformance. By contrast, content identified as [informative](#) or "non-normative" is not required for conformance.

Object

In the context of user interfaces, an item in the perceptual user experience, represented in markup languages by one or more [elements](#), and rendered by [user agents](#).

In the context of programming, the instantiation of one or more [classes](#) and interfaces which define the general characteristics of similar objects. An object in an [accessibility API](#) may represent one or more [DOM](#) objects. [Accessibility APIs](#) have defined interfaces that are distinct from [DOM](#) interfaces.

Ontology

A description of the characteristics of [classes](#) and how they relate to each other.

Operable

Usable by users in ways they can control. References in this document relate to [WCAG 2.1 Principle 2: Content must be operable](#) [WCAG21]. See [Keyboard Accessible](#).

Owned Element

An 'owned element' is any [DOM](#) descendant of the [element](#), any element specified as a child via [aria-owns](#), or any [DOM](#) descendant of the owned child.

Owning Element

An 'owning element' is any DOM ancestor of the element, or any element with an aria-owns attribute which references the ID of the element.

Perceivable

Presentable to users in ways they can sense. References in this document relate to WCAG 2.1 Principle 1: Content must be perceivable [WCAG21].

Property

Attributes that are essential to the nature of a given object, or that represent a data value associated with the object. A change of a property may significantly impact the meaning or presentation of an object. Certain properties (for example, aria-multiline) are less likely to change than states, but note that the frequency of change difference is not a rule. A few properties, such as aria-activedescendant, aria-valuenow, and aria-valuetext are expected to change often. See clarification of states versus properties.

Relationship

A connection between two distinct things. Relationships may be of various types to indicate which object labels another, controls another, etc.

Role

Main indicator of type. This semantic association allows tools to present and support interaction with the object in a manner that is consistent with user expectations about other objects of that type.

Root WAI-ARIA node

The primary element containing non-metadata content. In many languages, this is the document element but in HTML, it is the <body>.

Semantics

The meaning of something as understood by a human, defined in a way that computers can process a representation of an object, such as elements and attributes, and reliably represent the object in a way that various humans will achieve a mutually consistent understanding of the object.

State

A state is a dynamic property expressing characteristics of an object that may change in response to user action or automated processes. States do not affect the essential nature of the object, but represent data associated with the object or user interaction possibilities. See clarification of states versus properties.

Sub-document

Any document created from a <frame>, <iframe> or similar mechanism. A sub-document may contain a document, an application or any widget such as a calendar pulled in from another server. In the accessibility tree there are two accessible objects for this situation—one represents the <frame>/<iframe> element in the parent document, which parents a single accessible object child representing the spawned document contents.

Target Element

An element specified in a WAI-ARIA relation. For example, in <div aria-controls="elem1">,

where “elem1” is the ID for the target element.

Taxonomy

A hierarchical definition of how the characteristics of various *classes* relate to each other, in which classes inherit the properties of superclasses in the hierarchy. A taxonomy can comprise part of the formal definition of an *ontology*.

Text node

Type of DOM *node* that represents the textual content of an *attribute* or an *element*. A Text node has no child nodes.

Tooltip attribute

Any host language attribute that would result in a user agent generating a tooltip such as in response to a mouse hover in desktop user agents.

Understandable

Presentable to users in ways they can construct an appropriate meaning. References in this document relate to [WCAG 2.1 Principle 3: Information and the operation of user interface must be understandable](#) [WCAG21].

Unicode Braille Patterns

In Unicode, braille is represented in a block called Braille Patterns (U+2800..U+28FF). The block contains all 256 possible patterns of an 8-dot braille cell; this includes the complete 6-dot cell range which is represented by U+2800..U+283F. In all braille systems, the braille pattern dots-0 (U+2800) is used to represent a space or the lack of content; it is also called a blank Braille pattern. See [Braille Patterns on Wikipedia](#).

User Agent

Any software that retrieves, renders and facilitates end user interaction with Web content. This definition may differ from that used in other documents.

Valid IDREF

A reference to a target element in the same document that has a matching ID

Widget

Discrete user interface *object* with which the user can interact. Widgets range from simple objects that have one value or operation (e.g., check boxes and menu items), to complex objects that contain many managed sub-objects (e.g., trees and grids).

3. Conformance

The main content of Accessible Rich Internet Applications is *normative* and defines requirements that impact conformance claims. Introductory material, appendices, sections marked as "non-normative" and their

subsections, diagrams, examples, and notes are [informative](#) (non-normative). Non-normative material provides advisory information to help interpret the guidelines but does not create requirements that impact a conformance claim.

Normative sections provide requirements that [user agents](#) must follow for an implementation to conform to this specification. The keywords **MUST**, **MUST NOT**, **REQUIRED**, **SHALL**, **SHALL NOT**, **SHOULD**, **RECOMMENDED**, **MAY**, and **OPTIONAL** in this document are to be interpreted as described in [Keywords for use in RFCs to indicate requirement levels](#) [RFC2119]. RFC-2119 keywords are formatted in uppercase and contained in an element with `class="rfc2119"`. When the keywords shown above are used, but do not share this format, they do not convey formal information in the RFC 2119 sense, and are merely explanatory, i.e., informative. As much as possible, such usages are avoided in this specification.

Normative sections provide requirements that authors, user agents and assistive technologies **MUST** follow for an implementation to conform to this specification.

Non-normative (informative) sections provide information useful to understanding the specification. Such sections may contain examples of recommended practice, but it is not required to follow such recommendations in order to conform to this specification.

§ 3.1 Non-interference with the Host Language

WAI-ARIA processing by the [user agent](#) **MUST NOT** interfere with the normal operation of the built-in features of the host language.

If a CSS selector includes a WAI-ARIA attribute (e.g.,

```
input[aria-invalid="true"]
```

), user agents **MUST** update the visual display of any elements matching (or no longer matching) the selector any time the attribute is added/changed/removed in the [DOM](#). The user agent **MAY** alter the mapping of the host language features into an [accessibility API](#), but the user agent **MUST NOT** alter the [DOM](#) in order to remap WAI-ARIA markup into host language features.

§ 3.2 All WAI-ARIA in DOM

A conforming [user agent](#) which implements a document object model that does not conform to the [W3C DOM](#) specification **MUST** include the content attribute for role and its [WAI-ARIA role values](#), as well as the [WAI-ARIA States and Properties](#) in the [DOM](#) as specified by the author, even though processing may affect how the elements are exposed to accessibility APIs. Doing so ensures that each role attribute and all WAI-ARIA states and properties, including their values, are in the document in an unmodified form so other tools,

such as assistive technologies, can access them. A conforming W3C DOM meets this criterion.

3.3 Assistive Technology Notifications Communicated to Web Applications

Assistive technologies, such as speech recognition systems and alternate input devices for users with mobility impairments, require the ability to control a web application in a device-independent way. WAI-ARIA states and properties reflect the current state of rich internet application components. The ability for assistive technologies to notify web applications of necessary changes is essential because it allows these alternative input solutions to control an application without being dependent on the standard input device which the user is unable to effectively control directly.

User agents **MUST** provide a method to notify the web application when a change occurs to states or properties in the system accessibility API. Likewise, web application authors **SHOULD** update the web application accordingly when notified of a change request from the user agent or assistive technology.

NOTE

Many state and properties can be changed by assistive technologies through existing accessibility APIs by responding to a default action event. For example, the `aria-selected` state of a `tab` in a `tabpanel` can be changed by triggering the default action on the element.

3.4 Conformance Checkers

Any application or script verifying document conformance or validity **SHOULD** include a test for all of the normative author requirements in this specification. If testing for a given requirement, conformance checkers **MUST** issue an error if an author "**MUST**" requirement isn't met, and **MUST** issue a warning if an author "**SHOULD**" requirement isn't met.

3.5 Deprecated Requirements

As the technology evolves, sometimes new ways to meet a use case become available, that work better than a feature that was previously defined. But because of existing implementation of the older feature, that feature cannot be removed from the conformance model without rendering formerly conforming content non-conforming. In this case, the older feature is marked as "deprecated". This indicates that the feature is allowed in the conformance model and expected to be supported by user agents, but it is recommended that

authors do not use it for new content. In future versions of the specification, if the feature is no longer widely used, the feature could be removed and no longer expected to be supported by user agents.

4. Using WAI-ARIA

Complex web applications become inaccessible when [assistive technologies](#) cannot determine the [semantics](#) behind portions of a document or when the user is unable to effectively navigate to all parts of it in a usable way (see [WAI-ARIA Authoring Practices](#)). WAI-ARIA divides the semantics into [roles](#) (the type defining a user interface element) and [states](#) and [properties](#) supported by the roles.

Authors need to associate [elements](#) in the document to a [WAI-ARIA role](#) and the appropriate states and properties ([aria-* attributes](#)) during its life-cycle, unless the elements already have the appropriate [implicit WAI-ARIA semantics](#) for states and properties. In these instances the equivalent host language states and properties take precedence to avoid a conflict while the role attribute will take precedence over the implicit role of the host language element.

4.1 WAI-ARIA Roles

A [WAI-ARIA role](#) is set on an [element](#) using a [role attribute](#), similar to the [role attribute](#) defined in [Role Attribute \[ROLE-ATTRIBUTE\]](#).

EXAMPLE 1

```
<li role="menutem">Open file...</li>
```

The definition of each role in the model provides the following information :

- an informative description of the role;
- hierarchical information about related roles (e.g., a [searchbox](#) is a type of [textbox](#));
- context of the role (e.g., a [listitem](#) is contained inside a [list](#));
- references to related concepts in other specifications;
- supported [states](#) and [properties](#) for each role (e.g., a [checkbox](#) supports being checked via [aria-checked](#)).

Attaching a role gives [assistive technologies](#) information about how to handle each element. When WAI-

ARIA roles override host language semantics, there are no changes in the DOM, only in the *accessibility tree*.

User agents **MUST** use the first token in the sequence of tokens in the `role` attribute value that matches the name of any non-abstract WAI-ARIA role. The following steps will correctly identify the applicable WAI-ARIA role:

1. Use the rules of the host language to detect that an element has a role attribute and to identify the attribute value string for it.
2. Separate the attribute value string for that attribute into a sequence of whitespace-free substrings by separating on whitespace.
3. Compare the substrings to all the names of the non-abstract WAI-ARIA roles. Case-sensitivity of the comparison inherits from the case-sensitivity of the host language.
4. Use the first such substring in textual order that matches the name of a non-abstract WAI-ARIA role.

§ 4.2 WAI-ARIA States and Properties

WAI-ARIA provides a collection of accessibility states and properties which are used to support platform accessibility APIs on various operating system platforms. Assistive technologies may access this information through an exposed user agent DOM or through a mapping to the platform accessibility API. When combined with roles, the user agent can supply the assistive technologies with user interface information to convey to the user at any time. Changes in states or properties will result in a notification to assistive technologies, which could alert the user that a change has occurred.

In the following example, a list item (`html:li`) has been used to create a checkable menu item, and JavaScript events will capture mouse and keyboard events to toggle the value of `aria-checked`. A role is used to make the behavior of this simple widget known to the user agent. Attributes that change with user actions (such as `aria-checked`) are defined in the *states and properties* section.

EXAMPLE 2

```
<li role="menuitemcheckbox" aria-checked="true">Sort by Last Modified</li>
```

Some accessibility states, called *managed states*, are controlled by the user agent. Examples of managed state include keyboard focus and selection. Managed states often have corresponding CSS pseudo-classes (such as `:focus` and `::selection`) to define style changes. In contrast, the states in this specification are typically controlled by the author and are called *unmanaged states*. Some states are managed by the user agent, such as `aria-posinset` and `aria-setsize`, but the author can override them if the DOM is incomplete and would

cause the user agent calculation to be incorrect. User agents map both managed and unmanaged states to the platform accessibility APIs.

Most modern user agents support [CSS attribute selectors](#) ([CSS3-SELECTORS]), and can allow the author to create UI changes based on WAI-ARIA attribute information, reducing the amount of scripts necessary to achieve equivalent functionality. In the following example, a CSS selector is used to determine whether or not the text is bold and an image of a check mark is shown, based on the value of the [aria-checked](#) attribute.

EXAMPLE 3

```
[aria-checked="true"] { font-weight: bold; }
[aria-checked="true"]::before { background-image: url(checked.gif); }
```

If CSS is not used to toggle the visual representation of the check mark, the author could include additional markup and scripts to manage an image that represents whether or not the [menuitemcheckbox](#) is checked.

EXAMPLE 4

```
<li role="menuitemcheckbox" aria-checked="true">
  
  <!-- note: additional scripts required to toggle image source -->
  Sort by Last Modified
</li>
```

§ 4.3 Managing Focus and Supporting Keyboard Navigation

When using standard [HTML](#) interactive elements and simple [WAI-ARIA widgets](#), application developers can manipulate the tab order or associate keyboard shortcuts with elements in the document.

WAI-ARIA includes a number of "managing container" widgets, also known as "composite" widgets. When appropriate, the container is responsible for tracking the last descendant that was active (the default is usually the first item in the container). It is essential that a container maintain a usable and consistent strategy when focus leaves a container and is then later refocused. While there may be exceptions, it is recommended that when a previously focused container is refocused, the active descendant be the same element as the active descendant when the container was last focused. Exceptions include cases where the contents of a container widget have changed, and widgets like a menubar where the user expects to always return to the first item when focus leaves the menu bar. For example, if the second item of a tree group was the active descendant when the user tabbed out of the tree group, then the second item of the tree group remains the active

descendant when the tree group gets focus again. The user may also activate the container by clicking on one of the descendants within it. When the container or its active descendant has focus, the user may navigate through the container by pressing additional keys, such as the arrow keys, to change the currently active descendant. Any additional press of the main navigation key (generally the TAB key) will move out of the container to the next widget.

Usable keyboard navigation in a rich internet application is different from the tabbing paradigm among interactive elements, such as links and form controls, in a static document. In rich internet applications, the user tabs to significantly complex *widgets*, such as a menu or spreadsheet, and uses the arrow keys to navigate within the widget. The changes that WAI-ARIA introduces to keyboard navigation make this enhanced accessibility possible. In WAI-ARIA, any element can be keyboard focusable. In addition to host language mechanisms such as `tabindex`, `aria-activedescendant` provides another mechanism for keyboard operation. Most other aspects of WAI-ARIA widget development depend on keyboard navigation functioning properly.

When implementing `aria-activedescendant` as described below, the user agent keeps the `DOM` focus on the container element or on an input element that controls the container element. However, the user agent communicates `desktop focus events` and states to the assistive technology as if the element referenced by `aria-activedescendant` has focus. User agents are not expected to validate that the active descendant is a descendant of the container element. It is the responsibility of the user agent to ensure that keyboard events are processed at the `element` that has `DOM` focus. Any keyboard events directed at the active descendant bubble up to the `DOM` element with focus for processing.

4.3.1 Information for Authors

If the author removes the element with focus, the author **SHOULD** move focus to a logical element. Similarly, authors **SHOULD** not scroll the element with focus off screen unless the user performed a scrolling action.

Authors **SHOULD** ensure that all interactive `elements` are focusable and that all parts of composite widgets are either focusable or have a documented alternative method to achieve their function.

Authors **MUST** manage focus on the following container roles:

- `grid`
- `listbox`
- `menu`
- `menubar`
- `radiogroup`

- [tree](#)
- [treegrid](#)
- [tablist](#)

User agents that support WAI-ARIA expand the usage of host language mechanisms such as `tabindex`, `focus`, and `blur` to allow them on all [elements](#). Where the host language supports it, authors *MAY* add any element such as a `div`, `span`, or `img` to the default tab order by setting `tabindex="0"`. In addition, any item with `tabindex` equal to a negative integer is focusable via script or a mouse click, but is not part of the default tab order. This is supported in both [HTML] and [SVG2].

Authors *MAY* use [aria-activedescendant](#) to inform [assistive technologies](#) which descendant of a [widget](#) element is treated as having keyboard focus in the user interface if the role of the widget element supports `aria-activedescendant`. This is often a more convenient way of providing keyboard navigation within widgets, such as a [listbox](#), where the widget occupies only one stop in the page Tab sequence and other keys, typically arrow keys, are used to focus elements inside the widget.

Typically, the author will use host language [semantics](#) to put the widget in the Tab sequence (e.g., `tabindex="0"` in [HTML](#)) and `aria-activedescendant` to point to the ID of the currently active descendant. The author, not the user agent, is responsible for styling the currently active descendant to show it has keyboard focus. The author cannot use `:focus` to style the currently active descendant since the actual focus is on the container.

More information on managing focus can be found in the [Developing a Keyboard Interface](#) section of the [WAI-ARIA Authoring Practices](#).

4.3.2 Information for User Agents

The user agent **MUST** do the following to implement [aria-activedescendant](#):

1. Implement the host language method for keyboard navigation so that widgets that support `aria-activedescendant` may be included in the tab order.
2. For platforms that expose [desktop focus](#) or [accessibility API](#) focus separately from DOM focus, do not expose the focused state in the accessibility API for any element when it has DOM focus and also has [aria-activedescendant](#) which points to a valid [ID reference](#).
3. When the [aria-activedescendant](#) attribute changes on an element that currently has DOM focus, remove the focused state from the previously focused object and fire an accessibility API [desktop focus event](#) on the new element referenced by `aria-activedescendant`. If [aria-activedescendant](#) is cleared or does not point to an element in the current document, fire a desktop focus event for the [object](#) that had the attribute change.

4. Apply the following accessibility API states to any element with an ID attribute that can be referenced by an element with both an `aria-activedescendant` attribute and has DOM focus. There are two ways an element can be referenced by `aria-activedescendant`. One way is when it is `owned` by an element with `aria-activedescendant` and the other is when it is `owned` by an element that is controlled by an element with role of `combobox`, `textbox` or `searchbox` with an `aria-activedescendant` attribute:

- A. Focusable, if the element also has a WAI-ARIA `role`. The element needs to be focusable because it could be referenced by the `aria-activedescendant` attribute. Native elements that have no `role` attribute do not need to be checked; their native semantics determine the focusable state.
- B. Focused, whenever the element is the target of the `aria-activedescendant` attribute and the element with the `aria-activedescendant` attribute has DOM focus.

When an assistive technology uses its platform's accessibility API to request a change of focus, user agents **MUST** do the following:

1. Remove the platform's focused state from the previously focused object.

2. Set the DOM focus:

- A. If the `element` can take DOM focus, the `user agent` **MUST** set the DOM focus to it.
- B. Otherwise, if the current element has an ID and the ID is referenced by the `aria-activedescendant` attribute of an element that is focusable, the user agent **MUST** set DOM focus to the element that has the `aria-activedescendant` attribute.

NOTE

An element with an ID can be referenced when it is `owned` by a container element that has the `aria-activedescendant` attribute or by a container element that is controlled by an element that has the `aria-activedescendant` attribute (e.g. see `combobox`). Otherwise the `aria-activedescendant` attribute reference indicates an author error.

NOTE

The inability to set DOM focus to the containing element indicates an author error.

- C. Otherwise, the user agent **MAY** attempt to set DOM focus to the child element itself.

3. If the current element has an ID and is `owned` by either a container element with both an `aria-activedescendant` attribute and has DOM focus, or by a container element that is controlled by an element with both an `aria-activedescendant` attribute and has DOM focus, the user agent **MUST** set the accessibility API focused state and fire an accessibility API focus `event` on the element identified by the value of `aria-activedescendant`.

§ 5. The Roles Model

This section defines WAI-ARIA [roles](#) and describes their characteristics and properties.

The roles, their characteristics, the states and properties they support, and specification of how they may be used in markup, shall be considered normative.

In order to reflect the content in the [DOM](#), user agents **SHOULD** map the role attribute to the appropriate value in the implemented accessibility [API](#), and user agents **SHOULD** update the mapping when the role attribute changes.

§ 5.1 Relationships Between Concepts

The Roles Model uses the following relationships to relate WAI-ARIA roles to each other and to concepts from other specifications, such as [HTML](#).

§ 5.1.1 Superclass Role

The [role](#) that the current subclassed role extends in the Roles Model. This extension causes all the properties and constraints of the superclass role to propagate to the subclass role. Other than well known stable specifications, inheritance may be restricted to items defined inside this specification, so that external items cannot be changed and affect inherited [classes](#).

§ 5.1.2 Subclass Roles

Informative list of [roles](#) for which this role is the superclass. This is provided to facilitate reading of the specification but adds no new information.

§ 5.1.3 Related Concepts

Informative data about a similar or related idea from other specifications. Concepts that are related are not necessarily identical. Related concepts do not inherit properties from each other. Hence if the definition of

one concept changes, the properties, behavior, and definition of its related concept is not affected.

For example, a progress bar is like a status indicator. Therefore, the [progressbar widget](#) has a related concept which includes [status](#). However, if the definition of [status](#) is modified, the definition of a [progressbar](#) is not affected.

5.1.4 Base Concept

Informative data about [objects](#) that are considered prototypes for the [role](#). Base concept is similar to type, but without inheritance of limitations and properties. Base concepts are designed as a substitute for inheritance for external concepts. A base concept is like a [related concept](#) except that the base concept is almost identical to the role definition.

For example, the [checkbox](#) defined in this document has similar functionality and anticipated behavior to a `<input[type="checkbox"]>` defined in [HTML]. Therefore, a [checkbox](#) has an [HTML] checkbox as a [baseConcept](#). However, if the original [HTML] checkbox baseConcept definition is modified, the definition of a [checkbox](#) in this document will not be affected, because there is no actual inheritance of the respective type.

5.2 Characteristics of Roles

Roles are defined and described by their characteristics. Characteristics define the structural function of a role, such as what a role is, concepts behind it, and what instances the role can or must contain. In the case of [widgets](#) this also includes how it interacts with the [user agent](#) based on mapping to [HTML](#) forms. States and properties from [WAI-ARIA](#) that are supported by the role are also indicated.

Roles define the following characteristics.

5.2.1 Abstract Roles

Values

Boolean

Abstract [roles](#) are the foundation upon which all other [WAI-ARIA](#) roles are built. Content authors **MUST NOT** use abstract roles because they are not implemented in the [API](#) binding. User agents **MUST NOT** map abstract roles to the standard role mechanism of the accessibility [API](#). Abstract roles are provided to help with the following:

1. Organize the Roles Model and provide roles with a meaning in the context of known concepts.
2. Streamline the addition of roles that include necessary features.

§ 5.2.2 Required States and Properties

[States](#) and [properties](#) specifically required for the [role](#) and subclass roles. Content authors **MUST** provide a non-empty value for required states and properties. Content authors **MUST NOT** use the value [undefined](#) for required states and properties, unless [undefined](#) is an explicitly-supported value of that state or property.

When an [object](#) inherits from multiple ancestors and one ancestor indicates that property is supported while another ancestor indicates that it is required, the property is required in the inheriting object.

NOTE

A host language attribute with the appropriate [implicit WAI-ARIA semantic](#) fulfills this requirement.

§ 5.2.3 Supported States and Properties

[States](#) and [properties](#) specifically applicable to the [role](#) and child roles. Content authors **MAY** provide values for supported states and properties, but need not in cases where default values are sufficient. [User agents](#) **MUST** map all supported states and properties for the role to an accessibility [API](#). If the state or property is [undefined](#) and it has a default value for the role, [user agents](#) **SHOULD** expose the default value.

NOTE

A host language attribute with the appropriate [implicit WAI-ARIA semantic](#) fulfills this requirement.

§ 5.2.4 Inherited States and Properties

Informative list of properties that are inherited by a [role](#) from superclass roles. [States](#) and [properties](#) are inherited from superclass roles in the Roles Model, not from ancestor [elements](#) in the [DOM](#) tree. These properties are not explicitly defined on the role, as the inheritance of properties is automatic. This information is provided to facilitate reading of the specification. The set of supported states and properties combined with inherited states and properties forms the full set of states and properties supported by the role.

5.2.5 Prohibited States and Properties

List of states and properties that are prohibited on a [role](#). Authors **MUST NOT** specify a prohibited state or property.

NOTE

A host language attribute with the appropriate [implicit WAI-ARIA semantic](#) would also prohibit a state or property in this section.

5.2.6 Required Owned Elements

Any [element](#) that will be [owned](#) by the element with this [role](#). For example, an element with the role [list](#) will own at least one element with the role [listitem](#).

When multiple roles are specified as *required owned elements* for a role, at least one instance of one required [owned](#) element is expected. This specification does *not* require an instance of each of the listed owned roles. For example, a [menu](#) should have at least one instance of a [menuitem](#), [menuitemcheckbox](#), or [menuitemradio](#). The [menu](#) role does not require one instance of each.

There may be times that required [owned](#) elements are missing, for example, while editing or while loading a data set. When a widget is missing *required owned elements* due to script execution or loading, authors **MUST** mark a containing element with [aria-busy](#) equal to `true`. For example, until a page is fully initialized and complete, an author could mark the document element as busy.

NOTE

A role that has 'required owned elements' does not imply the reverse relationship. While processing of a role may be incomplete without elements of given roles present as descendants, elements with roles in this list do not always have to be found within elements of the given role. See [required context role](#) for requirements about the context where elements of a given role will be contained.

NOTE

An element with a [subclass role](#) of the 'required owned element' does not fulfill this requirement. For example, the [listbox](#) role requires ownership of an element using the [option](#) or [group](#) role. Although the [group](#) role is the superclass of [row](#), adding an [owned](#) element with a role of [row](#) will not fulfill the requirement that [listbox](#) owns an [option](#) or a [group](#).

NOTE

An element with the appropriate [implicit WAI-ARIA semantic](#) fulfills this requirement.

§ 5.2.7 Required Context Role

The required context role defines the owning container where this [role](#) is allowed. If a role has a required context, authors **MUST** ensure that an element with the role is contained inside (or [owned](#) by) an element with the required context role. For example, an element with role `listitem` is only meaningful when contained inside (or [owned](#) by) an element with role `list`.

NOTE

A role that has 'required context role' does not imply the reverse relationship. While an element with the given role needs to appear within an element of the listed role(s) in order to be meaningful, elements of the listed roles do not always need descendant elements of the given role in order to be meaningful. See [required owned elements](#) for requirements about elements that require presence of a given descendant to be processed properly.

NOTE

An element with the appropriate [implicit WAI-ARIA semantic](#) fulfills this requirement.

§ 5.2.8 Accessible Name Calculation

Values

One of the following values:

1. author: name comes from values provided by the author in explicit markup features such as the [aria-label](#) attribute, the [aria-labelledby](#) attribute, or the host language labeling mechanism, such as the `alt` or `title` attributes in [HTML](#), with [HTML title](#) attribute having the lowest precedence for specifying a text alternative.
2. contents: name comes from the text value of the [element node](#). Although this may be allowed in addition to "author" in some [roles](#), this is used in content only if higher priority "author" features are not provided. Priority is defined by the [accessible name and description computation](#) algorithm [ACCNAME-1.2].
3. prohibited: the element does not support name from author. Authors **MUST NOT** use the [aria-](#)

label or aria-labelledby attributes to name the element.

5.2.8.1 Name Computation

Name Computation is defined in the Accessible Name and Description specification.

5.2.8.2 Description Computation

Description Computation is defined in the Accessible Name and Description specification.

5.2.8.3 Accessible Name and Description Computation

Accessible Name and Description Computation is defined in the Accessible Name and Description specification.

5.2.8.4 Roles Supporting Name from Author

- alert
- alertdialog (name required)
- application (name required)
- article
- banner
- blockquote
- button (name required)
- cell
- checkbox (name required)
- columnheader (name required)
- combobox (name required)

- [command](#)
- [complementary](#)
- [composite](#)
- [contentinfo](#)
- [definition](#)
- [dialog](#) (name required)
- [directory](#)
- [document](#)
- [feed](#)
- [figure](#)
- [form](#)
- [grid](#) (name required)
- [gridcell](#)
- [group](#)
- [heading](#) (name required)
- [img](#) (name required)
- [input](#)
- [landmark](#)
- [link](#) (name required)
- [list](#)
- [listbox](#) (name required)
- [listitem](#)
- [log](#)
- [main](#)
- [marquee](#) (name required)
- [math](#)
- [meter](#) (name required)
- [menu](#)
- [menubar](#)

- menuitem (name required)
- menuitemcheckbox (name required)
- menuitemradio (name required)
- navigation
- note
- option (name required)
- progressbar (name required)
- radio (name required)
- radiogroup (name required)
- range
- region (name required)
- row
- rowgroup
- rowheader (name required)
- scrollbar
- search
- searchbox (name required)
- sectionhead
- select
- separator
- slider (name required)
- spinbutton (name required)
- status
- switch (name required)
- tab
- table (name required)
- tablist
- tabpanel (name required)
- term

- [textbox](#) (name required)
- [time](#)
- [timer](#)
- [toolbar](#)
- [tooltip](#) (name required)
- [tree](#) (name required)
- [treegrid](#) (name required)
- [treeitem](#) (name required)
- [window](#)

5.2.8.5 Roles Supporting Name from Content

- [button](#) (name required)
- [cell](#)
- [checkbox](#) (name required)
- [columnheader](#) (name required)
- [gridcell](#)
- [heading](#) (name required)
- [link](#) (name required)
- [menuitem](#) (name required)
- [menuitemcheckbox](#) (name required)
- [menuitemradio](#) (name required)
- [option](#) (name required)
- [radio](#) (name required)
- [row](#)
- [rowheader](#) (name required)
- [sectionhead](#)
- [switch](#) (name required)

- [tab](#)
- [tooltip](#) (name required)
- [treeitem](#) (name required)

5.2.8.6 Roles which cannot be named (*Name prohibited*)

- [caption](#)
- [code](#)
- [deletion](#)
- [emphasis](#)
- [generic](#)
- [insertion](#)
- [paragraph](#)
- [presentation](#)
- [strong](#)
- [subscript](#)
- [superscript](#)

5.2.9 Presentational Children

Values

Boolean (`true` | `false`)

The DOM descendants are presentational. [User agents](#) **SHOULD NOT** expose descendants of this [element](#) through the platform [accessibility API](#). If [user agents](#) do not hide the descendant nodes, some information may be read twice.

5.2.10 Implicit Value for Role

Many states and properties have default values. Occasionally, the default value when used on a given role

should be different from the usual default. Roles that require a state or property to have a non-standard default value indicate this in the "Implicit Value for Role". This is expressed in the form "Default for `state` or `property name` is new default value". Roles that define this have the new default value for the state or property if the author does not provide an explicit value.

§ 5.3 Categorization of Roles

To support the current user scenario, this specification categorizes [roles](#) that define user interface [widgets](#) (sliders, tree controls, etc.) and those that define page structure (sections, navigation, etc.). Note that some assistive technologies provide special modes of interaction for regions marked with role [application](#) or [document](#).

A visual description of the relationships among roles is available in the [ARIA 1.2 Class Diagram](#).

Roles are categorized as follows:

1. [Abstract Roles](#)
2. [Widget Roles](#)
3. [Document Structure Roles](#)
4. [Landmark Roles](#)
5. [Live Region Roles](#)
6. [Window Roles](#)

§ 5.3.1 Abstract Roles

The following [roles](#) are used to support the WAI-ARIA Roles Model for the purpose of defining general role concepts.

Abstract roles are used for the ontology. Authors **MUST NOT** use abstract roles in content.

- [command](#)
- [composite](#)
- [input](#)
- [landmark](#)
- [range](#)

- [roletype](#)
- [section](#)
- [sectionhead](#)
- [select](#)
- [structure](#)
- [widget](#)
- [window](#)

5.3.2 Widget Roles

The following roles act as standalone user interface widgets or as part of larger, composite widgets.

- [button](#)
- [checkbox](#)
- [gridcell](#)
- [link](#)
- [menuitem](#)
- [menuitemcheckbox](#)
- [menuitemradio](#)
- [option](#)
- [progressbar](#)
- [radio](#)
- [scrollbar](#)
- [searchbox](#)
- [separator](#) (when focusable)
- [slider](#)
- [spinbutton](#)
- [switch](#)
- [tab](#)

- [tabpanel](#)
- [textbox](#)
- [treeitem](#)

The following roles act as composite user interface widgets. These roles typically act as containers that manage other, contained widgets.

- [combobox](#)
- [grid](#)
- [listbox](#)
- [menu](#)
- [menubar](#)
- [radiogroup](#)
- [tablist](#)
- [tree](#)
- [treegrid](#)

§ 5.3.3 Document Structure Roles

The following [roles](#) describe structures that organize content in a page. Document structures are not usually interactive.

- [application](#)
- [article](#)
- [blockquote](#)
- [caption](#)
- [cell](#)
- [columnheader](#)
- [definition](#)
- [deletion](#)
- [directory](#)
- [document](#)

- [emphasis](#)
- [feed](#)
- [figure](#)
- [generic](#)
- [group](#)
- [heading](#)
- [img](#)
- [insertion](#)
- [list](#)
- [listitem](#)
- [math](#)
- [meter](#)
- [none](#)
- [note](#)
- [paragraph](#)
- [presentation](#)
- [row](#)
- [rowgroup](#)
- [rowheader](#)
- [separator](#) (when not focusable)
- [strong](#)
- [subscript](#)
- [superscript](#)
- [table](#)
- [term](#)
- [time](#)
- [toolbar](#)
- [tooltip](#)

§ 5.3.4 Landmark Roles

The following [roles](#) are regions of the page intended as navigational [landmarks](#). All of these roles inherit from the [Landmark](#) base type and all are imported from the [Role Attribute \[ROLE-ATTRIBUTE\]](#). The roles are included here in order to make them clearly part of the [WAI-ARIA Roles Model](#).

- [banner](#)
- [complementary](#)
- [contentinfo](#)
- [form](#)
- [main](#)
- [navigation](#)
- [region](#)
- [search](#)

↳ 5.3.5 Live Region Roles

The following [roles](#) are [live regions](#) and may be modified by [live region attributes](#).

- [alert](#)
- [log](#)
- [marquee](#)
- [status](#)
- [timer](#)

↳ 5.3.6 Window Roles

The following [roles](#) act as windows within the browser or application.

- [alertdialog](#)
- [dialog](#)

§ 5.4 Definition of Roles

Below is an alphabetical list of WAI-ARIA [roles](#) to be used by authors.

Abstract roles are used for the ontology. Authors **MUST NOT** use abstract roles in content.

[alert](#)

A type of [live region](#) with important, and usually time-sensitive, information. See related [alertdialog](#) and [status](#).

[alertdialog](#)

A type of dialog that contains an alert message, where initial focus goes to an [element](#) within the dialog. See related [alert](#) and [dialog](#).

[application](#)

A [structure](#) containing one or more focusable elements requiring user input, such as keyboard or gesture events, that do not follow a standard interaction pattern supported by a [widget](#) role.

[article](#)

A section of a page that consists of a composition that forms an independent part of a document, page, or site.

[banner](#)

A [landmark](#) that contains mostly site-oriented content, rather than page-specific content.

[blockquote](#)

A section of content that is quoted from another source.

[button](#)

An input that allows for user-triggered actions when clicked or pressed. See related [link](#).

[caption](#)

Visible content that names, and may also describe, a [figure](#), [table](#), [grid](#), or [treegrid](#).

[cell](#)

A cell in a tabular container. See related [gridcell](#).

[checkbox](#)

A checkable input that has three possible values: `true`, `false`, or `mixed`.

[code](#)

A section whose content represents a fragment of computer code.

[columnheader](#)

A cell containing header information for a column.

[combobox](#)

An [input](#) that controls another element, such as a [listbox](#) or [grid](#), that can dynamically pop up to help the user set the value of the [input](#).

[command](#)

A form of widget that performs an action but does not receive input data.

complementary

A [landmark](#) that is designed to be complementary to the main content at a similar level in the [DOM](#) hierarchy, but remaining meaningful when separated from the main content.

composite

A [widget](#) that may contain navigable descendants or [owned](#) children.

contentinfo

A [landmark](#) that contains information about the parent document.

definition

A definition of a term or concept. See related [term](#).

deletion

A deletion contains content that is marked as removed or content that is being suggested for removal.

See related [insertion](#).

dialog

A dialog is a descendant window of the primary window of a web application. For [HTML](#) pages, the primary application window is the entire web document, i.e., the [body](#) element.

directory

[Deprecated in ARIA 1.2] A list of references to members of a group, such as a static table of contents.

document

An [element](#) containing content that [assistive technology](#) users may want to browse in a reading mode.

emphasis

One or more emphasized characters. See related [strong](#).

feed

A scrollable [list](#) of [articles](#) where scrolling may cause [articles](#) to be added to or removed from either end of the list.

figure

A perceivable [section](#) of content that typically contains a [graphical document](#), images, code snippets, or example text. The parts of a [figure](#) **MAY** be user-navigable.

form

A [landmark](#) region that contains a collection of items and objects that, as a whole, combine to create a form. See related [search](#).

generic

A nameless container [element](#) that has no semantic meaning on its own.

grid

A composite [widget](#) containing a collection of one or more rows with one or more cells where some or all cells in the grid are focusable by using methods of two-dimensional navigation, such as directional arrow keys.

gridcell

A [cell](#) in a [grid](#) or [treegrid](#).

group

A set of user interface [objects](#) that is not intended to be included in a page summary or table of contents by [assistive technologies](#).

heading

A heading for a section of the page.

img

A container for a collection of [elements](#) that form an image.

input

A generic type of [widget](#) that allows user input.

insertion

An insertion contains content that is marked as added or content that is being suggested for addition. See related [deletion](#).

landmark

A perceivable [section](#) containing content that is relevant to a specific, author-specified purpose and sufficiently important that users will likely want to be able to navigate to the section easily and to have it listed in a summary of the page. Such a page summary could be generated dynamically by a user agent or assistive technology.

link

An interactive reference to an internal or external resource that, when activated, causes the user agent to navigate to that resource. See related [button](#).

list

A [section](#) containing [listitem](#) elements. See related [listbox](#).

listbox

A [widget](#) that allows the user to select one or more items from a list of choices. See related [combobox](#) and [list](#).

listitem

A single item in a list or directory.

log

A type of [live region](#) where new information is added in meaningful order and old information may disappear. See related [marquee](#).

main

A [landmark](#) containing the main content of a document.

marquee

A type of [live region](#) where non-essential information changes frequently. See related [log](#).

math

Content that represents a mathematical expression.

meter

An [element](#) that represents a scalar measurement within a known range, or a fractional value. See related [progressbar](#).

menu

A type of [widget](#) that offers a list of choices to the user.

menubar

A presentation of [menu](#) that usually remains visible and is usually presented horizontally.

menuitem

An option in a set of choices contained by a [menu](#) or [menubar](#).

menuitemcheckbox

A [menuitem](#) with a checkable state whose possible values are `true`, `false`, or `mixed`.

menuitemradio

A checkable [menuitem](#) in a set of elements with the same role, only one of which can be checked at a time.

navigation

A [landmark](#) containing a collection of navigational [elements](#) (usually links) for navigating the document or related documents.

none

An [element](#) whose implicit native role semantics will not be mapped to the [accessibility API](#). See synonym [presentation](#).

note

A section whose content is parenthetic or ancillary to the main content of the resource.

option

A selectable item in a [listbox](#).

paragraph

A paragraph of content.

presentation

An [element](#) whose implicit native role semantics will not be mapped to the [accessibility API](#). See synonym [none](#).

progressbar

An [element](#) that displays the progress status for tasks that take a long time.

radio

A checkable input in a group of elements with the same role, only one of which can be checked at a time.

radiogroup

A group of [radio](#) buttons.

range

An element representing a range of values.

region

A [landmark](#) containing content that is relevant to a specific, author-specified purpose and sufficiently important that users will likely want to be able to navigate to the section easily and to have it listed in a summary of the page. Such a page summary could be generated dynamically by a user agent or assistive

technology.

roletype

The base [role](#) from which all other roles inherit.

row

A row of cells in a tabular container.

rowgroup

A structure containing one or more row elements in a tabular container.

rowheader

A cell containing header information for a row.

scrollbar

A graphical object that controls the scrolling of content within a viewing area, regardless of whether the content is fully displayed within the viewing area.

search

A [landmark](#) region that contains a collection of items and objects that, as a whole, combine to create a search facility. See related [form](#) and [searchbox](#).

searchbox

A type of textbox intended for specifying search criteria. See related [textbox](#) and [search](#).

section

A renderable structural containment unit in a document or application.

sectionhead

A structure that labels or summarizes the topic of its related section.

select

A form widget that allows the user to make selections from a set of choices.

separator

A divider that separates and distinguishes sections of content or groups of menuitems.

slider

An input where the user selects a value from within a given range.

spinbutton

A form of [range](#) that expects the user to select from among discrete choices.

status

A type of [live region](#) whose content is advisory information for the user but is not important enough to justify an [alert](#), often but not necessarily presented as a status bar.

strong

Content that is important, serious, or urgent. See related [emphasis](#).

structure

A document structural [element](#).

subscript

One or more subscripted characters. See related [superscript](#).

superscript

One or more superscripted characters. See related [superscript](#).

switch

A type of checkbox that represents on/off values, as opposed to checked/unchecked values. See related [checkbox](#).

tab

A grouping label providing a mechanism for selecting the tab content that is to be rendered to the user.

table

A [section](#) containing data arranged in rows and columns. See related [grid](#).

tablist

A list of [tab](#) [elements](#), which are references to [tabpanel](#) elements.

tabpanel

A container for the resources associated with a [tab](#), where each [tab](#) is contained in a [tablist](#).

term

A word or phrase with a corresponding definition. See related [definition](#).

textbox

A type of input that allows free-form text as its value.

time

An element that represents a specific point in time.

timer

A type of [live region](#) containing a numerical counter which indicates an amount of elapsed time from a start point, or the time remaining until an end point.

toolbar

A collection of commonly used function buttons or controls represented in compact visual form.

tooltip

A contextual popup that displays a description for an element.

tree

A [widget](#) that allows the user to select one or more items from a hierarchically organized collection.

treegrid

A [grid](#) whose rows can be expanded and collapsed in the same manner as for a [tree](#).

treeitem

An option item of a [tree](#). This is an [element](#) within a tree that may be expanded or collapsed if it contains a sub-level group of tree item elements.

widget

An interactive component of a graphical user interface (GUI).

window

A browser or application window.

alert role

A type of [live region](#) with important, and usually time-sensitive, information. See related [alertdialog](#) and [status](#).

Alerts are used to convey messages that may be immediately important to users. In the case of audio warnings, alerts provide an accessible alternative for hearing-impaired users. The [alert role](#) is applied to the element containing the alert message. An [alert](#) is a specialized form of the [status](#) role, which is processed as an atomic [live region](#).

Alerts are assertive live regions, which means they cause immediate notification for assistive technology users. If the operating system allows, the [user agent](#) **SHOULD** fire a system alert [event](#) through the accessibility API when the WAI-ARIA alert is created.

Neither authors nor user agents are required to set or manage focus to an alert in order for it to be processed. Since alerts are not required to receive focus, authors **SHOULD NOT** require users to close an alert. If an author desires focus to move to a message when it is conveyed, the author **SHOULD** use [alertdialog](#) instead of [alert](#).

Elements with the role [alert](#) have an implicit [aria-live](#) value of [assertive](#), and an implicit [aria-atomic](#) value of [true](#).

Characteristics:

Characteristic	Value
Superclass Role:	section
Subclass Roles:	alertdialog
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto

Characteristic	Value
	<p>aria-grabbed (state)</p> <p>aria-haspopup (deprecated on this role in ARIA 1.2)</p> <p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	author
Implicit Value for Role:	<p>Default for aria-live is assertive.</p> <p>Default for aria-atomic is true.</p>

alertdialog role

A type of dialog that contains an alert message, where initial focus goes to an [element](#) within the dialog. See related [alert](#) and [dialog](#).

Alert dialogs are used to convey messages to alert the user. The [alertdialog role](#) goes on the node containing both the alert message and the rest of the dialog. Content authors **SHOULD** make alert dialogs modal by ensuring that, while the [alertdialog](#) is shown, keyboard and mouse interactions only operate within the dialog. See [aria-modal](#).

Unlike [alert](#), [alertdialog](#) can receive a response from the user. For example, to confirm that the user understands the alert being generated. When the alert dialog is displayed, authors **SHOULD** set focus to an active element within the alert dialog, such as a form control or confirmation button. The [user agent](#) **SHOULD** fire a system alert [event](#) through the accessibility [API](#) when the alert is created, provided one is specified by the intended [accessibility API](#).

Authors **SHOULD** use [aria-describedby](#) on an [alertdialog](#) to reference the alert message element in the dialog. If they do not, an [assistive technology](#) can resort to its internal recovery mechanism to determine the contents of the alert message.

Characteristics:

Characteristic	Value
Superclass Role:	alert dialog
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live aria-modal aria-owns aria-relevant aria-roledescription
Name From:	author

Characteristic	Value
Accessible Name Required:	True

application role

A [structure](#) containing one or more focusable elements requiring user input, such as keyboard or gesture events, that do not follow a standard interaction pattern supported by a [widget](#) role.

Some [user agents](#) and [assistive technologies](#) have a browse mode where standard input events, such as up and down arrow key events, are intercepted and used to control a reading cursor. This browse mode behavior prevents elements that do not have a [widget](#) role from receiving and using such keyboard and gesture events to provide interactive functionality.

When there is a need to create an element with an interaction model that is not supported by any of the [WAI-ARIA widget](#) roles, authors **MAY** give that element role [application](#). And, when a user navigates into an element with role [application](#), [assistive technologies](#) that intercept standard input events **SHOULD** switch to a mode that passes most or all standard input events through to the web application.

For example, a presentation slide editor uses arrow keys to change the positions of textbox and image elements on the slide. There are not any [WAI-ARIA widget](#) roles that correspond to such an interaction model so an author could give the slide container role [application](#), an [aria-roledescription](#) of "Slide Editor", and use [aria-describedby](#) to provide instructions.

Because only the focusable elements contained in an [application](#) element are accessible to users of some assistive technologies, authors **MUST** use one of the following techniques to ensure all non-decorative static text or image content inside an application is accessible:

1. Associate the content with a focusable element using [aria-labelledby](#) or [aria-describedby](#).
2. Place the content in a focusable element that has role [document](#) or [article](#).
3. Manage focus of [owned](#) elements as described in [Managing Focus](#), updating the value of [aria-activelement](#) to reference the [element](#) containing the focused content.

Characteristics:

Characteristic	Value
Superclass Role:	structure
Supported States and Properties:	aria-activelement aria-disabled aria-errormessage aria-expanded

Characteristic	Value
	<u>aria-haspopup</u> <u>aria-invalid</u>
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-dropeffect</u> <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-hidden</u> (state) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	author
Accessible Name Required:	True

article role

A section of a page that consists of a composition that forms an independent part of a document, page, or site.

An article is not a navigational [landmark](#), but may be nested to form a discussion where assistive technologies could pay attention to article nesting to assist the user in following the discussion. An article could be a forum post, a magazine or newspaper article, a web log entry, a user-submitted comment, or any other independent item of content. It is *independent* in that its contents could stand alone, for example in syndication. However, the [element](#) is still associated with its ancestors; for instance, contact information that applies to a parent body element still covers the article as well. When nesting articles, the child articles

represent content that is related to the content of the parent article. For instance, a web log entry on a site that accepts user-submitted comments could represent the comments as articles nested within the article for the web log entry. Author, heading, date, or other information associated with an article does not apply to nested articles.

When the user navigates to an element assigned the role of `article`, [assistive technologies](#) that typically intercept standard keyboard events **SHOULD** switch to document browsing mode, as opposed to passing keyboard events through to the web application. Assistive technologies **MAY** provide a feature allowing the user to navigate the hierarchy of any nested `article` elements.

When an `article` is in the context of a [feed](#), the author **MAY** specify values for [`aria-posinset`](#) and [`aria-setsize`](#).

Characteristics:

Characteristic	Value
Superclass Role:	document
Related Concepts:	<code><article></code> in [HTML]
Supported States and Properties:	<code>aria-posinset</code> <code>aria-setsize</code>
Inherited States and Properties:	<code>aria-atomic</code> <code>aria-busy</code> (state) <code>aria-controls</code> <code>aria-current</code> (state) <code>aria-describedby</code> <code>aria-details</code> <code>aria-disabled</code> (state) (deprecated on this role in ARIA 1.2) <code>aria-dropeffect</code> <code>aria-errormessage</code> (deprecated on this role in ARIA 1.2) <code>aria-flowto</code> <code>aria-grabbed</code> (state) <code>aria-haspopup</code> (deprecated on this role in ARIA 1.2) <code>aria-hidden</code> (state)

Characteristic	Value
	<p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	author

banner role

A [landmark](#) that contains mostly site-oriented content, rather than page-specific content.

Site-oriented content typically includes things such as the logo or identity of the site sponsor, and a site-specific search tool. A banner usually appears at the top of the page and typically spans the full width.

User agents **SHOULD** treat elements with the role of `banner` as navigational [landmarks](#).

Within any [document](#) or [application](#), the author **SHOULD** mark no more than one [element](#) with the [banner role](#).

NOTE

Because [document](#) and [application](#) elements can be nested in the [DOM](#), they may have multiple `banner` elements as [DOM](#) descendants, assuming each of those is associated with different document nodes, either by a [DOM](#) nesting (e.g., [document](#) within [document](#)) or by use of the [aria-owns attribute](#).

Characteristics:

Characteristic	Value
Superclass Role:	landmark
Inherited States and Properties:	<p>aria-atomic</p> <p>aria-busy (state)</p> <p>aria-controls</p>

Characteristic	Value
	<p><u>aria-current</u> (state)</p> <p><u>aria-describedby</u></p> <p><u>aria-details</u></p> <p><u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-dropeffect</u></p> <p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Name From:	author

blockquote role

A section of content that is quoted from another source.

Characteristics:

Characteristic	Value
Superclass Role:	<u>section</u>
Related Concepts:	<blockquote> in [HTML]

Characteristic	Value
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-dropeffect</u> <u>aria-errormessage</u> (deprecated on this role in ARIA 1.2) <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-haspopup</u> (deprecated on this role in ARIA 1.2) <u>aria-hidden</u> (state) <u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	author

button role

An input that allows for user-triggered actions when clicked or pressed. See related [link](#).

Buttons are mostly used for discrete actions. Standardizing the appearance of buttons enhances the user's

recognition of the [widgets](#) as buttons and allows for a more compact display in toolbars.

Buttons support the optional [attribute `aria-pressed`](#). Buttons with a non-empty [aria-pressed](#) attribute are toggle buttons. When [aria-pressed](#) is `true` the button is in a "pressed" [state](#), when [aria-pressed](#) is `false` it is not pressed. If the attribute is not present, the button is a simple command button.

Characteristics:

Characteristic	Value
Superclass Role:	command
Base Concept:	<button> in [HTML]
Related Concepts:	link
Supported States and Properties:	aria-disabled aria-haspopup aria-expanded aria-pressed
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live

Characteristic	Value
	<u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	contents author
Accessible Name Required:	True
Children Presentational:	True

caption role

Visible content that names, and may also describe, a [figure](#), [table](#), [grid](#), or [treegrid](#).

When using **caption** authors **SHOULD** ensure:

- The **caption** is a direct child of a [figure](#), [table](#), [grid](#), or [treegrid](#).
- The **caption** is the first child of a [table](#), [grid](#), or [treegrid](#).
- The **caption** is the first or last child of a [figure](#).

Authors **SHOULD** set [aria-labelledby](#) on the parent [figure](#), [table](#), [grid](#), or [treegrid](#) to reference the element with role **caption**. However, if a **caption** contains content that serves as both a name and description for its parent, authors **MAY** instead set [aria-labelledby](#) to reference an element within the **caption** that contains a concise name, and set [aria-describedby](#) to reference an element within the **caption** that contains the descriptive content.

EXAMPLE 5

```
<div role="table" aria-labelledby="name" aria-describedby="desc">
  <div role="caption">
    <div id="name">Contest Entrants</div>
    <div id="desc">
      This table shows the total number of entrants (500) the
      contest accepted over the past four weeks.
    </div>
  </div>
<!-- ... -->
```

Characteristics:

Characteristic	Value
----------------	-------

Characteristic	Value
Superclass Role:	section
Related Concepts:	<caption> in [HTML] <figcaption> in [HTML]
Required Context Role:	figure grid table treegrid
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-live aria-owns aria-relevant aria-roledescription

Characteristic	Value
Prohibited States and Properties:	aria-label aria-labelledby
Name From:	prohibited

cell role

A cell in a tabular container. See related [gridcell](#).

Authors **MUST** ensure [elements](#) with [role](#) cell are contained in, or owned by, an element with the [role row](#).

Characteristics:

Characteristic	Value
Superclass Role:	section
Subclass Roles:	columnheader gridcell rowheader
Base Concept:	<td> in [HTML]
Required Context Role:	row
Supported States and Properties:	aria-colindex aria-colspan aria-rowindex aria-rowspan
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect

Characteristic	Value
	<p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Name From:	contents author

checkbox role

A checkable input that has three possible values: `true`, `false`, or `mixed`.

The [aria-checked](#) attribute of a checkbox indicates whether the input is checked (`true`), unchecked (`false`), or represents a group of [elements](#) that have a mixture of checked and unchecked values (`mixed`). Many checkboxes do not use the `mixed` value, and thus are effectively boolean checkboxes.

Characteristics:

Characteristic	Value
Superclass Role:	<u>input</u>
Subclass Roles:	<u>switch</u>
Related Concepts:	<input[type="checkbox"]> in <u>[HTML]</u>

Characteristic	Value
	option
Required States and Properties:	aria-checked
Supported States and Properties:	aria-errormessage aria-expanded aria-invalid aria_READONLY aria-required
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-keyshortcuts aria-label aria-labelledby aria-live aria-owns aria-relevant aria-roledescription
Name From:	contents author

Characteristic	Value
Accessible Name Required:	True
Children Presentational:	True

code role

A section whose content represents a fragment of computer code.

The primary purpose of the code role is to inform assistive technologies that the content is computer code and thus may require special presentation, in particular with respect to synthesized speech. More specifically, screen readers and other tools which provide text-to-speech presentation of content **SHOULD** prefer full punctuation verbosity to ensure common symbols (e.g. "-") are spoken.

Characteristics:

Characteristic	Value
Superclass Role:	section
Related Concepts:	<code> in [HTML]
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2)

Characteristic	Value
	<u>aria-keyshortcuts</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Prohibited States and Properties:	<u>aria-label</u> <u>aria-labelledby</u>
Name From:	prohibited

columnheader role

A cell containing header information for a column.

The `columnheader` can be used as a column header in a table or grid. It could also be used in a pie chart to show a similar [relationship](#) in the data.

The `columnheader` establishes a relationship between it and all cells in the corresponding column. It is the structural equivalent to an [HTML th element](#) with a column scope.

Authors **MUST** ensure [elements](#) with [role](#) `columnheader` are contained in, or [owned](#) by, an element with the role [row](#).

Applying the [`aria-selected`](#) state on a `columnheader` **MUST** not cause the user agent to automatically propagate the [`aria-selected`](#) state to all the cells in the corresponding column. An author **MAY** choose to propagate selection in this manner depending on the specific application.

While the `columnheader` role can be used in both interactive grids and non-interactive tables, the use of [`aria-readonly`](#) and [`aria-required`](#) is only applicable to interactive elements. Therefore, authors **SHOULD NOT** use [`aria-required`](#) or [`aria-readonly`](#) in a `columnheader` that descends from a [table](#), and user agents **SHOULD NOT** expose either property to [assistive technologies](#) unless the `columnheader` descends from a [grid](#).

NOTE

Because cells are organized into rows, there is not a single container element for the column. The column is the set of [`gridcell`](#) elements in a particular position within their respective [row](#) containers.

NOTE: Usage of aria-disabled

While [aria-disabled](#) is currently supported on [columnheader](#), in a future version the working group plans to prohibit its use on elements with role [columnheader](#) except when the element is in the context of a [grid](#) or [treegrid](#).

Characteristics:

Characteristic	Value
Superclass Role:	cell gridcell sectionhead
Base Concept:	<th[scope="col"]> in [HTML]
Required Context Role:	row
Supported States and Properties:	aria-sort
Inherited States and Properties:	aria-atomic aria-busy (state) aria-colindex aria-colspan aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-errormessage aria-expanded (state) aria-flowto aria-grabbed (state) aria-haspopup aria-hidden (state) aria-invalid (state) aria-keyshortcuts

Characteristic	Value
	<u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria_READONLY</u> <u>aria-relevant</u> <u>aria-required</u> <u>aria-roledescription</u> <u>aria-rowindex</u> <u>aria-rowspan</u> <u>aria-selected</u> (state)
Name From:	contents author
Accessible Name Required:	True

combobox role

An [input](#) that controls another element, such as a [listbox](#) or [grid](#), that can dynamically pop up to help the user set the value of the [input](#).

EDITOR'S NOTE: Major Changes to combobox role in ARIA 1.2

The Guidance for [combobox](#) has changed significantly in ARIA 1.2 due to problems with implementation of the previous patterns. Authors and developers of User Agents, Assistive Technologies, and Conformance Checkers are advised to review this section carefully to understand the changes. Explanation of the changes is available in the [ARIA repository wiki](#).

A **combobox** functionally combines a named input field with the ability to assist value selection via a supplementary popup element. A **combobox** input **MAY** be either a single-line text field that supports editing and typing or an element that only displays the current value of the **combobox**. If the **combobox** supports text input and provides autocompletion behavior as described in [aria-autocomplete](#), authors **MUST** set [aria-autocomplete](#) on the **combobox** element to the value that corresponds to the provided behavior.

Typically, the initial state of a **combobox** is collapsed. In the collapsed state, only the **combobox** element and a separate, optional popup control [button](#) are visible. A **combobox** is said to be expanded when both the

combobox element showing its current value and its associated popup element are visible. Authors **MUST** set `aria-expanded` to `true` on an element with role `combobox` when it is expanded and `false` when it is collapsed.

Authors **MUST** ensure the popup element associated with a combobox has a role of `listbox`, `tree`, `grid`, or `dialog`. Authors **MUST** set `aria-controls` on a combobox element to a value that refers to the combobox popup element.

Elements with the role `combobox` have an implicit `aria-haspopup` value of `listbox`. If the combobox popup element has a role other than `listbox`, authors **MUST** specify a value for `aria-haspopup` that corresponds to the role of its popup.

If the user interface includes an additional icon that allows the visibility of the popup to be controlled via pointer and touch events, authors **SHOULD** ensure that element has role `button`, that it is focusable but not included in the page `Tab` sequence, and that it is not a descendant of the element with role `combobox`. In addition, to be keyboard accessible, authors **SHOULD** provide keyboard mechanisms for moving focus between the combobox element and elements contained in the popup. For example, one common convention is that Down Arrow moves focus from the input to the first focusable descendant of the popup element. If the popup element supports `aria-activedescendant`, in lieu of moving focus, such keyboard mechanisms can control the value of `aria-activedescendant` on the combobox element. When a descendant of the popup element is active, authors **MAY** set `aria-activedescendant` on the combobox to a value that refers to the active element within the popup while focus remains on the combobox element.

User agents **MUST** expose the value of elements with role `combobox` to [assistive technologies](#). The value of a combobox is represented by one of the following:

- If the combobox element is a host language element that provides a value, such as an [HTML input](#) element, the value of the combobox is the value of that element.
- Otherwise, the value of the combobox is represented by its descendant elements and can be determined using the same method used to compute the name of a [button](#) from its descendant content.

EXAMPLE 6

```
<label for="tag_combo">Tag</label>
<input type="text" id="tag_combo"
       role="combobox" aria-autocomplete="list"
       aria-haspopup="listbox" aria-expanded="true"
       aria-controls="popup_listbox" aria-activedescendant="selected_option">
<ul role="listbox" id="popup_listbox">
  <li role="option">Zebra</li>
  <li role="option" id="selected_option">Zoom</li>
</ul>
```

EDITOR'S NOTE: Validity changes combobox for ARIA 1.2

Please review the following carefully. As a result of these changes a combobox following the ARIA 1.1 combobox specification will no longer conform with the ARIA specification.

NOTE

The structural requirements for **combobox** defined by this version of the specification are different from the requirements defined by ARIA 1.0 and ARIA 1.1:

- The ARIA 1.0 specification required the input element with the **combobox** role to be a single-line text field and reference the popup element with [aria-owns](#) instead of [aria-controls](#).
- The ARIA 1.1 specification, which was not broadly supported by assistive technologies, required the **combobox** to be a non-focusable element with two required owned elements -- a focusable [textbox](#) and a popup element controlled by the [textbox](#).
- The changes introduced in ARIA 1.2 improve interoperability with assistive technologies and enable authors to create presentations of combobox that more closely imitate a native [HTML select](#) element.

The features and behaviors of combobox implementations vary widely. Consequently, there are many important authoring considerations. See the [WAI-ARIA Authoring Practices](#) for additional details on implementing combobox design patterns.

Characteristics:

Characteristic	Value
Superclass Role:	<u>input</u>
Related Concepts:	<select> in [HTML]
Required States and Properties:	<u>aria-controls</u> <u>aria-expanded</u>

Characteristic	Value
Supported States and Properties:	<u>aria-activedescendant</u> <u>aria-autocomplete</u> <u>aria-errormessage</u> <u>aria-haspopup</u> <u>aria-invalid</u> <u>aria-readonly</u> <u>aria-required</u>
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) <u>aria-dropeffect</u> <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-hidden</u> (state) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	author
Accessible Name Required:	True
Implicit Value for Role:	Default for <u>aria-haspopup</u> is listbox.

command role

A form of widget that performs an action but does not receive input data.

NOTE

command is an abstract role used for the ontology. Authors should not use this role in content.

Characteristics:

Characteristic	Value
Is Abstract:	True
Superclass Role:	widget
Subclass Roles:	button link menuitem
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label

Characteristic	Value
	aria-labelledby aria-live aria-owns aria-relevant aria-roledescription
Name From:	author

complementary role

A [landmark](#) that is designed to be complementary to the main content at a similar level in the [DOM](#) hierarchy, but remaining meaningful when separated from the main content.

There are various types of content that would appropriately have this [role](#). For example, in the case of a portal, this may include but not be limited to show times, current weather, related articles, or stocks to watch. The complementary role indicates that contained content is relevant to the main content. If the complementary content is completely separable from the main content, it may be appropriate to use a more general role.

User agents **SHOULD** treat elements with the role of [complementary](#) as navigational [landmarks](#).

Characteristics:

Characteristic	Value
Superclass Role:	landmark
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2)

Characteristic	Value
	<p>aria-flowto</p> <p>aria-grabbed (state)</p> <p>aria-haspopup (deprecated on this role in ARIA 1.2)</p> <p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	author

composite role

A [widget](#) that may contain navigable descendants or [owned](#) children.

Authors **SHOULD** ensure that a composite widget exists as a single navigation stop within the larger navigation system of the web page. Once the composite widget has focus, authors **SHOULD** provide a separate navigation mechanism for users to navigate to [elements](#) that are descendants or owned children of the composite element.

NOTE

composite is an abstract role used for the ontology. Authors should not use this role in content.

Characteristics:

Characteristic	Value
Is Abstract:	True
Superclass Role:	widget

Characteristic	Value
Subclass Roles:	<u>grid</u> <u>select</u> <u>spinbutton</u> <u>tablist</u>
Supported States and Properties:	<u>aria-activedescendant</u> <u>aria-disabled</u>
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-dropeffect</u> <u>aria-errormessage</u> (deprecated on this role in ARIA 1.2) <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-haspopup</u> (deprecated on this role in ARIA 1.2) <u>aria-hidden</u> (state) <u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	author

contentinfo role

A [landmark](#) that contains information about the parent document.

Examples of information included in this region of the page are copyrights and links to privacy statements.

User agents **SHOULD** treat elements with the role of **contentinfo** as navigational [landmarks](#).

Within any [document](#) or [application](#), the author **SHOULD** mark no more than one [element](#) with the **contentinfo** role.

NOTE

Because [document](#) and [application](#) elements can be nested in the [DOM](#), they may have multiple **contentinfo** elements as [DOM](#) descendants, assuming each of those is associated with different document nodes, either by a [DOM](#) nesting (e.g., [document](#) within [document](#)) or by use of the [aria-owns](#) attribute.

Characteristics:

Characteristic	Value
Superclass Role:	landmark
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state)

Characteristic	Value
	<u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	author

definition role

A definition of a term or concept. See related [term](#).

Authors **SHOULD** identify the [element](#) being defined by giving that element a role of [term](#) and referencing it with the [aria-labelledby](#) attribute or by making the element with role [term](#) a descendant of the element with role **definition**.

Characteristics:

Characteristic	Value
Superclass Role:	<u>section</u>
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-dropeffect</u> <u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)

Characteristic	Value
	<p>aria-flowto</p> <p>aria-grabbed (state)</p> <p>aria-haspopup (deprecated on this role in ARIA 1.2)</p> <p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	author

deletion role

A deletion contains content that is marked as removed or content that is being suggested for removal. See related [insertion](#).

Deletions are typically used to either mark differences between two versions of content or to designate content suggested for removal in scenarios where multiple people are revising content.

Characteristics:

Characteristic	Value
Superclass Role:	section
Related Concepts:	 in [HTML]
Inherited States and Properties:	<p>aria-atomic</p> <p>aria-busy (state)</p> <p>aria-controls</p> <p>aria-current (state)</p> <p>aria-describedby</p>

Characteristic	Value
	<p><u>aria-details</u></p> <p><u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-dropeffect</u></p> <p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Prohibited States and Properties:	<p><u>aria-label</u></p> <p><u>aria-labelledby</u></p>
Name From:	prohibited

dialog role

A dialog is a descendant window of the primary window of a web application. For HTML pages, the primary application window is the entire web document, i.e., the `body` element.

Dialogs are most often used to prompt the user to enter or respond to information. A dialog that is designed to interrupt workflow is usually modal. See related [alertdialog](#).

Authors **MUST** provide an accessible name for a dialog, which can be done with the [aria-label](#) or [aria-labelledby](#) attribute.

Authors **SHOULD** ensure that all dialogs (both modal and non-modal) have at least one focusable

descendant element. Authors **SHOULD** focus an element in the modal dialog when it is displayed, and authors **SHOULD** manage focus of modal dialogs.

NOTE

In the description of this role, the term "web application" does not refer to the [application](#) role, which specifies specific assistive technology behaviors.

Characteristics:

Characteristic	Value
Superclass Role:	window
Subclass Roles:	alertdialog
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live

Characteristic	Value
	aria-modal aria-owns aria-relevant aria-roledescription
Name From:	author
Accessible Name Required:	True

directory role

[Deprecated in ARIA 1.2] A list of references to members of a group, such as a static table of contents.

NOTE

As exposed by accessibility APIs, the `directory` role is essentially equivalent to the `list` role. So, using `directory` does not provide any additional benefits to assistive technology users. Authors are advised to treat `directory` as deprecated and to use `list`, or a host language's equivalent semantics instead.

A `directory` is a static table of contents, whether linked or unlinked. This includes tables of contents built with lists, including nested lists. Dynamic tables of contents, however, might use a `tree` role instead.

Characteristics:

Characteristic	Value
Superclass Role:	list
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2)

Characteristic	Value
	<p>ARIA 1.2)</p> <p>aria-flowto</p> <p>aria-grabbed (state)</p> <p>aria-haspopup (deprecated on this role in ARIA 1.2)</p> <p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	author

document role

An [element](#) containing content that [assistive technology](#) users may want to browse in a reading mode.

When [user agent](#) focus moves to an element assigned the role of [document](#), [assistive technologies](#) having a reading mode for browsing static content *MAY* switch to that reading mode and intercept standard input events, such as Up or Down arrow keyboard events, to control the reading cursor.

Because [assistive technologies](#) that have a reading mode default to that mode for all elements except for those with either a [widget](#) or [application](#) role, the only circumstance where the [document](#) role is useful for changing assistive technology behavior is when the element with role [document](#) is a focusable child element of a [widget](#) or [application](#). For example, given an [application](#) element which contains some static rich text, the author can apply role [document](#) to the element containing the text and give it a [tabindex](#) of *0*. When a screen reader user presses the Tab key and places focus on the [document](#) element, the user will be able to read the text with the screen reader's reading cursor.

Characteristics:

Characteristic	Value
----------------	-------

Characteristic	Value
Superclass Role:	structure
Subclass Roles:	article
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live aria-owns aria-relevant aria-roledescription
Name From:	author
Accessible Name Required:	False

emphasis role

One or more emphasized characters. See related [strong](#).

The purpose of the **emphasis** role is to stress or emphasize content. It is not for communicating changes in typographical presentation that do not impact the meaning of the content. Authors **SHOULD** use the **emphasis** role only if its absence would change the meaning of the content.

The **emphasis** role is not intended to convey importance; for that purpose, the [strong](#) role is more appropriate.

Characteristics:

Characteristic	Value
Superclass Role:	section
Related Concepts:	<code></code> in [HTML]
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-live aria-owns

Characteristic	Value
	aria-relevant aria-roledescription
Prohibited States and Properties:	aria-label aria-labelledby
Name From:	prohibited

feed role

A scrollable [list](#) of [articles](#) where scrolling may cause [articles](#) to be added to or removed from either end of the list.

A **feed** enables users of [assistive technologies](#) that have a document browse mode, such as screen readers, to use the browse mode reading cursor to both read and scroll through a stream of rich content that may continue scrolling infinitely by loading more content as the user reads. In a **feed**, [assistive technologies](#) provide a web application with signals of the user's reading cursor movement by moving [user agent](#) focus, enabling the application to both add new content and visually position content as the user browses the page. The **feed** also lets authors inform assistive technologies when additions and removals are occurring so assistive technologies can more reliably update their reading view without disrupting reading or degrading performance.

For example, a **feed** could be used to present a stream of news stories where each [article](#) contains a story with text, links, images, and comments as well as widgets for sharing and commenting. As a screen reader user reads and interacts with each story and moves the screen reader reading cursor from story to story, each story scrolls into view and, as needed, new stories are loaded.

A **feed** is a container element whose children have role [article](#). When [articles](#) are added or removed from either or both ends of a **feed**, authors **SHOULD** set [aria-busy](#) to **true** on the **feed** element before the changes are made and set it to **false** after the changes are complete. Authors **SHOULD** avoid inserting or removing [articles](#) in the middle of a **feed**. These requirements help [assistive technologies](#) gracefully respond to changes in the **feed** content that occur simultaneously with user commands to move the reading cursor within the **feed**.

Authors **SHOULD** make each [article](#) in a **feed** focusable and ensure that the application scrolls an [article](#) into view when [user agent](#) focus is set on the [article](#) or one of its descendant elements. For example, in [HTML](#), each [article](#) element should have a [tabindex](#) value of either **-1** or **0**.

When an [assistive technology](#) reading cursor moves from one [article](#) to another, [assistive technologies](#) **SHOULD** set user agent focus on the [article](#) that contains the reading cursor. If the reading cursor lands on a focusable element inside the [article](#), the assistive technology **MAY** set focus on that element in lieu of

setting focus on the containing [article](#).

Because the ability to scroll to another [article](#) with an [assistive technology](#) reading cursor depends on the presence of another [article](#) in the page, authors **SHOULD** attempt to load additional [articles](#) before [user agent](#) focus reaches an [article](#) at either end of the set of [articles](#) that has been loaded.

Alternatively, authors **MAY** include an [article](#) at either or both ends of the loaded set of [articles](#) that includes an element, such as a [button](#), that lets the user request more [articles](#) to be loaded.

In addition to providing a brief label, authors **MAY** apply [aria-describedby](#) to [article](#) elements in a feed to suggest to screen readers which elements to speak after the label when users navigate by [article](#). Screen readers **MAY** provide users with a way to quickly scan feed content by speaking both the label and [accessible description](#) when navigating by [article](#), enabling the user to ignore repetitive or less important elements, such as embedded interaction widgets, that the author has left out of the description.

Authors **SHOULD** provide keyboard commands for moving focus among [articles](#) in a feed so users who do not utilize an assistive technology that provides [article](#) navigation features can use the keyboard to navigate the feed.

If the number of articles available in a feed supply is static, authors **MAY** specify [aria-setsize](#) on [article](#) elements in that feed. However, if the total number is extremely large, indefinite, or changes often, authors **MAY** set [aria-setsize](#) to -1 to communicate the unknown size of the set.

See the [WAI-ARIA Authoring Practices](#) for additional details on implementing a feed design pattern.

Characteristics:

Characteristic	Value
Superclass Role:	list
Required Owned Elements:	article
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2)

Characteristic	Value
	<u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-haspopup</u> (deprecated on this role in ARIA 1.2) <u>aria-hidden</u> (state) <u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	author
Accessible Name Required:	False

figure role

A perceivable [section](#) of content that typically contains a [graphical document](#), images, code snippets, or example text. The parts of a **figure** *MAY* be user-navigable.

Authors **SHOULD** provide a reference to the **figure** from the main text, but the **figure** need not be displayed at the same location as the referencing element. Authors *MAY* reference text serving as a caption using [aria-describedby](#). Authors *MAY* provide a label using [aria-label](#) or *MAY* reference text serving as a label using [aria-labelledby](#).

[Assistive technologies](#) **SHOULD** enable users to quickly navigate to figures. Mainstream [user agents](#) *MAY* enable users to quickly navigate to figures.

Characteristics:

Characteristic	Value
Superclass Role:	<u>section</u>
Related Concepts:	<figure> in [HTML]

Characteristic	Value
Inherited States and Properties:	<p>aria-atomic</p> <p>aria-busy (state)</p> <p>aria-controls</p> <p>aria-current (state)</p> <p>aria-describedby</p> <p>aria-details</p> <p>aria-disabled (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-dropeffect</p> <p>aria-errormessage (deprecated on this role in ARIA 1.2)</p> <p>aria-flowto</p> <p>aria-grabbed (state)</p> <p>aria-haspopup (deprecated on this role in ARIA 1.2)</p> <p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	author
Accessible Name Required:	False

form role

A [landmark](#) region that contains a collection of items and objects that, as a whole, combine to create a form.

See related [search](#).

A form may contain a mix of host language form controls, scripted controls, and hyperlinks. Authors are reminded to use native host language semantics to create form controls whenever possible. If the purpose of a form is to submit search criteria, authors **SHOULD** use the [search](#) role instead of the generic [form](#) role.

Authors **MUST** give each element with role [form](#) a brief label that describes the purpose of the form.

Authors **SHOULD** reference a visible label with [aria-labelledby](#) if a visible label is present. Authors **SHOULD** include the label inside of a heading whenever possible. The heading **MAY** be an instance of the standard host language heading element or an instance of an element with role [heading](#).

If an author uses a script to submit a form based on a user action that would otherwise not trigger an `onsubmit` event (for example, a form submission triggered by the user changing a form element's value), the author **SHOULD** provide the user with advance notification of the behavior.

User agents **SHOULD** treat elements with the role of [form](#) as navigational [landmarks](#).

Characteristics:

Characteristic	Value
Superclass Role:	landmark
Base Concept:	<form> in [HTML]
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state)

Characteristic	Value
	<p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Name From:	author
Accessible Name Required:	true

generic role

A nameless container [element](#) that has no semantic meaning on its own.

The `generic` role is intended for use as the implicit role of generic elements in host languages (such as `HTML div` or `span`), so is primarily for implementors of user agents. Authors **SHOULD NOT** use this role in content. Authors **MAY** use [presentation](#) or [none](#) to remove implicit accessibility semantics, or a semantic container role such as [group](#) to semantically group descendants in a named container.

Like an element with role [presentation](#), an element with role `generic` can provide a limited number of accessible states and properties for its descendants, such as [aria-live](#) attributes. However, unlike elements with role [presentation](#), `generic` elements are exposed in [accessibility APIs](#) so that assistive technologies can gather certain properties such as layout and bounds.

Characteristics:

Characteristic	Value
Superclass Role:	<u>structure</u>
Related Concepts:	<u>HTML div</u> , <u>HTML span</u>
Inherited States and Properties:	<p><u>aria-atomic</u></p> <p><u>aria-busy</u> (state)</p> <p><u>aria-controls</u></p> <p><u>aria-current</u> (state)</p>

Characteristic	Value
	<p><u>aria-describedby</u></p> <p><u>aria-details</u></p> <p><u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-dropeffect</u></p> <p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p>
Prohibited States and Properties:	<p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-roledescription</u></p>
Name From:	prohibited

grid role

A composite [widget](#) containing a collection of one or more rows with one or more cells where some or all cells in the grid are focusable by using methods of two-dimensional navigation, such as directional arrow keys.

The **grid** role does not imply a specific visual, e.g., tabular, presentation. It describes [relationships](#) among [elements](#). It may be used for purposes as simple as grouping a collection of checkboxes or navigation links or as complex as creating a full-featured spreadsheet application.

The cell elements of a **grid** have role [gridcell](#). Authors *MAY* designate a cell as a row or column header

by using either the [rowheader](#) or [columnheader role](#) in lieu of the [gridcell](#) role. Authors **MUST** ensure elements with role [gridcell](#), [columnheader](#), or [rowheader](#) are [owned](#) by elements with role [row](#), which are in turn owned by an element with role [rowgroup](#), or [grid](#).

To be [keyboard accessible](#), authors **SHOULD** manage focus of descendants of a [grid](#) as described in [Managing Focus](#). When a user is navigating the [grid](#) content with a keyboard, authors **SHOULD** set focus as follows:

- If a [gridcell](#) contains a single interactive [widget](#) that will not consume arrow key presses when it receives focus, such as a [checkbox](#), [button](#), or [link](#), authors **MAY** set focus on the interactive element contained in that cell. This allows the contained widget to be directly operable.
- Otherwise, authors **SHOULD** ensure the element that receives focus is a [gridcell](#), [rowheader](#), or [columnheader](#) element.

Authors **SHOULD** provide a mechanism for changing to an interaction or edit mode that allows users to navigate and interact with content contained inside a focusable cell if that focusable cell contains any of the following:

- a widget that requires arrow keys to operate, e.g., a [combobox](#) or [radiogroup](#)
- multiple interactive elements
- editable content

For example, if a cell in a spreadsheet contains a [combobox](#) or editable text, the [Enter](#) key might be used to activate a cell interaction or editing mode when that cell has focus so the directional arrow keys can be used to operate the contained [combobox](#) or [textbox](#). Depending on the implementation, pressing [Enter](#) again, [Tab](#), [Escape](#), or another key may switch the application back to the grid navigation mode.

Authors **MAY** use a [gridcell](#) to display the result of a formula, which could be editable by the user. In a spreadsheet application, for example, a [gridcell](#) may show a value calculated from a formula until the user activates the [gridcell](#) for editing when a [textbox](#) appears in the [gridcell](#) containing the formula in an editable state.

If [aria_READONLY](#) is set on an element with role [grid](#), [user agents](#) **MUST** propagate the value to all [gridcell](#) elements [owned](#) by the [grid](#) and expose the value in the accessibility [API](#). An author **MAY** override the propagated value of [aria_READONLY](#) for an individual [gridcell](#) element.

In a [grid](#) that provides cell content editing functions, if the content of a focusable [gridcell](#) element is not editable, authors **MAY** set [aria_READONLY](#) to true on the [gridcell](#) element. However, the value of [aria_READONLY](#), whether specified for a [grid](#) or individual cells, only indicates whether the content contained in cells is editable. It does not represent availability of functions for navigating or manipulating the [grid](#) itself.

An unspecified value for [aria_READONLY](#) does not imply that a [grid](#) or a [gridcell](#) contains editable

content. For example, if a `grid` presents a collection of elements that are not editable, such as a collection of `link` elements representing dates in a datepicker, it is not necessary for the author to specify a value for `aria-readonly`.

Authors **MAY** indicate that a focusable `gridcell` is selectable as the object of an action with the `aria-selected` attribute. If the `grid` allows multiple `gridcells` to be selected, the author **SHOULD** set `aria-multiselectable` to `true` on the element with role `grid`.

Since WAI-ARIA can augment an element of the host language, a `grid` can reuse the elements and attributes of a native table, such as an `HTML table` element. For example, if an author applies the `grid` role to an `HTML table` element, the author does not need to apply the `row` and `gridcell` roles to the descendant `HTML tr` and `td` elements because the `user agent` will automatically make the appropriate translations. When the author is reusing a native host language table element and needs a `gridcell` element to span multiple rows or columns, the author **SHOULD** apply the appropriate host language attributes instead of WAI-ARIA `aria-rowspan` or `aria-colspan` properties.

See the [WAI-ARIA Authoring Practices](#) for additional details on implementing grid design patterns.

Characteristics:

Characteristic	Value
Superclass Role:	<code>composite</code> <code>table</code>
Subclass Roles:	<code>treegrid</code>
Base Concept:	<code><table></code> in [HTML]
Required Owned Elements:	<code>row</code> <code>rowgroup</code> → <code>row</code>
Supported States and Properties:	<code>aria-multiselectable</code> <code>aria-readonly</code>
Inherited States and Properties:	<code>aria-activedescendant</code> <code>aria-atomic</code> <code>aria-busy</code> (state) <code>aria-colcount</code> <code>aria-controls</code> <code>aria-current</code> (state) <code>aria-describedby</code>

Characteristic	Value
	<p>aria-details</p> <p>aria-disabled (state)</p> <p>aria-dropeffect</p> <p>aria-errormessage (deprecated on this role in ARIA 1.2)</p> <p>aria-flowto</p> <p>aria-grabbed (state)</p> <p>aria-haspopup (deprecated on this role in ARIA 1.2)</p> <p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p> <p>aria-rowcount</p>
Name From:	author
Accessible Name Required:	True

gridcell role

A [cell](#) in a [grid](#) or [treemap](#).

A gridcell may be focusable, editable, and selectable. A gridcell may have [relationships](#) such as [aria-controls](#) to address the application of functional relationships.

If an author intends a gridcell to have a row header, column header, or both, and if the relevant headers cannot be determined from the [DOM](#) structure, authors **SHOULD** explicitly indicate which header cells are relevant to the gridcell by applying [aria-describedby](#) on the gridcell and referencing [elements](#) with role [rowheader](#) or [columnheader](#).

In a [treegrid](#), authors **MAY** define a **gridcell** as expandable by using the [aria-expanded](#) attribute. If the [aria-expanded](#) attribute is provided, it applies only to the individual cell. It is not a proxy for the container [row](#), which also can be expanded. The main use case for providing this attribute on a **gridcell** is pivot table behavior.

Authors **MUST** ensure [elements](#) with [role](#) gridcell are contained in, or [owned](#) by, an element with the [role row](#).

Characteristics:

Characteristic	Value
Superclass Role:	cell widget
Subclass Roles:	columnheader rowheader
Base Concept:	<td> in [HTML]
Required Context Role:	row
Supported States and Properties:	aria-disabled aria-errormessage aria-expanded aria-haspopup aria-invalid aria_READONLY aria-required aria-selected
Inherited States and Properties:	aria-atomic aria-busy (state) aria-colindex aria-colspan aria-controls aria-current (state) aria-describedby aria-details

Characteristic	Value
	<u>aria-dropeffect</u> <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-hidden</u> (state) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u> <u>aria-rowindex</u> <u>aria-rowspan</u>
Name From:	contents author

group role

A set of user interface [objects](#) that is not intended to be included in a page summary or table of contents by [assistive technologies](#).

Contrast with [region](#), which is a grouping of user interface objects that will be included in a page summary or table of contents.

Authors **SHOULD** use a [group](#) to form a logical collection of items in a [widget](#), such as children in a tree widget forming a collection of siblings in a hierarchy. However, when a [group](#) is used in the context of a [listbox](#), authors **MUST** limit its children to [option](#) elements. Therefore, proper handling of [group](#) by authors and assistive technologies is determined by the context in which it is provided.

Authors **MAY** nest [group](#) elements. If a section is significant enough to warrant inclusion in the web page's table of contents, the author **SHOULD** assign it a [role](#) of [region](#) or a [standard landmark role](#).

Characteristics:

Characteristic	Value
----------------	-------

Characteristic	Value
Superclass Role:	section
Subclass Roles:	row select toolbar
Related Concepts:	<fieldset> in [HTML]
Supported States and Properties:	aria-activedescendant aria-disabled
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live aria-owns aria-relevant aria-roledescription

Characteristic	Value
Name From:	author

heading role

A heading for a section of the page.

To ensure elements with a role of **heading** are organized into a logical outline, authors **MUST** use the [aria-level](#) attribute to indicate the proper nesting level.

Characteristics:

Characteristic	Value
Superclass Role:	sectionhead
Related Concepts:	<h1>, <h2>, <h3>, <h4>, <h5>, and <h6> in [HTML]
Required States and Properties:	aria-level
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts

Characteristic	Value
	<u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	contents author
Accessible Name Required:	True

img role

A container for a collection of [elements](#) that form an image.

An `img` can contain captions and descriptive text, as well as multiple image files that when viewed together give the impression of a single image. An `img` represents a single graphic within a document, whether or not it is formed by a collection of drawing [objects](#). In order for elements with a [role](#) of `img` to be [perceivable](#), authors **MUST** provide a label using the [aria-label](#) or [aria-labelledby](#) attribute.

Characteristics:

Characteristic	Value
Superclass Role:	<u>section</u>
Related Concepts:	 in [HTML]
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-dropeffect</u>

Characteristic	Value
	<p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Name From:	author
Accessible Name Required:	True
Children Presentational:	True

input role

A generic type of [widget](#) that allows user input.

Characteristics:

Characteristic	Value
Is Abstract:	True
Superclass Role:	<u>widget</u>
Subclass Roles:	<p><u>checkbox</u></p> <p><u>combobox</u></p> <p><u>option</u></p> <p><u>radio</u></p>

Characteristic	Value
	slider spinbutton textbox
Supported States and Properties:	aria-disabled
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live aria-owns aria-relevant aria-roledescription
Name From:	author

insertion role

An insertion contains content that is marked as added or content that is being suggested for addition. See related [deletion](#).

Insertions are typically used to either mark differences between two versions of content or to designate content suggested for addition in scenarios where multiple people are revising content.

Characteristics:

Characteristic	Value
Superclass Role:	section
Related Concepts:	<code><ins></code> in [HTML]
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-live aria-owns aria-relevant aria-roledescription

Characteristic	Value
Prohibited States and Properties:	aria-label aria-labelledby
Name From:	prohibited

landmark role

A perceivable [section](#) containing content that is relevant to a specific, author-specified purpose and sufficiently important that users will likely want to be able to navigate to the section easily and to have it listed in a summary of the page. Such a page summary could be generated dynamically by a user agent or assistive technology.

Authors designate the purpose of the content by assigning a role that is a subclass of the landmark role and, when needed, by providing a brief, descriptive label.

Elements with a role that is a subclass of the landmark role are known as landmark regions or navigational landmark regions. [Assistive technologies](#) **SHOULD** enable users to quickly navigate to landmark regions. Mainstream [user agents](#) **MAY** enable users to quickly navigate to landmark regions.

NOTE

landmark is an abstract role used for the ontology. Authors should not use this role in content.

Characteristics:

Characteristic	Value
Is Abstract:	True
Superclass Role:	section
Subclass Roles:	banner complementary contentinfo form main navigation region search

Characteristic	Value
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-dropeffect</u> <u>aria-errormessage</u> (deprecated on this role in ARIA 1.2) <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-haspopup</u> (deprecated on this role in ARIA 1.2) <u>aria-hidden</u> (state) <u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	author
Accessible Name Required:	False

link role

An interactive reference to an internal or external resource that, when activated, causes the user agent to

navigate to that resource. See related [button](#).

If this is a native link in the host language (such as an [HTML](#) anchor with an `href` value), activating the link causes the [user agent](#) to navigate to that resource. If this is a simulated link, the web application author is responsible for managing navigation.

NOTE

If pressing the link triggers an action but does not change browser focus or page location, authors are advised to consider using the [button](#) role instead of the [link](#) role.

Characteristics:

Characteristic	Value
Superclass Role:	command
Related Concepts:	<code><a></code> in [HTML] <code><link></code> in [HTML]
Supported States and Properties:	aria-disabled aria-expanded aria-haspopup
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts

Characteristic	Value
	<u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	contents author
Accessible Name Required:	True

list role

A [section](#) containing [listitem](#) elements. See related [listbox](#).

Lists contain children whose [role](#) is [listitem](#).

Characteristics:

Characteristic	Value
Superclass Role:	<u>section</u>
Subclass Roles:	<u>directory</u> <u>feed</u>
Base Concept:	 in [HTML] in [HTML]
Required Owned Elements:	<u>listitem</u>
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) (deprecated on this role in

Characteristic	Value
	<p>ARIA 1.2)</p> <p>aria-dropeffect</p> <p>aria-errormessage (deprecated on this role in ARIA 1.2)</p> <p>aria-flowto</p> <p>aria-grabbed (state)</p> <p>aria-haspopup (deprecated on this role in ARIA 1.2)</p> <p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	author

listbox role

A [widget](#) that allows the user to select one or more items from a list of choices. See related [combobox](#) and [list](#).

Items within the list are static and, unlike standard HTML [select](#) [elements](#), may contain images. List boxes contain children whose [role](#) is [option](#) or elements whose [role](#) is [group](#) which in turn contains children whose [role](#) is [option](#).

To be [keyboard accessible](#), authors **SHOULD** manage focus of [option](#) descendants for all instances of this role, as described in [Managing Focus](#).

Elements with the role [listbox](#) have an implicit [aria-orientation](#) value of [vertical](#).

Characteristics:

Characteristic	Value
Superclass Role:	select
Related Concepts:	list <select> in [HTML]
Required Owned Elements:	group → option option
Supported States and Properties:	aria-errormessage aria-expanded aria-invalid aria-multiselectable aria_READONLY aria-required
Inherited States and Properties:	aria-activedescendant aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-keyshortcuts aria-label aria-labelledby

Characteristic	Value
	aria-live
	aria-orientation
	aria-owns
	aria-relevant
	aria-roledescription
Name From:	author
Accessible Name Required:	True
Implicit Value for Role:	Default for aria-orientation is <code>vertical</code> .

`listitem` role

A single item in a list or directory.

Authors **MUST** ensure [elements](#) whose [role](#) is `listitem` are contained in, or [owned](#) by, an [element](#) whose [role](#) is [list](#).

Characteristics:

Characteristic	Value
Superclass Role:	section
Subclass Roles:	treeitem
Base Concept:	 in [HTML]
Required Context Role:	directory list
Supported States and Properties:	aria-level aria-posinset aria-setsize
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby

Characteristic	Value
	<p><u>aria-details</u></p> <p><u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-dropeffect</u></p> <p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Name From:	author

log role

A type of [live region](#) where new information is added in meaningful order and old information may disappear. See related [marquee](#).

Examples include chat logs, messaging history, game log, or an error log. In contrast to other live regions, in this [role](#) there is a [relationship](#) between the arrival of new items in the log and the reading order. The log contains a meaningful sequence and new information is added only to the end of the log, not at arbitrary points.

Elements with the role **log** have an implicit [aria-live](#) value of **polite**.

Characteristics:

Characteristic	Value
Superclass Role:	section
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live aria-owns aria-relevant aria-roledescription
Name From:	author
Implicit Value for Role:	Default for aria-live is polite .

main role

A [landmark](#) containing the main content of a document.

This marks the content that is directly related to or expands upon the central topic of the document. The [main role](#) is a non-obtrusive alternative for "skip to main content" links, where the navigation option to go to the main content (or other [landmarks](#)) is provided by the [user agent](#) through a dialog or by [assistive technologies](#).

User agents **SHOULD** treat elements with the role of `main` as navigational landmarks.

Within any [document](#) or [application](#), the author **SHOULD** mark no more than one [element](#) with the `main` role.

NOTE

Because [document](#) and [application](#) elements can be nested in the [DOM](#), they may have multiple `main` elements as [DOM](#) descendants, assuming each of those is associated with different document nodes, either by a [DOM](#) nesting (e.g., [document](#) within [document](#)) or by use of the [aria-owns](#) attribute.

Characteristics:

Characteristic	Value
Superclass Role:	landmark
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2)

Characteristic	Value
	<p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	author

marquee role

A type of [live region](#) where non-essential information changes frequently. See related [log](#).

Common usages of `marquee` include stock tickers and ad banners. The primary difference between a `marquee` and a [log](#) is that logs usually have a meaningful order or sequence of important content changes.

Elements with the role `marquee` have an implicit [aria-live](#) value of `off`.

Characteristics:

Characteristic	Value
Superclass Role:	section
Inherited States and Properties:	<p>aria-atomic</p> <p>aria-busy (state)</p> <p>aria-controls</p> <p>aria-current (state)</p> <p>aria-describedby</p> <p>aria-details</p> <p>aria-disabled (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-dropeffect</p>

Characteristic	Value
	<p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Name From:	author
Accessible Name Required:	True

math role

Content that represents a mathematical expression.

Content with the role **math** is intended to be marked up in an accessible format such as [MathML](#) [MathML3], or with another type of textual representation such as TeX or LaTeX, which can be converted to an accessible format by native browser implementations or a polyfill library.

While it is not ideal to use an image of a mathematical expression, there exists a significant amount of legacy content where images are used to represent mathematical expressions. Authors **SHOULD** ensure that images of math are labeled by text that describes the mathematical expression as it might be spoken.

NOTE

Browsers that support native implementations of MathML are able to provide a more robust, accessible math experience than can be accomplished with plain text approximations of math. Some rendering engines have close integration with screen readers that allow spatial touch exploration of the formula and refreshable braille display output in the Nemeth Braille format. This level of integration is not supported with images of mathematical formulas, even if the author provides a plain text approximation.

At the time of this writing, some mainstream browsers do not support MathML natively, and must be retrofit using a JavaScript polyfill library. When authoring math content, use native MathML wherever possible, and test thoroughly. Use a polyfill library or provide a fallback image with a text alternative approximation if necessary.

§ **MathML Example with Embedded TeX Annotation**

EXAMPLE 7

```
<!-- Note: Use a JavaScript polyfill library to ensure  
this renders in user agents that do not support MathML. -->  
<!-- The math element has an implicit role="math". -->  
 $<mrow>  
    <mi>x</mi>  
    <mo>=</mo>  
    <mfrac>  
      <mrow>  
        <mo form="prefix">-</mo>  
        <mi>b</mi>  
        <mo>±</mo>  
        <msqrt>  
          <msup>  
            <mi>b</mi>  
            <mn>2</mn>  
          </msup>  
          <mo>-</mo>  
          <mn>4</mn>  
          <mo>&#x2062; <!-- &InvisibleTimes; --></mo>  
          <mi>a</mi>  
          <mo>&#x2062; <!-- &InvisibleTimes; --></mo>  
          <mi>c</mi>  
        </msqrt>  
      </mrow>  
      <mrow>  
        <mn>2</mn>  
        <mo>&#x2062; <!-- &InvisibleTimes; --></mo>  
        <mi>a</mi>  
      </mrow>  
    </mfrac>  
  </mrow>  
  <annotation encoding="TeX">  
    x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}  
  </annotation>  
</math>$ 
```

Plain HTML or Polyfill DOM Result of the MathML Quadratic Formula

If a rendering engine does not support a native math format such as MathML, authors **MAY** use JavaScript to downgrade the content to a format the browser can display, such as this HTML image using a data URI and plain text alternative.

EXAMPLE 8

```

```

Characteristics:

Characteristic	Value
Superclass Role:	<u>section</u>
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-dropeffect</u> <u>aria-errormessage</u> (deprecated on this role in ARIA 1.2) <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-haspopup</u> (deprecated on this role in ARIA 1.2) <u>aria-hidden</u> (state) <u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>

Characteristic	Value
Name From:	author
Children Presentational:	False

meter role

An [element](#) that represents a scalar measurement within a known range, or a fractional value. See related [progressbar](#).

Authors **MAY** set [aria-valuemin](#) and [aria-valuemax](#) to indicate the minimum and maximum values for the **meter**. Otherwise, their implicit values follow the same rules as `<input[type="range"]>` in [HTML]:

- If [aria-valuemin](#) is missing or not a [number](#), it defaults to 0 (zero).
- If [aria-valuemax](#) is missing or not a [number](#), it defaults to 100.

The value of [aria-valuenow](#) **MUST NOT** fall below or exceed the computed values of [aria-valuemin](#) and [aria-valuemax](#), respectively.

Authors **SHOULD NOT** use the **meter** role to indicate progress; the [progressbar](#) role exists to address that need.

NOTE

Presently, there are no WAI-ARIA properties corresponding to the `low`, `optimum`, and `high` attributes supported on the `<meter>` element in [HTML]. The addition of these properties will be considered for ARIA version 1.3.

Characteristics:

Characteristic	Value
Superclass Role:	range
Related Concepts:	<code><meter></code> in [HTML]
Required States and Properties:	aria-valuenow
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details

Characteristic	Value
	<p><u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-dropeffect</u></p> <p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p> <p><u>aria-valuemax</u></p> <p><u>aria-valuemin</u></p> <p><u>aria-valuetext</u></p>
Name From:	author
Accessible Name Required:	True
Children Presentational:	True
Implicit Value for Role:	<p>Default for <u>aria-valuemin</u> is <i>0</i>.</p> <p>Default for <u>aria-valuemax</u> is <i>100</i>.</p>

menu role

A type of [widget](#) that offers a list of choices to the user.

A menu is often a list of common actions or functions that the user can invoke. The [menu role](#) is appropriate

when a list of menu items is presented in a manner similar to a menu on a desktop application.

To be [keyboard accessible](#), authors **SHOULD** manage focus of descendants for all instances of this [role](#), as described in [Managing Focus](#).

Elements with the role `menu` have an implicit [aria-orientation](#) value of `vertical`.

Characteristics:

Characteristic	Value
Superclass Role:	select
Subclass Roles:	menubar
Related Concepts:	list
Required Owned Elements:	group → menuitem group → menuitemradio group → menuitemcheckbox menuitem menuitemcheckbox menuitemradio
Inherited States and Properties:	aria-activedescendant aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2)

Characteristic	Value
	<p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-orientation</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	author
Implicit Value for Role:	Default for aria-orientation is vertical .

menubar role

A presentation of [menu](#) that usually remains visible and is usually presented horizontally.

The menubar [role](#) is used to create a menu bar similar to those found in Windows, Mac, and Gnome desktop applications. A menu bar is used to create a consistent set of frequently used commands. Authors **SHOULD** ensure that menubar interaction is similar to the typical menu bar interaction in a desktop graphical user interface.

To be [keyboard accessible](#), authors **SHOULD** manage focus of descendants for all instances of this [role](#), as described in [Managing Focus](#).

Elements with the role menubar have an implicit [aria-orientation](#) value of **horizontal**.

Characteristics:

Characteristic	Value
Superclass Role:	menu
Related Concepts:	toolbar
Required Owned Elements:	<p>group → menuitem</p> <p>group → menuitemradio</p> <p>group → menuitemcheckbox</p>

Characteristic	Value
	<u>menuitem</u> <u>menuitemcheckbox</u> <u>menuitemradio</u>
Inherited States and Properties:	<u>aria-activedescendant</u> <u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) <u>aria-dropeffect</u> <u>aria-errormessage</u> (deprecated on this role in ARIA 1.2) <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-haspopup</u> (deprecated on this role in ARIA 1.2) <u>aria-hidden</u> (state) <u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-orientation</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	author

Characteristic	Value
Implicit Value for Role:	Default for aria-orientation is horizontal.

menuitem role

An option in a set of choices contained by a [menu](#) or [menubar](#).

Authors **MAY** disable a menu item with the [aria-disabled](#) attribute. If the menu item has its [aria-haspopup](#) attribute set to **true**, it indicates that the menu item may be used to launch a sub-level menu, and authors **SHOULD** display a new sub-level menu when the menu item is activated.

In order to identify that they are related [widgets](#), authors **MUST** ensure that menu items are [owned](#) by an element with role [menu](#) or [menubar](#). Authors **MAY** separate menu items into sets by use of a [separator](#) or an element with an equivalent role from the native markup language.

Characteristics:

Characteristic	Value
Superclass Role:	command
Subclass Roles:	menuitemcheckbox
Related Concepts:	listitem option
Required Context Role:	group menu menubar
Supported States and Properties:	aria-disabled aria-expanded aria-haspopup aria-posinset aria-setsize
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state)

Characteristic	Value
	<p><u>aria-describedby</u></p> <p><u>aria-details</u></p> <p><u>aria-dropeffect</u></p> <p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Name From:	contents author
Accessible Name Required:	True

menuitemcheckbox role

A [menuitem](#) with a checkable state whose possible values are `true`, `false`, or `mixed`.

The [aria-checked](#) attribute of a [menuitemcheckbox](#) indicates whether the menu item is checked (`true`), unchecked (`false`), or represents a sub-level menu of other menu items that have a mixture of checked and unchecked values (`mixed`).

In order to identify that they are related [widgets](#), authors **MUST** ensure that menu item checkboxes are [owned](#) by an element with role [menu](#) or [menubar](#). Authors **MAY** separate menu items into sets by use of a [separator](#) or an element with an equivalent role from the native markup language.

Characteristics:

Characteristic	Value
Superclass Role:	menuitem
Subclass Roles:	menuitemradio
Related Concepts:	menuitem
Required Context Role:	group menu menubar
Required States and Properties:	aria-checked
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-expanded (state) aria-flowto aria-grabbed (state) aria-haspopup aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live

Characteristic	Value
	aria-owns aria-posinset aria-relevant aria-roledescription aria-setsize
Name From:	contents author
Accessible Name Required:	True
Children Presentational:	True

menuitemradio role

A checkable [menuitem](#) in a set of elements with the same role, only one of which can be checked at a time.

Authors **SHOULD** enforce that only one [menuitemradio](#) in a group can be checked at the same time. When one item in the group is checked, the previously checked item becomes unchecked (its [aria-checked](#) attribute becomes `false`).

In order to identify that they are related [widgets](#), authors **MUST** ensure that menu item radios are [owned](#) by an element with role [menu](#) or [menubar](#), or by a role [group](#) which itself is [owned](#) by an element with role [menu](#) or [menubar](#).

If a [menu](#) or [menubar](#) contains more than one group of [menuitemradio](#) elements, or if the menu contains one group and other, unrelated menu items, authors **SHOULD** contain each set of related [menuitemradio](#) elements in an element using the [group](#) role. Authors **MAY** also delimit the group from other menu items with an element using the [separator](#) role, or an element with an equivalent role from the native markup language.

Characteristics:

Characteristic	Value
Superclass Role:	menuitemcheckbox
Related Concepts:	menuitem

Characteristic	Value
Required Context Role:	group menu menubar
Inherited States and Properties:	aria-atomic aria-busy (state) aria-checked (state) (required) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-expanded (state) aria-flowto aria-grabbed (state) aria-haspopup aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live aria-owns aria-posinset aria-relevant aria-roledescription aria-setsize

Characteristic	Value
Name From:	contents author
Accessible Name Required:	True
Children Presentational:	True

navigation role

A [landmark](#) containing a collection of navigational [elements](#) (usually links) for navigating the document or related documents.

User agents **SHOULD** treat elements with the role of `navigation` as navigational [landmarks](#).

Characteristics:

Characteristic	Value
Superclass Role:	landmark
Related Concepts:	<code><nav></code> in [HTML]
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2)

Characteristic	Value
	ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live aria-owns aria-relevant aria-roledescription
Name From:	author

none role

An [element](#) whose implicit native role semantics will not be mapped to the [accessibility API](#). See synonym [presentation](#).

NOTE

Note regarding the ARIA 1.1 none role.

In ARIA 1.1, the working group introduced **none** as a synonym to the [presentation](#) role, due to author confusion surrounding the intended meaning of the word "presentation" or "presentational." Many individuals erroneously consider `role="presentation"` to be synonymous with `aria-hidden="true"`, and we believe `role="none"` conveys the actual meaning more unambiguously.

note role

A section whose content is parenthetic or ancillary to the main content of the resource.

Characteristics:

Characteristic	Value
Superclass Role:	section

Characteristic	Value
Inherited States and Properties:	<p>aria-atomic</p> <p>aria-busy (state)</p> <p>aria-controls</p> <p>aria-current (state)</p> <p>aria-describedby</p> <p>aria-details</p> <p>aria-disabled (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-dropeffect</p> <p>aria-errormessage (deprecated on this role in ARIA 1.2)</p> <p>aria-flowto</p> <p>aria-grabbed (state)</p> <p>aria-haspopup (deprecated on this role in ARIA 1.2)</p> <p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	author

option role

A selectable item in a [listbox](#).

Authors **MUST** ensure [elements](#) with [role](#) option are contained in, or [owned](#) by, an element with the [role](#)

[listbox](#) or [group](#) within a [listbox](#). Options not associated with a [listbox](#) might not be correctly mapped to an [accessibility API](#).

Elements with the role option have an implicit [aria-selected](#) value of `false`.

Characteristics:

Characteristic	Value
Superclass Role:	input
Subclass Roles:	treeitem
Base Concept:	<code><option></code> in [HTML]
Related Concepts:	listitem
Required Context Role:	group listbox
Required States and Properties:	aria-selected
Supported States and Properties:	aria-checked aria-posinset aria-setsize
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state)

Characteristic	Value
	<p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Name From:	contents author
Accessible Name Required:	True
Children Presentational:	True
Implicit Value for Role:	Default for <u>aria-selected</u> is false .

paragraph role

A paragraph of content.

Characteristics:

Characteristic	Value
Superclass Role:	<u>section</u>
Related Concepts:	<p> in [HTML]

Characteristic	Value
Inherited States and Properties:	<u>aria-atomic</u>
	<u>aria-busy</u> (state)
	<u>aria-controls</u>
	<u>aria-current</u> (state)
	<u>aria-describedby</u>
	<u>aria-details</u>
	<u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2)

Characteristic	Value
	<p><u>aria-dropeffect</u></p> <p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Prohibited States and Properties:	<p><u>aria-label</u></p> <p><u>aria-labelledby</u></p>
Name From:	prohibited

presentation role

An [element](#) whose implicit native role semantics will not be mapped to the [accessibility API](#). See synonym [none](#).

NOTE

Note regarding the ARIA 1.1 none role.

In ARIA 1.1, the working group introduced none as a synonym to the presentation role, due to author confusion surrounding the intended meaning of the word "presentation" or "presentational." Many individuals erroneously consider `role="presentation"` to be synonymous with `aria-hidden="true"`, and we believe `role="none"` conveys the actual meaning more unambiguously.

Until implementations include sufficient support for `role="none"`, web authors are advised to use the presentation role alone `role="presentation"` or redundantly as a fallback to the none role `role="none presentation"`.

The intended use is when an element is used to change the look of the page but does not have all the functional, interactive, or structural relevance implied by the element type, or may be used to provide for an accessible fallback in older browsers that do not support WAI-ARIA.

Example use cases:

- An element whose content is completely presentational (like a spacer image, decorative graphic, or clearing element);
- An image that is in a container with the img role and where the full text alternative is available and is marked up with aria-labelledby and (if needed) aria-describedby;
- An element used as an additional markup "hook" for CSS; or
- A layout table and/or any of its associated rows, cells, etc.

For any element with a role of presentation and which is not focusable, the user agent **MUST NOT** expose the implicit native semantics of the element (the role and its states and properties) to accessibility APIs. However, the user agent **MUST** expose content and descendant elements that do not have an explicit or inherited role of presentation. Thus, the presentation role causes a given element to be treated as having no role or to be removed from the accessibility tree, but does not cause the content contained within the element to be removed from the accessibility tree.

For example, according to an accessibility API, the following markup elements would appear to have identical role semantics (no role) and identical content.

EXAMPLE 9

```
<!-- 1. [role="presentation"] negates the implicit 'heading' role semantics but does not affect the native semantic of h1 -->
<h1 role="presentation"> Sample Content </h1>

<!-- 2. There is no implicit role for span, so only the contents are exposed. -->
<span> Sample Content </span>

<!-- 3. Depending on styling and other factors, this role declaration is redundant in some cases -->
<span role="presentation"> Sample Content </span>

<!-- 4. In all cases, the element contents are exposed to accessibility APIs without any impact -->
<!-- <> --> Sample Content <!-- </> -->
```

The `presentation` role is used on an element that has implicit native semantics, meaning that there is a default accessibility API role for the element. Some elements are only complete when additional descendant elements are provided. For example, in `HTML`, table elements (matching the `table` role) require `tr` descendants (the `row` role), which in turn require `th` or `td` children (the `cell`, `columnheader`, `rowheader` roles). Similarly, lists require list item children. The descendant elements that complete the semantics of an element are described in [WAI-ARIA](#) as [required owned elements](#).

When an explicit or inherited role of `presentation` is applied to an element with the implicit semantic of a [WAI-ARIA](#) role that has [required owned elements](#), in addition to the element with the explicit role of `presentation`, the user agent **MUST** apply an inherited role of presentation to any owned elements that do not have an explicit role defined. Also, when an explicit or inherited role of presentation is applied to a host language element which has required children as defined by the host language specification, in addition to the element with the explicit role of presentation, the user agent **MUST** apply an inherited role of presentation to any required children that do not have an explicit role defined.

In `HTML`, the `` [element](#) is treated as a single entity regardless of the type of image file. Consequently, using `role="presentation"` or `role="none"` on an `HTML img` is equivalent to using `aria-hidden="true"`. In order to make the image contents accessible, authors can embed the object using an `<object>` or `<iframe>` [element](#), or use inline `SVG` code, and follow the accessibility guidelines for the image content.

For any element with an explicit or inherited role of `presentation` and which is not focusable, user agents **MUST** ignore role-specific [WAI-ARIA](#) states and properties for that element. For example, in `HTML`, a `ul` or `ol` element with a role of `presentation` will have the implicit native semantics of its `li` elements removed because the `list` role to which the `ul` or `ol` corresponds has a [required owned element](#) of `listitem`. Likewise, the implicit native semantics of an `HTML table` element's `thead/tbody/tfoot/tr/th/td` descendants will also be removed, because the `HTML` specification indicates that these are required structural descendants of the `table` element.

NOTE

Only the implicit native semantics of elements that correspond to [WAI-ARIA required owned elements](#) are removed. All other content remains intact, including nested tables or lists, unless those elements also have an explicit role of **presentation** applied.

For example, according to an accessibility API, the following markup elements would appear to have identical role semantics (no roles) and identical content.

EXAMPLE 10

```
<!-- 1. [role="presentation"] negates the implicit 'list' and 'listitem' role semantics but
<ul role="presentation">
  <li> Sample Content </li>
  <li> More Sample Content </li>
</ul>

<!-- 2. There is no implicit role for "foo", so only the contents are exposed. -->
<foo>
  <foo> Sample Content </foo>
  <foo> More Sample Content </foo>
</foo>
```

NOTE

There are other [WAI-ARIA](#) roles with required children for which this situation is applicable (e.g., radiogroups and listboxes), but tables and lists are the most common real-world cases in which the presentation inheritance is likely to apply.

For any element with an explicit or inherited role of **presentation**, user agents **MUST** apply an inherited role of **presentation** to all host-language-specific labeling elements for the presentational element. For example, a **table** element with a role of **presentation** will have the implicit native semantics of its **caption** element removed, because the caption is merely a label for the presentational table.

Authors **SHOULD NOT** provide meaningful alternative text (for example, use `alt=""` in HTML) when the **presentation** role is applied to an image.

In the following code sample, the containing [`img`](#) and is appropriately labeled by the caption paragraph. In this example the `img` element can be marked as presentation because the role and the text alternatives are provided by the containing element.

EXAMPLE 11

```
<div role="img" aria-labelledby="caption">
  
  <p id="caption">A visible text caption labeling the image.</p>
</div>
```

In the following code sample, because the anchor (HTML `a` element) is acting as the treeitem, the list item (HTML `li` element) is assigned an explicit WAI-ARIA role of presentation to override the user agent's implicit native semantics for list items.

EXAMPLE 12

```
<ul role="tree">
  <li role="presentation">
    <a role="treeitem" aria-expanded="true">An expanded tree node</a>
  </li>
  ...
</ul>
```

Presentational Roles Conflict Resolution

There are a number of ways presentational role conflicts are resolved.

User agents **MUST NOT** expose elements having explicit or inherited presentational role in the accessibility tree, with these exceptions:

- If an element is focusable, or otherwise interactive, user agents **MUST** ignore the `presentation` role and expose the element with its implicit role, in order to ensure that the element is operable.
- If a required owned element has an explicit non-presentational role, user agents **MUST** ignore an inherited presentational role and expose the element with its explicit role. If the action of exposing the explicit role causes the accessibility tree to be malformed, the expected results are undefined.
- If an element has global WAI-ARIA states or properties, user agents **MUST** ignore the `presentation` role and expose the element with its implicit role. However, if an element has only non-global, role-specific WAI-ARIA states or properties, the element **MUST NOT** be exposed unless the presentational role is inherited and an explicit non-presentational role is applied.

For example, `aria-describedby` is a global attribute and would always be applied; `aria-level` is not a global attribute and would therefore only apply if the element was not in a presentational state.

EXAMPLE 13

```
<!-- 1. [role="presentation"] is ignored due to the global aria-describedby property. -->
<h1 role="presentation" aria-describedby="comment-1"> Sample Content </h1>
<!-- 2. [role="presentation"] negates both the implicit 'heading' and the non-global aria-i
<h1 role="presentation" aria-level="2"> Sample Content </h1>
```

Characteristics:

Characteristic	Value
Superclass Role:	<u>structure</u>
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-dropeffect</u> <u>aria-errormessage</u> (deprecated on this role in ARIA 1.2) <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-haspopup</u> (deprecated on this role in ARIA 1.2) <u>aria-hidden</u> (state) <u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-keyshortcuts</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>

Characteristic	Value
Prohibited States and Properties:	aria-label aria-labelledby
Name From:	prohibited

progressbar role

An [element](#) that displays the progress status for tasks that take a long time.

A progressbar indicates that the user's request has been received and the application is making progress toward completing the requested action.

Authors **MAY** set [aria-valuemin](#) and [aria-valuemax](#) to indicate the minimum and maximum progress indicator values. Otherwise, their implicit values follow the same rules as `<input[type="range"]>` in [\[HTML\]](#):

- If `aria-valuemin` is missing or not a [number](#), it defaults to 0 (zero).
- If `aria-valuemax` is missing or not a [number](#), it defaults to 100.

The author **SHOULD** supply a value for [aria-valuenow](#) unless the value is indeterminate, in which case the author **SHOULD** omit the [aria-valuenow](#) attribute. Authors **SHOULD** update this value when the visual progress indicator is updated. If the **progressbar** is describing the loading progress of a particular region of a page, the author **SHOULD** use [aria-describedby](#) to point to the status, and set the [aria-busy](#) attribute to `true` on the region until it is finished loading. It is not possible for the user to alter the value of a **progressbar** because it is always read-only.

NOTE

Assistive technologies generally will render the value of [aria-valuenow](#) as a percent of a range between the value of [aria-valuemin](#) and [aria-valuemax](#), unless [aria-valuetext](#) is specified.

Characteristics:

Characteristic	Value
Superclass Role:	range widget
Related Concepts:	status
Inherited States and Properties:	aria-atomic

Characteristic	Value
	<p><u>aria-busy</u> (state)</p> <p><u>aria-controls</u></p> <p><u>aria-current</u> (state)</p> <p><u>aria-describedby</u></p> <p><u>aria-details</u></p> <p><u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-dropeffect</u></p> <p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p> <p><u>aria-valuemax</u></p> <p><u>aria-valuemin</u></p> <p><u>aria-valuenow</u></p> <p><u>aria-valuetext</u></p>
Name From:	author
Accessible Name Required:	True
Children Presentational:	True

Characteristic	Value
Implicit Value for Role:	Default for aria-valuemin is <i>0</i> . Default for aria-valuemax is <i>100</i> .

radio role

A checkable input in a group of elements with the same role, only one of which can be checked at a time.

Authors **SHOULD** ensure that [elements](#) with role **radio** are explicitly grouped in order to indicate which ones affect the same value. This is achieved by enclosing the radio elements in an element with role [radiogroup](#). If it is not possible to make the radio buttons [DOM](#) children of the [radiogroup](#), authors **SHOULD** use the [aria-owns attribute](#) on the [radiogroup](#) element to indicate the [relationship](#) to its children.

Characteristics:

Characteristic	Value
Superclass Role:	input
Related Concepts:	<input[type="radio"]> in [HTML]
Required States and Properties:	aria-checked
Supported States and Properties:	aria-posinset aria-setsize
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state)

Characteristic	Value
	<p>aria-haspopup (deprecated on this role in ARIA 1.2)</p> <p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	contents author
Accessible Name Required:	True
Children Presentational:	True

radiogroup role

A group of [radio](#) buttons.

A radiogroup is a type of [select](#) list that can only have a single entry checked at any one time. Authors **SHOULD** enforce that only one radio button in a group can be checked at the same time. When one item in the group is checked, the previously checked item becomes unchecked (its [aria-checked](#) attribute becomes `false`).

Characteristics:

Characteristic	Value
Superclass Role:	select
Related Concepts:	list
Required Owned Elements:	radio
Supported States and Properties:	aria-errormessage aria-invalid

Characteristic	Value
	<u>aria_READONLY</u> <u>aria_REQUIRED</u>
Inherited States and Properties:	<u>aria_ACTIVEDESCENDANT</u> <u>aria_ATOMIC</u> <u>aria_BUSY</u> (state) <u>aria_CONTROLS</u> <u>aria_CURRENT</u> (state) <u>aria_DESCIBEDBY</u> <u>aria_DETAILS</u> <u>aria_DISABLED</u> (state) <u>aria_DROPEFFECT</u> <u>aria_FLOWTO</u> <u>aria_GRABBED</u> (state) <u>aria_HASPOPUP</u> (deprecated on this role in ARIA 1.2) <u>aria_HIDDEN</u> (state) <u>aria_KEYSHORTCUTS</u> <u>aria_LABEL</u> <u>aria_LABELLEDBY</u> <u>aria_LIVE</u> <u>aria_ORIENTATION</u> <u>aria_OWNS</u> <u>aria_RELEVANT</u> <u>aria_ROLEDESCRIPTION</u>
Name From:	author
Accessible Name Required:	True

range role

An element representing a range of values.

NOTE

`range` is an abstract role used for the ontology. Authors should not use this role in content.

Characteristics:

Characteristic	Value
Is Abstract:	True
Superclass Role:	structure
Subclass Roles:	meter progressbar scrollbar slider spinbutton
Supported States and Properties:	aria-valuemax aria-valuemin aria-valuenow aria-valuetext
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2)

Characteristic	Value
	<p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	author

region role

A [landmark](#) containing content that is relevant to a specific, author-specified purpose and sufficiently important that users will likely want to be able to navigate to the section easily and to have it listed in a summary of the page. Such a page summary could be generated dynamically by a user agent or assistive technology.

Authors **SHOULD** limit use of the region role to sections containing content with a purpose that is not accurately described by one of the other [landmark](#) roles, such as [main](#), [complementary](#), or [navigation](#).

Authors **MUST** give each element with role region a brief label that describes the purpose of the content in the region. Authors **SHOULD** reference a visible label with [aria-labelledby](#) if a visible label is present. Authors **SHOULD** include the label inside of a heading whenever possible. The heading **MAY** be an instance of the standard host language heading element or an instance of an element with role [heading](#).

[Assistive technologies](#) **SHOULD** enable users to quickly navigate to elements with role region. Mainstream [user agents](#) **MAY** enable users to quickly navigate to elements with role region.

Characteristics:

Characteristic	Value
Superclass Role:	landmark
Related Concepts:	<section> in [HTML]
Inherited States and Properties:	aria-atomic

Characteristic	Value
	<p><u>aria-busy</u> (state)</p> <p><u>aria-controls</u></p> <p><u>aria-current</u> (state)</p> <p><u>aria-describedby</u></p> <p><u>aria-details</u></p> <p><u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-dropeffect</u></p> <p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Name From:	author
Accessible Name Required:	True

roletype role

The base [role](#) from which all other roles inherit.

Properties of this role describe the structural and functional purpose of [objects](#) that are assigned this role. A

role is a concept that can be used to understand and operate instances.

NOTE

roletype is an abstract role used for the ontology. Authors should not use this role in content.

Characteristics:

Characteristic	Value
Is Abstract:	True
Subclass Roles:	structure widget window
Supported States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (Global use deprecated in ARIA 1.2) aria-dropeffect aria-errormessage (Global use deprecated in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (Global use deprecated in ARIA 1.2) aria-hidden (state) aria-invalid (state) (Global use deprecated in ARIA 1.2) aria-keyshortcuts aria-label (Except where prohibited) aria-labelledby (Except where prohibited) aria-live

Characteristic	Value
	aria-owns aria-relevant aria-roledescription
Name From:	n/a

row role

A row of cells in a tabular container.

Rows contain [cell](#) or [gridcell](#) elements, and thus serve to organize a [table](#), [grid](#), or [treemap](#).

While the row role can be used in a [table](#), [grid](#), or [treemap](#), the semantics of [aria-expanded](#), [aria-posinset](#), [aria-setsize](#), and [aria-level](#) are only applicable to the hierarchical structure of an interactive tree grid. Therefore, authors **MUST NOT** apply [aria-expanded](#), [aria-posinset](#), [aria-setsize](#), and [aria-level](#) to a [row](#) that descends from a [table](#) or [grid](#), and user agents **SHOULD NOT** expose any of these four properties to assistive technologies unless the [row](#) descends from a [treemap](#).

Authors **MUST** ensure elements with [role](#) [row](#) are contained in, or [owned](#) by, an element with the role [table](#), [grid](#), [rowgroup](#), or [treemap](#).

NOTE: Usage of aria-disabled

While [aria-disabled](#) is currently supported on [row](#), in a future version the working group plans to prohibit its on elements with role [row](#) except when the element is in the context of a [grid](#) or [treemap](#).

Characteristics:

Characteristic	Value
Superclass Role:	group widget
Base Concept:	<tr> in [HTML]
Required Context Role:	grid rowgroup table treemap

Characteristic	Value
Required Owned Elements:	cell columnheader gridcell rowheader
Supported States and Properties:	aria-colindex aria-expanded aria-level aria-posinset aria-rowindex aria-setsize aria-selected
Inherited States and Properties:	aria-activedescendant aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts

Characteristic	Value
	<u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	contents author

rowgroup role

A structure containing one or more row elements in a tabular container.

The **rowgroup** role establishes a relationship between owned row elements. It is a structural equivalent to the **thead**, **tfoot**, and **tbody** elements in an HTML table element.

Authors **MUST** ensure elements with role **rowgroup** are contained in, or owned by, an element with the role grid, table, or treegrid.

NOTE

The **rowgroup** role exists, in part, to support role symmetry in HTML, and allows for the propagation of presentation inheritance on HTML table elements with an explicit presentation role applied.

NOTE

This role does not differentiate between types of row groups (e.g., **thead** vs. **tbody**), but an issue has been raised for WAI-ARIA 2.0.

Characteristics:

Characteristic	Value
Superclass Role:	<u>structure</u>
Base Concept:	tbody, tfoot and thead in [HTML]
Required Context Role:	<u>grid</u> <u>table</u>

Characteristic	Value
	treegrid
Required Owned Elements:	row
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live aria-owns aria-relevant aria-roledescription
Name From:	author

rowheader role

A cell containing header information for a row.

The [rowheader](#) role can be used to identify a cell as a header for a row in a [table](#), [grid](#), or [treemap](#). The rowheader establishes a [relationship](#) between it and all cells in the corresponding row. It is a structural equivalent to setting `scope="row"` on an [HTML th element](#).

Authors **MUST** ensure [elements](#) with [role](#) `rowheader` are contained in, or [owned](#) by, an element with the role [row](#).

Applying the [aria-selected](#) state on a rowheader **MUST NOT** cause the user agent to automatically propagate the [aria-selected](#) state to all the cells in the corresponding row. An author **MAY** choose to propagate selection in this manner depending on the specific application.

While the `rowheader` role can be used in both interactive grids and non-interactive tables, the use of [aria-expanded](#), [aria_READONLY](#), and [aria-required](#) is only applicable to interactive elements. Therefore, authors **SHOULD NOT** use [aria-expanded](#), [aria_READONLY](#), or [aria-required](#) in a `rowheader` that descends from a [table](#), and user agents **SHOULD NOT** expose these properties to [assistive technologies](#) unless the `rowheader` descends from a [grid](#) or [treemap](#).

NOTE: Usage of `aria-disabled`

While [aria-disabled](#) is currently supported on [rowheader](#), in a future version the working group plans to prohibit its use on elements with role [rowheader](#) except when the element is in the context of a [grid](#) or [treemap](#).

Characteristics:

Characteristic	Value
Superclass Role:	cell gridcell sectionhead
Base Concept:	<code><th[scope="row"]></code> in [HTML]
Required Context Role:	row
Supported States and Properties:	aria-expanded aria-sort
Inherited States and Properties:	aria-atomic aria-busy (state) aria-colindex aria-colspan

Characteristic	Value
	<u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) <u>aria-dropeffect</u> <u>aria-errormessage</u> <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-haspopup</u> <u>aria-hidden</u> (state) <u>aria-invalid</u> (state) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria_READONLY</u> <u>aria-relevant</u> <u>aria-required</u> <u>aria-roledescription</u> <u>aria-rowindex</u> <u>aria-rowspan</u> <u>aria-selected</u> (state)
Name From:	contents author
Accessible Name Required:	True

scrollbar role

A graphical object that controls the scrolling of content within a viewing area, regardless of whether the content is fully displayed within the viewing area.

A scrollbar represents the current value and range of possible values via the size of the scrollbar and position of the thumb with respect to the visible range of the orientation (horizontal or vertical) it controls. Its orientation represents the orientation of the scrollbar and the scrolling effect on the viewing area controlled by the scrollbar. It is typically possible to add or subtract to the current value by using directional keys such as arrow keys.

Authors **MUST** set the [aria-controls](#) attribute on the scrollbar element to reference the scrollable area it controls.

Authors **MAY** set [aria-valuemin](#) and [aria-valuemax](#) to indicate the minimum and maximum thumb position. Otherwise, their implicit values follow the same rules as `<input[type="range"]>` in [HTML]:

- If [aria-valuemin](#) is missing or not a [number](#), it defaults to 0 (zero).
- If [aria-valuemax](#) is missing or not a [number](#), it defaults to 100.

Authors **MUST** set the [aria-valuenow](#) attribute to indicate the current thumb position. If [aria-valuenow](#) is missing or has an unexpected value, browsers **MAY** implement the repair techniques specified in the [section describing handling author errors in states and properties](#), which are equivalent to the repair techniques for `<input[type="range"]>` in [HTML].

Elements with the role **scrollbar** have an implicit [aria-orientation](#) value of **vertical**.

NOTE

Assistive technologies generally will render the value of [aria-valuenow](#) as a percent of a range between the value of [aria-valuemin](#) and [aria-valuemax](#), unless [aria-valuetext](#) is specified. It is best to set the values for [aria-valuemin](#), [aria-valuemax](#), and [aria-valuenow](#) in a manner that is appropriate for this calculation.

Characteristics:

Characteristic	Value
Superclass Role:	range widget
Required States and Properties:	aria-controls aria-valuenow
Supported States and Properties:	aria-disabled

Characteristic	Value
	<u>aria-orientation</u> <u>aria-valuemax</u> <u>aria-valuemin</u>
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-dropeffect</u> <u>aria-errormessage</u> (deprecated on this role in ARIA 1.2) <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-haspopup</u> (deprecated on this role in ARIA 1.2) <u>aria-hidden</u> (state) <u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u> <u>aria-valuetext</u>
Name From:	author
Accessible Name Required:	False
Children Presentational:	True
Implicit Value for Role:	Default for <u>aria-orientation</u> is vertical . Default for <u>aria-valuemin</u> is 0 .

Characteristic	Value
	Default for aria-valuemax is 100.

search role

A [landmark](#) region that contains a collection of items and objects that, as a whole, combine to create a search facility. See related [form](#) and [searchbox](#).

A search region may be a mix of host language form controls, scripted controls, and hyperlinks.

User agents **SHOULD** treat elements with the role of **search** as navigational [landmarks](#).

Characteristics:

Characteristic	Value
Superclass Role:	landmark
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label

Characteristic	Value
	aria-labelledby aria-live aria-owns aria-relevant aria-roledescription
Name From:	author

searchbox role

A type of textbox intended for specifying search criteria. See related [textbox](#) and [search](#).

Characteristics:

Characteristic	Value
Superclass Role:	textbox
Base Concept:	<input[type="search"]> in [HTML]
Inherited States and Properties:	aria-activedescendant aria-atomic aria-autocomplete aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-errormessage aria-flowto aria-grabbed (state) aria-haspopup aria-hidden (state) aria-invalid (state)

Characteristic	Value
	<u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-multiline</u> <u>aria-owns</u> <u>aria-placeholder</u> <u>aria-readonly</u> <u>aria-relevant</u> <u>aria-required</u> <u>aria-roledescription</u>
Name From:	author
Accessible Name Required:	True

section role

A renderable structural containment unit in a document or application.

NOTE

section is an abstract role used for the ontology. Authors should not use this role in content.

Characteristics:

Characteristic	Value
Is Abstract:	True
Superclass Role:	<u>structure</u>
Subclass Roles:	<u>alert</u> <u>blockquote</u> <u>caption</u> <u>cell</u> <u>code</u> <u>definition</u>

Characteristic	Value
	<u>deletion</u> <u>emphasis</u> <u>figure</u> <u>group</u> <u>img</u> <u>insertion</u> <u>landmark</u> <u>list</u> <u>listitem</u> <u>log</u> <u>marquee</u> <u>math</u> <u>note</u> <u>paragraph</u> <u>status</u> <u>strong</u> <u>subscript</u> <u>superscript</u> <u>table</u> <u>tabpanel</u> <u>term</u> <u>time</u> <u>tooltip</u>
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) (deprecated on this role in

Characteristic	Value
	<p>ARIA 1.2)</p> <p><u>aria-dropeffect</u></p> <p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Name From:	n/a

sectionhead role

A structure that labels or summarizes the topic of its related section.

NOTE

sectionhead is an abstract role used for the ontology. Authors should not use this role in content.

Characteristics:

Characteristic	Value
Is Abstract:	True
Superclass Role:	<u>structure</u>

Characteristic	Value
Subclass Roles:	columnheader heading rowheader tab
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live aria-owns aria-relevant aria-roledescription
Name From:	contents

Characteristic	Value
	author

select role

A form widget that allows the user to make selections from a set of choices.

NOTE

`select` is an abstract role used for the ontology. Authors should not use this role in content.

Characteristics:

Characteristic	Value
Is Abstract:	True
Superclass Role:	composite group
Subclass Roles:	listbox menu radiogroup tree
Supported States and Properties:	aria-orientation
Inherited States and Properties:	aria-activedescendant aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto

Characteristic	Value
	<p>aria-grabbed (state)</p> <p>aria-haspopup (deprecated on this role in ARIA 1.2)</p> <p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	author

separator role

A divider that separates and distinguishes sections of content or groups of menuitems.

There are two types of separators: a static [structure](#) that provides only a visible boundary and a focusable, interactive [widget](#) that is also moveable. If a **separator** is not focusable, it is revealed to [assistive technologies](#) as a static structural element. For example, a static **separator** can be used to help visually divide two groups of menu items in a menu or to provide a horizontal rule between two sections of a page.

Authors **MAY** make a **separator** focusable to create a [widget](#) that both provides a visible boundary between two sections of content and enables the user to change the relative size of the sections by changing the position of the **separator**. A variable **separator** widget can be moved continuously within a range, whereas a fixed **separator** widget supports only two discrete positions. Typically, a fixed **separator** widget is used to toggle one of the sections between expanded and collapsed states.

If the **separator** is focusable, authors **MUST** set the value of [aria-valuenow](#) to a [number](#) reflecting the current position of the **separator** and update that value when it changes. Authors **SHOULD** also provide the value of [aria-valuemin](#) if it is not **0** and the value of [aria-valuemax](#) if it is not **100**. If missing or not a number, the implicit values of these attributes are as follows:

- The implicit value of [aria-valuemin](#) is **0**.

- The implicit value of `aria-valuemax` is 100.

In applications where there is more than one focusable `separator`, authors **SHOULD** provide an accessible name for each one.

Elements with the role `separator` have an implicit `aria-orientation` value of `horizontal`.

Characteristics:

Characteristic	Value
Superclass Role:	<code>structure</code> (if not focusable) <code>widget</code> (if focusable)
Related Concepts:	<code><hr></code> in [HTML]
Required States and Properties:	<code>aria-valuenow</code> (if focusable)
Supported States and Properties:	<code>aria-disabled</code> (if focusable) <code>aria-orientation</code> <code>aria-valuemax</code> (if focusable) <code>aria-valuemin</code> (if focusable) <code>aria-valuetext</code> (if focusable)
Inherited States and Properties:	<code>aria-atomic</code> <code>aria-busy</code> (state) <code>aria-controls</code> <code>aria-current</code> (state) <code>aria-describedby</code> <code>aria-details</code> <code>aria-dropeffect</code> <code>aria-errormessage</code> (deprecated on this role in ARIA 1.2) <code>aria-flowto</code> <code>aria-grabbed</code> (state) <code>aria-haspopup</code> (deprecated on this role in ARIA 1.2) <code>aria-hidden</code> (state) <code>aria-invalid</code> (state) (deprecated on this role in ARIA 1.2)

Characteristic	Value
	ARIA 1.2) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	author
Children Presentational:	True
Implicit Value for Role:	Default for <u>aria-orientation</u> is horizontal. Default for <u>aria-valuemin</u> is 0. Default for <u>aria-valuemax</u> is 100.

slider role

An input where the user selects a value from within a given range.

A slider represents the current value and range of possible values via the size of the slider and position of the thumb. It is typically possible to add or subtract to the value by using directional keys such as arrow keys.

Authors **MAY** set the [aria-valuemin](#) and [aria-valuemax](#) attributes. Otherwise, their implicit values follow the same rules as `<input[type="range"]>` in [HTML]:

- If `aria-valuemin` is missing or not a [number](#), it defaults to 0 (zero).
- If `aria-valuemax` is missing or not a [number](#), it defaults to 100.

Authors **MUST** set the [aria-valuenow](#) attribute. If `aria-valuenow` is missing or has an unexpected value, browsers **MAY** implement the repair techniques specified in the [section describing handling author errors in states and properties](#), which are equivalent to the repair techniques for `<input[type="range"]>` in [HTML].

Elements with the role `slider` have an implicit [aria-orientation](#) value of `horizontal`.

Characteristics:

Characteristic	Value
----------------	-------

Characteristic	Value
Superclass Role:	input range
Required States and Properties:	aria-valuenow
Supported States and Properties:	aria-errormessage aria-haspopup aria-invalid aria-orientation aria_READONLY aria-valuemax aria-valuemin
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-flowto aria-grabbed (state) aria-hidden (state) aria-keyshortcuts aria-label aria-labelledby aria-live aria-owns aria-relevant aria-roledescription aria-valuetext

Characteristic	Value
Name From:	author
Accessible Name Required:	True
Children Presentational:	True
Implicit Value for Role:	Default for aria-orientation is horizontal. Default for aria-valuemin is 0. Default for aria-valuemax is 100.

spinbutton role

A form of [range](#) that expects the user to select from among discrete choices.

A [spinbutton](#) typically allows users to change its displayed value by activating increment and decrement buttons that step through a set of allowed values. Some implementations display the value in an text field that allows editing and typing but typically limits input in ways that help prevent invalid values.

Although a [spinbutton](#) is similar in appearance to many presentations of [select](#), it is advisable to use [spinbutton](#) when working with known ranges (especially in the case of large ranges) as opposed to distinct options. For example, a [spinbutton](#) representing a range from 1 to 1,000,000 would provide much better performance than a [select](#) [widget](#) representing the same values.

Authors **MAY** create a [spinbutton](#) with children or owned elements, but **MUST** limit those elements to a [textbox](#) and/or two [buttons](#). Alternatively, authors **MAY** apply the [spinbutton](#) role to a text input and create sibling buttons to support the increment and decrement functions.

To be [keyboard accessible](#), authors **SHOULD** manage focus of descendants for all instances of this [role](#), as described in [Managing Focus](#). When a [spinbutton](#) receives focus, authors **SHOULD** ensure focus is placed on the [textbox](#) element if one is present, and on the [spinbutton](#) itself otherwise. Authors **SHOULD** also ensure the up and down arrows on a keyboard perform the increment and decrement functions and that the increment and decrement [button](#) elements are *NOT* included in the primary navigation ring, e.g., the Tab ring in [HTML](#).

Authors **SHOULD** set the [aria-valuenow](#) attribute when the [spinbutton](#) has a value. Authors **SHOULD** set the [aria-valuemin](#) attribute when there is a minimum value, and the [aria-valuemax](#) attribute when there is a maximum value.

Characteristics:

Characteristic	Value
Superclass Role:	composite input

Characteristic	Value
	<u>range</u>
Supported States and Properties:	<u>aria-errormessage</u> <u>aria-invalid</u> <u>aria_READONLY</u> <u>aria_REQUIRED</u> <u>aria-valuemax</u> <u>aria-valuemin</u> <u>aria-valuenow</u> <u>aria-valuetext</u>

Characteristic	Value
Inherited States and Properties:	<u>aria-activedescendant</u> <u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) <u>aria-dropeffect</u> <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-haspopup</u> (deprecated on this role in ARIA 1.2) <u>aria-hidden</u> (state) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	author
Accessible Name Required:	True
Implicit Value for Role:	Default for <u>aria-valuemin</u> is that there is no minimum value. Default for <u>aria-valuemax</u> is that there is no maximum value. Default for <u>aria-valuenow</u> is 0

status role

A type of [live region](#) whose content is advisory information for the user but is not important enough to justify

an [alert](#), often but not necessarily presented as a status bar.

Authors **SHOULD** ensure an element with role **status** does not receive focus as a result of change in status.

Status is a form of [live region](#). If another part of the page controls what appears in the status, authors **SHOULD** make the [relationship](#) explicit with the [aria-controls](#) attribute.

[Assistive technologies](#) **MAY** reserve some cells of a Braille display to render the status.

Elements with the role **status** have an implicit [aria-live](#) value of **polite** and an implicit [aria-atomic](#) value of **true**.

Characteristics:

Characteristic	Value
Superclass Role:	section
Subclass Roles:	timer
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label

Characteristic	Value
	aria-labelledby aria-live aria-owns aria-relevant aria-roledescription
Name From:	author
Implicit Value for Role:	Default for aria-live is polite. Default for aria-atomic is true.

strong role

Content that is important, serious, or urgent. See related [emphasis](#).

The purpose of the **strong** role is to communicate strong importance, seriousness, or urgency. It is not for communicating changes in typographical presentation that are not important to the meaning of the content. Authors **SHOULD** use the **strong** role only if its absence would change the meaning of the content.

The **strong** role is not intended to convey stress or emphasis; for that purpose, the [emphasis](#) role is more appropriate.

Characteristics:

Characteristic	Value
Superclass Role:	section
Related Concepts:	 in [HTML]
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2)

Characteristic	Value
	ARIA 1.2) <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-haspopup</u> (deprecated on this role in ARIA 1.2) <u>aria-hidden</u> (state) <u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-keyshortcuts</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Prohibited States and Properties:	<u>aria-label</u> <u>aria-labelledby</u>
Name From:	prohibited

structure role

A document structural [element](#).

[Roles](#) for document structure support the accessibility of dynamic web content by helping [assistive technologies](#) determine active content versus static document content. Structural roles by themselves do not all map to [accessibility APIs](#), but are used to create [widget](#) roles or assist content adaptation for assistive technologies.

NOTE

structure is an abstract role used for the ontology. Authors should not use this role in content.

Characteristics:

Characteristic	Value
Is Abstract:	True
Superclass Role:	roletype

Characteristic	Value
Subclass Roles:	application document generic presentation range rowgroup section sectionhead separator
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live

Characteristic	Value
	aria-owns aria-relevant aria-roledescription
Name From:	n/a

subscript role

One or more subscripted characters. See related [superscript](#).

The **subscript** role is intended to be used only to mark up typographical conventions that have specific meanings; not for typographical presentation for presentation's sake. In general, authors **SHOULD** use this role only if the absence of the subscript would change the meaning of the content.

Characteristics:

Characteristic	Value
Superclass Role:	section
Related Concepts:	<sub> and <sup> in [HTML]
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state)

Characteristic	Value
	<p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Prohibited States and Properties:	<p><u>aria-label</u></p> <p><u>aria-labelledby</u></p>
Name From:	prohibited

superscript role

One or more superscripted characters. See related [superscript](#).

The **superscript** role is intended to be used only to mark up typographical conventions that have specific meanings; not for typographical presentation for presentation's sake. In general, authors **SHOULD** use this role only if the absence of the superscript would change the meaning of the content.

Characteristics:

Characteristic	Value
Superclass Role:	<u>section</u>
Related Concepts:	<sub> and <sup> in [HTML]
Inherited States and Properties:	<p><u>aria-atomic</u></p> <p><u>aria-busy</u> (state)</p> <p><u>aria-controls</u></p> <p><u>aria-current</u> (state)</p> <p><u>aria-describedby</u></p> <p><u>aria-details</u></p> <p><u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-dropeffect</u></p>

Characteristic	Value
	<p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Prohibited States and Properties:	<p><u>aria-label</u></p> <p><u>aria-labelledby</u></p>
Name From:	prohibited

switch role

A type of checkbox that represents on/off values, as opposed to checked/unchecked values. See related [checkbox](#).

The [aria-checked](#) attribute of a switch indicates whether the input is on (`true`) or off (`false`). The `mixed` value is invalid, and user agents **MUST** treat a `mixed` value as equivalent to `false` for this role.

NOTE

A switch provides approximately the same functionality as a checkbox and toggle button, but makes it possible for assistive technologies to present the widget in a fashion consistent with its on-screen appearance.

Characteristics:

Characteristic	Value

Characteristic	Value
Superclass Role:	checkbox
Related Concepts:	button
Required States and Properties:	aria-checked
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-errormessage aria-expanded (state) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) aria-keyshortcuts aria-label aria-labelledby aria-live aria-owns aria_READONLY aria-relevant aria-required aria-roledescription

Characteristic	Value
Name From:	contents author
Accessible Name Required:	True
Children Presentational:	True

tab role

A grouping label providing a mechanism for selecting the tab content that is to be rendered to the user.

If a [tabpanel](#) or item in a [tabpanel](#) has focus, the associated [tab](#) is the currently active tab in the [tablist](#), as defined in [Managing Focus](#). [tablist](#) elements, which contain a set of associated [tab](#) elements, are typically placed near a series of [tabpanel](#) elements, usually preceding it. See the [WAI-ARIA Authoring Practices](#) for details on implementing a tab set design pattern.

Authors **MUST** ensure [elements with role tab](#) are contained in, or [owned](#) by, an element with the role [tablist](#).

Authors **SHOULD** ensure the [tabpanel](#) associated with the currently active tab is [perceivable](#) to the user.

For a single-selectable [tablist](#), authors **SHOULD** hide other [tabpanel](#) [elements](#) from the user until the user selects the tab associated with that [tabpanel](#). For a multi-selectable [tablist](#), authors **SHOULD** ensure that the [tab](#) for each visible [tabpanel](#) has the [aria-expanded](#) [attribute](#) set to [true](#), and that the [tabs](#) associated with the remaining hidden [tabpanel](#) elements have their [aria-expanded](#) [attributes](#) set to [false](#).

In either case, authors **SHOULD** ensure that a selected tab has its [aria-selected](#) [attribute](#) set to [true](#), that inactive tab elements have their [aria-selected](#) [attribute](#) set to [false](#), and that the currently selected tab provides a visual indication that it is selected. In the absence of an [aria-selected](#) [attribute](#) on the current tab, [user agents](#) **SHOULD** indicate to [assistive technologies](#) through the platform [accessibility API](#) that the currently focused tab is selected.

Characteristics:

Characteristic	Value
Superclass Role:	sectionhead widget
Required Context Role:	tablist
Supported States and Properties:	aria-disabled

Characteristic	Value
	<u>aria-expanded</u> <u>aria-haspopup</u> <u>aria-posinset</u> <u>aria-selected</u> <u>aria-setsize</u>
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-dropeffect</u> <u>aria-errormessage</u> (deprecated on this role in ARIA 1.2) <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-hidden</u> (state) <u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	contents author
Children Presentational:	True
Implicit Value for Role:	Default for <u>aria-selected</u> is false.

table role

A [section](#) containing data arranged in rows and columns. See related [grid](#).

The **table** role is intended for tabular containers which are not interactive. If the tabular container maintains a selection state, provides its own two-dimensional navigation, or allows the user to rearrange or otherwise manipulate its contents or the display thereof, authors **SHOULD** use [grid](#) or [treegrid](#) instead.

Authors **SHOULD** prefer the use of the host language's semantics for table whenever possible, such as the `<table>` element in [\[HTML\]](#).

Characteristics:

Characteristic	Value
Superclass Role:	section
Subclass Roles:	grid
Base Concept:	<code><table></code> in [HTML]
Required Owned Elements:	row rowgroup → row
Supported States and Properties:	aria-colcount aria-rowcount
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA)

Characteristic	Value
	1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live aria-owns aria-relevant aria-roledescription
Name From:	author
Accessible Name Required:	True

tablist role

A list of [tab](#) elements, which are references to [tabpanel](#) elements.

To be [keyboard accessible](#), authors **SHOULD** manage focus of descendants for all instances of this [role](#), as described in [Managing Focus](#).

For a single-selectable [tablist](#), authors **SHOULD** hide other [tabpanel](#) elements from the user until the user selects the tab associated with that [tabpanel](#). For a multi-selectable [tablist](#), authors **SHOULD** ensure each visible [tabpanel](#) has its [aria-expanded](#) attribute set to `true`, and that the remaining hidden [tabpanel](#) elements have their [aria-expanded](#) attributes set to `false`.

[tablist](#) elements are typically placed near usually preceding, a series of [tabpanel](#) elements. See the [WAI-ARIA Authoring Practices](#) for details on implementing a tab set design pattern.

Elements with the role [tablist](#) have an implicit [aria-orientation](#) value of `horizontal`.

Characteristics:

Characteristic	Value
Superclass Role:	composite
Required Owned Elements:	tab

Characteristic	Value
Supported States and Properties:	<u>aria-multiselectable</u> <u>aria-orientation</u>
Inherited States and Properties:	<u>aria-activedescendant</u> <u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) <u>aria-dropeffect</u> <u>aria-errormessage</u> (deprecated on this role in ARIA 1.2) <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-haspopup</u> (deprecated on this role in ARIA 1.2) <u>aria-hidden</u> (state) <u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-keyshortcuts</u> <u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	author
Implicit Value for Role:	Default for <u>aria-orientation</u> is horizontal.

tabpanel role

A container for the resources associated with a [tab](#), where each [tab](#) is contained in a [tablist](#).

Authors **SHOULD** associate a [tabpanel element](#) with its [tab](#), either by using the [aria-controls](#) attribute on the tab to reference the tab panel, or by using the [aria-labelledby](#) attribute on the tab panel to reference the tab.

[tablist](#) elements are typically placed near, usually preceding, a series of [tabpanel](#) elements. See the [WAI-ARIA Authoring Practices](#) for details on implementing a tab set design pattern.

Characteristics:

Characteristic	Value
Superclass Role:	section
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby

Characteristic	Value
	aria-live aria-owns aria-relevant aria-roledescription
Name From:	author
Accessible Name Required:	True

term role

A word or phrase with a corresponding definition. See related [definition](#).

The **term** role is used to explicitly identify a word or phrase for which a [definition](#) has been provided by the author or is expected to be provided by the user.

Authors **SHOULD NOT** use the **term** role on interactive elements such as links because doing so could prevent users of [assistive technologies](#) from interacting with those elements.

Characteristics:

Characteristic	Value
Superclass Role:	section
Related Concepts:	<dfn> in [HTML]
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state)

Characteristic	Value
	<p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Name From:	author

textbox role

A type of input that allows free-form text as its value.

If the [aria-multiline](#) attribute is true, the [widget](#) accepts line breaks within the input, as in an [HTML textarea](#). Otherwise, this is a simple text box. The intended use is for languages that do not have a text input [element](#), or cases in which an element with different [semantics](#) is repurposed as a text field.

NOTE

In most user agent implementations, the default behavior of the ENTER or RETURN key is different between the single-line and multi-line text fields in [HTML](#). When user has focus in a single-line `<input type="text">` element, the keystroke usually submits the form. When user has focus in a multi-line `<textarea>` element, the keystroke inserts a line break. The WAI-ARIA `textbox` role differentiates these types of boxes with the [aria-multiline](#) attribute, so authors are advised to be aware of this distinction when designing the field.

Characteristics:

Characteristic	Value
Superclass Role:	<u>input</u>
Subclass Roles:	<u>searchbox</u>

Characteristic	Value
Related Concepts:	<textarea> in [HTML] <input[type="text"]> in [HTML]
Supported States and Properties:	aria-activedescendant aria-autocomplete aria-errormessage aria-haspopup aria-invalid aria-multiline aria-placeholder aria-readonly aria-required
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-flowto aria-grabbed (state) aria-hidden (state) aria-keyshortcuts aria-label aria-labelledby aria-live aria-owns aria-relevant aria-roledescription

Characteristic	Value
Name From:	author
Accessible Name Required:	True

time role

An element that represents a specific point in time.

NOTE

At the present time, there are no WAI-ARIA properties corresponding to the `datetime` attribute supported on `<time>` in [HTML]. The addition of this property will be considered for ARIA version 1.3.

Authors **SHOULD** limit text contents to a valid date- or time-related string, or apply this future `datetime`-equivalent property to the element which has role `time`.

Examples of valid date- or time-related strings as text contents of an element with the `time` role:

- A valid month string: `2019-11`
- A valid date string: `2019-11-18`
- A valid yearless date string: `11-18`
- A valid time string: `09:54:39`
- A valid floating date and time string: `2019-11-18T14:54`
- A valid time-zone offset string: `-08:00`
- A valid global date and time string: `2019-11-18T14:54Z`
- A valid week string: `2019-W47`
- Four or more ASCII digits, at least one of which is not U+0030 DIGIT ZERO (0): `0001`
- A valid duration string: `4h 18m 3s`

Characteristics:

Characteristic	Value
Superclass Role:	section
Related Concepts:	<code><time></code> in [HTML]
Inherited States and Properties:	aria-atomic aria-busy (state)

Characteristic	Value
	<p>aria-controls</p> <p>aria-current (state)</p> <p>aria-describedby</p> <p>aria-details</p> <p>aria-disabled (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-dropeffect</p> <p>aria-errormessage (deprecated on this role in ARIA 1.2)</p> <p>aria-flowto</p> <p>aria-grabbed (state)</p> <p>aria-haspopup (deprecated on this role in ARIA 1.2)</p> <p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	author

timer role

A type of [live region](#) containing a numerical counter which indicates an amount of elapsed time from a start point, or the time remaining until an end point.

The text contents of the timer [object](#) indicate the current time measurement, and are updated as that amount changes. The timer value is not necessarily machine parsable, but authors **SHOULD** update the text contents at fixed intervals, except when the timer is paused or reaches an end-point.

Elements with the role **timer** have an implicit [**aria-live**](#) value of off.

Characteristics:

Characteristic	Value
Superclass Role:	status
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby aria-live aria-owns aria-relevant aria-roledescription
Name From:	author

toolbar role

A collection of commonly used function buttons or controls represented in compact visual form.

The toolbar is often a subset of functions found in a [menubar](#), designed to reduce user effort in using these functions. Authors **MUST** supply a label on each toolbar when the application contains more than one toolbar.

Authors **MAY** manage focus of descendants for all instances of this [role](#), as described in [Managing Focus](#).

Elements with the role **toolbar** have an implicit [aria-orientation](#) value of **horizontal**.

Characteristics:

Characteristic	Value
Superclass Role:	group
Related Concepts:	menubar
Supported States and Properties:	aria-orientation
Inherited States and Properties:	aria-activedescendant aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts

Characteristic	Value
	<u>aria-label</u> <u>aria-labelledby</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-relevant</u> <u>aria-roledescription</u>
Name From:	author
Implicit Value for Role:	Default for <u>aria-orientation</u> is horizontal.

tooltip role

A contextual popup that displays a description for an element.

The **tooltip** typically becomes visible, after a short delay, in response to a mouse hover, or after the owning element receives keyboard focus. The use of a WAI-ARIA tooltip is a supplement to the normal tooltip behavior of the user agent.

NOTE

Typical tooltip delays last from one to five seconds.

Authors **SHOULD** ensure that elements with the role **tooltip** are referenced through the use of aria-describedby before or at the time the tooltip is displayed.

Characteristics:

Characteristic	Value
Superclass Role:	<u>section</u>
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) (deprecated on this role in

Characteristic	Value
	<p>ARIA 1.2)</p> <p>aria-dropeffect</p> <p>aria-errormessage (deprecated on this role in ARIA 1.2)</p> <p>aria-flowto</p> <p>aria-grabbed (state)</p> <p>aria-haspopup (deprecated on this role in ARIA 1.2)</p> <p>aria-hidden (state)</p> <p>aria-invalid (state) (deprecated on this role in ARIA 1.2)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-owns</p> <p>aria-relevant</p> <p>aria-roledescription</p>
Name From:	contents author
Accessible Name Required:	True

tree role

A [widget](#) that allows the user to select one or more items from a hierarchically organized collection.

To be [keyboard accessible](#), authors **SHOULD** manage focus of descendants for all instances of this [role](#), as described in [Managing Focus](#).

Elements with the role **tree** have an implicit [aria-orientation](#) value of **vertical**.

Characteristics:

Characteristic	Value
----------------	-------

Characteristic	Value
Superclass Role:	select
Subclass Roles:	treegrid
Required Owned Elements:	group → treeitem treeitem
Supported States and Properties:	aria-errormessage aria-invalid aria-multiselectable aria-required
Inherited States and Properties:	aria-activedescendant aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) aria-dropeffect aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-keyshortcuts aria-label aria-labelledby aria-live aria-orientation aria-owns aria-relevant

Characteristic	Value
	aria-roledescription
Name From:	author
Accessible Name Required:	True
Implicit Value for Role:	Default for aria-orientation is vertical.

treegrid role

A [grid](#) whose rows can be expanded and collapsed in the same manner as for a [tree](#).

If [aria_READONLY](#) is set on an [element](#) with [role treegrid](#), [user agents](#) **MUST** propagate the value to all [gridcell](#) elements owned by the [treegrid](#) and expose the value in the accessibility [API](#). An author **MAY** override the propagated value of [aria_READONLY](#) for an individual [gridcell](#) element.

When the [aria_READONLY](#) attribute is applied to a focusable [gridcell](#), it indicates whether the content contained in the [gridcell](#) is editable. The [aria_READONLY](#) attribute does not represent availability of functions for navigating or manipulating the [treegrid](#) itself.

In a [treegrid](#) that provides content editing functions, if the content of a focusable [gridcell](#) element is not editable, authors **MAY** set [aria_READONLY](#) to true on the [gridcell](#) element. However, if a [treegrid](#) presents a collection of elements that do not support [aria_READONLY](#), such as a collection of [link](#) elements, it is not necessary for the author to specify a value for [aria_READONLY](#).

To be [keyboard accessible](#), authors **SHOULD** manage focus of descendants for all instances of this [role](#), as described in [Managing Focus](#).

Characteristics:

Characteristic	Value
Superclass Role:	grid tree
Required Owned Elements:	row rowgroup → row
Inherited States and Properties:	aria-activedescendant aria-atomic aria-busy (state) aria-colcount

Characteristic	Value
	<p>aria-controls</p> <p>aria-current (state)</p> <p>aria-describedby</p> <p>aria-details</p> <p>aria-disabled (state)</p> <p>aria-dropeffect</p> <p>aria-errormessage</p> <p>aria-flowto</p> <p>aria-grabbed (state)</p> <p>aria-haspopup (deprecated on this role in ARIA 1.2)</p> <p>aria-hidden (state)</p> <p>aria-invalid (state)</p> <p>aria-keyshortcuts</p> <p>aria-label</p> <p>aria-labelledby</p> <p>aria-live</p> <p>aria-multiselectable</p> <p>aria-orientation</p> <p>aria-owns</p> <p>aria_READONLY</p> <p>aria-relevant</p> <p>aria-required</p> <p>aria-roledescription</p> <p>aria-rowcount</p>
Name From:	author
Accessible Name Required:	True

treeitem role

An option item of a [tree](#). This is an [element](#) within a tree that may be expanded or collapsed if it contains a

sub-level group of tree item elements.

A collection of `treeitem` elements to be expanded and collapsed are enclosed in an element with the [group role](#).

Authors **MUST** ensure [elements with role treeitem](#) are contained in, or [owned](#) by, an element with the role [group](#) or [tree](#).

Characteristics:

Characteristic	Value
Superclass Role:	<u>listitem</u> <u>option</u>
Required Context Role:	<u>group</u> <u>tree</u>
Supported States and Properties:	<u>aria-expanded</u> <u>aria-haspopup</u>
Inherited States and Properties:	<u>aria-atomic</u> <u>aria-busy</u> (state) <u>aria-checked</u> (state) <u>aria-controls</u> <u>aria-current</u> (state) <u>aria-describedby</u> <u>aria-details</u> <u>aria-disabled</u> (state) <u>aria-dropeffect</u> <u>aria-errormessage</u> (deprecated on this role in ARIA 1.2) <u>aria-flowto</u> <u>aria-grabbed</u> (state) <u>aria-hidden</u> (state) <u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2) <u>aria-keyshortcuts</u>

Characteristic	Value
	<u>aria-label</u> <u>aria-labelledby</u> <u>aria-level</u> <u>aria-live</u> <u>aria-owns</u> <u>aria-posinset</u> <u>aria-relevant</u> <u>aria-roledescription</u> <u>aria-selected</u> (state) (required) <u>aria-setsize</u>
Name From:	contents author
Accessible Name Required:	True

widget role

An interactive component of a graphical user interface (GUI).

Widgets are discrete user interface objects with which the user can interact. Widget [roles](#) map to standard features in [accessibility APIs](#). When the user navigates an element assigned any of the non-abstract subclass roles of [widget](#), [assistive technologies](#) that typically intercept standard keyboard events **SHOULD** switch to an application browsing mode, and pass keyboard events through to the web application. The intent is to hint to certain [assistive technologies](#) to switch from normal browsing mode into a mode more appropriate for interacting with a web application; some [user agents](#) have a browse navigation mode where keys, such as up and down arrows, are used to browse the document, and this native behavior prevents the use of these keys by a web application.

NOTE

[widget](#) is an abstract role used for the ontology. Authors should not use this role in content.

Characteristics:

Characteristic	Value
Is Abstract:	True

Characteristic	Value
Superclass Role:	roletype
Subclass Roles:	command composite gridcell input progressbar row scrollbar separator tab
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby aria-details aria-disabled (state) (deprecated on this role in ARIA 1.2) aria-dropeffect aria-errormessage (deprecated on this role in ARIA 1.2) aria-flowto aria-grabbed (state) aria-haspopup (deprecated on this role in ARIA 1.2) aria-hidden (state) aria-invalid (state) (deprecated on this role in ARIA 1.2) aria-keyshortcuts aria-label aria-labelledby

Characteristic	Value
	aria-live aria-owns aria-relevant aria-roledescription
Name From:	n/a

window role

A browser or application window.

Elements with this [role](#) have a window-like behavior in a graphical user interface (GUI) context, regardless of whether they are implemented as a native window in the operating system, or merely as a section of the document styled to look like a window.

NOTE

In the description of this role, the term "application" does not refer to the [application](#) role, which specifies specific assistive technology behaviors.

NOTE

window is an abstract role used for the ontology. Authors should not use this role in content.

Characteristics:

Characteristic	Value
Is Abstract:	True
Superclass Role:	roletype
Subclass Roles:	dialog
Supported States and Properties:	aria-modal
Inherited States and Properties:	aria-atomic aria-busy (state) aria-controls aria-current (state) aria-describedby

Characteristic	Value
	<p><u>aria-details</u></p> <p><u>aria-disabled</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-dropeffect</u></p> <p><u>aria-errormessage</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-flowto</u></p> <p><u>aria-grabbed</u> (state)</p> <p><u>aria-haspopup</u> (deprecated on this role in ARIA 1.2)</p> <p><u>aria-hidden</u> (state)</p> <p><u>aria-invalid</u> (state) (deprecated on this role in ARIA 1.2)</p> <p><u>aria-keyshortcuts</u></p> <p><u>aria-label</u></p> <p><u>aria-labelledby</u></p> <p><u>aria-live</u></p> <p><u>aria-owns</u></p> <p><u>aria-relevant</u></p> <p><u>aria-roledescription</u></p>
Name From:	author

§ 6. Supported States and Properties

§ 6.1 Clarification of States versus Properties

The terms "states" and "properties" refer to similar features. Both provide specific information about an [object](#), and both form part of the definition of the nature of [roles](#). In this document, states and properties are both treated as aria-prefixed markup [attributes](#). However, they are maintained conceptually distinct to clarify subtle differences in their meaning. One major difference is that the values of properties (such as [aria-](#)

labelledby) are often less likely to change throughout the application life-cycle than the values of states (such as aria-checked) which may change frequently due to user interaction. Note that the frequency of change difference is not a rule; a few properties, such as aria-valuetext are expected to change often. Because the distinction between states and properties is of little consequence to most web content authors, this specification refers to both "states" and "properties" simply as "attributes" whenever possible. See the definitions of *state* and *property* for more information.

6.2 Characteristics of States and Properties

States and properties have the characteristics described in the following sections.

6.2.1 Related Concepts

Advisory information about features from this or other languages that correspond to this state or property. While the correspondence may not be exact, it is useful to help understand the intent of the state or property.

6.2.2 Used in Roles

Advisory information about roles that use this state or property. This information is provided to help understand the appropriate usage of the state or property. Use of a given state or property is not defined when used on roles other than those listed.

6.2.3 Inherits into Roles

Advisory information about roles that inherit the state or property from an ancestor role.

6.2.4 Value

Value type of the state or property. The value may be one of the following types:

true/false

Value representing either `true` or `false`. The default value for this value type is `false` unless otherwise specified.

tristate

Value representing `true`, `false`, `mixed`, or `undefined` values. The default value for this value type is `undefined` unless otherwise specified.

true/false/undefined

Value representing `true`, `false`, or `undefined` (not applicable). The default value for this value type is `undefined` unless otherwise specified. For example, an element with `aria-expanded` set to `false` is not currently expanded; an element with `aria-expanded` set to `undefined` is not expandable.

ID reference

Reference to the ID of another `element` in the same document

ID reference list

A list of one or more ID references.

integer

A numerical value without a fractional component.

number

Any real numerical value.

string

Unconstrained value type.

token

One of a limited set of allowed values. The default value is defined in each attribute's Values table, as specified in the [Attribute Values](#) section.

token list

A list of one or more tokens.

These are generic types for states and properties, but do not define specific representation. See [State and Property Attribute Processing](#) for details on how these values are expressed and handled in host languages.

6.3 ARIA Attributes

6.3.1 Multi-value Attribute Values

When the ARIA attribute definition includes a table listing the attribute's allowed values, that attribute is a multi-value nullable attribute. Each value in the table is a keyword for the attribute, mapping to a state of the same name.

6.3.2 IDL reflection of ARIA attributes

All ARIA attributes reflect in IDL as [nullable DOMString](#) attributes. This includes the boolean-like [true/false](#) type, and all other ARIA attributes.

Default values from the ARIA values tables **MUST NOT** reflect to IDL as the [missing value default](#) or the [invalid value default](#) for the attribute. On getting, a missing ARIA attribute will return `null`. ARIA attributes are not validated on get. If an ARIA value is invalid, on getting, it will return its set value as a literal string, and will not return an invalid value default.

6.3.3 Operating System Accessibility API mapping of multi-value ARIA attributes

Unlike IDL reflection, operating system accessibility API mappings of ARIA attributes can have defaults. Any default values from the ARIA values tables are exposed to the operating system accessibility API as described in [5.2.3 Supported States and Properties](#), and in [Core Accessibility API Mappings 1.1](#).

6.3.4 ARIA nullable DOMString Attributes

As noted in [A. Mapping WAI-ARIA Value types to languages](#), attributes are included in host languages, and the syntax for representation of WAI-ARIA types is governed by the host language.

The following algorithm should be used for ARIA nullable [DOMString](#) attributes in [HTML](#):

On getting, if the corresponding content attribute is not present, then the IDL attribute must return `null`, otherwise, the IDL attribute must get the value in a transparent, case-preserving manner. On setting, if the new value is `null`, the content attribute must be removed, and otherwise, the content attribute must be set to the specified new value in a transparent, case-preserving manner.

NOTE

Note: As of ARIA 1.2, all ARIA attributes exposed via IDL are defined as nullable [DOMStrings](#). This matches the current implementation of all major rendering engines. This specification change should result in no implementation changes; it will merely represent the current reality of web engines. However, in a future draft, the ARIA Working Group intends to change several ARIA attributes to non-nullable [DOMStrings](#), and seek implementations. The proposed change will bring ARIA into alignment with the [HTML's usage of enumerated attributes](#).

6.3.4.1 Example Attribute Usage

This section is non-normative.

EXAMPLE 14

```
// HTML hidden="" example (not aria-hidden="true")
// Actual boolean type; defaults to false.

// Note: Actual boolean assignment and return value.
el.hidden = true;
el.hidden; // true

// Removal of content attribute results in missing value default: boolean false.
el.removeAttribute("hidden");
el.hidden; // false
```

EXAMPLE 15

```
// aria-busy example
// true/false ~ boolean-like nullable string; returns null unless set

el.ariaBusy; // null

// Note: String assignment and return value.
el.ariaBusy = "true";
el.ariaBusy; // "true"

// Removal of content attribute results in missing value default: string "false".
el.removeAttribute("aria-busy");
el.ariaBusy; // null

// Assignment of invalid "busy" value. Not validated on set or get and the literal string \
el.setAttribute("aria-busy", "busy");
el.ariaBusy; // "busy"
```

EXAMPLE 16

```
// aria-pressed example
// Tristate ~ true/false/mixed/undefined string; null if unspecified

// no value has been defined
button.ariaPressed; // null

// A value of "true", "false", or "mixed" for aria-pressed on a button denotes a toggle button.
button.setAttribute("aria-pressed", "true"); // Content attribute assignment.
button.ariaPressed; // "true"
button.ariaPressed = "false"; // DOM property assignment.
button.ariaPressed; // "false"

// Assignment of invalid "foo" value. Not validated on set or get and the literal string value remains.
button.ariaPressed = "foo";
button.ariaPressed; // "foo" (Note: button is no longer a toggle button.)

// Removal of content attribute results in a null value
button.removeAttribute("aria-pressed");
button.ariaPressed; // null
```

§ 6.4 Translatable States and Properties

The [HTML](#) specification states that other specifications can define [translatable attributes](#). In order to be understandable by assistive technology users, the values of the following [states](#) and [properties](#) are [translatable attributes](#) and should be translated when a page is localized:

- [aria-label](#)
- [aria-placeholder](#)
- [aria-roledescription](#)
- [aria-valuetext](#)

§ 6.5 Global States and Properties

Some [states](#) and [properties](#) are applicable to all host language [elements](#) regardless of whether a [role](#) is applied. The following global states and properties are supported by all roles and by all base markup elements unless otherwise prohibited. If a role prohibits use of any global states or properties, those states or properties are listed as prohibited in the characteristics table included in the section that defines the role.

- [aria-atomic](#)
- [aria-busy \(state\)](#)
- [aria-controls](#)
- [aria-current \(state\)](#)
- [aria-describedby](#)
- [aria-details](#)
- [aria-disabled \(state\)](#) (Global use deprecated in ARIA 1.2)
- [aria-dropeffect](#)
- [aria-errormessage](#) (Global use deprecated in ARIA 1.2)
- [aria-flowto](#)
- [aria-grabbed \(state\)](#)
- [aria-haspopup](#) (Global use deprecated in ARIA 1.2)
- [aria-hidden \(state\)](#)
- [aria-invalid \(state\)](#) (Global use deprecated in ARIA 1.2)
- [aria-keyshortcuts](#)
- [aria-label](#) (Except where prohibited)
- [aria-labelledby](#) (Except where prohibited)
- [aria-live](#)
- [aria-owns](#)
- [aria-relevant](#)
- [aria-roledescription](#)

§ 6.6 Taxonomy of WAI-ARIA States and Properties

States and properties are categorized as follows:

1. [Widget Attributes](#)
2. [Live Region Attributes](#)
3. [Drag-and-Drop Attributes](#)
4. [Relationship Attributes](#)

6.6.1 Widget Attributes

This section contains [attributes](#) specific to common user interface [elements](#) found on [GUI](#) systems or in rich internet applications which receive user input and process user actions. These attributes are used to support the [widget roles](#).

- [aria-autocomplete](#)
- [aria-checked](#)
- [aria-disabled](#)
- [aria-errormessage](#)
- [aria-expanded](#)
- [aria-haspopup](#)
- [aria-hidden](#)
- [aria-invalid](#)
- [aria-label](#)
- [aria-level](#)
- [aria-modal](#)
- [aria-multiline](#)
- [aria-multiselectable](#)
- [aria-orientation](#)
- [aria-placeholder](#)
- [aria-pressed](#)
- [aria-readonly](#)
- [aria-required](#)
- [aria-selected](#)
- [aria-sort](#)
- [aria-valuemax](#)
- [aria-valuemin](#)
- [aria-valuenow](#)

- [aria-valuetext](#)

Widget attributes might be mapped by a [user agent](#) to platform [accessibility API state](#), for access by [assistive technologies](#), or they might be accessed directly from the [DOM](#). User agents **MUST** provide a way for assistive technologies to be notified when states change, either through [DOM attribute change events](#) or platform accessibility API events.

6.6.2 Live Region Attributes

This section contains [attributes](#) specific to [live regions](#) in rich internet applications. These attributes may be applied to any [element](#). The purpose of these attributes is to indicate that content changes may occur without the element having focus, and to provide [assistive technologies](#) with information on how to process those content updates. Some [roles](#) specify a default value for the [aria-live](#) attribute specific to that role. An example of a live region is a ticker section that lists updating stock quotes.

- [aria-atomic](#)
- [aria-busy](#)
- [aria-live](#)
- [aria-relevant](#)

6.6.3 Drag-and-Drop Attributes

This section lists [attributes](#) which indicate information about drag-and-drop interface [elements](#), such as draggable elements and their drop targets. Drop target information will be rendered visually by the author and provided to [assistive technologies](#) through an alternate modality.

- [aria-dropeffect](#)
- [aria-grabbed](#)

6.6.4 Relationship Attributes

This section lists [attributes](#) that indicate [relationships](#) or associations between [elements](#) which cannot be readily determined from the document structure.

- [aria-activedescendant](#)
- [aria-colcount](#)
- [aria-colindex](#)
- [aria-colspan](#)
- [aria-controls](#)
- [aria-describedby](#)
- [aria-details](#)
- [aria-errormessage](#)
- [aria-flowto](#)
- [aria-labelledby](#)
- [aria-owns](#)
- [aria-posinset](#)
- [aria-rowcount](#)
- [aria-rowindex](#)
- [aria-rowspan](#)
- [aria-setsize](#)

6.7 Definitions of States and Properties (all aria-* attributes)

Below is an alphabetical list of WAI-ARIA [states](#) and [properties](#) to be used by rich internet application authors. A detailed definition of each WAI-ARIA state and [property](#) follows this compact list.

[aria-activedescendant](#)

Identifies the currently active element when DOM focus is on a [composite](#) widget, [combobox](#), [textbox](#), [group](#), or [application](#).

[aria-atomic](#)

Indicates whether [assistive technologies](#) will present all, or only parts of, the changed region based on the change notifications defined by the [aria-relevant](#) attribute.

[aria-autocomplete](#)

Indicates whether inputting text could trigger display of one or more predictions of the user's intended value for a [combobox](#), [searchbox](#), or [textbox](#) and specifies how predictions would be presented if they were made.

[aria-busy](#)

Indicates an element is being modified and that assistive technologies **MAY** want to wait until the modifications are complete before exposing them to the user.

aria-checked

Indicates the current "checked" state of checkboxes, radio buttons, and other widgets. See related [aria-pressed](#) and [aria-selected](#).

aria-colcount

Defines the total number of columns in a [table](#), [grid](#), or [treemap](#). See related [aria-colindex](#).

aria-colindex

Defines an [element's](#) column index or position with respect to the total number of columns within a [table](#), [grid](#), or [treemap](#). See related [aria-colcount](#) and [aria-colspan](#).

aria-colspan

Defines the number of columns spanned by a cell or gridcell within a [table](#), [grid](#), or [treemap](#). See related [aria-colindex](#) and [aria-rowspan](#).

aria-controls

Identifies the [element](#) (or elements) whose contents or presence are controlled by the current element.

See related [aria-owns](#).

aria-current

Indicates the [element](#) that represents the current item within a container or set of related elements.

aria-describedby

Identifies the [element](#) (or elements) that describes the [object](#). See related [aria-labelledby](#).

aria-details

Identifies the [element](#) that provides a detailed, extended description for the [object](#). See related [aria-describedby](#).

aria-disabled

Indicates that the [element](#) is [perceivable](#) but disabled, so it is not editable or otherwise [operable](#). See related [aria-hidden](#) and [aria-readonly](#).

aria-dropeffect

[Deprecated in ARIA 1.1] Indicates what functions can be performed when a dragged object is released on the drop target.

aria-errormessage

Identifies the [element](#) that provides an error message for an [object](#). See related [aria-invalid](#) and [aria-describedby](#).

aria-expanded

Indicates whether a grouping element owned or controlled by this element is expanded or collapsed.

aria-flowto

Identifies the next [element](#) (or elements) in an alternate reading order of content which, at the user's discretion, allows assistive technology to override the general default of reading in document source order.

aria-grabbed

[Deprecated in ARIA 1.1] Indicates an element's "grabbed" [state](#) in a drag-and-drop operation.

[aria-haspopup](#)

Indicates the availability and type of interactive popup element, such as menu or dialog, that can be triggered by an [element](#).

[aria-hidden](#)

Indicates whether the [element](#) is exposed to an accessibility API. See related [aria-disabled](#).

[aria-invalid](#)

Indicates the entered value does not conform to the format expected by the application. See related [aria-errormessage](#).

[aria-keyshortcuts](#)

Indicates keyboard shortcuts that an author has implemented to activate or give focus to an element.

[aria-label](#)

Defines a string value that labels the current element. See related [aria-labelledby](#).

[aria-labelledby](#)

Identifies the [element](#) (or elements) that labels the current element. See related [aria-describedby](#).

[aria-level](#)

Defines the hierarchical level of an [element](#) within a structure.

[aria-live](#)

Indicates that an [element](#) will be updated, and describes the types of updates the [user agents](#), [assistive technologies](#), and user can expect from the [live region](#).

[aria-modal](#)

Indicates whether an [element](#) is modal when displayed.

[aria-multiline](#)

Indicates whether a text box accepts multiple lines of input or only a single line.

[aria-multiselectable](#)

Indicates that the user may select more than one item from the current selectable descendants.

[aria-orientation](#)

Indicates whether the element's orientation is horizontal, vertical, or unknown/ambiguous.

[aria-owns](#)

Identifies an [element](#) (or elements) in order to define a visual, functional, or contextual parent/child [relationship](#) between [DOM](#) elements where the [DOM](#) hierarchy cannot be used to represent the relationship. See related [aria-controls](#).

[aria-placeholder](#)

Defines a short hint (a word or short phrase) intended to aid the user with data entry when the control has no value. A hint could be a sample value or a brief description of the expected format.

[aria-posinset](#)

Defines an [element](#)'s number or position in the current set of listitems or treeitems. Not required if all elements in the set are present in the [DOM](#). See related [aria-setsize](#).

[aria-pressed](#)

Indicates the current "pressed" state of toggle buttons. See related [aria-checked](#) and [aria-selected](#).

[aria-readonly](#)

Indicates that the element is not editable, but is otherwise operable. See related [aria-disabled](#).

[aria-relevant](#)

Indicates what notifications the user agent will trigger when the accessibility tree within a live region is modified. See related [aria-atomic](#).

[aria-required](#)

Indicates that user input is required on the element before a form may be submitted.

[aria-roledescription](#)

Defines a human-readable, author-localized description for the role of an element.

[aria-rowcount](#)

Defines the total number of rows in a [table](#), [grid](#), or [treemap](#). See related [aria-rowindex](#).

[aria-rowindex](#)

Defines an element's row index or position with respect to the total number of rows within a [table](#), [grid](#), or [treemap](#). See related [aria-rowcount](#) and [aria-rowspan](#).

[aria-rowspan](#)

Defines the number of rows spanned by a cell or gridcell within a [table](#), [grid](#), or [treemap](#). See related [aria-rowindex](#) and [aria-colspan](#).

[aria-selected](#)

Indicates the current "selected" state of various widgets. See related [aria-checked](#) and [aria-pressed](#).

[aria-setsize](#)

Defines the number of items in the current set of listitems or treeitems. Not required if all elements in the set are present in the DOM. See related [aria-posinset](#).

[aria-sort](#)

Indicates if items in a table or grid are sorted in ascending or descending order.

[aria-valuemax](#)

Defines the maximum allowed value for a range widget.

[aria-valuemin](#)

Defines the minimum allowed value for a range widget.

[aria-valuenow](#)

Defines the current value for a range widget. See related [aria-valuetext](#).

[aria-valuetext](#)

Defines the human readable text alternative of [aria-valuenow](#) for a range widget.

aria-activedescendant property

Identifies the currently active element when DOM focus is on a [composite](#) widget, [combobox](#), [textbox](#), [group](#), or [application](#).

The `aria-activedescendant` property provides an alternative method of managing focus for interactive elements that may contain multiple focusable descendants, such as menus, grids, and toolbars. Instead of moving DOM focus among [owned](#) elements, authors *MAY* set DOM focus on a container [element](#) that supports `aria-activedescendant` and then use `aria-activedescendant` to refer to the element that is active.

Authors **MUST** ensure that one of the following two sets of conditions is met when setting the value of `aria-activedescendant` on an element with DOM focus:

1. The value of `aria-activedescendant` refers to an [owned](#) element. An owned element is either a descendant of the element with DOM focus or a logical descendant as indicated by the [aria-owns](#) attribute.
2. The element with DOM focus is a [combobox](#), [textbox](#) or [searchbox](#) with [aria-controls](#) referring to an element that supports `aria-activedescendant`, and the value of `aria-activedescendant` refers to an owned element of the controlled element. For example, in a [combobox](#), focus may remain on the [combobox](#) while the value of `aria-activedescendant` on the [combobox](#) element refers to a descendant of a popup [listbox](#) that is controlled by the [combobox](#).

Authors **SHOULD** also ensure that the currently active descendant is visible and in view (or scrolls into view) when focused.

Characteristics:

Characteristic	Value
Related Concepts:	SVG [SVG2] and DOM [DOM] active
Used in Roles:	application combobox composite group textbox
Inherits into Roles:	grid listbox menu menubar radiogroup

Characteristic	Value
	row searchbox select spinbutton tablist toolbar tree treegrid
Value:	ID reference

aria-atomic property

Indicates whether [assistive technologies](#) will present all, or only parts of, the changed region based on the change notifications defined by the [aria-relevant](#) attribute.

Both [accessibility APIs](#) and the [Document Object Model \[DOM\]](#) provide events to allow the assistive technologies to determine changed areas of the document.

When the content of a [live region](#) changes, user agents **SHOULD** examine the changed [element](#) and traverse the ancestors to find the first element with [aria-atomic](#) set, and apply the appropriate behavior for the cases below.

1. If none of the ancestors have explicitly set [aria-atomic](#), the default is that [aria-atomic](#) is **false**, and assistive technologies will only present the changed node to the user.
2. If [aria-atomic](#) is explicitly set to **false**, assistive technologies will stop searching up the ancestor chain and present only the changed node to the user.
3. If [aria-atomic](#) is explicitly set to **true**, assistive technologies will present the entire contents of the element, including the author-defined live region label if one exists.

When [aria-atomic](#) is **true**, assistive technologies **MAY** choose to combine several changes and present the entire changed region at once.

Characteristics:

Characteristic	Value
Used in Roles:	All elements of the base markup
Value:	true/false

Values:

Value	Description
false (default)	Assistive technologies will present only the changed node or nodes.
true	Assistive technologies will present the entire changed region as a whole, including the author-defined label if one exists.

aria-autocomplete property

Indicates whether inputting text could trigger display of one or more predictions of the user's intended value for a [combobox](#), [searchbox](#), or [textbox](#) and specifies how predictions would be presented if they were made.

The `aria-autocomplete` property describes the type of interaction model a [textbox](#), [searchbox](#), or [combobox](#) employs when dynamically helping users complete text input. It distinguishes between two models: the inline model (`aria-autocomplete="inline"`) that presents a value completion prediction inside the text input and the list model (`aria-autocomplete="list"`) that presents a collection of possible values in a separate element that pops up adjacent to the text input. It is possible for an input to offer both models at the same time (`aria-autocomplete="both"`).

The `aria-autocomplete` property is limited to describing predictive behaviors of an input element. Authors **SHOULD** either omit specifying a value for `aria-autocomplete` or set `aria-autocomplete` to `none` if an input element provides one or more input proposals where none of the proposals are dependent on the specific input provided by the user. For instance, a combobox where the value of `aria-autocomplete` would be `none` is a search field that displays suggested values by listing the 5 most recently used search terms without any filtering of the list based on the user's input. Elements with a role that supports `aria-autocomplete` have a default value for `aria-autocomplete` of `none`.

When an inline suggestion is made as a user types in an input, suggested text for completing the value of the field dynamically appears in the field after the input cursor, and the suggested value is accepted as the value of the input if the user performs an action that causes focus to leave the field. When an element has `aria-autocomplete` set to `inline` or `both`, authors **SHOULD** ensure that the automatically suggested portion of the text is presented as selected text. This enables assistive technologies to distinguish between a user's input and the automatic suggestion and, in the event that the suggestion is not the desired value, enables the user to easily delete the suggestion or replace it by continuing to type.

If an element has `aria-autocomplete` set to `list` or `both`, authors **MUST** ensure both of the following conditions are met:

1. The element has a value specified for [aria-controls](#) that refers to the element that contains the

collection of suggested values.

2. The element has a value for [aria-haspopup](#) that matches the role of the element that contains the collection of suggested values.

Some implementations of the list model require the user to perform an action, such as moving focus to the suggestion with the Down Arrow or clicking on the suggestion, in order to choose the suggestion. In such implementations, authors **MAY** manage focus by either using [aria-activedescendant](#) if the collection container supports it or by moving DOM focus to the suggestion. However, other implementations of the list model automatically highlight one suggestion as the selected value that will be accepted when the field loses focus, e.g., when the user presses the Tab key or clicks on a different field. If an element has [aria-autocomplete](#) set to list or both, and if a suggestion is automatically selected as the user provides input, authors **MUST** ensure all the following conditions are met:

1. The collection of suggestions is presented in an element with a role that supports [aria-activedescendant](#).
2. The value of [aria-activedescendant](#) set on the input field is dynamically adjusted to refer to the element containing the selected suggestion as described in the definition of [aria-activedescendant](#).
3. DOM focus remains on the text input while the suggestions are displayed.

The [aria-autocomplete](#) property is not intended to indicate the presence of a completion suggestion, and authors **SHOULD NOT** dynamically change its value in order to communicate the presence of a suggestion. When an element has [aria-autocomplete](#) set to list or both, authors **SHOULD** use the [aria-expanded](#) state to communicate whether the element that presents the suggestion collection is displayed.

Characteristics:

Characteristic	Value
Used in Roles:	<u>combobox</u> <u>textbox</u>
Inherits into Roles:	<u>searchbox</u>
Value:	<u>token</u>

Values:

Value	Description
inline	When a user is providing input, text suggesting one way to complete the provided input may be dynamically inserted after the caret.
list	When a user is providing input, an element containing a collection of values that could complete the provided

Value	Description
	input may be displayed.
both	When a user is providing input, an element containing a collection of values that could complete the provided input may be displayed. If displayed, one value in the collection is automatically selected, and the text needed to complete the automatically selected value appears after the caret in the input.
none (default)	When a user is providing input, an automatic suggestion that attempts to predict how the user intends to complete the input is not displayed.

aria-busy state

Indicates an element is being modified and that assistive technologies **MAY** want to wait until the modifications are complete before exposing them to the user.

The default value of `aria-busy` is `false` for all elements. When `aria-busy` is `true` for an element, assistive technologies **MAY** ignore changes to content owned by that element and then process all changes made during the busy period as a single, atomic update when `aria-busy` becomes `false`.

If it is necessary to make multiple additions, modifications, or removals within a container element that is already either partially or fully rendered, authors **MAY** set `aria-busy` to `true` on the container element before the first change, and then set it to `false` when the last change is complete. For example, if multiple changes to a [live region](#) should be spoken as a single unit of speech, authors **MAY** set `aria-busy` to `true` while the changes are being made and then set it to `false` when the changes are complete and ready to be spoken.

If an element with role [feed](#) is marked busy, assistive technologies **MAY** defer rendering changes that occur inside the `feed` with the exception of user-initiated changes that occur inside the [article](#) that the user is reading during the busy period.

If changes to a rendered [widget](#) would create a state where the [widget](#) is missing [required owned elements](#) during script execution, authors **MUST** set `aria-busy` to `true` on the [widget](#) during the update process. For example, if a rendered tree grid required a set of simultaneous updates to multiple discontiguous branches, an alternative to replacing the complete tree element with a single update would be to mark the tree busy while each of the branches are modified.

Characteristics:

Characteristic	Value

Characteristic	Value
Used in Roles:	All elements of the base markup
Value:	true/false

Values:

Value	Description
false (default):	There are no expected updates for the element.
true	The element is being updated.

aria-checked state

Indicates the current "checked" [state](#) of checkboxes, radio buttons, and other [widgets](#). See related [aria-pressed](#) and [aria-selected](#).

The [aria-checked](#) [attribute](#) indicates whether the [element](#) is checked ([true](#)), unchecked ([false](#)), or represents a group of other elements that have a mixture of checked and unchecked values ([mixed](#)). Most inputs only support values of [true](#) and [false](#), but the [mixed](#) value is supported by certain tri-state inputs such as a [checkbox](#) or [menuitemcheckbox](#).

The [mixed](#) value is *not* supported on [radio](#), [menuitemradio](#), [switch](#) or any element that inherits from these, and [user agents](#) **MUST** treat a [mixed](#) value as equivalent to [false](#) for those [roles](#).

Examples using the [mixed](#) value of tri-state inputs are covered in the [WAI-ARIA Authoring Practices](#).

Characteristics:

Characteristic	Value
Used in Roles:	checkbox menuitemcheckbox option radio switch
Inherits into Roles:	menuitemradio switch treeitem
Value:	tristate

Values:

Value	Description
false	The element supports being checked but is not currently checked.
mixed	Indicates a mixed mode value for a tri-state checkbox or menuitemcheckbox.
true	The element is checked.
undefined (default)	The element does not support being checked.

aria-colcount property

Defines the total number of columns in a [table](#), [grid](#), or [treemap](#). See related [aria-colindex](#).

If all of the columns are present in the [DOM](#), it is not necessary to set this [attribute](#) as the [user agent](#) can automatically calculate the total number of columns. However, if only a portion of the columns is present in the [DOM](#) at a given moment, this attribute is needed to provide an explicit indication of the number of columns in the full table.

Authors **MUST** set the value of [aria-colcount](#) to an integer equal to the number of columns in the full table. If the total number of columns is unknown, authors **MUST** set the value of [aria-colcount](#) to -1 to indicate that the value should not be calculated by the user agent.

The following example shows a grid with 16 columns, of which columns 2, 3, 4, and 9 are displayed to the user.

EXAMPLE 17

```
<div role="grid" aria-colcount="16">
  <div role="rowgroup">
    <div role="row">
      <span role="columnheader" aria-colindex="2">First Name</span>
      <span role="columnheader" aria-colindex="3">Last Name</span>
      <span role="columnheader" aria-colindex="4">Company</span>
      <span role="columnheader" aria-colindex="9">Phone</span>
    </div>
  </div>
  <div role="rowgroup">
    <div role="row">
      <span role="gridcell" aria-colindex="2">Fred</span>
      <span role="gridcell" aria-colindex="3">Jackson</span>
      <span role="gridcell" aria-colindex="4">Acme, Inc.</span>
      <span role="gridcell" aria-colindex="9">555-1234</span>
    </div>
    <div role="row">
      <span role="gridcell" aria-colindex="2">Sara</span>
      <span role="gridcell" aria-colindex="3">James</span>
      <span role="gridcell" aria-colindex="4">Acme, Inc.</span>
      <span role="gridcell" aria-colindex="9">555-1235</span>
    </div>
    ...
  </div>
</div>
```

Characteristics:

Characteristic	Value
Used in Roles:	table
Inherits into Roles:	grid treegrid
Value:	integer

aria-colindex property

Defines an [element's](#) column index or position with respect to the total number of columns within a [table](#), [grid](#), or [treegrid](#). See related [aria-colcount](#) and [aria-colspan](#).

If all of the columns are present in the [DOM](#), it is not necessary to set this [attribute](#) as the [user agent](#) can automatically calculate the column index of each cell or [gridcell](#). However, if only a portion of the columns is present in the [DOM](#) at a given moment, this attribute is needed to provide an explicit indication of the column of each cell or gridcell with respect to the full table.

Authors **MUST** set the value for `aria-colindex` to an integer greater than or equal to 1, greater than the `aria-colindex` value of any previous elements within the same row, and less than or equal to the number of columns in the full table. For a cell or gridcell which spans multiple columns, authors **MUST** set the value of `aria-colindex` to the start of the span.

If the set of columns which is present in the DOM is contiguous, and if there are no cells which span more than one row or column in that set, then authors **MAY** place `aria-colindex` on each row, setting the value to the index of the first column of the set. Otherwise, authors **SHOULD** place `aria-colindex` on all of the children or `owned` elements of each row.

The following example shows a grid with 16 columns, of which columns 2 through 5 are displayed to the user. Because the set of columns is contiguous, `aria-colindex` can be placed on each row.

EXAMPLE 18

```
<div role="grid" aria-colcount="16">
  <div role="rowgroup">
    <div role="row" aria-colindex="2">
      <span role="columnheader">First Name</span>
      <span role="columnheader">Last Name</span>
      <span role="columnheader">Company</span>
      <span role="columnheader">Address</span>
    </div>
  </div>
  <div role="rowgroup">
    <div role="row" aria-colindex="2">
      <span role="gridcell">Fred</span>
      <span role="gridcell">Jackson</span>
      <span role="gridcell">Acme, Inc.</span>
      <span role="gridcell">123 Broad St.</span>
    </div>
    <div role="row" aria-colindex="2">
      <span role="gridcell">Sara</span>
      <span role="gridcell">James</span>
      <span role="gridcell">Acme, Inc.</span>
      <span role="gridcell">123 Broad St.</span>
    </div>
    ...
  </div>
</div>
```

The following example shows a grid with 16 columns, of which columns 2 through 5 are displayed to the user. While the set of columns is contiguous, some of the cells span multiple rows. As a result, `aria-colindex` needs to be placed on all of the owned elements of each row.

EXAMPLE 19

```
<div role="grid" aria-colcount="16">
  <div role="rowgroup">
    <div role="row">
      <span role="columnheader" aria-colindex="2">First Name</span>
      <span role="columnheader" aria-colindex="3">Last Name</span>
      <span role="columnheader" aria-colindex="4">Company</span>
      <span role="columnheader" aria-colindex="5">Address</span>
    </div>
  </div>
  <div role="rowgroup">
    <div role="row">
      <span role="gridcell" aria-colindex="2">Fred</span>
      <span role="gridcell" aria-colindex="3">Jackson</span>
      <span role="gridcell" aria-colindex="4" aria-rowspan="2">Acme, Inc.</span>
      <span role="gridcell" aria-colindex="5" aria-rowspan="2">123 Broad St.</span>
    </div>
    <div role="row">
      <span role="gridcell" aria-colindex="2">Sara</span>
      <span role="gridcell" aria-colindex="3">James</span>
    </div>
    ...
  </div>
</div>
```

The following example shows a grid with 16 columns, of which columns 2, 3, 4, and 9 are displayed to the user. Because the set of columns is non-contiguous, aria-colindex needs to be placed on all of the owned elements of each row.

EXAMPLE 20

```
<div role="grid" aria-colcount="16">
  <div role="rowgroup">
    <div role="row">
      <span role="columnheader" aria-colindex="2">First Name</span>
      <span role="columnheader" aria-colindex="3">Last Name</span>
      <span role="columnheader" aria-colindex="4">Company</span>
      <span role="columnheader" aria-colindex="9">Phone</span>
    </div>
  </div>
  <div role="rowgroup">
    <div role="row">
      <span role="gridcell" aria-colindex="2">Fred</span>
      <span role="gridcell" aria-colindex="3">Jackson</span>
      <span role="gridcell" aria-colindex="4">Acme, Inc.</span>
      <span role="gridcell" aria-colindex="9">555-1234</span>
    </div>
    <div role="row">
      <span role="gridcell" aria-colindex="2">Sara</span>
      <span role="gridcell" aria-colindex="3">James</span>
      <span role="gridcell" aria-colindex="4">Acme, Inc.</span>
      <span role="gridcell" aria-colindex="9">555-1235</span>
    </div>
    ...
  </div>
</div>
```

Characteristics:

Characteristic	Value
Used in Roles:	cell row
Inherits into Roles:	columnheader gridcell rowheader
Value:	integer

aria-colspan property

Defines the number of columns spanned by a cell or gridcell within a [table](#), [grid](#), or [treegrid](#). See related [aria-colindex](#) and [aria-rowspan](#).

This [attribute](#) is intended for cells and gridcells which are not contained in a native table. When defining the

column span of cells or gridcells in a native table, authors **SHOULD** use the host language's attribute instead of [aria-colspan](#). If [aria-colspan](#) is used on an element for which the host language provides an equivalent attribute, [user agents](#) **MUST** ignore the value of [aria-colspan](#) and instead expose the value of the host language's attribute to [assistive technologies](#).

Authors **MUST** set the value of [aria-colspan](#) to an integer greater than or equal to 1 and less than the value which would cause the cell or gridcell to overlap the next cell or gridcell in the same row.

Characteristics:

Characteristic	Value
Used in Roles:	cell
Inherits into Roles:	columnheader rowheader
Value:	integer

aria-controls property

Identifies the [element](#) (or elements) whose contents or presence are controlled by the current element. See related [aria-owns](#).

For example:

- A table of contents tree view may control the content of a neighboring document pane.
- A group of checkboxes may control what commodity prices are tracked live in a table or graph.
- A tab controls the display of its associated tab panel.

Characteristics:

Characteristic	Value
Used in Roles:	All elements of the base markup
Value:	ID reference list

aria-current state

Indicates the [element](#) that represents the current item within a container or set of related elements.

The [aria-current](#) attribute is a token type. Any value not included in the list of allowed values **SHOULD** be treated by [assistive technologies](#) as if the value `true` had been provided. If the attribute is not present or

its value is an empty string or `undefined`, the default value of `false` applies and the [`aria-current` state](#) **MUST NOT** be exposed by user agents or assistive technologies.

The [`aria-current`](#) attribute is used when an element within a set of related elements is visually styled to indicate it is the current item in the set. For example:

- A `page` token used to indicate a link within a set of pagination links, where the link is visually styled to represent the currently-displayed page.
- A `step` token used to indicate a link within a step indicator for a step-based process, where the link is visually styled to represent the current step.
- A `location` token used to indicate the image that is visually highlighted as the current component of a flow chart.
- A `date` token used to indicate the current date within a calendar.
- A `time` token used to indicate the current time within a timetable.

Authors **SHOULD** only mark one element in a set of elements as current with [`aria-current`](#).

Authors **SHOULD NOT** use the [`aria-current`](#) attribute as a substitute for [`aria-selected`](#) in widgets where [`aria-selected`](#) has the same meaning. For example, in a [`tablist`](#), [`aria-selected`](#) is used on a [`tab`](#) to indicate the currently-displayed [`tabpanel`](#).

NOTE

In some use cases for widgets that support [`aria-selected`](#), current and selected can have different meanings and can both be used within the same set of elements. For example, `aria-current="page"` can be used in a navigation [`tree`](#) to indicate which page is currently displayed, while `aria-selected="true"` indicates which page will be displayed if the user activates the [`treeitem`](#). Furthermore, the same tree may support operating on one or more selected pages (treeitems) by way of a context menu containing options such as "delete" and "move."

Characteristics:

Characteristic	Value
Used in Roles:	All elements of the base markup
Value:	<code>token</code>

Values:

Value	Description
<code>page</code>	Represents the current page within a set of pages.
<code>step</code>	Represents the current step within a process.

Value	Description
location	Represents the current location within an environment or context.
date	Represents the current date within a collection of dates.
time	Represents the current time within a set of times.
true	Represents the current item within a set.
false (default)	Does not represent the current item within a set.

aria-describedby property

Identifies the [element](#) (or elements) that describes the [object](#). See related [aria-labelledby](#).

The [aria-labelledby](#) attribute is similar to the [aria-describedby](#) in that both reference other elements to calculate a text alternative, but a label should be concise, where a description is intended to provide more verbose information.

The element or elements referenced by the aria-describedby comprise the entire description. Include ID references to multiple elements if necessary, or enclose a set of elements (e.g., paragraphs) with the element referenced by the ID.

Characteristics:

Characteristic	Value
Related Concepts:	<label> in [HTML] online help HTML table cell headers
Used in Roles:	All elements of the base markup
Value:	ID reference list

aria-details property

Identifies the [element](#) that provides a detailed, extended description for the [object](#). See related [aria-describedby](#).

The [aria-details](#) attribute references a single element that provides more detailed information than would normally be provided by [aria-describedby](#). It enables [assistive technologies](#) to make users aware of the availability of an extended description as well as navigate to it. Authors **SHOULD** ensure the element referenced by [aria-details](#) is visible to all users.

Unlike elements referenced by `aria-describedby`, the element referenced by `aria-details` is not used in either the Accessible [Name Computation](#) or the Accessible [Description Computation](#) as defined in the Accessible Name and Description specification. Thus, the content of an element referenced by `aria-details` is not flattened to a string when presented to assistive technology users. This makes `aria-details` particularly useful when converting the information to a string would cause a loss of information or make the extended description more difficult to understand.

In some user agents, multiple reference relationships for descriptive information are not supported by the accessibility API. In such cases, if both `aria-describedby` and `aria-details` are provided on an element, `aria-details` takes precedence.

A common use for `aria-details` is in digital publishing where an extended description needs to be conveyed in a book that requires structural markup or the embedding of other technology to provide illustrative content. The following example demonstrates this scenario.

[EXAMPLE 21](#)

```
<!-- Provision of an extended description -->

<details id="det">
  <summary>Example</summary>
  <p>
    The Pythagorean Theorem is a relationship in Euclidean Geometry between the three sides of a right triangle, where the square of the hypotenuse is the sum of the squares of the two opposing sides.
  </p>
  <p>
    The following drawing illustrates an application of the Pythagorean Theorem when used to construct a skateboard ramp.
  </p>
  <object data="skatebd-ramp.svg" type="image/svg+xml"></object>
  <p>
    In this example you will notice a skateboard with a base and vertical board whose width is the width of the ramp. To compute how long the ramp must be, simply calculate the base length, square it, sum it with the square of the height of the ramp, and take the square root of the sum.
  </p>
</details>
```

Alternatively, `aria-details` may refer to a link to a web page having the extended description, as shown in the following example.

EXAMPLE 22

```
<!-- Provision of an extended description -->

<p>
  See an <a href="http://foo.com/pt.html" id="det">Application of the Pythagorean Theorem</a>
</p>
```

Characteristics:

Characteristic	Value
Used in Roles:	All elements of the base markup
Value:	ID reference

aria-disabled state

Indicates that the [element](#) is [perceivable](#) but disabled, so it is not [editable](#) or otherwise [operable](#). See related [aria-hidden](#) and [aria_READONLY](#).

For example, irrelevant options in a radio group may be disabled. Disabled elements might not receive focus from the tab order. For some disabled elements, applications might choose not to support navigation to descendants. In addition to setting the [aria-disabled](#) attribute, authors **SHOULD** change the appearance (grayed out, etc.) to indicate that the item has been disabled.

The [state](#) of being disabled applies to the current element and all focusable descendant elements of the element on which the [aria-disabled](#) attribute is applied.

NOTE

While [aria-disabled](#) and proper scripting can successfully disable an element with role [link](#), fully disabling a host language equivalent can be problematic. Authors are advised not to use [aria-disabled](#) on elements that cannot be disabled through features of the host language alone.

NOTE: Usage on columnheader, rowheader and row

While [aria-disabled](#) is currently supported on [columnheader](#), [rowheader](#), and [row](#), in a future version the working group plans to prohibit its use on elements with any of those three roles except when they are in the context of a [grid](#) or [treegrid](#).

NOTE

This state is being deprecated as a global state in ARIA 1.2. In future versions it will only be allowed on roles where it is specifically supported.

Characteristics:

Characteristic	Value
Used in Roles:	application button composite gridcell group input link menuitem scrollbar separator tab
Inherits into Roles:	checkbox columnheader combobox grid listbox menu menubar menuitemcheckbox menuitemradio option radio radiogroup row rowheader

Characteristic	Value
	searchbox
	select
	slider
	spinbutton
	switch
	tablist
	textbox
	toolbar
	tree
	treegrid
	treeitem
Value:	true/false

Values:

Value	Description
false (default)	The element is enabled.
true	The element and all focusable descendants are disabled and its value cannot be changed by the user.

aria-dropeffect property

[Deprecated in ARIA 1.1] Indicates what functions can be performed when a dragged object is released on the drop target.

NOTE

The `aria-dropeffect` property is expected to be replaced by a new feature in a future version of WAI-ARIA. Authors are therefore advised to treat `aria-dropeffect` as [deprecated](#).

This [property](#) allows assistive technologies to convey the possible drag options available to users, including whether a pop-up menu of choices is provided by the application. Typically, drop effect functions can only be provided once an object has been grabbed for a drag operation as the drop effect functions available are dependent on the object being dragged.

More than one drop effect may be supported for a given [element](#). Therefore, the value of this [attribute](#) is a

space-separated set of tokens indicating the possible effects, or none if there is no supported operation. In addition to setting the [aria-dropeffect](#) attribute, authors **SHOULD** show a visual indication of potential drop targets.

Characteristics:

Characteristic	Value
Used in Roles:	All elements of the base markup
Value:	token list

Values:

Value	Description
copy	A duplicate of the source object will be dropped into the target.
execute	A function supported by the drop target is executed, using the drag source as an input.
link	A reference or shortcut to the dragged object will be created in the target object.
move	The source object will be removed from its current location and dropped into the target.
none (default)	No operation can be performed; effectively cancels the drag operation if an attempt is made to drop on this object. Ignored if combined with any other token value. e.g., 'none copy' is equivalent to a 'copy' value.
popup	There is a popup menu or dialog that allows the user to choose one of the drag operations (copy, move, link, execute) and any other drag functionality, such as cancel.

aria-errormessage property

Identifies the [element](#) that provides an error message for an [object](#). See related [aria-invalid](#) and [aria-describedby](#).

The [aria-errormessage](#) attribute references another element that contains error message text. Authors **MUST** use [aria-invalid](#) in conjunction with [aria-errormessage](#).

When the value of an object is not valid, [aria-invalid](#) is set to **true**, which indicates that the message contained by an element referenced by [aria-errormessage](#) is pertinent.

When an object is in a valid state, it has either `aria-invalid` set to `false` or it does not have the `aria-invalid` attribute. Authors *MAY* use `aria-errormessage` on an object that is currently valid, but only if the element referenced by `aria-errormessage` is `hidden`, because the message it contains is not pertinent.

When `aria-errormessage` is pertinent, authors *MUST* ensure the content is not hidden so users can navigate to and examine the error message. Similarly, when `aria-errormessage` is not pertinent, authors *MUST* either ensure the content is `hidden` or remove the `aria-errormessage` attribute or its value.

User agents *MUST NOT* expose `aria-errormessage` for an object with an `aria-invalid` value of `false`.

Authors *MAY* call attention to a newly rendered error message with a live region by either applying an `aria-live` property or using one of the `live region roles`, such as `alert`. A live region is appropriate when an error message is displayed to users after they have provided an invalid value.

A typical message describes what is wrong and informs users what is required. For example, an error message might be, “Invalid time: the time must be between 9:00 AM and 5:00 PM.” The following example code shows markup for an initial valid state and for a subsequent invalid state. Note the changes to `aria-invalid` on the text input `object`, and to `aria-live` on the `element` containing the text of the error message:

EXAMPLE 23

```
<!-- Initial valid state -->
<label for="startTime"> Please enter a start time for the meeting: </label>
<input id="startTime" type="text" aria-errormessage="msgID" value="" aria-invalid="false">
<span id="msgID" aria-live="assertive"><span style="visibility:hidden">Invalid time: the ti

<!-- User has input an invalid value -->
<label for="startTime"> Please enter a start time for the meeting: </label>
<input id="startTime" type="text" aria-errormessage="msgID" aria-invalid="true" value="11:3
<span id="msgID" aria-live="assertive"><span style="visibility:visible">Invalid time: the t
```

NOTE

This example uses `aria-live="assertive"` to indicate that assistive technologies should immediately announce the error message rather than completing other queued announcements first. This increases the likelihood that users are aware of the error message before they move focus out of the input.

NOTE

This state is being deprecated as a global state in ARIA 1.2. In future versions it will only be allowed on roles where it is specifically supported.

Characteristics:

Characteristic	Value
Used in Roles:	application checkbox combobox gridcell listbox radiogroup slider spinbutton textbox tree
Inherits into Roles:	columnheader rowheader searchbox switch treegrid
Value:	ID reference

aria-expanded state

Indicates whether a grouping element owned or controlled by this element is expanded or collapsed.

The [aria-expanded](#) attribute is applied to a focusable, interactive element that toggles visibility of content in another element. For example, it is applied to a parent [treeitem](#) to indicate whether its child branch of the tree is shown. Similarly, it can be applied to a [button](#) that controls visibility of a section of page content.

If a grouping container that can be expanded or collapsed is not [owned](#) by the element that has the [aria-expanded](#) attribute, the author **SHOULD** identify the controlling relationship by referencing the container from the element that has [aria-expanded](#) with the [aria-controls](#) property.

Characteristics:

Characteristic	Value
Used in Roles:	application

Characteristic	Value
	button checkbox combobox gridcell link listbox menuitem row rowheader tab treeitem
Inherits into Roles:	columnheader menuitemcheckbox menuitemradio rowheader switch
Value:	true/false/undefined

Values:

Value	Description
false	The grouping element this element owns or controls is collapsed.
true	The grouping element this element owns or controls is expanded.
undefined (default)	The element does not own or control a grouping element that is expandable.

aria-flowto property

Identifies the next [element](#) (or elements) in an alternate reading order of content which, at the user's discretion, allows assistive technology to override the general default of reading in document source order.

When [aria-flowto](#) has a single ID reference, it allows [assistive technologies](#) to, at the user's request,

forego normal document reading order and go to the targeted [object](#). However, when [aria-flowto](#) is provided with multiple ID references, assistive technologies **SHOULD** present the referenced elements as path choices.

In the case of one or more ID references, [user agents](#) or assistive technologies **SHOULD** give the user the option of navigating to any of the targeted elements. The name of the path can be determined by the name of the target element of the [aria-flowto](#) attribute. [Accessibility APIs](#) can provide named path [relationships](#).

Characteristics:

Characteristic	Value
Used in Roles:	All elements of the base markup
Value:	ID reference list

aria-grabbed state

[Deprecated in ARIA 1.1] Indicates an element's "grabbed" [state](#) in a drag-and-drop operation.

NOTE

The [aria-grabbed](#) state is expected to be replaced by a new feature in a future version of [WAI-ARIA](#). Authors are therefore advised to treat [aria-grabbed](#) as [deprecated](#).

Setting [aria-grabbed](#) to [true](#) indicates that the [element](#) has been selected for dragging. Setting [aria-grabbed](#) to [false](#) indicates that the element can be grabbed for a drag-and-drop operation, but is not currently grabbed. If [aria-grabbed](#) is unspecified or set to [undefined](#) (default), the element cannot be grabbed.

When [aria-grabbed](#) is set to [true](#), authors **SHOULD** update the [aria-dropeffect](#) attribute of all potential drop targets. When an element is not grabbed (the value is set to [false](#) or [undefined](#), or the attribute is removed), authors **SHOULD** revert the [aria-dropeffect](#) attributes of the associated drop targets to [none](#).

Characteristics:

Characteristic	Value
Used in Roles:	All elements of the base markup
Value:	true/false/undefined

Values:

Value	Description
-------	-------------

Value	Description
false	Indicates that the element supports being dragged.
true	Indicates that the element has been "grabbed" for dragging.
undefined (default)	Indicates that the element does not support being dragged.

aria-haspopup property

Indicates the availability and type of interactive popup element, such as menu or dialog, that can be triggered by an [element](#).

A popup element usually appears as a block of content that is on top of other content. Authors **MUST** ensure that the role of the element that serves as the container for the popup content is [menu](#), [listbox](#), [tree](#), [grid](#), or [dialog](#), and that the value of `aria-haspopup` matches the role of the popup container.

For the popup element to be keyboard accessible, authors **SHOULD** ensure that the element that can trigger the popup is focusable, that there is a keyboard mechanism for opening the popup, and that the popup element manages focus of all its descendants as described in [Managing Focus](#).

The `aria-haspopup` property is a token type. [User agents](#) **MUST** treat any value of `aria-haspopup` that is not included in the list of allowed values, including an empty string, as if the value `false` had been provided. To provide backward compatibility with ARIA 1.0 content, user agents **MUST** treat an `aria-haspopup` value of `true` as equivalent to a value of `menu`.

Assistive technologies **SHOULD NOT** expose the `aria-haspopup` property if it has a value of `false`.

NOTE

A [tooltip](#) is not considered to be a popup in this context.

NOTE

`aria-haspopup` is most relevant to use when there is a visual indicator in the element that triggers the popup. For example, many controls styled with a downward pointing triangle, chevron, or ellipsis (three consecutive dots) have become standard visual indicators that a popup will display when the control is activated. If some functional difference is relevant to display to a sighted user by means of a different visual style, that functional difference is usually relevant to convey to users of assistive technology. If there is no visual indication that an element will trigger a popup, authors are advised to consider whether use of `aria-haspopup` is necessary, and avoid using it when it's not.

NOTE

This property is being deprecated as a global property in ARIA 1.2. In future versions it will only be allowed on roles where it is specifically supported.

Characteristics:

Characteristic	Value
Related Concepts:	aria-controls
Used in Roles:	application button combobox gridcell link menuitem slider tab textbox treeitem
Inherits into Roles:	columnheader menuitemcheckbox menuitemradio rowheader searchbox
Value:	token

Values:

Value	Description
false (default)	Indicates the element does not have a popup.
true	Indicates the popup is a menu .
menu	Indicates the popup is a menu .
listbox	Indicates the popup is a listbox .
tree	Indicates the popup is a tree .

Value	Description
<code>grid</code>	Indicates the popup is a grid .
<code>dialog</code>	Indicates the popup is a dialog .

aria-hidden state

Indicates whether the [element](#) is exposed to an accessibility API. See related [aria-disabled](#).

User agents determine an element's [hidden](#) status based on whether it is rendered, and the rendering is usually controlled by [CSS](#). For example, an element whose `display` property is set to `none` is not rendered. An element is considered [hidden](#) if it, or any of its ancestors are not rendered or have their `aria-hidden` attribute value set to `true`.

Authors **MAY**, with caution, use `aria-hidden` to hide visibly rendered content from assistive technologies *only* if the act of hiding this content is intended to improve the experience for users of assistive technologies by removing redundant or extraneous content. Authors using `aria-hidden` to hide visible content from screen readers **MUST** ensure that identical or equivalent meaning and functionality is exposed to assistive technologies.

NOTE

Authors are advised to use extreme caution and consider a wide range of disabilities when hiding visibly rendered content from assistive technologies. For example, a sighted, dexterity-impaired individual may use voice-controlled assistive technologies to access a visual interface. If an author hides visible link text "Go to checkout" and exposes similar, yet non-identical link text "Check out now" to the accessibility API, the user may be unable to access the interface they perceive using voice control. Similar problems may also arise for screen reader users. For example, a sighted telephone support technician may attempt to have the blind screen reader user click the "Go to checkout" link, which they may be unable to find using a type-ahead item search ("Go to...").

NOTE

At the time of this writing, `aria-hidden="false"` is known to work inconsistently in browsers. As future implementations improve, use caution and test thoroughly before relying on this approach.

Characteristics:

Characteristic	Value
Used in Roles:	All elements of the base markup
Value:	true/false/undefined

Values:

Value	Description
false	The element is exposed to the accessibility API as if it was rendered.
true	The element is hidden from the accessibility API.
undefined (default)	The element's hidden state is determined by the user agent based on whether it is rendered.

aria-invalid state

Indicates the entered value does not conform to the format expected by the application. See related [aria-errormessage](#).

If the value is computed to be invalid or out-of-range, the application author **SHOULD** set this [attribute](#) to **true**. [User agents](#) **SHOULD** inform the user of the error. Application authors **SHOULD** provide suggestions for corrections if they are known.

When the user attempts to submit data involving a field for which [aria-required](#) is **true**, authors **MAY** use the [aria-invalid](#) attribute to signal there is an error. However, if the user has not attempted to submit the form, authors **SHOULD NOT** set the [aria-invalid](#) attribute on required [widgets](#) simply because the user has not yet entered data.

For future expansion, the [aria-invalid](#) attribute is a token type. Any value not recognized in the list of allowed values **MUST** be treated by user agents as if the value **true** had been provided. If the attribute is not present, or its value is **false**, or its value is an empty string, the default value of **false** applies.

NOTE

This state is being deprecated as a global state in ARIA 1.2. In future versions it will only be allowed on roles where it is specifically supported.

Characteristics:

Characteristic	Value
Used in Roles:	application checkbox combobox gridcell listbox

Characteristic	Value
	radiogroup slider spinbutton textbox tree
Inherits into Roles:	columnheader rowheader searchbox switch treagrid
Value:	token

Values:

Value	Description
grammar	A grammatical error was detected.
false (default)	There are no detected errors in the value.
spelling	A spelling error was detected.
true	The value entered by the user has failed validation.

aria-keyshortcuts property

Indicates keyboard shortcuts that an author has implemented to activate or give focus to an element.

The value of the `aria-keyshortcuts` attribute is a space-separated list of keyboard shortcuts that can be pressed to activate a command or textbox widget. The keys defined in the shortcuts represent the physical keys pressed and not the actual characters generated. Each keyboard shortcut consists of one or more tokens delimited by the plus sign ("+") representing zero or more modifier keys and exactly one non-modifier key that must be pressed simultaneously to activate the given shortcut.

Authors **MUST** specify modifier keys exactly according to the [UI Events KeyboardEvent key Values](#) spec [`uiEvents-key`] - for example, "Alt", "Control", "Shift", "Meta", or "AltGraph". Note that Meta corresponds to the Command key, and Alt to the Option key, on Apple computers.

The valid names for non-modifier keys are any printable character such as "A", "B", "1", "2", "\$", "Plus" for a plus sign, "Space" for the spacebar, or the names of any other non-modifier key specified in the [UI Events](#)

[KeyboardEvent key Values](#) spec [uievents-key] - for example, "Enter", "Tab", "ArrowRight", "PageDown", "Escape", or "F1". The use of "Space" for the spacebar is an exception to the [UI Events KeyboardEvent key Values](#) spec [uievents-key] as the space or spacebar key is encoded as ' ' and would be treated as a whitespace character.

Authors **MUST** ensure modifier keys come first when they are part of a keyboard shortcut. Authors **MUST** ensure that required non-modifier keys come last when they are part of a shortcut. The order of the modifier keys is not otherwise significant, so "Alt+Shift+T" and "Shift+Alt+T" are equivalent, but "T+Shift+Alt" is not valid because all of the modifier keys don't come first, and "Alt" is not valid because it doesn't include at least one non-modifier key.

When specifying an alphabetic key, both the uppercase and lowercase variants are considered equivalent: "a" and "A" are the same.

When implementing keyboard shortcuts authors should consider the keyboards they intend to support to avoid unintended results. Keyboard designs vary significantly based on the device used and the languages supported. For example, many modifier keys are used in conjunction with other keys to create common punctuation symbols, create number characters, swap keyboard sides on bilingual keyboards to switch languages, and perform a number of other functions.

For many supported keyboards, authors can prevent conflicts by avoiding keys other than ASCII letters, as number characters and common punctuation often require modifiers. Here, the keyboard shortcut entered does not equate to the key generated. For example, in French keyboard layouts, the number characters are not available until you press the Control key, so a keyboard shortcut defined as "Control+2" would be ambiguous as this is how one would type the "2" character on a French keyboard.

If the character used is determined by a modifier key, the author **MUST** specify the actual key used to generate the character, that is generated by the key, and not the resulting character. This convention enables the assistive technology to accurately convey what keys must be used to generate the shortcut. For example, on most U.S. English keyboards, the percent sign "%" can be input by pressing Shift+5. The correct way to specify this shortcut is "Shift+5". It is incorrect to specify "%" or "Shift+%" . However, note that on some international keyboards the percent sign may be an unmodified key, in which case "%" and "Shift+%" could be correct on those keyboards.

If the key that needs to be specified is illegal in the host language or would cause a string to be terminated, authors **MUST** use the string escaping sequence of the host language to specify it. For example, the double-quote character can be encoded as "Shift+'" in [HTML](#).

Examples of valid keyboard shortcuts include:

- "A"
- "Shift+Space"
- "Control+Alt+."

- "Control+Shift+;"
- "Alt+Shift+P Control+F"
- "Meta+C Meta+Shift+C"

User agents **MUST NOT** change keyboard behavior in response to the `aria-keyshortcuts` attribute.

Authors **MUST** handle scripted keyboard events to process `aria-keyshortcuts`. The `aria-keyshortcuts` attribute exposes the existence of these shortcuts so that assistive technologies can communicate this information to users.

Authors **SHOULD** provide a way to expose keyboard shortcuts so that all users may discover them, such as through the use of a tooltip. Authors **MUST** ensure that `aria-keyshortcuts` applied to disabled elements are unavailable.

Authors **SHOULD** avoid implementing shortcut keys that inhibit operating system, user agent, or assistive technology functionality. This requires the author to carefully consider both which keys to assign and the contexts and conditions in which the keys are available to the user. For guidance, see the keyboard shortcuts section of the [WAI-ARIA Authoring Practices](#).

Characteristics:

Characteristic	Value
Related Concepts:	Keyboard shortcut
Used in Roles:	All elements of the base markup
Value:	string

aria-label property

Defines a string value that labels the current element. See related [aria-labelledby](#).

The purpose of [aria-label](#) is the same as that of [aria-labelledby](#). It provides the user with a recognizable name of the object. The most common [accessibility API](#) mapping for a label is the [accessible name](#) property.

If the label text is available in the [DOM](#) (i.e. typically visible text content), authors **SHOULD** use [aria-labelledby](#) and **SHOULD NOT** use [aria-label](#). There may be instances where the name of an element cannot be determined programmatically from the [DOM](#), and there are cases where referencing [DOM](#) content is not the desired user experience. Most host languages provide an attribute that could be used to name the element (e.g., the `title` attribute in [\[HTML\]](#)), yet this could present a browser tooltip. In the cases where [DOM](#) content or a tooltip is undesirable, authors **MAY** set the accessible name of the element using [aria-label](#). As required by the [accessible name and description computation](#), user agents give precedence to [aria-labelledby](#) over [aria-label](#) when computing the accessible name property.

Characteristics:

Characteristic	Value
Related Concepts:	title attribute in [HTML]
Used in Roles:	All elements of the base markup except for the following roles: caption , code , deletion , emphasis , generic , insertion , paragraph , presentation , strong , subscript , superscript
Value:	string

aria-labelledby property

Identifies the [element](#) (or elements) that labels the current element. See related [aria-describedby](#).

The purpose of [aria-labelledby](#) is the same as that of [aria-label](#). It provides the user with a recognizable name of the object. The most common [accessibility API](#) mapping for a label is the [accessible name](#) property.

If the interface is such that it is not possible to have a visible label on the screen, authors **SHOULD** use [aria-label](#) and **SHOULD NOT** use [aria-labelledby](#). As required by the [accessible name and description computation](#), user agents give precedence to [aria-labelledby](#) over [aria-label](#) when computing the accessible name property.

The [aria-labelledby](#) attribute is similar to [aria-describedby](#) in that both reference other elements to calculate a text alternative, but a label should be concise, where a description is intended to provide more verbose information.

NOTE

The expected spelling of this property in U.S. English is "labeledby." However, the [accessibility API](#) features to which this property is mapped have established the "labelledby" spelling. This property is spelled that way to match the convention and minimize the difficulty for developers.

Characteristics:

Characteristic	Value
Related Concepts:	<label> in [HTML]
Used in Roles:	All elements of the base markup except for the following roles: caption , code , deletion , emphasis , generic , insertion , paragraph , presentation , strong , subscript , superscript

Characteristic	Value
Value:	ID reference list

aria-level property

Defines the hierarchical level of an [element](#) within a structure.

This can be applied inside trees to tree items, to headings inside a document, to nested grids, nested tablists and to other structural items that may appear inside a container or participate in an ownership hierarchy. The value for [aria-level](#) is an integer greater than or equal to 1.

Levels increase with depth. If the [DOM](#) ancestry does not accurately represent the level, authors **SHOULD** explicitly define the [aria-level](#) attribute.

This attribute is applied to elements that act as leaf nodes within the orientation of the set, for example, on elements with role [treeitem](#) rather than elements with role [group](#). This means that multiple elements in a set may have the same value for this attribute. Although it would be less repetitive to provide a single value on the container, restricting this to leaf nodes ensures that there is a single way for [assistive technologies](#) to use the attribute.

If the [DOM](#) ancestry accurately represents the level, the [user agent](#) can calculate the level of an item from the document structure. This attribute can be used to provide an explicit indication of the level when that is not possible to calculate from the document structure or the [aria-owns](#) attribute. User agent support for automatic calculation of level may vary; authors **SHOULD** test with [user agents](#) and assistive technologies to determine whether this attribute is needed. If the author intends for the user agent to calculate the level, the author **SHOULD** omit this attribute.

NOTE

In the case of a [treegrid](#), [aria-level](#) is supported on elements with the role [row](#), not elements with role [gridcell](#). At first glance, this may seem inconsistent with the application of [aria-level](#) on [treeitem](#) elements, but it is consistent in that the [row](#) acts as the leaf node within the vertical orientation of the [grid](#), whereas the [gridcell](#) is a leaf node within the horizontal orientation of each [row](#). Level is not supported on sets of cells within rows, so the [aria-level](#) attribute is applied to the element with the role [row](#).

Characteristics:

Characteristic	Value
Used in Roles:	heading listitem

Characteristic	Value
	row
Inherits into Roles:	treeitem
Value:	integer

aria-live property

Indicates that an [element](#) will be updated, and describes the types of updates the [user agents](#), [assistive technologies](#), and user can expect from the [live region](#).

The values of this [attribute](#) are expressed in degrees of importance. When regions are specified as [polite](#), assistive technologies will notify users of updates but generally do not interrupt the current task, and updates take low priority. When regions are specified as [assertive](#), assistive technologies will immediately notify the user, and could potentially clear the speech queue of previous updates.

Politeness levels are essentially an ordering mechanism for updates and serve as a strong suggestion to user agents or assistive technologies. The value may be overridden by user agents, assistive technologies, or the user. For example, if assistive technologies can determine that a change occurred in response to a key press or a mouse click, the assistive technologies may present that change immediately even if the value of the [aria-live](#) attribute states otherwise.

Since different users have different needs, it is up to the user to tweak his or her assistive technologies' response to a live region with a certain politeness level from the commonly defined baseline. Assistive technologies may choose to implement increasing and decreasing levels of granularity so that the user can exercise control over queues and interruptions.

When the [property](#) is not set on an [object](#) that needs to send updates, the politeness level is the value of the nearest ancestor that sets the [aria-live](#) attribute.

The [aria-live](#) attribute is the primary determination for the order of presentation of changes to live regions. Implementations will also consider the default level of politeness in a [role](#) when the [aria-live](#) attribute is not set in the ancestor chain (e.g., [log](#) changes are [polite](#) by default). Items which are [assertive](#) will be presented immediately, followed by [polite](#) items. User agents or assistive technologies **MAY** choose to clear queued changes when an assertive change occurs. (e.g., changes in an assertive region may remove all currently queued changes)

When live regions are marked as [polite](#), assistive technologies **SHOULD** announce updates at the next graceful opportunity, such as at the end of speaking the current sentence or when the user pauses typing.

When live regions are marked as [assertive](#), assistive technologies **SHOULD** notify the user immediately. Because an interruption may disorient users or cause them to not complete their current task, authors **SHOULD NOT** use the assertive value unless the interruption is imperative.

Characteristics:

Characteristic	Value
Used in Roles:	All elements of the base markup
Value:	token

Values:

Value	Description
assertive	Indicates that updates to the region have the highest priority and should be presented to the user immediately.
off (default)	Indicates that updates to the region should not be presented to the user unless the user is currently focused on that region.
polite	Indicates that updates to the region should be presented at the next graceful opportunity, such as at the end of speaking the current sentence or when the user pauses typing.

aria-modal property

Indicates whether an [element](#) is modal when displayed.

The [aria-modal attribute](#) is used to indicate that the presence of a "modal" element precludes usage of other content on the page. For example, when a modal dialog is displayed, it is expected that the user's interaction is limited to the contents of the dialog, until the modal dialog loses focus or is no longer displayed.

When a modal element is displayed, assistive technologies **SHOULD** navigate to the element unless focus has explicitly been set elsewhere. Assistive technologies **MAY** limit navigation to the modal element's contents. If focus moves to an element outside the modal element, assistive technologies **SHOULD NOT** limit navigation to the modal element.

When a modal element is displayed, authors **MUST** ensure the interface can be controlled using only descendants of the modal element. In other words, if a modal dialog has a close button, the button should be a descendant of the dialog. When a modal element is displayed, authors **SHOULD** mark all other contents as inert (such as "inert subtrees" in [HTML](#)) if the ability to do so exists in the host language.

Characteristics:

Characteristic	Value

Characteristic	Value
Used in Roles:	window
Inherits into Roles:	alertdialog dialog
Value:	true/false

Values:

Value	Description
false (default)	Element is not modal.
true	Element is modal.

aria-multiline property

Indicates whether a text box accepts multiple lines of input or only a single line.

NOTE

In most user agent implementations, the default behavior of the ENTER or RETURN key is different between the single-line and multi-line text fields in [HTML](#). When user has focus in a single-line `<input type="text">` element, the keystroke usually submits the form. When user has focus in a multi-line `<textarea>` element, the keystroke inserts a line break. The [WAI-ARIA textbox role](#) differentiates these types of boxes with the [aria-multiline](#) attribute, so authors are advised to be aware of this distinction when designing the field.

Characteristics:

Characteristic	Value
Used in Roles:	textbox
Inherits into Roles:	searchbox
Value:	true/false

Values:

Value	Description
false (default)	This is a single-line text box.
true	This is a multi-line text box.

aria-multiselectable property

Indicates that the user may select more than one item from the current selectable descendants.

Authors **SHOULD** ensure that selected descendants have the [aria-selected](#) attribute set to `true`, and selectable descendant have the [aria-selected](#) attribute set to `false`. Authors **SHOULD NOT** use the [aria-selected](#) attribute on descendants that are not selectable.

NOTE

Lists and trees are examples of roles that might allow users to select more than one item at a time.

Characteristics:

Characteristic	Value
Used in Roles:	grid listbox tablist tree
Inherits into Roles:	treegrid
Value:	true/false

Values:

Value	Description
false (default)	Only one item can be selected.
true	More than one item in the widget may be selected at a time.

aria-orientation property

Indicates whether the element's orientation is horizontal, vertical, or unknown/ambiguous.

NOTE

In ARIA 1.1, the default value for [aria-orientation](#) changed from `horizontal` to `undefined`. Implicit defaults are defined on some roles (e.g., [slider](#) defaults to horizontal; [scrollbar](#) defaults to vertical) but remain undefined on roles where an expected default orientation is ambiguous (e.g., [radiogroup](#)).

Characteristics:

Characteristic	Value
Used in Roles:	scrollbar select separator slider tablist toolbar
Inherits into Roles:	listbox menu menubar radiogroup tree treegrid
Value:	token

Values:

Value	Description
horizontal	The element is oriented horizontally.
undefined (default)	The element's orientation is unknown/ambiguous.
vertical	The element is oriented vertically.

aria-owns property

Identifies an [element](#) (or elements) in order to define a visual, functional, or contextual parent/child [relationship](#) between [DOM](#) elements where the [DOM](#) hierarchy cannot be used to represent the relationship. See related [aria-controls](#).

The value of the [aria-owns attribute](#) is a space-separated ID reference list that references one or more elements in the document by ID. The reason for adding [aria-owns](#) is to expose a parent/child contextual relationship to [assistive technologies](#) that is otherwise impossible to infer from the [DOM](#).

If an element has both [aria-owns](#) and [DOM](#) children then the order of the child elements with respect to the parent/child relationship is the [DOM](#) children first, then the elements referenced in [aria-owns](#). If the author

intends that the DOM children are not first, then list the DOM children in [aria-owns](#) in the desired order. Authors **SHOULD NOT** use [aria-owns](#) as a replacement for the DOM hierarchy. If the relationship is represented in the DOM, do not use [aria-owns](#). Authors **MUST** ensure that an element's ID is not specified in more than one other element's [aria-owns](#) attribute at any time. In other words, an element can have only one explicit owner.

Characteristics:

Characteristic	Value
Used in Roles:	All elements of the base markup
Value:	ID reference list

aria-placeholder property

Defines a short hint (a word or short phrase) intended to aid the user with data entry when the control has no value. A hint could be a sample value or a brief description of the expected format.

Authors **SHOULD NOT** use [aria-placeholder](#) instead of a label as their purposes are different: The label indicates what kind of information is expected. The placeholder text is a hint about the expected value. See related [aria-labelledby](#) and [aria-label](#).

Authors **SHOULD** present this hint to the user by displaying the hint text at any time the control's value is the empty string. This includes cases where the control first receives focus, and when users remove a previously-entered value.

NOTE

As is the case with the related `placeholder` attribute in [HTML], use of placeholder text as a replacement for a displayed label can reduce the accessibility and usability of the control for a range of users including older users and users with cognitive, mobility, fine motor skill or vision impairments. While the hint given by the control's label is shown at all times, the short hint given in the `placeholder` attribute is only shown before the user enters a value. Furthermore, placeholder text may be mistaken for a pre-filled value, and as commonly implemented the default color of the placeholder text provides insufficient contrast and the lack of a separate visible label reduces the size of the hit region available for setting focus on the control.

NOTE

The following examples do not use the `HTML label` element as it cannot be used to label `HTML` elements with `contenteditable`.

The following example shows a [searchbox](#) in which the user has entered a value:

EXAMPLE 24

```
<span id="label">Birthday:</span>
<div contenteditable role="searchbox" aria-labelledby="label" aria-placeholder="MM-DD-YYYY">
```

The following example shows the same [searchbox](#) in which the user has not yet entered a value or has removed a previously-entered value:

EXAMPLE 25

```
<span id="label">Birthday:</span>
<div contenteditable role="searchbox" aria-labelledby="label" aria-placeholder="MM-DD-YYYY">
```

Characteristics:

Characteristic	Value
Related Concepts:	placeholder attribute in [HTML]
Used in Roles:	textbox
Inherits into Roles:	searchbox
Value:	string

aria-posinset property

Defines an [element](#)'s number or position in the current set of listitems or treeitems. Not required if all elements in the set are present in the [DOM](#). See related [aria-setsize](#).

If all items in a set are present in the document structure, it is not necessary to set this [attribute](#), as the [user agent](#) can automatically calculate the set size and position for each item. However, if only a portion of the set is present in the document structure at a given moment, this [property](#) is needed to provide an explicit indication of an element's position.

The following example shows items 5 through 8 in a set of 16.

EXAMPLE 26

```
<h2 id="label_fruit"> Available Fruit </h2>
<ul role="listbox" aria-labelledby="label_fruit">
  <li role="option" aria-setsize="16" aria-posinset="5"> apples </li>
  <li role="option" aria-setsize="16" aria-posinset="6"> bananas </li>
  <li role="option" aria-setsize="16" aria-posinset="7"> cantaloupes </li>
  <li role="option" aria-setsize="16" aria-posinset="8"> dates </li>
</ul>
```

Authors **MUST** set the value for `aria-posinset` to an integer greater than or equal to 1, and less than or equal to the size of the set when that size is known. Authors **SHOULD** use `aria-setsize`.

When exposing `aria-posinset` on a `menuitem`, `menuitemcheckbox`, or `menuitemradio`, authors **SHOULD** set the value of `aria-posinset` with respect to the total number of items in the `menu`, excluding any separators.

Characteristics:

Characteristic	Value
Used in Roles:	<code>article</code> <code>listitem</code> <code>menuitem</code> <code>option</code> <code>radio</code> <code>row</code> <code>tab</code>
Inherits into Roles:	<code>menuitemcheckbox</code> <code>menuitemradio</code> <code>treeitem</code>
Value:	<code>integer</code>

aria-pressed state

Indicates the current "pressed" `state` of toggle buttons. See related `aria-checked` and `aria-selected`.

Toggle buttons require a full press-and-release cycle to change their value. Activating it once changes the value to `true`, and activating it another time changes the value back to `false`. A value of `mixed` means that

the values of more than one item controlled by the button do not all share the same value. If the [attribute](#) is not present, the button is not a toggle button.

The [aria-pressed](#) attribute is similar but not identical to the [aria-checked](#) attribute. Operating systems support **pressed** on buttons and **checked** on checkboxes.

Characteristics:

Characteristic	Value
Used in Roles:	button
Value:	tristate

Values:

Value	Description
false	The element supports being pressed but is not currently pressed.
mixed	Indicates a mixed mode value for a tri-state toggle button.
true	The element is pressed.
undefined (default)	The element does not support being pressed.

ariareadonly property

Indicates that the [element](#) is not editable, but is otherwise [operable](#). See related [aria-disabled](#).

This means the user can read but not set the value of the [widget](#). Readonly elements are relevant to the user, and application authors **SHOULD NOT** restrict navigation to the element or its focusable descendants. Other actions such as copying the value of the element are also supported. This is in contrast to disabled elements, to which applications might not allow user navigation to descendants.

Examples include:

- A form element which represents a constant.
- Row or column headers in a spreadsheet grid.
- The result of a calculation such as a shopping cart total.

Characteristics:

Characteristic	Value
Related Concepts:	readonly attribute in [HTML]

Characteristic	Value
Used in Roles:	checkbox combobox grid gridcell listbox radiogroup slider spinbutton textbox
Inherits into Roles:	columnheader rowheader searchbox switch treegrid
Value:	true/false

Values:

Value	Description
false (default)	The user can set the value of the element.
true	The user cannot change the value of the element.

aria-relevant property

Indicates what notifications the user agent will trigger when the accessibility tree within a live region is modified. See related [aria-atomic](#).

The [attribute](#) is represented as a space-separated list of the following values: `additions`, `removals`, `text`; or a single catch-all value `all`.

This is used to describe [semantically](#) meaningful changes, as opposed to merely presentational ones. For example, nodes that are removed from the top of a log are merely removed for purposes of creating room for other entries, and the removal of them does not have meaning. However, in the case of a buddy list, removal of a buddy name indicates that they are no longer online, and this is a meaningful [event](#). In that case [aria-](#)

[relevant](#) will be set to `all`. When the [aria-relevant](#) attribute is not provided, the default value, `additions text`, indicates that text modifications and node additions are relevant, but that node removals are irrelevant.

NOTE

[aria-relevant](#) values of `removals` or `all` are to be used sparingly. Assistive technologies only need to be informed of content removal when its removal represents an important change, such as a buddy leaving a chat room.

NOTE

Text removals should only be considered relevant if one of the specified values is '`removals`' or '`all`'. For example, for a text change from '`foo`' to '`bar`' in a live region with a default [aria-relevant](#) value, the text addition ('`bar`') would be spoken, but the text removal ('`foo`') would not.

[aria-relevant](#) is an optional attribute of live regions. This is a suggestion to [assistive technologies](#), but assistive technologies are not required to present changes of all the relevant types.

When [aria-relevant](#) is not defined, an element's value is inherited from the nearest ancestor with a defined value. Although the value is a [token list](#), inherited values are not additive; the value provided on a descendant element completely overrides any inherited value from an ancestor element.

When text changes are denoted as relevant, user agents **MUST** monitor any descendant node change that affects the [accessible name and description computation](#) of the live region as if the accessible name were determined from contents ([nameFrom: contents](#)). For example, a text change would be triggered if the [HTML alt](#) attribute of a contained image changed. However, no change would be triggered if there was a text change to a node outside the live region, even if that node was referenced (via [aria-labelledby](#)) by an element contained in the live region.

Characteristics:

Characteristic	Value
Used in Roles:	All elements of the base markup
Value:	<u>token list</u>

Values:

Value	Description
additions	Element nodes are added to the accessibility tree within the live region.

Value	Description
additions text (default)	Equivalent to the combination of values, "additions text".
all	Equivalent to the combination of all values, "additions removals text".
removals	Text content, a text alternative, or an element node within the live region is removed from the accessibility tree.
text	Text content or a text alternative is added to any descendant in the accessibility tree of the live region.

aria-required property

Indicates that user input is required on the [element](#) before a form may be submitted.

For example, if the user needs to fill in an address field, the author will need to set the field's [aria-required](#) attribute to true.

NOTE

The fact that the element is required is often presented visually (such as a sign or symbol after the [widget](#)). Using the [aria-required attribute](#) allows the author to explicitly convey to [assistive technologies](#) that an element is required.

Unless an exactly equivalent native attribute is available, host languages **SHOULD** allow authors to use the [aria-required](#) attribute on host language form elements that require input or selection by the user.

Characteristics:

Characteristic	Value
Related Concepts:	required attribute in [HTML]
Used in Roles:	checkbox combobox gridcell listbox radiogroup spinbutton textbox

Characteristic	Value
	tree
Inherits into Roles:	columnheader rowheader searchbox switch treemap
Value:	true/false

Values:

Value	Description
false (default)	User input is not necessary to submit the form.
true	Users need to provide input on an element before a form is submitted.

aria-roledescription property

Defines a human-readable, author-localized description for the [role](#) of an [element](#).

Some [assistive technologies](#), such as screen readers, present the role of an element as part of the user experience. Such assistive technologies typically localize the name of the role, and they may customize it as well. Users of these assistive technologies depend on the presentation of the role name, such as "region," "button," or "slider," for an understanding of the purpose of the element and, if it is a widget, how to interact with it.

The [aria-roledescription](#) property gives authors the ability to override how assistive technologies localize and express the name of a role. Thus inappropriately using [aria-roledescription](#) may inhibit users' ability to understand or interact with an element. Authors **SHOULD** limit use of [aria-roledescription](#) to clarifying the purpose of non-interactive container roles like [group](#) or [region](#), or to providing a *more specific* description of a [widget](#).

When using [aria-roledescription](#), authors **SHOULD** also ensure that:

1. The element to which [aria-roledescription](#) is applied has a valid [WAI-ARIA](#) role or has an implicit [WAI-ARIA](#) role semantic.
2. The value of [aria-roledescription](#) is not empty or does not contain only whitespace characters.

User agents **MUST NOT** expose the [aria-roledescription](#) property if any of the following conditions

exist:

1. The element to which `aria-roledescription` is applied does not have a valid WAI-ARIA role or does not have an implicit WAI-ARIA role semantic.
2. The element to which `aria-roledescription` is applied has an explicit or implicit WAI-ARIA role where `aria-roledescription` is prohibited.
3. The value of `aria-roledescription` is empty or contains only whitespace characters.

Assistive technologies **SHOULD** use the value of `aria-roledescription` when presenting the role of an element, but **SHOULD NOT** change other functionality based on the role of an element that has a value for `aria-roledescription`. For example, an assistive technology that provides functions for navigating to the next region or button **SHOULD** allow those functions to navigate to regions and buttons that have an `aria-roledescription`.

The following two examples show the use of `aria-roledescription` to indicate that a non-interactive container is a "slide" in a web-based presentation application.

EXAMPLE 27

```
<div role="region" aria-roledescription="slide" id="slide42" aria-labelledby="slide42headir">
<h1 id="slide42heading">Quarterly Report</h1>
<!-- remaining slide contents -->
</div>
```

EXAMPLE 28

```
<section aria-roledescription="slide" id="slide42" aria-labelledby="slide42headir">
<h1 id="slide42heading">Quarterly Report</h1>
<!-- remaining slide contents -->
</section>
```

In the previous examples, a screen reader user may hear "Quarterly Report, slide" rather than the more vague "Quarterly Report, region" or "Quarterly Report, group."

Characteristics:

Characteristic	Value
Used in Roles:	All elements of the base markup
Value:	<u>string</u>

aria-rowcount property

Defines the total number of rows in a [table](#), [grid](#), or [treemap](#). See related [aria-rowindex](#).

If all of the rows are present in the DOM, it is not necessary to set this [attribute](#) as the [user agent](#) can automatically calculate the total number of rows. However, if only a portion of the rows is present in the DOM at a given moment, this attribute is needed to provide an explicit indication of the number of rows in the full table.

Authors **MUST** set the value of [aria-rowcount](#) to an integer equal to the number of rows in the full table. If the total number of rows is unknown, authors **MUST** set the value of [aria-rowcount](#) to -1 to indicate that the value should not be calculated by the user agent.

The following example shows a grid with 2000 rows, of which the first row and rows 100 through 102 are displayed to the user.

EXAMPLE 29

```
<div role="grid" aria-rowcount="2000">
  <div role="rowgroup">
    <div role="row" aria-rowindex="1">
      <span role="columnheader">First Name</span>
      <span role="columnheader">Last Name</span>
      <span role="columnheader">Company</span>
      <span role="columnheader">Phone</span>
    </div>
  </div>
  <div role="rowgroup">
    <div role="row" aria-rowindex="100">
      <span role="gridcell">Fred</span>
      <span role="gridcell">Jackson</span>
      <span role="gridcell">Acme, Inc.</span>
      <span role="gridcell">555-1234</span>
    </div>
    <div role="row" aria-rowindex="101">
      <span role="gridcell">Sara</span>
      <span role="gridcell">James</span>
      <span role="gridcell">Acme, Inc.</span>
      <span role="gridcell">555-1235</span>
    </div>
    <div role="row" aria-rowindex="102">
      <span role="gridcell">Taylor</span>
      <span role="gridcell">Johnson</span>
      <span role="gridcell">Acme, Inc.</span>
      <span role="gridcell">555-1236</span>
    </div>
  </div>
</div>
```

Characteristics:

Characteristic	Value
----------------	-------

Characteristic	Value
Used in Roles:	table
Inherits into Roles:	grid treegrid
Value:	integer

aria-rowindex property

Defines an [element's](#) row index or position with respect to the total number of rows within a [table](#), [grid](#), or [treegrid](#). See related [aria-rowcount](#) and [aria-rowspan](#).

If all of the rows are present in the [DOM](#), it is not necessary to set this [attribute](#) as the [user agent](#) can automatically calculate the index of each row. However, if only a portion of the rows is present in the [DOM](#) at a given moment, this attribute is needed to provide an explicit indication of each row's position with respect to the full table.

Authors **MUST** set the value for [aria-rowindex](#) to an integer greater than or equal to 1, greater than the [aria-rowindex](#) value of any previous rows, and less than or equal to the number of rows in the full table. For a cell or gridcell which spans multiple rows, authors **MUST** set the value of [aria-rowindex](#) to the start of the span.

Authors **SHOULD** place [aria-rowindex](#) on each row. Authors **MAY** also place [aria-rowindex](#) on all of the children or [owned elements](#) of each row.

The following example shows a grid with 2000 rows, of which the first row and rows 100 through 102 are displayed to the user.

EXAMPLE 30

```
<div role="grid" aria-rowcount="2000">
  <div role="rowgroup">
    <div role="row" aria-rowindex="1">
      <span role="columnheader">First Name</span>
      <span role="columnheader">Last Name</span>
      <span role="columnheader">Company</span>
      <span role="columnheader">Phone</span>
    </div>
  </div>
  <div role="rowgroup">
    <div role="row" aria-rowindex="100">
      <span role="gridcell">Fred</span>
      <span role="gridcell">Jackson</span>
      <span role="gridcell">Acme, Inc.</span>
      <span role="gridcell">555-1234</span>
    </div>
    <div role="row" aria-rowindex="101">
      <span role="gridcell">Sara</span>
      <span role="gridcell">James</span>
      <span role="gridcell">Acme, Inc.</span>
      <span role="gridcell">555-1235</span>
    </div>
    <div role="row" aria-rowindex="102">
      <span role="gridcell">Taylor</span>
      <span role="gridcell">Johnson</span>
      <span role="gridcell">Acme, Inc.</span>
      <span role="gridcell">555-1236</span>
    </div>
  </div>
</div>
</div>
```

The following example shows the grid from the previous example with `aria-rowindex` also placed on all of the owned elements of each row.

EXAMPLE 31

```
<div role="grid" aria-rowcount="2000">
  <div role="rowgroup">
    <div role="row" aria-rowindex="1">
      <span role="columnheader" aria-rowindex="1">First Name</span>
      <span role="columnheader" aria-rowindex="1">Last Name</span>
      <span role="columnheader" aria-rowindex="1">Company</span>
      <span role="columnheader" aria-rowindex="1">Phone</span>
    </div>
  </div>
  <div role="rowgroup">
    <div role="row" aria-rowindex="100">
      <span role="gridcell" aria-rowindex="100">Fred</span>
      <span role="gridcell" aria-rowindex="100">Jackson</span>
      <span role="gridcell" aria-rowindex="100">Acme, Inc.</span>
      <span role="gridcell" aria-rowindex="100">555-1234</span>
    </div>
    <div role="row" aria-rowindex="101">
      <span role="gridcell" aria-rowindex="101">Sara</span>
      <span role="gridcell" aria-rowindex="101">James</span>
      <span role="gridcell" aria-rowindex="101">Acme, Inc.</span>
      <span role="gridcell" aria-rowindex="101">555-1235</span>
    </div>
    <div role="row" aria-rowindex="102">
      <span role="gridcell" aria-rowindex="102">Taylor</span>
      <span role="gridcell" aria-rowindex="102">Johnson</span>
      <span role="gridcell" aria-rowindex="102">Acme, Inc.</span>
      <span role="gridcell" aria-rowindex="102">555-1236</span>
    </div>
  </div>
</div>
```

Characteristics:

Characteristic	Value
Used in Roles:	<u>cell</u> <u>row</u>
Inherits into Roles:	<u>columnheader</u> <u>gridcell</u> <u>rowheader</u>
Value:	<u>integer</u>

aria-rowspan property

Defines the number of rows spanned by a cell or gridcell within a table, grid, or treegrid. See related

[aria-rowindex](#) and [aria-colspan](#).

This [attribute](#) is intended for cells and gridcells which are not contained in a native table. When defining the row span of cells or gridcells in a native table, authors **SHOULD** use the host language's attribute instead of [aria-rowspan](#). If [aria-rowspan](#) is used on an element for which the host language provides an equivalent attribute, [user agents](#) **MUST** ignore the value of [aria-rowspan](#) and instead expose the value of the host language's attribute to [assistive technologies](#).

Authors **MUST** set the value of [aria-rowspan](#) to an integer greater than or equal to 0 and less than the value which would cause the cell or gridcell to overlap the next cell or gridcell in the same column. Setting the value to 0 indicates that the cell or gridcell is to span all the remaining rows in the row group.

Characteristics:

Characteristic	Value
Used in Roles:	<u>cell</u>
Inherits into Roles:	<u>columnheader</u> <u>rowheader</u>
Value:	<u>integer</u>

[aria-selected state](#)

Indicates the current "selected" [state](#) of various [widgets](#). See related [aria-checked](#) and [aria-pressed](#).

This [attribute](#) is used with single-selection and multiple-selection widgets:

1. Single-selection containers where the currently focused item is not selected. The selection normally follows the focus, and is managed by the [user agent](#).
2. Multiple-selection containers. Authors **SHOULD** ensure that any selectable descendant of a container in which the [aria-multiselectable](#) attribute is `true` specifies a value of either `true` or `false` for the [aria-selected](#) attribute.

Any explicit assignment of [aria-selected](#) takes precedence over the implicit selection based on focus. If no [DOM](#) element in the widget is explicitly marked as selected, assistive technologies **MAY** convey implicit selection which follows the keyboard focus of the [managed focus](#) widget. If any [DOM](#) element in the widget is explicitly marked as selected, the user agent **MUST NOT** convey implicit selection for the widget.

Characteristics:

Characteristic	Value
Used in Roles:	<u>gridcell</u>

Characteristic	Value
	option
	row
	tab
Inherits into Roles:	columnheader rowheader treeitem
Value:	true/false/undefined

Values:

Value	Description
false	The selectable element is not selected.
true	The selectable element is selected.
undefined (default)	The element is not selectable.

aria-setsize property

Defines the number of items in the current set of listitems or treeitems. Not required if all elements in the set are present in the DOM. See related [aria-posinset](#).

This [property](#) is marked on the members of a set, not the container element that collects the members of the set. To orient the user by saying an element is "item X out of Y," the [assistive technologies](#) would use X equal to the [aria-posinset](#) attribute and Y equal to the [aria-setsize](#) attribute.

If all items in a set are present in the document structure, it is not necessary to set this property, as the [user agent](#) can automatically calculate the set size and position for each item. However, if only a portion of the set is present in the document structure at a given moment (in order to reduce document size), this property is needed to provide an explicit indication of set size.

Authors **MUST** set the value of [aria-setsize](#) to an integer equal to the number of items in the set. If the total number of items is unknown, authors **SHOULD** set the value of [aria-setsize](#) to -1.

When exposing [aria-setsize](#) on a [menuitem](#), [menuitemcheckbox](#), or [menuitemradio](#), authors **SHOULD** set the value of [aria-setsize](#) based on the total number of items in the [menu](#), excluding any separators.

The following example shows items 5 through 8 in a set of 16.

EXAMPLE 32

```
<h2 id="label_fruit"> Available Fruit </h2>
<ul role="listbox" aria-labelledby="label_fruit">
  <li role="option" aria-setsize="16" aria-posinset="5"> apples </li>
  <li role="option" aria-setsize="16" aria-posinset="6"> bananas </li>
  <li role="option" aria-setsize="16" aria-posinset="7"> cantaloupes </li>
  <li role="option" aria-setsize="16" aria-posinset="8"> dates </li>
</ul>
```

The following example shows items 5 through 8 in a set whose total size is unknown.

EXAMPLE 33

```
<h2 id="label_fruit"> Available Fruit </h2>
<ul role="listbox" aria-labelledby="label_fruit">
  <li role="option" aria-setsize="-1" aria-posinset="5"> apples </li>
  <li role="option" aria-setsize="-1" aria-posinset="6"> bananas </li>
  <li role="option" aria-setsize="-1" aria-posinset="7"> cantaloupes </li>
  <li role="option" aria-setsize="-1" aria-posinset="8"> dates </li>
</ul>
```

Characteristics:

Characteristic	Value
Used in Roles:	article listitem menuitem option radio row tab
Inherits into Roles:	menuitemcheckbox menuitemradio treeitem
Value:	integer

aria-sort property

Indicates if items in a table or grid are sorted in ascending or descending order.

Authors **SHOULD** only apply this [property](#) to table headers or grid headers. If the property is not provided, there is no defined sort order. For each table or grid, authors **SHOULD** apply [aria-sort](#) to only one header at a time.

Characteristics:

Characteristic	Value
Used in Roles:	columnheader rowheader
Value:	token

Values:

Value	Description
ascending	Items are sorted in ascending order by this column.
descending	Items are sorted in descending order by this column.
none (default)	There is no defined sort applied to the column.
other	A sort algorithm other than ascending or descending has been applied.

aria-valuemax property

Defines the maximum allowed value for a range [widget](#).

Authors **MUST** ensure the value of [aria-valuemax](#) is greater than or equal to the value of [aria-valuenow](#). If the [aria-valuenow](#) has a known maximum and minimum, the author **SHOULD** provide properties for [aria-valuemax](#) and [aria-valuenow](#).

NOTE

A range widget starts with a given value, which can be increased until reaching the maximum value, defined by this [property](#). Declaring the minimum and maximum values allows assistive technology to convey the size of the range to users.

Characteristics:

Characteristic	Value
Related Concepts:	<range> element max attribute in [HTML]

Characteristic	Value
Used in Roles:	range scrollbar separator slider spinbutton
Inherits into Roles:	meter progressbar scrollbar slider spinbutton
Value:	number

aria-valuemin property

Defines the minimum allowed value for a range [widget](#).

Authors **MUST** ensure the value of [aria-valuemin](#) is less than or equal to the value of [aria-valuemax](#). If the [aria-valuenow](#) has a known maximum and minimum, the author **SHOULD** provide properties for [aria-valuemax](#) and [aria-valuemin](#).

NOTE

A range widget starts with a given value, which can be decreased until reaching the minimum value, defined by this [property](#). Declaring the minimum and maximum values allows assistive technology to convey the size of the range to users.

Characteristics:

Characteristic	Value
Related Concepts:	<range> element min attribute in [HTML]
Used in Roles:	range scrollbar separator slider

Characteristic	Value
	spinbutton
Inherits into Roles:	meter progressbar scrollbar slider spinbutton
Value:	number

aria-valuenow property

Defines the current value for a range [widget](#). See related [aria-valuetext](#).

This property is used, for example, on a range widget such as a slider or progress bar.

If the current value is not known (for example, an indeterminate progress bar), the author **SHOULD NOT** set the [aria-valuenow](#) [attribute](#). If the [aria-valuenow](#) attribute is absent, no information is implied about the current value. If the [aria-valuenow](#) has a known maximum and minimum, the author **SHOULD** provide properties for [aria-valuemax](#) and [aria-valuemin](#).

The value of [aria-valuenow](#) is a decimal number. If the range is a set of numeric values, then [aria-valuenow](#) is one of those values. For example, if the range is [0, 1], a valid [aria-valuenow](#) is 0.5. A value outside the range, such as -2.5 or 1.1, is invalid.

For [progressbar](#) elements and [scrollbar](#) elements, assistive technologies **SHOULD** render the value to users as a percent, calculated as a position on the range from [aria-valuemin](#) to [aria-valuemax](#) if both are defined, otherwise the actual value with a percent indicator. For elements with role [slider](#) and [spinbutton](#), assistive technologies **SHOULD** render the actual value to users.

When the rendered value cannot be accurately represented as a number, authors **SHOULD** use the [aria-valuetext](#) attribute in conjunction with [aria-valuenow](#) to provide a user-friendly representation of the range's current value. For example, a slider may have rendered values of [small](#), [medium](#), and [large](#). In this case, the values of [aria-valuetext](#) would be one of the strings: [small](#), [medium](#), or [large](#).

NOTE

If [aria-valuetext](#) is specified, assistive technologies render that instead of the value of [aria-valuenow](#).

Characteristics:

Characteristic	Value
Related Concepts:	<range> element value attribute in [HTML]
Used in Roles:	meter range scrollbar separator slider spinbutton
Inherits into Roles:	meter progressbar scrollbar slider spinbutton
Value:	number

aria-valuetext property

Defines the human readable text alternative of [aria-valuenow](#) for a range [widget](#).

This property is used, for example, on a range widget such as a slider or progress bar.

If the [aria-valuetext](#) attribute is set, authors **SHOULD** also set the [aria-valuenow](#) attribute, unless that value is unknown (for example, on an indeterminate [progressbar](#)).

Authors **SHOULD** only set the [aria-valuetext](#) attribute when the rendered value cannot be meaningfully represented as a number. For example, a slider may have rendered values of `small`, `medium`, and `large`. In this case, the values of [aria-valuenow](#) could range from 1 through 3, which indicate the position of each value in the value space, but the [aria-valuetext](#) would be one of the strings: `small`, `medium`, or `large`. If the [aria-valuetext](#) attribute is absent, the [assistive technologies](#) will rely solely on the [aria-valuenow](#) attribute for the current value.

If [aria-valuetext](#) is specified, assistive technologies **SHOULD** render that value instead of the value of [aria-valuenow](#).

Characteristics:

Characteristic	Value
Used in Roles:	range separator spinbutton
Inherits into Roles:	meter progressbar scrollbar slider spinbutton
Value:	string

§ 7. Accessibility Tree

The [accessibility tree](#) and the [DOM tree](#) are parallel structures. The [accessibility tree](#) includes the user interface objects of the [user agent](#) and the objects of the document. [Accessible objects](#) are created in the accessibility tree for every [DOM element](#) that should be exposed to an [assistive technology](#), either because it may fire an accessibility [event](#) or because it has a [property](#), [relationship](#) or feature which needs to be exposed.

§ 7.1 Excluding Elements from the Accessibility Tree

The following [elements](#) are not exposed via the [accessibility API](#) and user agents **MUST NOT** include them in the [accessibility tree](#):

- Elements, including their descendent elements, that have host language semantics specifying that the element is not displayed, such as [CSS display:none](#), [visibility:hidden](#), or the [HTML hidden](#) attribute.
- Elements with [none](#) or [presentation](#) as the first role in the role attribute. However, their exclusion is conditional. In addition, the element's descendants and text content are generally included. These exceptions and conditions are documented in the [presentation \(role\)](#) section.

If not already excluded from the accessibility tree per the above rules, user agents **SHOULD NOT** include the following elements in the accessibility tree:

- Elements, including their descendants, that have `aria-hidden` set to `true`. In other words, `aria-hidden="true"` on a parent overrides `aria-hidden="false"` on descendants.
- Any descendants of elements that have the characteristic "[Children Presentational: True](#)" unless the descendant is not allowed to be presentational because it meets one of the conditions for exception described in [Presentational Roles Conflict Resolution](#). However, the text content of any excluded descendants is included.

Elements with the following roles have the characteristic "Children Presentational: True":

- [button](#)
- [checkbox](#)
- [img](#)
- [menuitemcheckbox](#)
- [menuitemradio](#)
- [meter](#)
- [option](#)
- [progressbar](#)
- [radio](#)
- [scrollbar](#)
- [separator](#)
- [slider](#)
- [switch](#)
- [tab](#)

§ 7.2 Including Elements in the Accessibility Tree

If not excluded from or marked as hidden in the accessibility tree per the rules above in [Excluding Elements in the Accessibility Tree](#), user agents **MUST** provide an [accessible object](#) in the [accessibility tree](#) for [DOM elements](#) that meet any of the following criteria:

- Elements that are not [hidden](#) and may fire an [accessibility API event](#), including:
 - Elements that are currently focused, even if the element or one of its ancestor elements has its `aria-hidden` attribute set to `true`.

- Elements that are a valid target of an [aria-activedescendant](#) attribute.
- Elements that have an explicit role or a global WAI-ARIA attribute and do not have [aria-hidden](#) set to true. (See [Excluding Elements in the Accessibility Tree](#) for additional guidance on [aria-hidden](#).)
- Elements that are not [hidden](#) and have an ID that is referenced by another element via a WAI-ARIA property.

NOTE

Text equivalents for hidden referenced objects may still be used in the [name and description computation](#) even when not included in the accessibility tree.

§ 8. Implementation in Host Languages

The [roles](#), [state](#), and [properties](#) defined in this specification do not form a complete web language or format. They are intended to be used in the context of a host language. This section discusses how host languages are to implement WAI-ARIA, to ensure that the markup specified here will integrate smoothly and effectively with the host language markup.

Although markup languages look alike superficially, they do not share language definition infrastructure. To accommodate differences in language-building approaches, the requirements are both general and modularization-specific. While allowing for differences in how the specifications are written, the intent is to maintain consistency in how the WAI-ARIA information looks to authors and how it is manipulated in the DOM by scripts.

WAI-ARIA roles, states, and properties are implemented as [attributes](#) of [elements](#). Roles are applied by placing their names among the tokens appearing in the value of a host-language-provided [role](#) attribute. States and properties each get their own attribute, with values as defined for each particular state or property in this specification. The name of the attribute is the aria-prefixed name of the state or property.

§ 8.1 Role Attribute

An implementing host language will provide an [attribute](#) with the following characteristics:

- The attribute name **MUST** be [role](#);
- The attribute value **MUST** allow a token list as the value;
- The appearance of the name literal of any concrete WAI-ARIA [role](#) as one of these tokens **MUST NOT**

in and of itself make the attribute value illegal in the host-language syntax; and

- The first name literal of a non-abstract WAI-ARIA role in the list of tokens in the role attribute defines the role according to which the user agent **MUST** process the element. User Agent processing for roles is defined in the [Core Accessibility API Mappings](#) [CORE-AAM-1.2].

8.2 State and Property Attributes

An implementing host language **MUST** allow [attributes](#) with the following characteristics:

- The attribute name is the name of any state or property identified in the [Supported States and Properties](#) section, such as [aria-busy](#), [aria-selected](#), [aria-activedescendant](#), [aria-valuetext](#);
- The syntax does **NOT** prevent the attribute from appearing anywhere that it is applicable, as specified in this specification;
- When these attributes appear in a document instance, the attributes will be processed as defined in this specification.

Host languages that support [XML Namespaces](#) [XML-NAMES] **MAY** require that WAI-ARIA attributes be used with a namespace. In this case, the namespace for WAI-ARIA state and property attributes **MUST** be <http://www.w3.org/ns/wai-aria/>. To use WAI-ARIA in host languages that do not explicitly describe support for it, authors **SHOULD** use this namespace as well, if the host language supports namespaces and there is expectation that user agents will recognize the WAI-ARIA namespace. The namespace prefix is not defined by this specification but generally is expected to be "aria".

NOTE

The WAI-ARIA state and property attributes have a naming convention such that they all begin with the string "aria-". This is *not* a namespace prefix, it is a part of the state or property name. Therefore, when using WAI-ARIA states and properties with namespace prefixes, the complete attribute name will be like "aria:aria-foo".

Some host languages do not use namespaces with WAI-ARIA state and property attributes, either because the host language does not support namespaces or because the designers wish to incorporate WAI-ARIA into the core feature set. In these host languages, the namespace name for these attributes has no value. The names of these attributes do not have a prefix offset by a colon; in the terms of namespaces they are unprefixed attribute names. The ECMAScript binding of the DOM interface `getAttributeNS` for example, treats an empty string ("") as representing this condition, so that both `getAttribute("aria-busy")` and `getAttributeNS("", "aria-busy")` access the same [aria-busy](#) attribute in the DOM.

NOTE

According to the requirements of this section, some user agents recognize WAI-ARIA state and property attributes *with* namespaces, some *without* namespaces, and some might recognize both. Authors are advised to be aware of which form is supported for the host language they are using. Unless the host language and supporting user agents explicitly indicate that the namespace is required, authors are advised to use the attribute without namespaces. Even user agents that support namespaces generally do not publish namespaced WAI-ARIA states and properties to accessibility APIs. In particular, current implementations of HTML, including XHTML, do not support this namespace.

8.3 Focus Navigation

An implementing host language **MUST** provide support for the author to make all interactive elements focusable, that is, any renderable or event-receiving elements. An implementing host language **MUST** provide a facility to allow web authors to define whether these focusable, interactive elements appear in the default tab navigation order. The `tabindex` attribute in HTML is an example of one implementation.

8.4 Implicit WAI-ARIA Semantics

WAI-ARIA is designed to provide semantic information about objects when host languages lack native semantics for the object. WAI-ARIA is designed, however, to provide additional semantics for many host languages. Furthermore, host languages over time can evolve and provide new native features that correspond to WAI-ARIA features. Therefore, there are many situations in which WAI-ARIA semantics are redundant with host language semantics.

These host language features can be viewed as having "implicit WAI-ARIA semantics". User agent processing of features with implicit WAI-ARIA semantics would be similar to the processing for the WAI-ARIA feature. The processing might not be identical because of lexical differences between the host language feature and the WAI-ARIA feature, but generally the user agent would expose the same information to the accessibility API. Features with implicit WAI-ARIA semantics satisfy WAI-ARIA structural requirements such as required owned elements, required states and properties, etc. and do not require explicit WAI-ARIA semantics to be provided. On elements with implicit WAI-ARIA roles, authors can also use WAI-ARIA states and properties supported by those roles *without* requiring explicit indication of the WAI-ARIA role.

For example, if an element with the functionality already exists, such as a checkbox or radio button, use the native semantics of the host language. WAI-ARIA markup is only intended to be used to enhance the native

semantics (e.g., indicating that the element is required with [aria-required](#)), or to change the semantics to a different purpose from the standard functionality of the element.

Implicit WAI-ARIA semantics affect the conflict resolution procedures in the following section, Conflicts with Host Language Semantics. Therefore, implicit WAI-ARIA semantics need to be defined in a normative specification, such as the host language specification or the [Core Accessibility API Mappings](#).

8.5 Conflicts with Host Language Semantics

WAI-ARIA roles, states, and properties are intended to add [semantic](#) information when native host language elements with these semantics are not available, and are generally used on elements that have no native semantics of their own. They can also be used on elements that have similar but non-identical semantics (for example, a nested list could be used to represent a tree structure). This method can be part of a fallback strategy for older browsers that have no WAI-ARIA implementation, or because native presentation of the repurposed element reduces the amount of style and/or script needed. Except for the cases outlined below, user agents **MUST** always use the WAI-ARIA semantics to define how it exposes the element to accessibility APIs, rather than using the host language semantics.

In addition to these normal situations in which WAI-ARIA is expected to override native semantics, there are elements that are inappropriate to override with WAI-ARIA. This could be because identical host language semantics exist, so WAI-ARIA is not needed, or because semantics from WAI-ARIA directly conflict with host language semantics. When a feature in the host language with identical role semantics and values is available, and the author has no compelling reason to avoid using the host language feature, authors **SHOULD** use the host language features rather than repurpose other elements with WAI-ARIA.

Host languages can have features that have implicit WAI-ARIA semantics corresponding to roles. When a WAI-ARIA role is provided, user agents **MUST** use the semantic of the WAI-ARIA role for processing, not the native semantic, unless the role requires WAI-ARIA states and properties whose attributes are explicitly forbidden on the native element by the host language. Values for roles do not conflict in the same way as values for states and properties (for example, the HTML 'checked' attribute and the 'aria-checked' attribute could have conflicting values), and authors are expected to have valid reason to provide a WAI-ARIA role even on elements that would not normally be repurposed.

When WAI-ARIA states and properties correspond to host language features that have the same [implicit WAI-ARIA semantic](#), it can be particularly problematic to use the WAI-ARIA feature. If the WAI-ARIA feature and the host language feature are both provided but their values are not kept in sync, user agents and assistive technologies cannot know which value to use. Therefore, to prevent providing conflicting states and properties to assistive technologies, host languages **MUST** explicitly declare where the use of WAI-ARIA attributes on each host language element conflicts with native attributes for that element. When a host language declares a WAI-ARIA attribute to be in direct semantic conflict with a native attribute for a given

element, user agents **MUST** ignore the WAI-ARIA attribute and instead use the host language attribute with the same implicit semantic.

Host languages **MAY** document features that cannot be overridden with WAI-ARIA (these are called "strong native semantics"). These can be features that have implicit WAI-ARIA semantics, as well as features where the processing would be uncertain if the semantics were changed with WAI-ARIA. Conformance checkers **MAY** signal an error or warning when a WAI-ARIA role is used on elements with strong native semantics, but as described above, user agents **MUST** still use the value of the semantic of the WAI-ARIA role when exposing the element to accessibility APIs unless the native host language semantic is permanently presentational.

The opportunity for host languages to create exceptions to the WAI-ARIA override of native features is meant to avoid potential author errors or problems with intrinsic processing of host language features. Author errors could happen when a host language and WAI-ARIA provide similar but not identical features, where it might not be clear how changing one but not the other affects the accessibility API. Intrinsic processing refers to the way a feature is processed, beyond simple rendering and exposure to the Accessibility API, that cannot reasonably be changed in response to an ARIA feature, and would lead to unpredictable results were ARIA allowed. In these situations, there is good reason for host languages to limit the scope of WAI-ARIA. However, this provision does not give blanket permission for host languages to forbid the use of WAI-ARIA simply by documenting, feature by feature, that it may not be used. Host languages should create restrictions on the use of ARIA only when it is critical to effective processing of content.

Certain ARIA features are critical to building a complete model in the accessibility API. Such features are not expected to conflict with native host language semantics (though they may complement them). Therefore, host languages **MUST NOT** declare strong native semantics that prevent use of the following ARIA features:

- aria-describedby
- aria-label
- aria-labelledby

8.6 State and Property Attribute Processing

State and property attributes are included in host languages, and therefore syntax for representation of their value types is governed by the host language. For each of the value types defined in Value, an appropriate value type from the host language is used. Recommended correspondences between WAI-ARIA value types and various host language value types are listed in Mapping WAI-ARIA Value types to languages. This is a non-normative mapping in order to accommodate new host languages supporting WAI-ARIA.

The list value types—ID reference list and token list—allow more than one value of the given type to be provided. The values are separated by delimiter characters recognized by the host language for list attributes,

such as space characters, commas, etc. Some languages may require a specific, single delimiter, while others may allow various delimiters.

Global states and properties are supported on any element in the host language. However, authors **MUST** only use non-global states and properties on elements with a role supporting the state or property; either defined as an explicit WAI-ARIA role, or as defined by the host language implicit WAI-ARIA semantic matching an appropriate WAI-ARIA role. When a role attribute is added to an element, the [semantics](#) and behavior of the element, including support for WAI-ARIA states and properties, are augmented or overridden by the role behavior. User agents **MUST** ignore non-global states and properties used on an element without a role supporting the state or property; either defined as an explicit WAI-ARIA role, or as defined by the host language WAI-ARIA semantic matching an appropriate WAI-ARIA role. For example, the [aria-valuetext](#) attribute may be used on a [progressbar](#).

WAI-ARIA roles have associated states and properties that are qualified as "supported" or "required". An example of a property *supported* by the [combobox](#) role is [aria-autocomplete](#). The property is designated "supported" in this case because a given [combobox](#) might or might not implement auto completion. In contrast, the [combobox](#) role *requires* the [aria-expanded](#) state in order to indicate that it is expandable. Comboboxes have a controlled popup element, such as a [listbox](#), that is either open or closed. If the [listbox](#) is open, the [combobox](#) is in its expanded state; otherwise it is collapsed.

When WAI-ARIA roles are used, *supported* states and properties that are not present in the [DOM](#) are treated according to their default value. Keeping with the [combobox](#) example, a missing [aria-autocomplete](#) attribute is equivalent to [aria-autocomplete="none"](#), meaning the [combobox](#) does not offer auto completion.

However, *required* states and properties that are absent are an author error. Missing required states and properties are treated as if they were present and have an implicit neutral value that is not necessarily their default value. For example, the default value of [aria-expanded](#) is [undefined](#), meaning neither expandable nor collapsible. But that does not apply to the case of a [combobox](#). In this case, [aria-expanded](#) is needed to convey the expandable/collapsible nature of the [combobox](#). Thus, the implicit value of [aria-expanded](#) for the [combobox](#) role is [false](#), meaning expandable (and currently collapsed). The characteristics table associated with each WAI-ARIA role has an "[Implicit Value for Role](#)" entry that specifies the value of a state or property to use in the context of that role when the state or property is missing.

Elements that have implicit WAI-ARIA semantics support the full set of WAI-ARIA states and properties supported by the corresponding role. Therefore, authors **MAY** omit the role when setting states and properties. The role is only needed when the implicit WAI-ARIA role of the element needs to be changed.

Sometimes states and properties are present in the [DOM](#) but have a zero-length string ("") as their value. Authors **MAY** specify a zero-length string ("") for any supported (but not required) state or property. User agents **SHOULD** treat state and property attributes with a value of "" the same as they treat an absent attribute. For supported states and properties, this corresponds to the default value, but if it is a required

attribute, it signals an author error, and the implicit value for the role is used.

8.6.1 ID Reference Error Processing

User agents **SHOULD** ignore ID references that do not match the ID of another [element](#) in the same document.

It is the web author's responsibility to ensure that IDs are unique. If more than one element has the same ID, the user agent **SHOULD** use the first element found with the given ID. The behavior will be the same as `getElementById`.

If the same element is specified multiple times in a single WAI-ARIA relation, user agents **SHOULD** return multiple pointers to the same [element](#).

[**aria-activedescendant**](#) is defined as referencing only a single ID reference. Any [**aria-activedescendant**](#) value that does not match an existing ID reference exactly is an author error and will not match any element in the [DOM](#).

8.7 CSS Selectors

NOTE

This section might be removed in a future version.

Support for [**attribute**](#) selectors **MUST** include WAI-ARIA attributes. For example, `.fooMenuItem[aria-haspopup="true"]` would select all [**elements**](#) with class `fooMenuItem`, and WAI-ARIA property [**aria-haspopup**](#) with value of `true`. The presentation **MUST** be updated for dynamic changes to WAI-ARIA attributes. This allows authors to match styling with WAI-ARIA [**semantics**](#).

9. Handling Author Errors

9.1 Roles

User agents are expected to perform validation of WAI-ARIA [**roles**](#).

As stated in the [Definition of Roles](#) section, it is considered an authoring error to use [abstract roles](#) in content. User agents **MUST NOT** map abstract roles via the standard role mechanism of the accessibility API.

If the `role` attribute contains no tokens matching the name of a non-abstract WAI-ARIA role, the user agent **MUST** treat the element as if no `role` had been provided. For example, `<table role="foo">` should be exposed in the same way as `<table>` and `<input type="text" role="structure">` in the same way as `<input type="text">`.

§ 9.2 States and Properties

In general, [user agents](#) do not do much validation of WAI-ARIA [properties](#). User agents **MAY** do some minor validation on request, such as making sure [valid IDs](#) are specified for WAI-ARIA relations, and enforcing things like [aria-posinset](#) being within 1 and [aria-setsize](#), inclusive. User agents are not responsible for logical validation, such as the following:

1. Circular references created by relations, such as specifying that two [elements](#) own each other.
2. Correct usage with regard to [DOM](#) tree structure, such as an [element](#) being owned by more than one other element.
3. Elements with [WAI-ARIA roles](#) correctly implement the behavior of the specified role. For example, user agents do not verify that an element with a role of [checkbox](#) actually behaves like a checkbox.
4. Elements that do not correctly observe required child / parent role relationships or that appear elsewhere than in their required parent.
5. Determining whether [aria-activedescendant](#) actually points to an [owned](#) element of the container widget.
6. Determining implicit values of [aria-setsize](#) and [aria-posinset](#) when they are specified on some but not all the elements of the set.

If the author specifies a non-numeric value for a decimal or integer value type, the user agent **SHOULD** do the following:

- When asked for the string version of the property, return the string if specified by the author.
- When asked for the numeric version:
 - Follow the guidance in the [Fallback values for missing required attributes](#) table below, if applicable.
 - Otherwise, return a fallback value of 0.0 for decimal value types and 0 for integer value types.

If a WAI-ARIA property contains an unknown or disallowed value, the user agent **SHOULD** expose to platform [accessibility APIs](#) as follows:

- When exposing as a platform accessibility API attribute, expose the unknown value — do not yet it against possible values.
- When exposing as a platform API Boolean state:
 - For values of "" (empty string), "undefined" or no *attribute* present:
 - Follow the guidance in the [Fallback values for missing required attributes](#) table below, if applicable.
 - Otherwise, treat as false.
 - Treat any other value as true.
- Otherwise, ignore the value and treat the property as not present.

NOTE

In UIA, the user agent might leave the corresponding property set to "unsupported."

User agents **MUST NOT** expose WAI-ARIA attributes that reference unresolved IDs. For example:

- When the state or property has only one ID reference that cannot be resolved, treat as if the state or property is not present.
- When the state or property has a list of ID references, ignore any that can't be resolved. If none in the list can be resolved, treat as if the state or property is not present.

User Agents **MUST NOT** expose [aria-roledescription](#) when:

- The element it is applied to has an invalid WAI-ARIA role, or
- The element does not have an implicit WAI-ARIA role

If a required WAI-ARIA attribute for a given role is missing, user agents **SHOULD** process the attribute as if the values given in the following table were provided.

Fallback values for missing required attributes

WAI-ARIA role	Required Attribute	Fallback value
checkbox	aria-checked	false
combobox	aria-controls	no mapping
combobox	aria-expanded	false

WAI-ARIA role	Required Attribute	Fallback value
<u>heading</u>	<u>aria-level</u>	2
<u>menuitemcheckbox</u>	<u>aria-checked</u>	false
<u>menuitemradio</u>	<u>aria-checked</u>	false
<u>radio</u>	<u>aria-checked</u>	false
<u>scrollbar</u>	<u>aria-controls</u>	no mapping
<u>scrollbar</u>	<u>aria-valuenow</u>	If missing or not a <u>number</u> , (<u>aria-valuemax</u> - <u>aria-valuemin</u>) / 2. If present but less than <u>aria-valuemin</u> , the value of <u>aria-valuemin</u> . If present but greater than <u>aria-valuemax</u> , the value of <u>aria-valuemax</u> .
<u>separator</u> (if focusable)	<u>aria-valuenow</u>	If missing or not a <u>number</u> , (<u>aria-valuemax</u> - <u>aria-valuemin</u>) / 2. If present but less than <u>aria-valuemin</u> , the value of <u>aria-valuemin</u> . If present but greater than <u>aria-valuemax</u> , the value of <u>aria-valuemax</u> .
<u>slider</u>	<u>aria-valuenow</u>	If missing or not a <u>number</u> , (<u>aria-valuemax</u> - <u>aria-valuemin</u>) / 2. If present but less than <u>aria-valuemin</u> , the value of <u>aria-valuemin</u> . If present but greater than <u>aria-valuemax</u> , the value of <u>aria-valuemax</u> .
<u>switch</u>	<u>aria-checked</u>	false
<u>meter</u>	<u>aria-valuenow</u>	A value matching the implicit or explicitly set <u>aria-valuemin</u> .

NOTE

Implicit Values for non-required states and properties appear in the characteristics table for each role. These are not considered fallback values so are not included here.

§ 10. IDL Interface

Conforming user agents **MUST** implement the following IDL interface.

§ 10.1 Interface Mixin *ARIAMixin*

WebIDL

```
interface mixin ARIAMixin {
    attribute DOMString? role;

    attribute DOMString? ariaAtomic;
    attribute DOMString? ariaAutoComplete;
    attribute DOMString? ariaBusy;
    attribute DOMString? ariaChecked;
    attribute DOMString? ariaColCount;
    attribute DOMString? ariaColIndex;

    attribute DOMString? ariaColSpan;

    attribute DOMString? ariaCurrent;

    attribute DOMString? ariaDisabled;

    attribute DOMString? ariaExpanded;

    attribute DOMString? ariaHasPopup;
    attribute DOMString? ariaHidden;
    attribute DOMString? ariaInvalid;
    attribute DOMString? ariaKeyShortcuts;
    attribute DOMString? ariaLabel;

    attribute DOMString? ariaLevel;
    attribute DOMString? ariaLive;
    attribute DOMString? ariaModal;
    attribute DOMString? ariaMultiLine;
    attribute DOMString? ariaMultiSelectable;
    attribute DOMString? ariaOrientation;

    attribute DOMString? ariaPlaceholder;
    attribute DOMString? ariaPosInSet;
    attribute DOMString? ariaPressed;
    attribute DOMString? ariaReadOnly;

    attribute DOMString? ariaRequired;
    attribute DOMString? ariaRoleDescription;
    attribute DOMString? ariaRowCount;
    attribute DOMString? ariaRowIndex;

    attribute DOMString? ariaRowSpan;
    attribute DOMString? ariaSelected;
    attribute DOMString? ariaSetSize;
    attribute DOMString? ariaSort;
    attribute DOMString? ariaValueMax;
    attribute DOMString? ariaValueMin;
    attribute DOMString? ariaValueNow;
```

```
    attribute DOMString? ariaValueText;
};
```

Interfaces that include `ARIAMixin` must provide the following algorithms:

- **`ARIAMixin` getter steps**, which take the host interface instance, IDL attribute name, and content attribute name, and must return a string value; and
- **`ARIAMixin` setter steps**, which take the host interface instance, IDL attribute name, content attribute name, and string value, and must return nothing.

For every IDL attribute `idlAttribute` defined in `ARIAMixin`, on getting, it must perform the following steps:

1. Let `contentAttribute` be the ARIA content attribute determined by looking up `idlAttribute` in the ARIA Attribute Correspondence table.
2. Return the result of running the **`ARIAMixin` getter steps**, given this, `idlAttribute`, and `contentAttribute`.

Similarly, on setting, it must perform the following steps:

1. Let `contentAttribute` be the ARIA content attribute determined by looking up `idlAttribute` in the ARIA Attribute Correspondence table.
2. Run the **`ARIAMixin` setter steps**, given this, `idlAttribute`, `contentAttribute`, and the given value.

NOTE

This very general framework is motivated by the desire for different host interfaces, such as `Element` and `ElementInternals`, to give these IDL attributes different behaviors. The alternative is requiring each host interface to duplicate the IDL attributes independently, so that they can specify independent behaviors, but that comes with a high risk of them getting out of sync.

§ 10.2 ARIA Attribute Correspondence

The following table provides a correspondence between IDL attribute names and content attribute names, for use by `ARIAMixin`.

IDL Attribute	Reflected ARIA Content Attribute
<code>role</code>	<code>role</code>
<code>ariaAtomic</code>	<code>aria-atomic</code>
<code>ariaAutoComplete</code>	<code>aria-autocomplete</code>

<i>ariaBusy</i>	aria-busy
<i>ariaChecked</i>	aria-checked
<i>ariaColCount</i>	aria-colcount
<i>ariaColIndex</i>	aria-colindex
<i>ariaColSpan</i>	aria-colspan
<i>ariaCurrent</i>	aria-current
<i>ariaDisabled</i>	aria-disabled
<i>ariaExpanded</i>	aria-expanded
<i>ariaHasPopup</i>	aria-haspopup
<i>ariaHidden</i>	aria-hidden
<i>ariaInvalid</i>	aria-invalid
<i>ariaKeyShortcuts</i>	aria-keyshortcuts
<i>ariaLabel</i>	aria-label
<i>ariaLevel</i>	aria-level
<i>ariaLive</i>	aria-live
<i>ariaModal</i>	aria-modal
<i>ariaMultiLine</i>	aria-multiline
<i>ariaMultiSelectable</i>	aria-multiselectable
<i>ariaOrientation</i>	aria-orientation
<i>ariaPlaceholder</i>	aria-placeholder
<i>ariaPosInSet</i>	aria-posinset
<i>ariaPressed</i>	aria-pressed
<i>ariaReadOnly</i>	aria-readonly
<i>ariaRequired</i>	aria-required
<i>ariaRoleDescription</i>	aria-roledescription
<i>ariaRowCount</i>	aria-rowcount
<i>ariaRowIndex</i>	aria-rowindex
<i>ariaRowSpan</i>	aria-rowspan
<i>ariaSelected</i>	aria-selected
<i>ariaSetSize</i>	aria-setsize
<i>ariaSort</i>	aria-sort
<i>ariaValueMax</i>	aria-valuemax
<i>ariaValueMin</i>	aria-valuemin
<i>ariaValueNow</i>	aria-valuenow

ariaValueText**aria-valuetext****NOTE**

Note: Attributes [aria-dropeffect](#) and [aria-grabbed](#) were deprecated in ARIA 1.1 and do not have corresponding IDL attributes.

§ 10.2.1 Disambiguation Pattern

This section is non-normative.

Though specification authors may make exceptions to this pattern, the following rules were used to disambiguate names and case of the IDL attributes listed above.

- Any attribute name referencing concepts that are combinations of two or more words (such as "described by") becomes a camel-cased IDL attribute capitalizing each word boundary. For example, [aria-describedby](#) becomes `ariaDescribedBy` with both the D and B capitalized.
- Likewise, any attribute name referencing concepts that can be hyphenated (such as "multi-selectable") becomes a camel-cased IDL attribute capitalizing each hyphenation boundary. For example, the only valid spelling for "multi-selectable" is hyphenated, so [aria-multiselectable](#) becomes `ariaMultiSelectable` with both the M and S capitalized.
- When trusted dictionary sources list both hyphenated or non-hyphenated spellings (e.g. "multi-line" and "multiline" are both valid spellings) use the hyphenated version and apply the hyphenation rule above. For example, [aria-multiline](#) becomes `ariaMultiLine` with both the M and L capitalized.
- If all trusted dictionary sources list a single spelling of a compound word with no spaces or hyphens, only the first letter of the term is capitalized. For example, neither "place-holder" nor "place holder" are considered valid spellings of the term "placeholder," so [aria-placeholder](#) becomes `ariaPlaceholder` with only the P capitalized.
- There are currently no acronym-based ARIA attributes, but if future attributes include acronym usage, attempt to match existing DOM conventions (e.g. ID becomes Id).

§ 10.2.2 IDL Attribute Name Notes or Exceptions

This section is non-normative.

Any notes or exceptions for specific attribute names will be listed here.

- **ariaPosInSet:** The [aria-posinset](#) attribute refers to an item's position in a set (two words: "in set") rather than the "inset" of an item from the beginning of the collection. Therefore the IDL attribute name is `ariaPosInSet` with the P, I, and second S capitalized, *not ariaPosInset*.

§ 10.3 ARIAMixin Mixed in to Element

User agents **MUST** include ARIAMixin on Element:

WebIDL

[Element](#) includes [ARIAMixin](#);

For Element:

- The [ARIAMixin getter steps](#) given `element`, `idlAttribute`, and `contentAttribute` are to return the result of the getter algorithm for `idlAttribute reflecting contentAttribute` on `element`.
- The [ARIAMixin setter steps](#) given `element`, `idlAttribute`, `contentAttribute`, and `value` are to perform the setter algorithm for `idlAttribute reflecting contentAttribute` on `element`, given `value`.

NOTE

In practice, this means that, e.g., the role IDL on Element reflects the role content attribute; the `ariaValueMin` IDL attribute reflects the `aria-valuemin` content attribute; etc.

§ 10.4 Example IDL Attribute Usage

This section is non-normative.

The primary purpose of ARIA IDL attribute reflection is to ease JavaScript-based manipulation of values. The following examples demonstrate its usage.

EXAMPLE 34

```
<div id="inaccessibleButton">
  <!-- Use semantic markup instead. This is just a retrofit example. -->
</div>

// Get a reference to the element.
let el = document.getElementById('inaccessibleButton');
el.tabIndex = 0; // Make it focusable.

// Set the role and label.
el.role = "button";
el.ariaLabel = "Edit";

// Get the role and label.
el.role; // Returns "button"
el.ariaLabel; // Returns "Edit"

// These are interchangeable with the more verbose setAttribute and getAttribute methods.
el.setAttribute("role", "button");
el.setAttribute("aria-label", "Edit");
el.getAttribute("role"); // Returns "button"
el.getAttribute("aria-label"); // Returns "Edit"

// Changes via either interface are reflected by the other.
el.setAttribute("aria-label", "Delete");
el.ariaLabel; // Returns "Delete"
el.ariaLabel = "Publish";
el.getAttribute("aria-label"); // Returns "Publish"
```

§ 11. Privacy and Security Considerations

This section is non-normative.

This specification introduces no new security considerations.

In accordance with [Web Platform Design Principles](#), this specification provides no programmatic interface to determine if information is being used by Assistive Technologies. However, this specification does allow an author to present different information to users of Assistive Technologies from the information available to users who do not use Assistive Technologies. This is possible using many features of the ARIA specification, just as this is possible using many other parts of the web technology stack. This content disparity could be abused to perform [active fingerprinting](#) of users of Assistive Technologies.

A. Mapping WAI-ARIA Value types to languages

This section is non-normative.

NOTE

The HTML column of the table below is advisory. Guidance on use of WAI-ARIA state and properties in HTML is provided in [Allowed ARIA roles, states and properties \(\[HTML-ARIA\]\)](#).

NOTE

The suggested mappings for true/false values in HTML use [Keyword and enumerated attributes](#) with allowed values of `true` and `false`, instead of using the HTML boolean value type.

The table below provides recommended mappings between WAI-ARIA state and property types and attribute types from [\[HTML\]](#), [XML Schema Datatypes](#) [XMLSHEMA11-2], [\[SVG2\]](#), and SGML.

Languages not listed below might have appropriate value types defined in the language. If they do not, we recommend XML Schema Datatypes for general purpose XML languages. Documents using DTDs instead of schemas will not be able to validate automatically and require additional processing on WAI-ARIA attributes.

<u>WAI-ARIA</u> type	<u>HTML</u>	<u>XML Schema</u>
true/false	Keyword and enumerated attributes with allowed values of "true" and "false"	<u>boolean</u>
true/false/ undefined	Keyword and enumerated attributes with allowed values of <code>true</code> , <code>false</code> , and <code>undefined</code>	<u>NMTOKEN</u> with an enumeration constraint allowing values of <code>true</code> , <code>false</code> , and <code>undefined</code>
tristate	Keyword and enumerated attributes with allowed values of "true", "false", and "mixed"	<u>NMTOKEN</u> with an enumeration constraint allowing values of "true", "false", and "mixed"
number	Floating-point numbers	<u>decimal</u>
integer	Non-negative integer	<u>integer</u>
token	Keyword and enumerated attributes	<u>NMTOKEN</u> with an enumeration constraint allowing values listed in the state or property definition
token list	Space-separated tokens	<u>NMTOKENS</u> with an enumeration constraint allowing values listed in the state or property definition

ID reference	The value of a defined id attribute on another element	IDREF
ID reference list	The value of one or more defined id attributes on other element(s), represented as Space-separated tokens	IDREFS
string	No value constraints	string

§ B. Substantive changes since the [WAI-ARIA 1.1 Recommendation](#)

- 16-Feb-2023: Resolved At-Risk items from CR
- 16-Feb-2023: Reverted [spinbutton](#): Change the default value of [aria-valuenow](#) from `0` to "there is no current value." Also add [aria-valuetext](#) as a supported property.
- 17-Sep-2021: Revised IDL and enumerated attribute section to reflect implementations
- 30-Aug-2021: removed ariaDescription from IDL section as was added erroneously
- 14-May-2021: Added Privacy and Security Considerations section
- 05-May-2021: clarify accessible name prohibited definition
- 10-Feb-2021: clarify including elements in accessibility tree to only require elements when actually focused
- 08-Sep-2020: remove [aria-level](#) from [tablist](#)
- 08-Sep-2020: Remove contents as a supported name source for [rowgroup](#).
- 08-Sep-2020: prohibit [aria-roledescription](#) on [generic](#)
- 08-Sep-2020: Require user agents to expose a value for [combobox](#) elements
- 08-Sep-2020: Remove multiple inheritance from [menuitemcheckbox](#) and [menuitemradio](#)
- 08-Sep-2020: Add missing implicit value for [progressbar](#)
- 27-Jul-2020: Update to define owned and container for
- 10-Jul-2020: Re-add [aria-haspopup](#) on links
- 15-May-2020: Remove nullable from IDL [DOMStrings](#), add enumerated attributes prose and examples, and remove ariaRelevant IDL until Issue #1267 can be resolved.
- 07-May-2020: Deprecate [aria-disabled](#), [aria-errormessage](#), [aria-haspopup](#) and [aria-invalid](#) as globals rather than removing them.
- 03-Apr-2020: Clarify default values

- 03-Apr-2020: Revise [meter](#) authoring advice
- 26-Mar-2020: remove recommendation to use `role="none presentation"`
- 26-Mar-2020: Add info about layout and bounds to [generic](#)
- 03-Mar-2020: Clean up of Presentational roles conflict resolution section
- 20-Feb-2020: Update [combobox](#) to remove aria-multiline reference
- 01-Nov-2019: Modify [combobox](#) to new ARIA 1.2 pattern.
- 25-Oct-2019: Modify [caption](#) authoring advice
- 25-Oct-2019: Change [aria-disabled](#), [aria-errormessage](#), [aria-haspopup](#) and [aria-invalid](#) from global to widget specific.
- 24-Oct-2019: Prohibits Labeling of [caption](#), [code](#), [deletion](#), [emphasis](#), [insertion](#), [paragraph](#), [presentation](#), [strong](#), [subscript](#), [superscript](#)
- 24-Oct-2019: Remove accessible name required from [log](#) and [timer](#)
- 24-Oct-2019: Allow group as child of [listbox](#)
- 24-Oct-2019: Add [code](#) role
- 24-Oct-2019: Add [time](#) role
- 24-Oct-2019: Add [subscript](#) and [superscript](#) roles
- 24-Oct-2019: Add [meter](#) role
- 23-Oct-2019: Resolve inconsistencies around group ownership of [menuitem](#), [menuitemcheckbox](#) and [menuitemradio](#).
- 23-Oct-2019: Add [generic](#) role
- 22-Oct-2019: Clarify use of [alertdialog](#) and [alert](#) roles
- 22-Oct-2019: Add [insertion](#) and [deletion](#) roles
- 18-Oct-2019: Remove references to taxonomy file
- 18-Oct-2019: Remove implicit value from [aria-checked](#) on [checkbox](#)
- 17-Oct-2019: Add [strong](#) and [emphasis](#) roles
- 11-Oct-2019: Deprecate [directory](#) role
- 11-Oct-2019: Make [form](#) role accessible name required true
- 11-Oct-2019: Remove allowance of [group](#) in [list](#)
- 04-Sep-2019: Add [aria-required](#) as a supported property of [checkbox](#)
- 04-Sep-2019: Allow [aria-posinset](#) and [aria-setsize](#) on [row](#) when used in a [treemap](#).

- 04-Sep-2019: Add [aria-expanded](#) support to [application](#) and [checkbox](#) roles.
- 04-Sep-2019: Remove [aria-expanded](#) support from the following roles: [alert](#), [alertdialog](#), [article](#), [banner](#), [blockquote](#), [caption](#), [cell](#), [complementary](#), [contentinfo](#), [definition](#), [deletion](#), [dialog](#), [directory](#), [feed](#), [figure](#), [form](#), [grid](#), [group](#), [heading](#), [img](#), [insertion](#), [landmark](#), [list](#), [listitem](#), [log](#), [main](#), [marquee](#), [math](#), [menu](#), [menubar](#), [navigation](#), [note](#), [paragraph](#), [radiogroup](#), [region](#), [search](#), [select](#), [status](#), [subscript](#), [superscript](#), [table](#), [tabpanel](#), [term](#), [time](#), [timer](#), [toolbar](#), [tooltip](#), [tree](#), [treegrid](#).
- 04-Sep-2019: Remove children-presentational=true from [math](#) role
- 22-Aug-2019: Remove [aria-level](#) from [grid](#)
- 23-Jul-2019: Add [generic](#) role
- 11-Jul-2019: Remove advice against changing roles
- 11-Jul-2019: Set Accessible Name Required to false on [gridcell](#)
- 04-Jun-2019: Make [aria-valuemin](#) and [aria-valuemax](#) supported, rather than required, properties of focusable [separator](#), [slider](#), and [scrollbar](#). Make [aria-orientation](#) a supported, rather than required, property of [scrollbar](#).
- 27-Mar-2019: Add Translatable States and Properties Section
- 31-Jan-2019: Change the superclass of [range](#) from widget to structure.
- 23-Jan-2019: Removed Default value of [aria-checked](#) from [menuitemcheckbox](#) and [menuitemradio](#) roles
- 09-Jan-2019: Removed Default value of [aria-checked](#) from [switch](#) and [checkbox](#) roles
- 05-Oct-2018: Role [spinbutton](#): Change the default value of [aria-valuenow](#) from `0` to "there is no current value." Also add [aria-valuetext](#) as a supported property.
- 05-Oct-2018: Role [spinbutton](#): allow empty values, no min, no max, and structure with sibling steppers
- 21-Aug-2018: Correct normative language in [rowheader](#) to be consistent with required states and properties.
- 21-Jun-2018: Allow [group](#) as child of [listbox](#).
- 31-May-2018: Add [blockquote](#), [caption](#), and [paragraph](#) roles.
- 01-Apr-2018: Added ARIA IDL Section (JavaScript interfaces).
- 06-Dec-2017: Make [aria-expanded](#) a supported state of [menuitem](#). This change also makes it a supported property of [menuitemcheckbox](#) and [menuitemradio](#) via inheritance.
- 06-Dec-2017: When `aria-errormessage` is not pertinent, authors **MUST** either ensure the content is not rendered or remove the `aria-errormessage` attribute or its value. User agents **MUST NOT** expose `aria-`

`errormessage` for an object with an `aria-invalid` value of `false`.

§ C. Acknowledgments

This section is non-normative.

The following people contributed to the development of this document.

§ C.1 Participants active in the ARIA WG at the time of publication

- Sina Bahram (Invited Expert)
- Curt Bellew (Oracle Corporation)
- Zoë Bijl (Invited Expert)
- Shari Butler (Pearson plc)
- Dominic Cooney (Meta)
- Michael Cooper (W3C Staff)
- James Craig (Apple Inc.)
- Joanmarie Diggs (Igalia)
- Isaac Durazo (Bocoup)
- Howard Edwards (Bocoup)
- Frank Elavsky (Invited Expert)
- Mayuri Faldu (Navy Federal Credit Union)
- Steve Faulkner (TPGi)
- Reinaldo Ferraz (NIC.br)
- Alexander Flenniken (Bocoup)
- Bryan Garaventa (Level Access)
- Rashmi Garimella (Google LLC)
- Matt Garrish (DAISY Consortium)
- Jaunita George (Navy Federal Credit Union)

- Ariella Gilmore (IBM Corporation)
- Raghavendra Giriappa (IBM Corporation)
- Michael Goddard (Bocoup)
- Glen Gordon (TPGi)
- Shirisha Gubba (Google LLC)
- Jon Gunderson (University of Illinois at Urbana-Champaign)
- Markku Hakkinen (Educational Testing Service)
- Sarah Higley (Microsoft Corporation)
- Hans Hillen (TPGi)
- Isabel Holdsworth (TPGi)
- Stanley Hon (Microsoft Corporation)
- Patrick Hung (University of Ontario Institute of Technology)
- Matthew King (Meta)
- Greta Krafsig (The Washington Post)
- Peter Krautzberger (Invited Expert)
- JaEun Jemma Ku (University of Illinois at Urbana-Champaign)
- Christopher Lane (VMWare)
- Charles LaPierre (Benetech)
- Gez Lemon (TPGi)
- Aaron Leventhal (Google LLC)
- Brian Liu Xu (Microsoft Corporation)
- David MacDonald (Invited Expert)
- Carolyn MacLeod (IBM Corporation)
- Mark McCarthy (University of Illinois at Urbana-Champaign)
- Jan McSorley (Pearson plc)
- Erika Miguel (Bocoup)
- Daniel Montalvo (W3C)
- Sheila Moussavi (Bocoup)
- James Nurthen (Adobe)

- Scott O'Hara (Microsoft Corporation)
- Adam Page (Intel Corporation)
- Michael Pennisi (Bocoup)
- Roberto Perez (Microsoft Corporation)
- Janina Sajka (Invited Expert, The Linux Foundation)
- Trish Salas (Level Access)
- Stefan Schnabel (SAP SE)
- Harris Schneiderman (Deque Systems, Inc.)
- Boaz Sender (Bocoup)
- Cynthia Shelly (Google LLC)
- Tzviya Siegman (Wiley)
- Avneesh Singh (DAISY Consortium)
- Neil Soiffer (Invited Expert)
- Francis Storr (Intel Corporation)
- Melanie Sumner (Invited Expert)
- Alexander Surkov (Igalia)
- James Teh (Mozilla Foundation)
- Seth Thompson (Bocoup)
- Jan Williams (TPGi)
- Benjamin Young (Wiley)
- Valerie Young (Igalia)
- Helen Zhou (University of Illinois)
- 邹杨 (Shenzhen Accessibility Research Association)

C.2 Other ARIA contributors, commenters, and previously active participants

Ann Abbott (Invited Expert), Shadi Abou-Zahra (W3C), Irfan Ali (Educational Testing Service), Jim Allan (TSB), CB Averitt (Deque Systems, Inc), Jonny Axelsson (Opera Software), David Baron (Mozilla Foundation), Art Barstow (Nokia Corporation), Simon Bates,

Amelia Bellamy-Royds (Invited Expert), Alex Bernier (Association BrailleNet),
Jorge Blazquez Alonso (IBM Corporation), Christy Blew (University of Illinois at Urbana-Champaign),
Chris Blouch (AOL), David Bolter (Mozilla Foundation), Alice Boxhall (Igalia),
Judy Brewer ([W3C/MIT](#)), Mark Birbeck (Sidewinder Labs), Matthew Brennan (Facebook),
Bogdan Brinza (Microsoft Corporation), Kim Bunge (TPGi),
Sally Cain (Royal National Institute of Blind People (RNIB)), Ben Caldwell (Trace),
Thaddeus Cambron (Invited Expert), Tammy Campoverde (UnitedHealth Group),
Gerardo Capiel (Benetech), David Caro (Wikimedia Foundation), Sofia Celic-Li,
Jaesik Chang (Samsung Electronics Co., Ltd.), Alex Qiang Chen (University of Manchester),
Charles Chen (Google, Inc.), Gerard K. Cohen, Christian Cohrs,
Timothy Cole (University of Illinois at Urbana-Champaign), Jory Cunningham (Salesforce),
Deborah Dahl, Erik Dahlström (Opera Software), Jes Daigle (Bocoup),
Dimitar Denev (Frauenhofer Gesellschaft), Jason Duan (IBM Corporation),
Micah Dubinko (Invited Expert), Mandana Eibegger, Beth Epperson (Websense),
Fred Esch (IBM Corporation), Donald Evans (AOL), Chris Fleizach (Apple Inc.),
John Foliot (Deque Systems, Inc.), Kelly Ford (Microsoft Corporation),
Geoff Freed (Invited Expert, NCAM), Kentarou Fukuda (IBM Corporation),
Christopher Gallelo (Microsoft Corporation), Billy Gregory (The Paciello Group, LLC),
Karl Groves (The Paciello Group, LLC), Birkir Gunnarsson (Deque Systems, Inc.), Guido Geloso,
Ali Ghassemi, Becky Gibson (Invited Expert), Alfred S. Gilman,
Andres Gonzalez (Adobe Systems Inc.), Scott González (JQuery Foundation), James Graham,
Georgios Grigoriadis (SAP AG), Jeff Grimes (Oracle), Loretta Guarino Reid (Google, Inc.),
Markus Gylling (DAISY Consortium), Katie Haritos-Shea (Knowbility), Barbara Hartel,
James Hawkins (Google, Inc.), Benjamin Hawkes-Lewis, Sean Hayes (Microsoft Corporation),
Mona Heath (University of Illinois at Urbana-Champaign), Jan Heck, Shawn Henry, Tina Homboe,
Nicholas Hoyt (University of Illinois at Urbana-Champaign), John Hrvatin (Microsoft Corporation),
Takahiro Inada, Masayasu Ishikawa ([W3C](#)), Jim Jewitt, Kenny Johar (Microsoft Corporation),
Earl Johnson (Sun), Masahiko Kaneko (Microsoft Corporation),
Shilpi Kapoor (BarrierBreak Technologies), Marjolein Katsma, Susann Keohane (IBM Corporation),
George Kerscher (International Digital Publishing Forum),
Jason Kiss (Department of Internal Affairs, New Zealand Government), Todd Kloots,
Jamie Knight (British Broadcasting Corporation), Johannes Koch, Sam Kuper, Jael Kurz,
Rajesh Lal (Nokia Corporation),
Diego La Monica (International Webmasters Association / [HTML](#) Writers Guild (IWA-HWG)),
Lori Lane (University of Illinois at Urbana-Champaign), Alex Li (SAP), Chris Lilley,
Thomas Logan (HiSoftware Inc.), Brian Loh, William Loughborough (Invited Expert),
Krzysztof Maczyński , Linda Mao (Microsoft), Anders Markussen (Opera Software),
Daniel Marques (WIRIS Science), Matthew May (Adobe Systems Inc.),
Dominic Mazzoni (Google LLC), Shane McCarron (Invited Expert, Aptest),
Charles McCathie Nevile (Yandex), Juliette McShane (Access2online Inc.),
Heather Migliorisi (Invited Expert), Mary Jo Mueller (IBM Corporation), Alexandre Morgaut (4D),

Ann Navarro (Invited Expert), Rich Noah (Bocoup), Joshue O Connor (Invited Expert, CFIT), Achraf Othman (MADA Center), Artur Ortega (Microsoft Corporation), Sailesh Panchang (Deque), Lisa Pappas (Society for Technical Communication (STC)), Marta Pawlowska (Samsung Electronics Co., Ltd.), Dave Pawson (RNIB), Steven Pemberton (CWI Amsterdam), Vijaya Gowri Perumal (Newgen Knowledgeworks), Christos Petrou (Centre for Inclusive Design), Simon Pieters (Bocoup), Jean-Bernard Piot (4D), David Poehlman (Opera Software), Ian Pouncey (TetraLogical Services Ltd), Sarah Pulis (Media Access Australia), T.V. Raman (Google, Inc.), Ruoxi Ran ([W3C Staff](#)), Melanie Richards (Microsoft Corporation), Jan Richards, Adrian Roselli (TPGi), Gregory Rosmaita (Invited Expert), Tony Ross (Microsoft Corporation), Alex Russell (Dojo Foundation), Mark Sadecki (Invited Expert), Mario Sánchez Prada (Samsung Electronics Co., Ltd. and Gnome Foundation), Martin Schaus (SAP AG), Doug Schepers ([W3C](#)), Cynthia Shelly (Microsoft Corporation) , Joseph Scheuhammer (Invited Expert, Inclusive Design Research Centre, OCAD University) , Matthias Schmitt , Richard Schwerdtfeger (IBM, Knowbility), Lisa Seeman-Kestenbaum (Invited Expert) , Marc Silbey (Microsoft Corporation), Leif Halvard Sili, Henri Sivonen (Mozilla), Ville Skyttä , Sharon Snider (IBM Corporation), Michael Smith ([W3C](#)), Andi Snow-Weaver (IBM Corporation), Volker Sorge (Invited Expert), Vitaly Sourikov, Mike Squillace (IBM), Maciej Stachowiak (Apple Inc.), Christophe Strobbe, Henny Swan (BBC), Suzanne Taylor (Pearson plc), William Tennis (Navy Federal Credit Union), Terrill Thompson, David Todd, Gregg Vanderheiden (Invited Expert, Trace), Job van Achterberg (Invited Expert), Anne van Kesteren, Scott Vinkle (Shopify), Wen He (Tencent), Can Wang (Zhejiang University), Wei Wang (Zhejiang University), Léonie Watson (TetraLogical Services Ltd), Wu Wei ([W3C / RITT](#)), Jason White (Educational Testing Service), Sam White (Apple Inc.), Ryan Williams (Oracle), Tom Wlodkowski, Evan Yamanishi (W. W. Norton), Marco Zehe (Mozilla Foundation), Gottfried Zimmermann (Invited Expert, Access Technologies Group)

C.3 Enabling funders

This publication has been funded in part with U.S. Federal funds from the Department of Education, National Institute on Disability, Independent Living, and Rehabilitation Research (NIDILRR), initially under contract number ED-OSE-10-C-0067, then under contract number HHSP23301500054C, and now under HHS75P00120P00168. The content of this publication does not necessarily reflect the views or policies of the U.S. Department of Education, nor does mention of trade names, commercial products, or organizations imply endorsement by the U.S. Government.

D. References

D.1 Normative references

[ACCNAME-1.2]

[*Accessible Name and Description Computation 1.2*](#). Bryan Garaventa; Joanmarie Diggs; Michael Cooper. W3C. 11 July 2019. W3C Working Draft. URL: <https://www.w3.org/TR/accname-1.2/>

[CORE-AAM]

[*Core Accessibility API Mappings 1.1*](#). Joanmarie Diggs; Joseph Scheuhammer; Richard Schwerdtfeger; Michael Cooper; Andi Snow-Weaver; Aaron Leventhal. W3C. 14 December 2017. W3C Recommendation. URL: <https://www.w3.org/TR/core-aam-1.1/>

[CORE-AAM-1.2]

[*Core Accessibility API Mappings 1.2*](#). Valerie Young; Alexander Surkov; Michael Cooper. W3C. 18 May 2023. W3C Candidate Recommendation. URL: <https://www.w3.org/TR/core-aam-1.2/>

[CSS3-SELECTORS]

[*Selectors Level 3*](#). Tantek Çelik; Elika Etemad; Daniel Glazman; Ian Hickson; Peter Linss; John Williams. W3C. 6 November 2018. W3C Recommendation. URL: <https://www.w3.org/TR/selectors-3/>

[DOM]

[*DOM Standard*](#). Anne van Kesteren. WHATWG. Living Standard. URL: <https://dom.spec.whatwg.org/>

[HTML]

[*HTML Standard*](#). Anne van Kesteren; Domenic Denicola; Ian Hickson; Philip Jägenstedt; Simon Pieters. WHATWG. Living Standard. URL: <https://html.spec.whatwg.org/multipage/>

[MathML3]

[*Mathematical Markup Language \(MathML\) Version 3.0 2nd Edition*](#). David Carlisle; Patrick D F Ion; Robert R Miner. W3C. 10 April 2014. W3C Recommendation. URL: <https://www.w3.org/TR/MathML3/>

[RFC2119]

[*Key words for use in RFCs to Indicate Requirement Levels*](#). S. Bradner. IETF. March 1997. Best Current Practice. URL: <https://www.rfc-editor.org/rfc/rfc2119>

[ROLE-ATTRIBUTE]

[*Role Attribute 1.0*](#). Shane McCarron et al. W3C. 28 March 2013. W3C Recommendation. URL: <https://www.w3.org/TR/role-attribute/>

[SVG2]

[*Scalable Vector Graphics \(SVG\) 2*](#). Amelia Bellamy-Royds; Bogdan Brinza; Chris Lilley; Dirk Schulze; David Storey; Eric Willigers. W3C. 4 October 2018. W3C Candidate Recommendation. URL: <https://www.w3.org/TR/SVG2/>

[uievents-key]

[UI Events KeyboardEvent key Values](#). Travis Leithead; Gary Kacmarcik. W3C. 30 May 2023. W3C Candidate Recommendation. URL: <https://www.w3.org/TR/uievents-key/>

[webidl]

[Web IDL Standard](#). Edgar Chen; Timothy Gu. WHATWG. Living Standard. URL: <https://webidl.spec.whatwg.org/>

[XML-NAMES]

[Namespaces in XML 1.0 \(Third Edition\)](#). Tim Bray; Dave Hollander; Andrew Layman; Richard Tobin; Henry Thompson et al. W3C. 8 December 2009. W3C Recommendation. URL: <https://www.w3.org/TR/xml-names/>

D.2 Informative references

[AT-SPI]

[Assistive Technology Service Provider Interface](#). The GNOME Project. URL: <https://developer-old.gnome.org/libatspi/stable/>

[ATK]

[ATK - Accessibility Toolkit](#). The GNOME Project. URL: <https://developer.gnome.org/atk/stable/>

[AXAPI]

[The NSAccessibility Protocol for macOS](#). Apple, Inc. URL: <https://developer.apple.com/documentation/appkit/nsaccessibility>

[HTML-ARIA]

[ARIA in HTML](#). Steve Faulkner; Scott O'Hara; Patrick Lauke. W3C. 31 May 2023. W3C Recommendation. URL: <https://www.w3.org/TR/html-aria/>

[IAccessible2]

[IAccessible2](#). Linux Foundation. URL: <https://wiki.linuxfoundation.org/accessibility/iaccessible2/>

[MSAA]

[Microsoft Active Accessibility \(MSAA\)](#). Microsoft Corporation. URL: <https://docs.microsoft.com/en-us/windows/win32/winauto/microsoft-active-accessibility>

[UI-AUTOMATION]

[UI Automation](#). Microsoft Corporation. URL: <https://docs.microsoft.com/en-us/windows/win32/winauto/ui-automation-specification>

[UIA-EXPRESS]

[The IAccessibleEx Interface](#). Microsoft Corporation. URL: <https://docs.microsoft.com/en-us/windows/win32/winauto/iaccessibleex>

[wai-aria-1.1]

[Accessible Rich Internet Applications \(WAI-ARIA\) 1.1](#). Joanmarie Diggs; Shane McCarron; Michael Cooper; Richard Schwerdtfeger; James Craig. W3C. 14 December 2017. W3C Recommendation. URL:

<https://www.w3.org/TR/wai-aria-1.1/>

[WAI-ARIA-PRACTICES-1.2]

WAI-ARIA Authoring Practices 1.2. Matthew King; JaEun Jemma Ku; James Nurthen; Zoë Bijl; Michael Cooper. W3C. 19 May 2022. W3C Working Group Note. URL: <https://www.w3.org/TR/wai-aria-practices-1.2/>

[WCAG21]

Web Content Accessibility Guidelines (WCAG) 2.1. Andrew Kirkpatrick; Joshue O'Connor; Alastair Campbell; Michael Cooper. W3C. 5 June 2018. W3C Recommendation. URL: <https://www.w3.org/TR/WCAG21/>

[XMLSHEMA11-2]

W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes. David Peterson; Sandy Gao; Ashok Malhotra; Michael Sperberg-McQueen; Henry Thompson; Paul V. Biron et al. W3C. 5 April 2012. W3C Recommendation. URL: <https://www.w3.org/TR/xmlschema11-2/>

↑

Supplemental Guidance to WCAG 2

*Additional ways to improve accessibility, not required to meet
WCAG*

[About](#)

[Supplemental](#)

[Web Accessibility](#)

[Initiative](#) WAI

[WCAG](#)



Design Patterns:

- [Make the Purpose of Your Page Clear](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o1p01-clear-purpose/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o1p01-clear-purpose/>)
- [Use a Familiar Hierarchy and Design](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o1p02-familiar-design/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o1p02-familiar-design/>)
- [Use a Consistent Visual Design](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o1p03-consistent-design/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o1p03-consistent-design/>)
- [Make Each Step Clear](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o1p04-clear-steps/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o1p04-clear-steps/>)
- [Clearly Identify Controls and Their Use](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o1p05-clear-controls/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o1p05-clear-controls/>)
- [Make the Relationship Clear Between Controls and the Content They Affect](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o1p06-control-actions/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o1p06-control-actions/>)
- [Use Icons that Help the User](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o1p07-icons-used/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o1p07-icons-used/>)

Help Users Find What They Need

Objective: [Help Users Find What They Need](https://www.w3.org/WAI/WCAG2/supplemental/objectives/o2-find/) (<https://www.w3.org/WAI/WCAG2/supplemental/objectives/o2-find/>)

Design Patterns:

- [Make it Easy to Find the Most Important Tasks and Features](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o2p01-site-important/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o2p01-site-important/>)
- [Make the Site Hierarchy Easy to Understand and Navigate](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o2p02-site-structure/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o2p02-site-structure/>)
- [Use a Clear and Understandable Page Structure](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o2p03-page-structure/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o2p03-page-structure/>)
- [Make it easy to find the most important actions and information on the page](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o2p04-page-important/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o2p04-page-important/>)
- [Break Media into Chunks](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o2p05-chunked-media/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o2p05-chunked-media/>)

- [Provide Search](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o2p06-search/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o2p06-search/>)

Use Clear and Understandable Content

Objective: [Use Clear and Understandable Content](https://www.w3.org/WAI/WCAG2/supplemental/objectives/o3-clear-content/) (<https://www.w3.org/WAI/WCAG2/supplemental/objectives/o3-clear-content/>)

Design Patterns:

- [Use Clear Words](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p01-clear-words/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p01-clear-words/>)
- [Use a Simple Tense and Voice](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p02-simple-tense/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p02-simple-tense/>)
- [Avoid Double Negatives or Nested Clauses](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p03-double-negatives/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p03-double-negatives/>)
- [Use Literal Language](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p04-literal-language/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p04-literal-language/>)
- [Keep Text Succinct](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p05-succinct-text/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p05-succinct-text/>)
- [Use Clear, Unambiguous Formatting and Punctuation](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p06-format-punctuation/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p06-format-punctuation/>)
- [Include Symbols and Letters Necessary to Decipher the Words](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p07-symbols-letters/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p07-symbols-letters/>)
- [Provide Summary of Long Documents and Media](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p08-summary-provided/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p08-summary-provided/>)
- [Separate Each Instruction](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p09-separated-instructions/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p09-separated-instructions/>)
- [Use White Spacing](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p10-whitespace/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p10-whitespace/>)
- [Ensure Foreground Content is not Obscured by Background](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p11-unobsured-foreground/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p11-unobsured-foreground/>)
- [Explain Implied Content](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p12-implicit-explained/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p12-implicit-explained/>)
- [Provide Alternatives for Numerical Concepts](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p13-alternatives-numerical-concepts/) (<https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p13-alternatives-numerical-concepts/>)

[supplemental/patterns/o3p13-numerical-alternatives/](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o3p13-numerical-alternatives/))

- Help Users Avoid Mistakes and Know How to Correct Them

Objective: [Help Users Avoid Mistakes and Know How to Correct Them \(https://www.w3.org/WAI/WCAG2/supplemental/objectives/o4-minimize-mistakes/\)](https://www.w3.org/WAI/WCAG2/supplemental/objectives/o4-minimize-mistakes/)

Design Patterns:

- [Ensure Controls and Content Do Not Move Unexpectedly \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p01-unexpected-movement/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p01-unexpected-movement/)
- [Let Users Go Back \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p02-back-undo/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p02-back-undo/)
- [Notify Users of Fees and Charges at the Start of a Task \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p03-declared-charges/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p03-declared-charges/)
- [Design Forms to Prevent Mistakes \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p04-supportive-forms/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p04-supportive-forms/)
- [Make it Easy to Undo Form Errors \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p05-form-undo/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p05-form-undo/)
- [Use Clear Visible Labels \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p06-clear-labels/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p06-clear-labels/)
- [Use Clear Step-by-step Instructions \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p07-step-instructions/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p07-step-instructions/)
- [Accept different input formats \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p08-input-formats/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p08-input-formats/)
- [Avoid Data Loss and “Timeouts” \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p09-data-loss/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p09-data-loss/)
- [Provide Feedback \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p10-status-feedback/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p10-status-feedback/)
- [Help the user stay safe \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p11-user-safety/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p11-user-safety/)
- [Use Familiar Metrics and Units \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p12-familiar-metrics/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o4p12-familiar-metrics/)

- Help Users Focus

Objective: [Help Users Focus \(https://www.w3.org/WAI/WCAG2/supplemental/objectives/o5-user-focus/\)](https://www.w3.org/WAI/WCAG2/supplemental/objectives/o5-user-focus/)

Design Patterns:

- [Limit Interruptions \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o5p01-minimal-interruptions/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o5p01-minimal-interruptions/)
- [Make Short Critical Paths \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o5p02-short-paths/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o5p02-short-paths/)
- [Avoid Too Much Content \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o5p03-manageable-quantity/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o5p03-manageable-quantity/)
- [Provide Information So a User Can Complete and Prepare for a Task \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o5p04-task-expectations/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o5p04-task-expectations/)

- Ensure Processes Do Not Rely on Memory

Objective: [Ensure Processes Do Not Rely on Memory \(https://www.w3.org/WAI/WCAG2/supplemental/objectives/o6-memory/\)](https://www.w3.org/WAI/WCAG2/supplemental/objectives/o6-memory/)

Design Patterns:

- [Provide a Login that Does Not Rely on Memory or Other Cognitive Skills \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o6p01-login-cognition/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o6p01-login-cognition/)
- [Allow the User a Simple, Single Step, Login \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o6p02-singlestep-login/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o6p02-singlestep-login/)
- [Provide a Login Alternative with Less Words \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o6p03-concise-login/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o6p03-concise-login/)
- [Let Users Avoid Navigating Voice Menus \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o6p04-voice-menus/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o6p04-voice-menus/)
- [Do Not Rely on Users Calculations or Memorizing Information \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o6p05-low-cognition/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o6p05-low-cognition/)

- Provide Help and Support

Objective: [Provide Help and Support \(https://www.w3.org/WAI/WCAG2/supplemental/objectives/o7-support/\)](https://www.w3.org/WAI/WCAG2/supplemental/objectives/o7-support/)

Design Patterns:

- [Provide Human Help \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o7p01-human-help/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o7p01-human-help/)
- [Provide Alternative Content for Complex Information and Tasks \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o7p02-alternative-content/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o7p02-alternative-content/)
- [Clearly State the Results and Disadvantages of Actions, Options, and Selections \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o7p03-supported-choice/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o7p03-supported-choice/)
- [Provide Help for Forms and Non-standard Controls \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o7p04-described-interactions/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o7p04-described-interactions/)
- [Make It Easy to Find Help and Give Feedback \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o7p05-findable-support/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o7p05-findable-support/)
- [Provide Help with Directions \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o7p06-supported-wayfinding/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o7p06-supported-wayfinding/)
- [Provide Reminders \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o7p07-reminders/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o7p07-reminders/)

- **Support Adaptation and Personalization**

Objective: [Support Adaptation and Personalization \(https://www.w3.org/WAI/WCAG2/supplemental/objectives/o8-personalization/\)](https://www.w3.org/WAI/WCAG2/supplemental/objectives/o8-personalization/)

Design Patterns:

- [Let Users Control When the Content Moves or Changes \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o8p01-motion/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o8p01-motion/)
- [Enable APIs and Extensions \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o8p02-apis/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o8p02-apis/)
- [Support Simplification \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o8p03-complexity/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o8p03-complexity/)
- [Support a Personalized and Familiar Interface \(https://www.w3.org/WAI/WCAG2/supplemental/patterns/o8p04-interface/\)](https://www.w3.org/WAI/WCAG2/supplemental/patterns/o8p04-interface/)

[– Collapse All Sections](#)



Audio Content and Video Content

in [Making Audio and Video Media Accessible](#) (<https://www.w3.org/WAI/media/av/>)



Summary

This page describes accessibility considerations when planning, scripting, storyboarding, recording, and producing audio and video.

It covers **common accessibility barriers** including:

- [missing description of visual information](#) (↳ [#plan-description](#)) (such as text in the video) for people who cannot see the video
- [requiring sight to understand the content](#) (↳ [#sensory](#)) of the video
- [making text in the video hard for some people to see](#) (↳ [#readable](#)) because there is not enough contrast between the text and the background colors

Page Contents

- [Introduction](#) (↳ [#introduction](#))
- [Audio](#) (↳ [#audio](#))
 - [Create high-quality audio - recording setup](#) (↳ [#create-high-quality-audio--recording-setup](#))
 - [Use low background audio - recording, post-production \(WCAG AAA\)](#) (↳ [#use-low-background-audio--recording-post-production-wcag-aaa](#))
 - [Speak clearly and slowly - speakers](#) (↳ [#speak-clearly-and-slowly--speakers](#))
 - [Give people time to process information - speakers, post-production](#) (↳ [#give-people-time-to-process-information--speakers-post-production](#))
 - [Use clear language - script](#) (↳ [#use-clear-language--script](#))
 - [Provide redundancy for sensory characteristics - script \(WCAG A\)](#) (↳ [#sensory](#))
- [Video](#) (↳ [#video](#))
 - [Avoid causing seizures - storyboarding, post-production \(WCAG A\)](#) (↳ [#avoid-](#))

causing-seizures-storyboarding-post-production-wcag-a)

- Consider speaker visibility – *storyboarding, recording, post-production*([#consider-speaker-visibility--storyboarding-recording-post-production](#))
- Make overlay text readable – *storyboarding, post-production* (WCAG AA, AAA)([#readable](#))
- Plan for sign language – *storyboarding, script, recording* (WCAG AAA)([#plan-for-sign-language](#))
- Plan for description of visual information – *storyboarding, recording* (WCAG A, AA)([#plan-description](#))

Introduction

This page addresses accessibility considerations when planning, scripting, storyboarding, recording, and producing audio and video.

Some of the guidance below is related to requirements in Web Content Accessibility Guidelines (WCAG) and has links to a separate resource. (*The Planning page of this resource introduces the WCAG Standard (<https://www.w3.org/WAI/media/av/planning/#wcag-standard>).*) Other guidance is good practice.

Additional guidance is in the resource Making Events Accessible - Checklist for meetings, conferences, training, and presentations that are remote/virtual, in-person, or hybrid (<https://www.w3.org/WAI/teach-advocate/accessible-presentations/>):

- Preparing Slides and Projected Material (<https://www.w3.org/WAI/teach-advocate/accessible-presentations/#preparing-slides-and-projected-material-speakers>)
- During the Presentation (<https://www.w3.org/WAI/teach-advocate/accessible-presentations/#during-the-presentation-speakers>)

Audio

Create high-quality audio – *recording setup*

- Use high-quality microphone(s) and recording software.
- When feasible, record in a room that is isolated from all external sounds.

- Avoid rooms with hard surfaces, such as tile or wood floors.

Use low background audio – *recording, post-production (WCAG AAA)*

When the main audio is a person speaking and you have background music, set the levels so people with hearing or cognitive disabilities can easily distinguish the speaking from the background.

Specifically, make the background sounds at least 20 decibels lower than the foreground speech content (with the exception of occasional sounds that last for only one or two seconds).

Avoid sounds that can be distracting or irritating, such as some high pitches and repeating patterns.

More information is in Understanding Success Criterion 1.4.7: Low or No Background Audio (AAA) (<https://www.w3.org/WAI/WCAG22/Understanding/low-or-no-background-audio.html>).

Speak clearly and slowly – *speakers*

Speak clearly. This is important for people wanting to understand the content, and for captioners.

Speak as slowly as appropriate. This will enable listeners to understand better, and make the timing better for captions and sign language.

Give people time to process information – *speakers, post-production*

Pause between topics.

Use clear language – *script*

Avoid or explain jargon, acronyms, and idioms. For example, expressions such as “raising the bar” can be interpreted literally by some people with cognitive disabilities and can be confusing.

Provide redundancy for sensory characteristics – *script* (WCAG A)

Make your information work for people who cannot see and/or cannot hear.

For example, instead of saying:

Attach this to the green end.

Say:

Attach the small ring to the green end, which is the larger end.

More information that primarily addresses web pages, yet is relevant to audio and video, is in [Understanding Success Criterion 1.3.3: Sensory Characteristics \(A\) \(https://www.w3.org/WAI/WCAG22/Understanding/sensory-characteristics.html\)](#).

Video

Avoid causing seizures – *storyboarding, post-production* (WCAG A)

Avoid anything that flashes more than three times in any one second period.

More information is in [Understanding Success Criterion 2.3.2: Three Flashes \(AAA\) \(https://www.w3.org/WAI/WCAG22/Understanding/three-flashes.html\)](#) and [Understanding Success Criterion 2.3.1: Three Flashes or Below Threshold \(A\) \(https://www.w3.org/WAI/WCAG22/Understanding/three-flashes-or-below-threshold.html\)](#)

Consider speaker visibility – *storyboarding, recording, post-production*

Some people use mouth movement to help understand spoken language. When feasible, ensure that the speaker's face is visible and in good light.

Make overlay text readable – *storyboarding, post-production* (WCAG AA, AAA)

For any text, consider the font family, size, and contrast between the text and background.

More information is in [Understanding Success Criterion 1.4.3: Contrast \(Minimum\) \(AA\) \(https://www.w3.org/WAI/WCAG22/Understanding/contrast-minimum.html\)](https://www.w3.org/WAI/WCAG22/Understanding/contrast-minimum.html) and [Understanding Success Criterion 1.4.6: Contrast \(Enhanced\) \(AAA\) \(https://www.w3.org/WAI/WCAG22/Understanding/contrast-enhanced.html\)](https://www.w3.org/WAI/WCAG22/Understanding/contrast-enhanced.html).

Plan for sign language – *storyboarding, script, recording (WCAG AAA)*

Often sign languages are provided as an overlay in the bottom right corner of videos. For example: [NHS 111 British Sign Language \(BSL\) Advert \(YouTube\)](#)



(<https://www.youtube.com/watch?v=TCq3ru9HQSc>)

Plan for the video not to include important information that would be obstructed by a sign language overlay.

For other guidance including recording, see another page of this resource: [Sign Languages](#) (<https://www.w3.org/WAI/media/av/sign-languages/>)

Plan for description of visual information – *storyboarding, recording (WCAG A, AA)*

Description provides content to people who are blind and others who cannot see the video adequately. It describes the visual information needed to understand the content, **including text displayed in the video.**

Plan to either:

- Integrate description into the main audio content,
or
- Record the audio and video with timing to accommodate separate description.

Integrate description

For some videos, such as presentations and instructional videos, the best way to handle

description is not to need it at all — that is, all the visual information that users need to understand the content is integrated in the main audio. This is called “integrated description”. When planned in advance, this is fairly simple for many videos on the web. For example:

Instead of the speaker saying:	The speaker can say:
As you can see on this chart, sales increased significantly from the first quarter to the second quarter.	This chart shows that sales increased significantly, from 1 million in the first quarter to 1.3 million in the second quarter.
Whip the mixture until it looks like this.	Whip the mixture until the oil, vinegar, and spices are well combined.
Attach this to the green end.	Attach the small ring to the green end, which is the larger end.

Here is an example training video with the description integrated in what the trainer is saying (YouTube)



(<https://www.youtube.com/watch?v=jUfmCvdzqbM>)

If you want guidance on what to include in description, see the “Description of Visual Information” page, Tips for Writing Description section (<https://www.w3.org/WAI/media/av/description/#writing>).

Time for description

For some types of videos, such as dramas, the description of the visual information cannot be smoothly handled by the speakers in the main video. For those videos, the description will be separate.

Where the description is fairly short, plan space in the audio to add the description.

Where the description is longer than you want to leave space in the main audio, you can record extra time in the scene to accommodate the description without having to pause the

scene. That is, the same scene is shorter in the main video. In the described version, that same scene is a little longer at the beginning or the end of it. For example:

Narration	Main Video Scene Duration	Described Video Scene Duration	Description
“Captions are also handy for people who want to watch video in loud environments.”	3 seconds	7 seconds	A man is watching the captioned video with a group of people chatting away next to him.
“Or where you need to be very, very quiet.”	2 seconds	5 seconds	Turns out that they are in a library. The group is shushed by the librarian.

An example of this is the [Web Accessibility Perspectives: Video Captions](https://www.w3.org/WAI/perspective-videos/captions/) (<https://www.w3.org/WAI/perspective-videos/captions/>) video. The main video is 48 seconds long. The described version is 1 minute and 18 seconds long, yet there are no pauses in the visual aspect of the video.

More about description

More information is in the next page of this resource: [Description of Visual Information](https://www.w3.org/WAI/media/av/description/) (<https://www.w3.org/WAI/media/av/description/>).

◀ Previous: Planning	(https://www.w3.org/WAI/media/av/planning/)	(https://www.w3.org/WAI/media/av/description/)	Next: ▶ Description
--	--	--	--

Updated: 17 September 2024. [Latest changes](https://www.w3.org/WAI/media/av/changelog/) (<https://www.w3.org/WAI/media/av/changelog/>).
First published September 2019.

Editor: [Shawn Lawton Henry](https://www.w3.org/WAI/media/av/acknowledgements/). [Acknowledgements](https://www.w3.org/WAI/media/av/acknowledgements/) (<https://www.w3.org/WAI/media/av/acknowledgements/>) lists contributors and credits.

Developed by the Education and Outreach Working Group (EOWG). Originally drafted as part of the [WCAG TA Project](https://www.w3.org/WAI/WCAGTA/) (<https://www.w3.org/WAI/WCAGTA/>) funded by the U.S. Access Board. Revised as part of the [WAI Expanding Access project](https://www.w3.org/WAI/expanding-access/) (<https://www.w3.org/WAI/expanding-access/>) funded by the Ford Foundation.

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities.

Copyright © 2025 World Wide Web Consortium ([W3C®](#)).
See [Permission to Use WAI Material](#).



Captions/Subtitles

in *Making Audio and Video Media Accessible* (<https://www.w3.org/WAI/media/av/>)



Summary

Captions (called “subtitles” in some areas) provide content to people who are Deaf and hard-of-hearing. Captions are a text version of the speech and non-speech audio information needed to understand the content. They are synchronized with the audio and usually shown in a media player when users turn them on.

This page helps you understand and create captions and subtitles.

Page Contents

- [Introduction](#)(`↳ #introduction`)
 - [Captions and Subtitles](#)(`↳ #captions-and-subtitles`)
 - [Live Captions](#)(`↳ #live-captions`)
- [Does My Media Need Captions?](#)(`↳ #checklist`)
- [Skills and Tools](#)(`↳ #skills-and-tools`)
- [Automatic Captions are Not Sufficient](#)(`↳ #automatic-captions-are-not-sufficient`)
- [Creating Captions](#)(`↳ #creating-captions`)
 - [Caption File Format](#)(`↳ #caption-file-format`)
 - [Caption Tools](#)(`↳ #caption-tools`)
 - [Transcribing Audio to Text](#)(`↳ #transcribing-audio-to-text`)
- [Positioning and Styling Captions](#)(`↳ #positioning-and-styling-captions`)

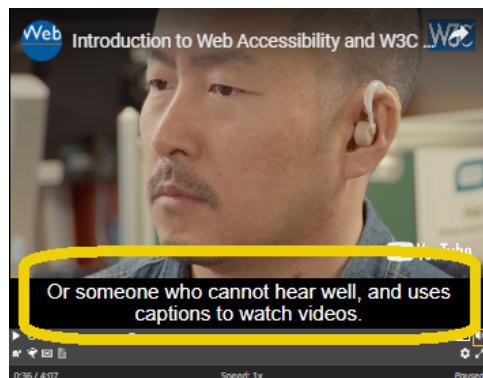
Introduction

Who: Captions (also called “intralingual subtitles”) provide content to people who are Deaf

and others who cannot hear the audio. They are also used by people who process written information better than audio.

What: Captions are a text version of the speech and non-speech audio information needed to understand the content. They are displayed within the media player and are synchronized with the audio.

Most are “closed captions” that can be hidden or shown by people watching the video. They can be “open captions” that are always displayed and cannot be turned off.



Captions and Subtitles

The terms “captions” and “subtitles” are used for the same thing in different regions of the world. This resource uses:

- *Captions* for the same language as the spoken audio.
- *Subtitles* for spoken audio translated into another language.

Some regions use *subtitles* for both the same language as the audio and for the translation. Sometimes they are distinguished as *intralingual subtitles* (same language) and *interlingual subtitles* (different language).

Subtitles are implemented the same way as captions. Subtitles/interlingual subtitles are usually only the spoken audio (for people who can hear the audio but do not know the spoken language). They can be a translation of the caption content, including non-speech audio information.

Captions are needed for accessibility, whereas subtitles in other languages are not directly an accessibility accommodation.

Live Captions

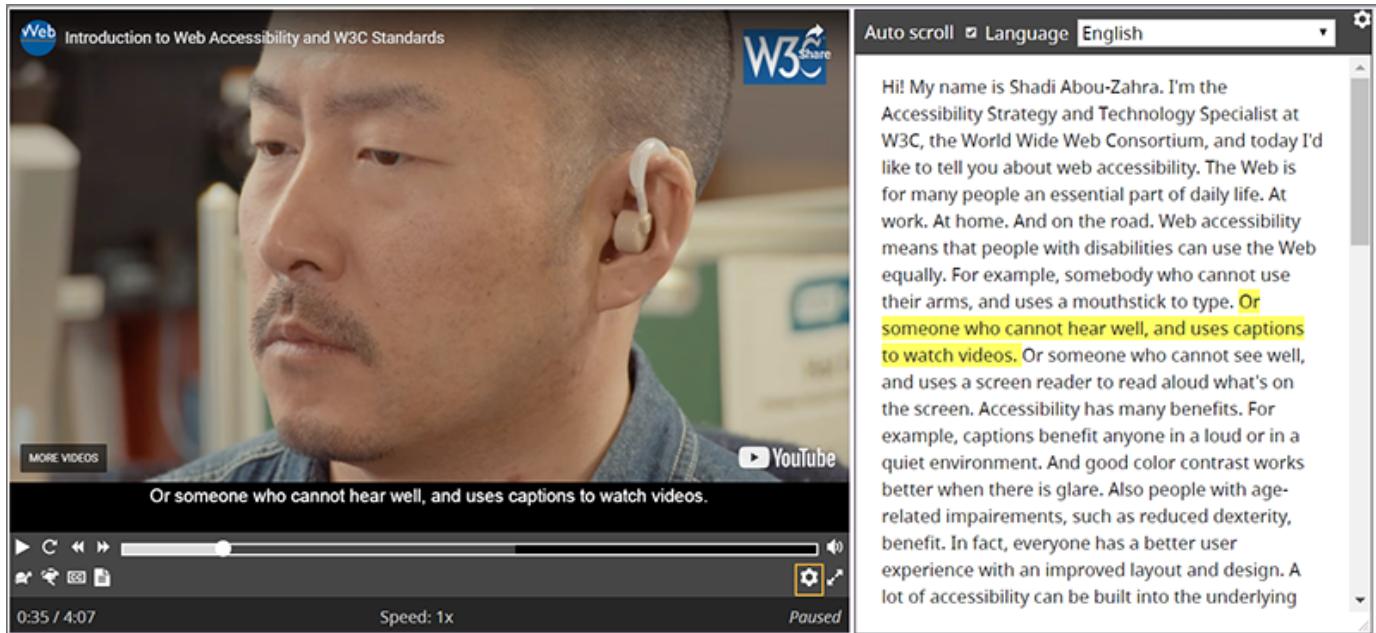
Live captions are usually done by professional real-time captioners or Communication Access Realtime Translation (CART) providers. Live captions can be done in-person or remotely. That is, the person doing the captioning/CART does not have to be at the same location as the live action; they can be doing the live captions by listening to the audio over a phone or Internet connection.

If you have live captions and you post a recording, you will probably need to do minor editing for accuracy.

This rest of this page addresses developing captions for pre-recorded media.

Interactive Transcripts from Captions

Caption files are used by some media players to provide *interactive transcripts*. Interactive transcripts highlight text phrases as they are spoken. Users can select text in the transcript and go to that point in the video. Some players provide interactive transcript functionality.



Notes

For optimum accessibility, provide a separate caption file of the description of visual information (called audio description, video description, or described video).

Captions and transcripts include the same text, so one can be used to develop the other.

Does My Media Need Captions?

This section tells you:

- What is required in the WCAG standard at Level A, AA, and AAA. ([WCAG \(https://www.w3.org/WAI/media/av/planning/#wcag-standard\)](https://www.w3.org/WAI/media/av/planning/#wcag-standard) is introduced in the Planning page of this resource.)
- What is needed to meet user needs, beyond WCAG. If there are no "A"s, then it is not required in WCAG.

Audio-only (e.g., podcast):

- For pre-recorded:
 - Captions are useful for people who are hard of hearing to get the richness of listening to the audio and fill in what they don't hear well by reading the captions.
Captions are not required to meet WCAG. (Transcripts are at Level A.)
- For live:
 - Captions are useful for people who are hard of hearing to get the richness of listening to the audio and fill in what they don't hear well by reading the captions.
Live text stream or accurate script of the audio available when live is in WCAG at Level AAA.

Video-only (no audio content):

- For pre-recorded and live:
 - Captions are not needed because there is no audio information.

Video with audio content:

Does the video have audio information that is needed to understand what the video is communicating?

- If no (for example, it is just background music):
 - Captions are not needed because there is no important audio content. Consider [informing users \(https://www.w3.org/WAI/media/av/planning/#none\)](https://www.w3.org/WAI/media/av/planning/#none).
- If yes:
 - For pre-recorded:
 - Captions are needed to provide the audio content to people who are Deaf or hard of hearing.

Captions are **required** in WCAG at Level A.

- For live:
 - Captions are needed to provide the audio content to people who are Deaf or hard of hearing.
Captions are **required** in WCAG at Level AA.

WCAG excerpts with links to more information in “Understanding WCAG”:

- [A 1.2.2 Captions](https://www.w3.org/WAI/WCAG22/Understanding/captions-prerecorded.html) (<https://www.w3.org/WAI/WCAG22/Understanding/captions-prerecorded.html>) (Prerecorded): Captions are provided for all prerecorded audio content in synchronized media...
- [AA 1.2.4 Captions](https://www.w3.org/WAI/WCAG22/Understanding/captions-live.html) (<https://www.w3.org/WAI/WCAG22/Understanding/captions-live.html>) (Live): Captions are provided for all live audio content in synchronized media.

Skills and Tools

Creating captions requires typing up the audio (“transcribing”) and formatting it in a file with timestamps. Transcribing an audio file is fairly difficult and takes quite a bit of time for people who don’t have the software and skill for it. The file format for captions are simple, yet it’s tedious to add timestamps, especially without software or service for developing caption files.

Creating high-quality captions requires knowledge of which non-speech audio information should be included in the captions. It’s more art than science — for example, it’s not always clear which non-speech audio information to include and how to communicate it in text.

Even correcting an automatic caption files takes quite a bit of time for people who don’t do it regularly.

However, people who have the software, skills, and experience in developing captions, can develop them much faster.

For these reasons, many organizations choose to outsource their captions.

Automatic Captions are Not Sufficient

Automatically-generated captions do not meet user needs or accessibility requirements, unless they are confirmed to be fully accurate. Usually they need significant editing.

There are tools that use speech recognition technology to turn a soundtrack into a timed caption file. For example, some common video websites provide automatic captions. However, often the automatic caption text is wrong and does not match the spoken audio — sometimes in ways that change the meaning (or are embarrassing). For example, missing just one word such as “not” can make the captions contradict the actual audio content.

Example of bad automatic captions (that cause a fire)

Spoken audio:

"Broil on high for **4 to 5 minutes**. You should **not** preheat the oven."

Automatic caption:

"Broil on high for **45 minutes**. You should **know to** preheat the oven."



Automatic captions can be used as a starting point for developing accurate captions and transcripts.

Creating Captions

Caption File Format

The most common format for captions on the web is [WebVTT](https://www.w3.org/TR/webvtt/) (<https://www.w3.org/TR/webvtt/>): The Web Video Text Tracks Format.

Example VTT file with speakers identified

WEBVTT

00:11.000 --> 00:13.000

<v Rajwinder Kaur>Welcome to the podcast.

00:13.000 --> 00:17.000

<v Shawn Henry>Thank you for this opportunity to share information about accessibility.

00:17.000 --> 00:20.000

<v Rajwinder>Would you start by telling us a little about your role at W3C?

00:20.000 --> 00:24.000

<v Shawn>I work within the Web Accessibility Initiative, W-A-I, pronounced "way".

Other caption formats are: SRT and Timed Text Markup Language ([TTML \(https://www.w3.org/TR/ttml2/\)](https://www.w3.org/TR/ttml2/)).

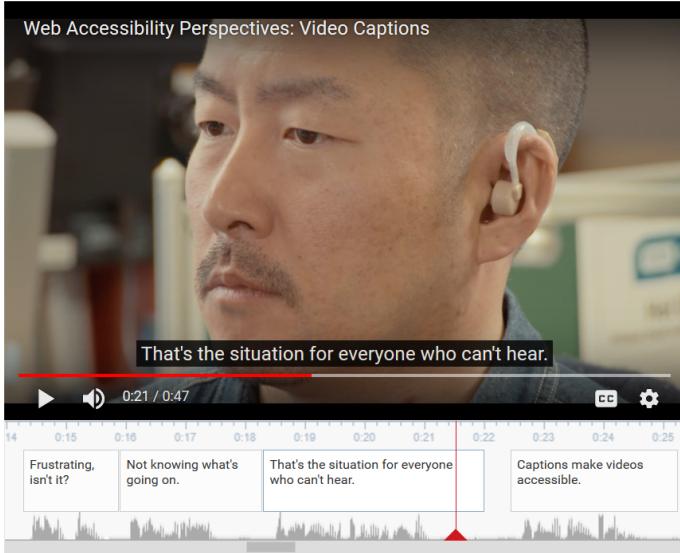
Caption Tools

Most people use software or services to help develop captions. There are several free captioning software programs and online services available.

Several free and fee-based tools create automatic captions that you can use as a starting point. For example, a common video website includes automatic captions and tools for you to edit the captions. **You will need to edit automatic captions for accuracy.**

If you already have transcription of the audio into text, there are free tools that will generate a captions file with timestamps. You will need to edit it for line breaks as described in another page of this resource, Transcribing Audio to Text: [More on Captions \(https://www.w3.org/WAI/media/av/transcribing/#more-on-captions\)](https://www.w3.org/WAI/media/av/transcribing/#more-on-captions).

Most caption-editing tools can export a plain text transcript.



The screen capture shows one tool for editing captions, in the area underneath the video.

Transcribing Audio to Text

For specific guidance on what to type up, see another page in this resource: [Transcribing Audio to Text](https://www.w3.org/WAI/media/av/transcribing/) (<https://www.w3.org/WAI/media/av/transcribing/>).

Positioning and Styling Captions

There are options for authors to position and style captions. Support in browsers and other media players is inconsistent and sometimes unreliable. Most web videos just use the player's default presentation style, which is usually white characters in a black box.

Some media players enable users to set preferences for how and where captions are displayed, including text style, text size, colors, and position of the captions.

← Previous:
[Description](#)

(<https://www.w3.org/https://www.w3.org/WAI/media/av/description/>)

Next:
[Transcripts](#) →

Updated: 17 September 2024. [Latest changes](#) (<https://www.w3.org/WAI/media/av/changelog/>).
First published September 2019.

Editor: [Shawn Lawton Henry](#). [Acknowledgements](#) (<https://www.w3.org/WAI/media/av/acknowledgements/>) lists contributors and credits.

Developed by the Education and Outreach Working Group ([EOWG](#)). Originally drafted as part of the [WCAG TA Project](#) (<https://www.w3.org/WAI/WCAGTA/>) funded by the U.S. Access Board. Revised as part of the [WAI Expanding Access project](#) (<https://www.w3.org/WAI/ExpandingAccess/>).

[WAI/expand-access/](#)) funded by the Ford Foundation.

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities.

Copyright © 2025 World Wide Web Consortium ([W3C®](#)).

See [Permission to Use WAI Material](#).



Carousels Tutorial

in [Tutorials](#) (<https://www.w3.org/WAI/tutorials/>)

Implement an accessible carousel widget by providing a robust structure and user control:

- **Structure (<https://www.w3.org/WAI/tutorials/carousels/structure/>):** Use semantic structure for the carousel to support proper function of assistive technology.
- **Functionality (<https://www.w3.org/WAI/tutorials/carousels/functionality/>):** Add functionality to display and announce carousel items.
- **Animations (<https://www.w3.org/WAI/tutorials/carousels/animations/>):** Add a transition animation between items and ensure users can stop and resume it.
- **Styling (<https://www.w3.org/WAI/tutorials/carousels/styling/>):** Style the carousel to make sure it's usable and readable by everyone.

See also the [complete working example](#) (<https://www.w3.org/WAI/tutorials/carousels/working-example/>) and [full code](#) (<https://www.w3.org/WAI/tutorials/carousels/full-code/>) of the example carousel.

What are carousels?

Carousels show a collection of items one at a time. They are also known as “slideshows” and “sliders”. Typical uses of carousels include scrolling news headlines, featured articles on home pages, and image galleries.

What makes a carousel accessible?

- Users must be able to pause carousel movement because it can be too fast or distracting, making text hard to read.
- All functionality, including navigating between carousel items, must be operable by

keyboard.

- Changes to carousel items must be communicated to all users, including screen reader users.
- The keyboard position (“focus”) is managed in a reasonable and comprehensible fashion.

Note: Carousels are disputed from a usability perspective because their content can be hard to discover. Ensuring accessibility can also improve usability.

Why is this important?

Typically, carousels are prominently located and are used to provide navigation or show page content. Accessible carousels are essential for many website users including:

- **People using keyboard navigation and voice input software** can navigate between individual items.
- **People using screen readers** will understand which item is currently shown and how to navigate between items.
- **People who are distracted by movement** can pause animations.
- **People who need more time to read** can pause animations, providing them with sufficient time to read and understand carousel content.

* Related WCAG resources

These tutorials provide best-practice guidance on implementing accessibility in different situations. This page combined the following WCAG success criteria and techniques from different conformance levels:

Success Criteria:

- **1.3.1 Info and Relationships:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-content-structure-separation-programmatic>) Information, structure, and relationships conveyed through presentation can be programmatically determined or are available in text. (Level A)
- **2.1.1 Keyboard:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-keyboard-operation-keyboard-operable>) All functionality of the content is operable through a keyboard interface without requiring specific timings for individual keystrokes,

except where the underlying function requires input that depends on the path of the user's movement and not just the endpoints. (Level A)

- **2.2.2 Pause, Stop, Hide:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-time-limits-pause>) For moving, blinking, scrolling, or auto-updating information, all of the following are true:

- **Moving, blinking, scrolling:** For any moving, blinking or scrolling information that (1) starts automatically, (2) lasts more than five seconds, and (3) is presented in parallel with other content, there is a mechanism for the user to pause, stop, or hide it unless the movement, blinking, or scrolling is part of an activity where it is essential; and
- **Auto-updating:** For any auto-updating information that (1) starts automatically and (2) is presented in parallel with other content, there is a mechanism for the user to pause, stop, or hide it or to control the frequency of the update unless the auto-updating is part of an activity where it is essential.

(Level A)

- **4.1.2 Name, Role, Value:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-ensure-compat-rsv>) For all user interface components (including but not limited to: form elements, links and components generated by scripts), the name and role can be programmatically determined; states, properties, and values that can be set by the user can be programmatically set; and notification of changes to these items is available to user agents, including assistive technologies. (Level A)

[\(https://www.w3.org/WAI/tutorials/carousels/structure/\)](https://www.w3.org/WAI/tutorials/carousels/structure/)

Next:
Carousel Structure ➔

Status: Updated 13 April 2017 (first published May 2015)

Editors: [Eric Eggert](https://www.w3.org/People/yatil/) (<https://www.w3.org/People/yatil/>) and [Shadi Abou-Zahra](https://www.w3.org/People/shadi/) (<https://www.w3.org/People/shadi/>). Update Editor: Brian Elton. Contributors: see [Acknowledgements](https://www.w3.org/WAI/tutorials/acknowledgements/) (<https://www.w3.org/WAI/tutorials/acknowledgements/>).

Developed by the Education and Outreach Working Group (EOWG (<https://www.w3.org/groups/wg/eowg>)). Developed with support from the [WAI-ACT project](https://www.w3.org/WAI/ACT/) (<https://www.w3.org/WAI/ACT/>), co-funded by the [European Commission IST Programme](#).

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities.

Copyright © 2025 World Wide Web Consortium ([W3C®](#)).
See [Permission to Use WAI Material](#).



Description of Visual Information

in [Making Audio and Video Media Accessible](#) (<https://www.w3.org/WAI/media/av/>)



Summary

Description of visual information is called ***audio description***, ***video description***, or ***described video*** in different areas.

Description provides content to people who are blind and others who cannot see the video adequately. It describes visual information needed to understand the content, including text displayed in the video.

This page helps you understand and create description of visual information for new and existing videos. (Description does not apply to audio-only, such as podcasts.)

Page Contents

- [Introduction](#)(`↳ #introduction`)
 - [Terminology](#)(`↳ #terminology`)
- [Does My Media Need Description?](#)(`↳ #checklist`)
- [Description Considerations, Skills, and Tools](#)(`↳ #description-considerations-skills-and-tools`)
- [What Method of Description?](#)(`↳ #what-method-of-description`)
- [Options for Creating Description](#)(`↳ #options-for-creating-description`)
 - [Integrated – Creating Integrated Description](#)(`↳ #integrated--creating-integrated-description`)
 - [Text – Creating Description in a Text File](#)(`↳ #text--creating-description-in-a-text-file`)
 - [Audio Only – Creating Description in a Separate Audio File Only](#)(`↳ #audio-only--creating-description-in-a-separate-audio-file-only`)
 - [Video with Space – Creating a Separate Described Video - If Descriptions Fit in](#)

[Audio Spaces\(↳ #video-with-space--creating-a-separate-described-video---if-descriptions-fit-in-audio-spaces\)](#)

- [Video Without Space – Creating a Separate Described Video - If Descriptions Do Not Fit in Audio Spaces\(↳ #video-without-space--creating-a-separate-described-video---if-descriptions-do-not-fit-in-audio-spaces\)](#)
- [Other Options\(↳ #other-options\)](#)
- [Tips for Doing Description Yourself\(↳ #tips-for-doing-description-yourself\)](#)
 - [Tips for Writing Descriptions\(↳ #writing\)](#)
 - [Tips for Recording Descriptions\(↳ #recording\)](#)
 - [Tips for Combining Audio Files\(↳ #combining\)](#)
 - [VTT File\(↳ #vtt-file\)](#)

Introduction

Who: Description provides content to people who are blind and others who cannot see the video adequately.

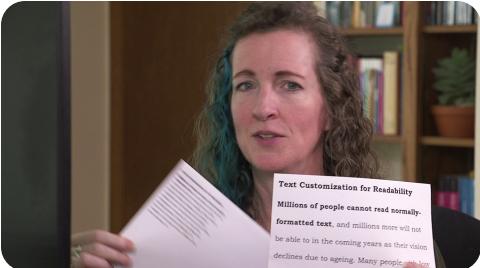
What: Description explains visual information needed to understand the content. (For example, "Pat opens a small box, looks at a diamond engagement ring, and cries".) It can be provided as:

- **integrated description** — description is included in the main speakers' scripts
- **alternative video** — description is included in a separate video
- **separate file** — description is in a timed text file or synced audio file; must be supported by the media player

 More about types of description:

Examples:

- Integrated: [Training video with the description integrated in what the trainer is saying \(YouTube\)](#)



(<https://www.youtube.com/watch?v=JUfmCvdzqbM>)

- Alternative video: [Alternative story video with audio description in a different voice \(YouTube\)](#)



(<https://www.youtube.com/watch?v=F3A1VffiOH4>)

These videos are also available from the W3C website: [training video \(web page\)](#) (<https://www.w3.org/2020/10/TPAC/w3cx-challenging-assumptions.html#talk>), [story video \(MP4 file size 28MB\)](#).

Terminology

Description of visual information provided via audio is called “audio description” in Web Content Accessibility Guidelines (WCAG). In some regions and documents it is called “video description” or “described video”.

This resource uses “described video” in some places as a shortened form of “a video that includes description of the visual information in audio”.

Does My Media Need Description?

This section tells you:

- What is required in the WCAG standard at Level A, AA, and AAA. ([WCAG \(<https://www.w3.org/WAI/media/av/planning/#wcag-standard>\) is introduced in the Planning page of this resource.](#))
- What is needed to meet user needs, beyond WCAG. If there are no “A’s, then it is not required in WCAG.

Audio-only (e.g., podcast):

- For pre-recorded and live:
 - Description is not needed because there is no visual information.

Video:

Does the video have visual information that is needed to understand what the video is communicating?

- If no (for example, it is only a person talking):
 - Description is not needed. Consider [informing users](https://www.w3.org/WAI/media/av/planning/#none) (<https://www.w3.org/WAI/media/av/planning/#none>).
- If yes:
 - For pre-recorded:
 - Description is needed to provide the important visual information to people who are blind and listen to the video.
 - Description **or** a [descriptive transcript](https://www.w3.org/WAI/media/av/transcripts/) (<https://www.w3.org/WAI/media/av/transcripts/>) is **required** in WCAG at Level A.
 - Description is **required** in WCAG at Level AA.
 - For live:
 - Description is needed to provide the important visual information to people who are blind.
 - Description is not required to meet WCAG.

WCAG excerpts with emphasis added, additions in [brackets], and links to more information in “Understanding WCAG”:

- [A 1.2.1 Audio-only and Video-only](https://www.w3.org/WAI/WCAG22/Understanding/audio-only-and-video-only-prerecorded.html) (<https://www.w3.org/WAI/WCAG22/Understanding/audio-only-and-video-only-prerecorded.html>) (Prerecorded): For... Prerecorded Video-only: Either an alternative for time-based media [descriptive transcript] **or** an audio track [of description] is provided that presents equivalent information for prerecorded video-only content.
- [A 1.2.3 Audio Description or Media Alternative](https://www.w3.org/WAI/WCAG22/Understanding/audio-description-or-media-alternative-prerecorded.html) (<https://www.w3.org/WAI/WCAG22/Understanding/audio-description-or-media-alternative-prerecorded.html>) (Prerecorded): An alternative for time-based media [transcript] **or** audio description of the prerecorded video content is provided for synchronized media...

- [AA 1.2.5 Audio Description \(https://www.w3.org/WAI/WCAG22/Understanding/audio-description-prerecorded.html\)](https://www.w3.org/WAI/WCAG22/Understanding/audio-description-prerecorded.html) (Prerecorded): Audio description is provided for all prerecorded video content in synchronized media.
- [AAA 1.2.7 Extended Audio Description \(https://www.w3.org/WAI/WCAG22/Understanding/extended-audio-description-prerecorded.html\)](https://www.w3.org/WAI/WCAG22/Understanding/extended-audio-description-prerecorded.html) (Prerecorded): Where pauses in foreground audio are insufficient to allow audio descriptions to convey the sense of the video, extended audio description is provided for all prerecorded video content in synchronized media.

Description Considerations, Skills, and Tools

When accessibility is considered *before* videos are produced, it significantly cuts down on cost and effort to develop description. For some types of video (such as some training videos), description of the visual information can be seamlessly integrated by the speakers as the video is planned and created, and you don't need separate description, thus there is no additional cost.

Information on planning for description in *new videos* is in the "Creating Audio and Video Content" page, [Plan for Description of Visual Information section \(https://www.w3.org/WAI/media/av/av-content/#plan-description\)](https://www.w3.org/WAI/media/av/av-content/#plan-description).

To add description to *an existing video*, you'll either need skills and tools to:

- write it
- create a VTT file with the timed descriptions

Or:

- write it
- narrate it
- record it
- integrate it in new audio and/or video files

Many organizations choose to outsource their description.

What Method of Description?

What method to use for description depends on your video content and the media player that you use. First, figure out these issues about your video content, timing, and player:

- **Integrated** — For new videos, can the speakers describe the relevant visual information as the video is recorded? This works well for some videos, such as presentations and instructional videos. For examples, see the “Creating Audio and Video Content” page, [Integrated Description section \(https://www.w3.org/WAI/media/av/av-content/#integrate-description\)](https://www.w3.org/WAI/media/av/av-content/#integrate-description).
- **Media player support** — Information about media player functionality is in the Accessible Media Players page under [Existing Players \(https://www.w3.org/WAI/media/av/player/#existing-players\)](https://www.w3.org/WAI/media/av/player/#existing-players). Does the media player, platform, or plug-in that you are using provide functionality for:
 - description from a text file?
 - a separate audio track for description?
- **Space in audio** — Is there enough space in the main audio for the description? That is, are there sufficient pauses throughout the narration or speaking where the relevant description will fit? For example,
 - If the only description needed is at the beginning of the video where these is a text title and background music, then: Yes, there is enough space.
 - If the speaker talks continually without pausing, then: No, there is not enough space for description.

Use the information from above to answer the following questions in order to determine what method to provide description for your video. The options listed first are usually the best, yet you can choose another option.

Description method:

- Is it a new video *and* can the speakers describe the visual information in the main audio?
 - If yes, provide **integrated description** (no separate description is needed), **or** another option below.
 - If no, will you use a media player that provides functionality for description from a text file?
 - If yes, provide description in a **timed text file**, **or** another option below.
 - If no, will you use a media player that supports a separate audio track for the description, *and* is there enough space in the main audio for

the description?

- If yes, provide description in a **separate audio file**,
or provide a **separate described video**.
- If no, provide a **separate described video**.

Options for Creating Description

Depending on your video situation, do one of the options below — as determined from the “What Description to Provide for My Video?” section above.

Integrated – Creating Integrated Description

This approach works for some new videos. The process to develop a video with integrated description is basically:

1. When writing the script, make sure all relevant visual information is included. See [Tips for Writing Descriptions below](#)([#writing](#)) and examples in Accessible Audio and Video Content, [Integrate description section](#) (<https://www.w3.org/WAI/media/av/av-content/#integrate-description>).
2. Before finalizing the video, check to confirm that all relevant visual information is covered in the audio.

Text – Creating Description in a Text File

This approach only works when the media player that you’re using supports text-based description that is read aloud. And, either the description fits in the space of the main audio, or the player provides functionality to pause during description. It requires someone to create a timed text file — minimal skills are needed; tools are not required, yet, tools make it faster and easier.

The process to develop descriptions in a text file is basically:

1. Write out the descriptions. See [Tips for Writing Descriptions below](#)([#writing](#)).
2. Add the timestamps for the descriptions in the file format used by the media player. It is usually [WebVTT like the example below](#)([#vtt](#)).

(Make sure the descriptions file is included with the video.)

If the descriptions do not fit into the main audio space, provide instructions to users to set their player to pause the video during the description. For example:

To set the video to pause for description of visual information:

- Select “Preferences”, then “Descriptions”.
The “Description Preferences” box opens.
- Under “Text-based description”, select the checkbox for
“Automatically pause video when description starts”.
- Select the Save button.



Audio Only – Creating Description in a Separate Audio File Only

This approach only works when there is enough space in the main audio for the description, *and* the media player supports a separate audio track for the description. This requires skills and software for audio recording and audio editing.

The process to develop description in a separate audio file is basically:

1. Write out the descriptions. See [Tips for Writing Descriptions below](#)([#writing](#)).
2. Record the descriptions. See [Tips for Recording Descriptions below](#)([#recording](#)).
3. Ensure the descriptions play in the audio spaces with the main video.
4. Provide a caption file of the description. [Example VTT file of audio description](#)([#vtt](#))

(Ensure the files are integrated with the player on the web page.)

Video with Space – Creating a Separate Described Video - If Descriptions Fit in Audio Spaces

This applies if the descriptions do fit in the spaces, as described in [Space in audio above](#)([#space](#)). It requires skills and software for audio recording and video editing. Depending on the player that you are using, you might need video software to regenerate the video.

The process to develop a separate audio file is basically:

1. Write out the descriptions. See [Tips for Writing Descriptions below](#)([#writing](#)).

2. Record the descriptions. See [Tips for Recording Descriptions below](#)([#recording](#)).

3. Create a new audio file by combining the original audio and the new description audio. See [Tips for Combining Audio Files below](#)([#combining](#)).

4. Provide the file(s):

- **If** your player uses separate video and audio tracks,
you're done.
- **If** your player uses a single video file that includes the audio,
generate the new described video with the audio file that you just created.

(Make sure on the web page where the video is available, the described version uses the correct version that you just created.)

Video Without Space – Creating a Separate Described Video - If Descriptions Do Not Fit in Audio Spaces

If all the descriptions do **not** fit in the spaces (*as described in Space in audio above*([#space](#))), you'll need to develop a separate audio file and also edit the visual track. This requires skills and software for audio recording, audio editing, and video editing.

The process to develop a separate audio file and edit the visual track is basically:

1. Write out the descriptions. See [Tips for Writing Descriptions below](#)([#writing](#)).

2. Record the descriptions. See [Tips for Recording Descriptions below](#)([#recording](#)).

3. Create a new audio file by combining the original audio and the new description audio. See [Tips for Combining Audio Files below](#)([#combining](#)).

4. Create a new video:

- **If** you have source video with longer scenes (*as described in Accessible Audio and Video Content, Time for description section* (<https://www.w3.org/WAI/media/av/av-content/#time-for-description>)), recut the scenes longer to fill in the visual space where you need to accommodate the time for the description.
- **If not or if you're adding to an existing video**, you will need to leave a static image in the video while the description is playing in the audio.

(Make sure on the web page where the video is available, the described version uses the correct version that you just created.)

Other Options

- Extended Description with SMIL — The only markup-based method for providing extended audio descriptions is to use SMIL 3.0. Support for SMIL is very limited. Implementations would most likely require plug-ins and/or heavily customized approaches.
- Provide functionality for the video to pause for the description — This is not suggested in most cases because it requires extra coding and provides a less-than-optimum user experience.

Tips for Doing Description Yourself

VTT File

Example VTT file of audio description

WEBVTT

00:00:04.000 --> 00:00:07.980

<v Audio Descriptions>A man sitting at a desk starts watching a video on his computer.

00:00:17.260 --> 00:00:20.780

<v Audio Descriptions>The video on his computer shows a person speaking to the camera.

00:00:20.780 --> 00:00:23.140

<v Audio Descriptions>It is playing with no audio.

00:00:26.880 --> 00:00:29.620

<v Audio Descriptions>The man watching the video has a hearing aid.

Previous:

(<https://www.w3.org/WAI/media/av/description/>)

(<https://www.w3.org/WAI/media/av/description/>)

Next:

 Media Player[www.w3.org/WAI/
media/av/player/](https://www.w3.org/WAI/media/av/player/)[www.w3.org/WAI/
media/av/captions/](https://www.w3.org/WAI/media/av/captions/)Captions/Subtitles 

Updated: 17 September 2024. [Latest changes \(https://www.w3.org/WAI/media/av/changelog/\).](https://www.w3.org/WAI/media/av/changelog/)

First published September 2019.

Editor: [Shawn Lawton Henry](#). [Acknowledgements \(https://www.w3.org/WAI/media/av/acknowledgements/\)](https://www.w3.org/WAI/media/av/acknowledgements/) lists contributors and credits.

Developed by the Education and Outreach Working Group ([EOWG](#)). Originally drafted as part of the [WCAG TA Project \(https://www.w3.org/WAI/WCAGTA/\)](#) funded by the U.S. Access Board. Revised as part of the [WAI Expanding Access project \(https://www.w3.org/WAI/expand-access/\)](#) funded by the Ford Foundation.

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities. Copyright © 2025 World Wide Web Consortium ([W3C®](#)). See [Permission to Use WAI Material](#).



Designing for Web Accessibility

in [Tips for Getting Started](https://www.w3.org/WAI/tips/) (<https://www.w3.org/WAI/tips/>)

Summary

This page introduces some basic considerations to help you get started making your user interface design and visual design more accessible to people with disabilities. These tips are good practice to help you meet Web Content Accessibility Guidelines (WCAG) requirements. Follow the links to the related WCAG requirements, detailed background in the “Understanding” document, guidance from Tutorials, user stories, and more.

Page Contents

- [Provide sufficient contrast between foreground and background](#)([#provide-sufficient-contrast-between-foreground-and-background](#))
- [Don't use color alone to convey information](#)([#dont-use-color-alone-to-convey-information](#))
- [Ensure that interactive elements are easy to identify](#)([#ensure-that-interactive-elements-are-easy-to-identify](#))
- [Provide clear and consistent navigation options](#)([#provide-clear-and-consistent-navigation-options](#))
- [Ensure that form elements include clearly associated labels](#)([#ensure-that-form-elements-include-clearly-associated-labels](#))
- [Provide easily identifiable feedback](#)([#provide-easily-identifiable-feedback](#))
- [Use headings and spacing to group related content](#)([#use-headings-and-spacing-to-group-related-content](#))
- [Create designs for different viewport sizes](#)([#create-designs-for-different-viewport-sizes](#))
- [Include image and media alternatives in your design](#)([#include-image-and-media-alternatives-in-your-design](#))

- [Provide controls for content that starts automatically \(» #provide-controls-for-content-that-starts-automatically\)](#)

Provide sufficient contrast between foreground and background

Foreground text needs to have sufficient contrast with background colors. This includes text on images, background gradients, buttons, and other elements. This does not apply for logos, or incidental text, such as text that happens to be in a photograph. The links below provide more information on the minimum contrast ratio as required by the WCAG and how to check contrast. “Contrast ratio” is a short version of the more technically correct term “luminance contrast ratio”.

Example: Contrast ratio

Insufficient

Some people cannot read text if there is not sufficient contrast between the text and background. For others, bright colors (high luminance) are not readable; they need low luminance.

Sufficient

Some people cannot read text if there is not sufficient contrast between the text and background. For others, bright colors (high luminance) are not readable; they need low luminance.

More Information

- **WCAG**
 - [Contrast \(Minimum\) 1.4.3 \(<https://www.w3.org/WAI/WCAG21/quickref/#contrast-minimum>\) \(Understanding 1.4.3 \(<https://www.w3.org/WAI/WCAG21/Understanding/contrast-minimum>\)\)](#)
- **User Stories**
 - [Lexie, online shopper who cannot distinguish between certain colors \(color blindness\) \(<https://www.w3.org/WAI/people-use-web/user-stories/story-four/>\)](#)
 - [Elias, retiree with low vision, hand tremor, and mild short-term memory loss](#)

(<https://www.w3.org/WAI/people-use-web/user-stories/story-nine/>)

- **Easy Check**

- [How to check contrast ratio \(https://www.w3.org/WAI/test-evaluate/preliminary/#contrast\)](https://www.w3.org/WAI/test-evaluate/preliminary/#contrast)

- **Support Tools**

- [List of tools to help determine contrast ratio](#)

Don't use color alone to convey information

While color can be useful to convey information, color should not be the only way information is conveyed. When using color to differentiate elements, also provide additional identification that does not rely on color perception. For example, use an asterisk in addition to color to indicate required form fields, and use labels to distinguish areas on graphs.

Example: Using color to convey meaning

Color only

Required fields are in red

Name

Email

Color and symbol

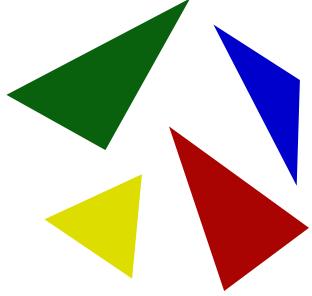
Required fields are in red and marked with an *

Name

Email *

Example: Refer to something using color alone

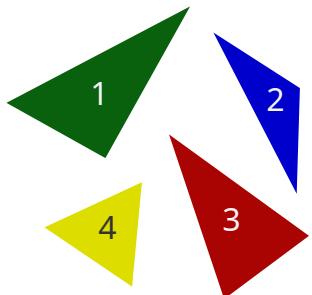
Color only



Which is the right-angled triangle?

- Green
- Blue
- Red
- Yellow
- Don't know

Color and number



Which is the right-angled triangle?

- Green (1)
- Blue (2)
- Red (3)
- Yellow (4)
- Don't know

More Information

- **WCAG**
 - [Use of Color 1.4.1](https://www.w3.org/WAI/WCAG21/quickref/#use-of-color) (<https://www.w3.org/WAI/WCAG21/quickref/#use-of-color>)
[Understanding 1.4.1](https://www.w3.org/WAI/WCAG21/Understanding/use-of-color) (<https://www.w3.org/WAI/WCAG21/Understanding/use-of-color>)
- **User Story**
 - [Lexie, online shopper who cannot distinguish between certain colors \(color blindness\)](https://www.w3.org/WAI/people-use-web/user-stories/story-four/) (<https://www.w3.org/WAI/people-use-web/user-stories/story-four/>)

Ensure that interactive elements are easy to identify

Provide distinct styles for interactive elements, such as links and buttons, to make them easy to identify. For example, change the appearance of links on mouse hover, keyboard focus, and touch-screen activation. Ensure that styles and naming for interactive elements are used consistently throughout the website.

Example: Unique styles for different link states

Style links to stand out from text

Some people can't use a mouse and use only a [keyboard to navigate](#) through web pages.

It is important that users can reach all interactive elements using the keyboard, and that it is clear which element has focus.

Visible keyboard focus could be a border or highlight that moves as you tab through the web page.

Mouse hover style

[keyboard to navigate](#)



Keyboard focus style

[keyboard to navigate](#)



Touch or click style

[keyboard to navigate](#)



More Information

• WCAG

- Focus Visible 2.4.7 (<https://www.w3.org/WAI/WCAG21/quickref/#focus-visible>)
([Understanding 2.4.7](https://www.w3.org/WAI/WCAG21/Understanding/#focus-visible) (<https://www.w3.org/WAI/WCAG21/Understanding/#focus-visible>))
- Consistent Identification 3.2.4 (<https://www.w3.org/WAI/WCAG21/quickref/#consistent-identification>)
([Understanding 3.2.4](https://www.w3.org/WAI/Understanding/3.2.4) (<https://www.w3.org/WAI/Understanding/3.2.4>))

WCAG21/Understanding/consistent-identification))

- **User Stories**

- Ade, reporter with limited use of his arms (<https://www.w3.org/WAI/people-use-web/user-stories/story-one/>)
- Ian, data entry clerk with autism (<https://www.w3.org/WAI/people-use-web/user-stories/story-two/>)
- Lexie, online shopper who cannot distinguish between certain colors (color blindness) (<https://www.w3.org/WAI/people-use-web/user-stories/story-four/>)
- Sophie, basketball fan with Down syndrome (<https://www.w3.org/WAI/people-use-web/user-stories/story-five/>)
- Stefan, student with attention deficit hyperactivity disorder and dyslexia (<https://www.w3.org/WAI/people-use-web/user-stories/story-eight/>)
- Elias, retiree with low vision, hand tremor, and mild short-term memory loss (<https://www.w3.org/WAI/people-use-web/user-stories/story-nine/>)

Provide clear and consistent navigation options

Ensure that navigation across pages within a website has consistent naming, styling, and positioning. Provide more than one method of website navigation, such as a site search or a site map. Help users understand where they are in a website or page by providing orientation cues, such as breadcrumbs and clear headings.

More Information

- **WCAG**

- Consistent Navigation 3.2.3 (<https://www.w3.org/WAI/WCAG21/quickref/#consistent-navigation>) (Understanding 3.2.3 (<https://www.w3.org/WAI/WCAG21/Understanding/consistent-identification>))
- Multiple Ways 2.4.5 (<https://www.w3.org/WAI/WCAG21/quickref/#multiple-ways>) (Understanding 2.4.5 (<https://www.w3.org/WAI/WCAG21/Understanding/multiple-ways>))

- **User Stories**

- Ian, data entry clerk with autism (<https://www.w3.org/WAI/people-use-web/user-stories/story-two/>)

- [Sophie, basketball fan with Down syndrome \(https://www.w3.org/WAI/people-use-web/user-stories/story-five/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-five/)
- [Marta, marketing assistant who is deaf and blind \(https://www.w3.org/WAI/people-use-web/user-stories/story-seven/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-seven/)
- [Elias, retiree with low vision, hand tremor, and mild short-term memory loss \(https://www.w3.org/WAI/people-use-web/user-stories/story-nine/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-nine/)

Ensure that form elements include clearly associated labels

Ensure that all fields have a descriptive label adjacent to the field. For left-to-right languages, labels are usually positioned to the left or above the field, except for checkboxes and radio buttons where they are usually to the right. Avoid having too much space between labels and fields.

✓ Example: Labels and input fields associated by proximity

Add a comment

Your E-mail

I am happy for you to contact me

Your Website

Comment

More Information

- WCAG

- [Labels or Instructions 3.3.2 \(https://www.w3.org/WAI/WCAG21/quickref/#labels-or-instructions\)](https://www.w3.org/WAI/WCAG21/quickref/#labels-or-instructions) ([Understanding 3.3.2 \(https://www.w3.org/WAI/Understanding_3.3.2\)](https://www.w3.org/WAI/Understanding_3.3.2))

WCAG21/Understanding/labels-or-instructions))

- Headings and Labels 2.4.6 (<https://www.w3.org/WAI/WCAG21/quickref/#headings-and-labels>) (Understanding 2.4.6 (<https://www.w3.org/WAI/WCAG21/Understanding/headings-and-labels>))
- **Tutorial**
 - Visual position of label text (<https://www.w3.org/WAI/tutorials/forms/labels/#visual-position-of-label-text>)

Provide easily identifiable feedback

Provide feedback for interactions, such as confirming form submission, alerting the user when something goes wrong, or notifying the user of changes on the page. Instructions should be easy to identify. Important feedback that requires user action should be presented in a prominent style.

✓ Example: Using error list, icon, and background color to make errors stand out

Please correct the following errors:

1.  Email address is invalid
2.  A Comment is required

Add a comment

Required fields are in red and marked with an *

Name

Superbear

 E-mail *

superbear@hq.example.com

Website

 Comment *

More Information

- **WCAG**
 - [Error Identification 3.3.1 \(https://www.w3.org/WAI/WCAG21/quickref/#error-identification\)](https://www.w3.org/WAI/WCAG21/quickref/#error-identification) ([Understanding 3.3.1 \(https://www.w3.org/WAI/WCAG21/Understanding/error-identification\)\)](https://www.w3.org/WAI/WCAG21/Understanding/error-identification)
 - [Labels or Instructions 3.3.2 \(https://www.w3.org/WAI/WCAG21/quickref/#labels-or-instructions\)](https://www.w3.org/WAI/WCAG21/quickref/#labels-or-instructions) ([Understanding 3.3.2 \(https://www.w3.org/WAI/WCAG21/Understanding/labels-or-instructions\)\)](https://www.w3.org/WAI/WCAG21/Understanding/labels-or-instructions)
 - [Error Suggestion 3.3.3 \(https://www.w3.org/WAI/WCAG21/quickref/#error-suggestion\)](https://www.w3.org/WAI/WCAG21/quickref/#error-suggestion) ([Understanding 3.3.3 \(https://www.w3.org/WAI/WCAG21/Understanding/error-suggestion\)\)](https://www.w3.org/WAI/WCAG21/Understanding/error-suggestion)
- **Tutorial**
 - [User Notifications \(https://www.w3.org/WAI/tutorials/forms/notifications/\)](https://www.w3.org/WAI/tutorials/forms/notifications/)
- **User Stories**
 - [Ian, data entry clerk with autism \(https://www.w3.org/WAI/people-use-web/user-stories/story-two/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-two/)
 - [Stefan, student with attention deficit hyperactivity disorder and dyslexia \(https://www.w3.org/WAI/people-use-web/user-stories/story-eight/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-eight/)
 - [Elias, retiree with low vision, hand tremor, and mild short-term memory loss \(https://www.w3.org/WAI/people-use-web/user-stories/story-nine/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-nine/)

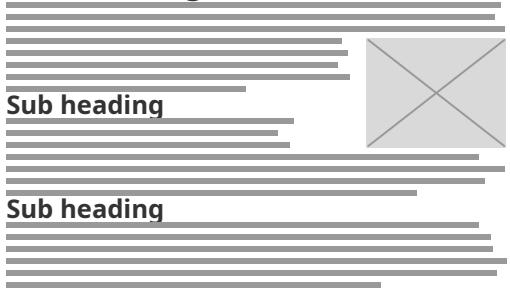
Use headings and spacing to group related content

Use whitespace and proximity to make relationships between content more apparent. Style headings to group content, reduce clutter, and make it easier to scan and understand.

Example: Spacing highlights relationship between content

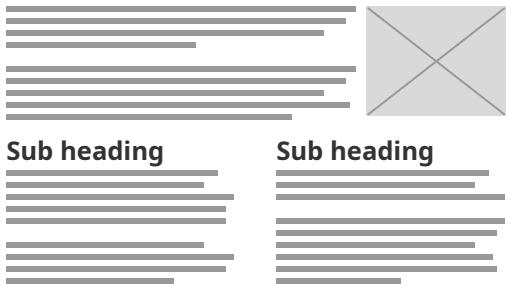
✖ Little spacing and unclear relationship

Main heading



More spacing and clearer relationship

Main heading



More Information

- **WCAG**

- [Headings and Labels 2.4.6 \(https://www.w3.org/WAI/WCAG21/quickref/#headings-and-labels\)](https://www.w3.org/WAI/WCAG21/quickref/#headings-and-labels) ([Understanding 2.4.6 \(https://www.w3.org/WAI/WCAG21/Understanding/headings-and-labels\)](https://www.w3.org/WAI/WCAG21/Understanding/headings-and-labels))
- [Section Headings 2.4.10 \(https://www.w3.org/WAI/WCAG21/quickref/#section-headings\)](https://www.w3.org/WAI/WCAG21/quickref/#section-headings) ([Understanding 2.4.10 \(https://www.w3.org/WAI/WCAG21/Understanding/section-headings\)](https://www.w3.org/WAI/WCAG21/Understanding/section-headings))

- **Tutorial**

- [Headings \(https://www.w3.org/WAI/tutorials/page-structure/headings/\)](https://www.w3.org/WAI/tutorials/page-structure/headings/)

- **User Stories**

- [Stefan, student with attention deficit hyperactivity disorder and dyslexia \(https://www.w3.org/WAI/people-use-web/user-stories/story-eight/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-eight/)
- [Elias, retiree with low vision, hand tremor, and mild short-term memory loss \(https://www.w3.org/WAI/people-use-web/user-stories/story-nine/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-nine/)

Create designs for different viewport sizes

Consider how page information is presented in different sized viewports, such as mobile phones or zoomed browser windows. Position and presentation of main elements, such as header and navigation can be changed to make best use of the space. Ensure that text size and line width are set to maximize readability and legibility.

Example: Content and navigation adapt to smaller mobile screen

The screenshot illustrates the responsive design of the W3C Web Accessibility Tutorials. On the desktop version, the layout includes:

- A top navigation bar with back/forward buttons, a refresh icon, and a search bar.
- The W3C logo and "Web Accessibility initiative" text.
- A sidebar on the left containing a navigation menu:
 - Images Tutorial
 - Images Concepts (selected)
 - 1 Informative Images
 - 2 Decorative Images
 - 3 Functional Images
 - 4 Images of Text
 - 5 Complex Images
 - 6 Groups of Images
 - 7 Image Maps
 - An alt Decision Tree
 - Tips and Tricks
- The main content area showing the "Images Concepts" page with the following text:

Images must have text alternatives that describe the information or function represented by the images. This ensures that images can be used by people with various disabilities. This tutorial demonstrates how to provide appropriate text alternatives based on the purpose of the image:

 - Informative Images:** Images that graphically represent concepts and information, typically pictures, photos and illustrations. The text alternative should be at least a short description conveying the essential information presented by the image.
- A "On this page" sidebar with links to "Why is this important?" and "Related WCAG 2.0 resources".
- A "Technologies covered in this Tutorial:" section listing CSS Fonts, CSS Transforms, HTML5, MathML, and WAI-ARIA.

On the mobile version, the layout is simplified:

- The W3C logo and "Web Accessibility initiative" text.
- The "Web Accessibility Tutorials" header and "Guidance on how to create websites that meet WCAG" text.
- A "Jump to the navigation" button.
- The main content area showing the "Images Concepts" page with the same text as the desktop version.
- A "On this page" sidebar with links to "Why is this important?" and "Related WCAG 2.0 resources".
- A "Technologies covered in this Tutorial:" section listing CSS Fonts.

Display in a wide window with small text uses multiple columns for primary content, visible navigation options, and visible secondary information.

Display in a narrow window, such as a mobile phone, or with large text uses single column for primary content, navigation options are revealed using an icon, and secondary information is also revealed via icon.

More Information

- **Background**
 - [Small Screen Size](#)
 - [Mobile considerations related to Understandability](#)

Include image and media alternatives in your design

Provide a place in your design for alternatives for images and media. For example, you might need:

- Visible links to transcripts of audio
- Visible links to audio described versions of videos
- Text along with icons and graphical buttons
- Captions and descriptions for tables or complex graphs

Work with content authors and developers to provide alternatives for non-text content.

Example: Design includes links to a transcript and to an audio described video



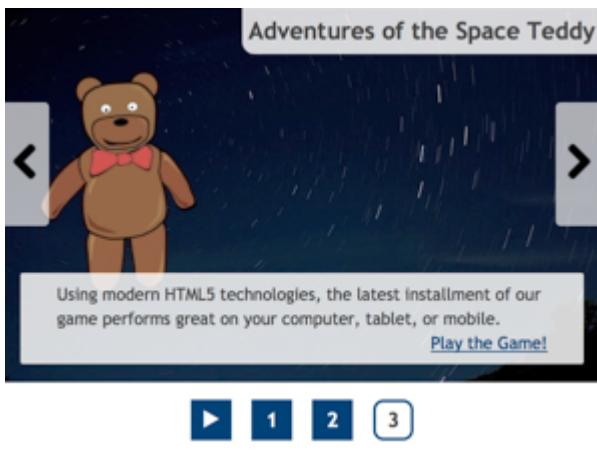
More Information

- **WCAG**
 - [Non-text Content 1.1.1 \(https://www.w3.org/WAI/WCAG21/quickref/#non-text-content\)](https://www.w3.org/WAI/WCAG21/quickref/#non-text-content) ([Understanding 1.1.1 \(https://www.w3.org/WAI/WCAG21/Understanding/non-text-content\)](https://www.w3.org/WAI/WCAG21/Understanding/non-text-content))
- **Tutorial**
 - [Images \(https://www.w3.org/WAI/tutorials/images/\)](https://www.w3.org/WAI/tutorials/images/)
- **User Story**
 - [Dhruv, older adult student who is deaf \(https://www.w3.org/WAI/people-use-web/user-stories/story-six/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-six/)

Provide controls for content that starts automatically

Provide visible controls to allow users to stop any animations or auto-playing sound. This applies to carousels, image sliders, background sound, and videos.

Example: Show play/stop and selection controls in carousel design



More Information

- **WCAG**
 - [Audio Control 1.4.2 \(https://www.w3.org/WAI/WCAG21/quickref/#audio-control\)](https://www.w3.org/WAI/WCAG21/quickref/#audio-control) ([Understanding 1.4.2 \(https://www.w3.org/WAI/WCAG21/Understanding/audio-control\)](https://www.w3.org/WAI/WCAG21/Understanding/audio-control))

- [Pause, Stop, Hide 2.2.2 \(https://www.w3.org/WAI/WCAG21/quickref/#pause-stop-hide\)](https://www.w3.org/WAI/WCAG21/quickref/#pause-stop-hide) ([Understanding 2.2.2 \(https://www.w3.org/WAI/WCAG21/Understanding/pause-stop-hide\)](https://www.w3.org/WAI/WCAG21/Understanding/pause-stop-hide))

- **Tutorial**

- [Carousel Concepts \(https://www.w3.org/WAI/tutorials/carousels/\)](https://www.w3.org/WAI/tutorials/carousels/)

- **User Story**

- [Dhruv, older adult student who is deaf \(https://www.w3.org/WAI/people-use-web/user-stories/story-six/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-six/)

Learn More About Accessibility

These tips are a few of the things you need to consider for web accessibility. The following resources help you learn why accessibility is important, and about guidelines for making the web more accessible to people with disabilities.

- [Introduction to Web Accessibility \(https://www.w3.org/WAI/fundamentals/accessibility-intro/\)](https://www.w3.org/WAI/fundamentals/accessibility-intro/) — covers broad issues, such as the business case, and links to helpful resources
- [Accessibility Principles \(https://www.w3.org/WAI/fundamentals/accessibility-principles/\)](https://www.w3.org/WAI/fundamentals/accessibility-principles/) — introduces the concepts behind the web accessibility requirements
- [How people with disabilities use the web \(https://www.w3.org/WAI/people-use-web/\)](https://www.w3.org/WAI/people-use-web/) — explores the impact of accessible design with real-life examples
- [Web Accessibility Tutorials \(https://www.w3.org/WAI/tutorials/\)](https://www.w3.org/WAI/tutorials/) — includes some guidance related to designing, for example, [providing alternative text for images \(https://www.w3.org/WAI/tutorials/images/\)](https://www.w3.org/WAI/tutorials/images/)
- [Before and After Demonstration \(https://www.w3.org/WAI/demos/bad/\)](https://www.w3.org/WAI/demos/bad/) — shows an inaccessible and accessible version of the same website, with annotations on accessibility barriers and repairs
- [How to Meet WCAG \(Quick Reference\) \(https://www.w3.org/WAI/WCAG21/quickref/\)](https://www.w3.org/WAI/WCAG21/quickref/) — customizable reference of all WCAG requirements and techniques
- [Web Accessibility Evaluation Tools List \(https://www.w3.org/WAI/ER/tools/\)](https://www.w3.org/WAI/ER/tools/) — includes tools to help explore contrast ratio

[tips/writing/](#)[tips/developing/](#)

Updated: 16 July 2024. [Latest changes \(https://www.w3.org/WAI/tips/changelog/\).](https://www.w3.org/WAI/tips/changelog/)

Date: Minor update 16 July 2024. Updated 16 July 2024. First published September 2015.

Editors: [Kevin White](https://www.w3.org/People/kevin) (<https://www.w3.org/People/kevin>), [Shadi Abou-Zahra](https://www.w3.org/People/shadi) (<https://www.w3.org/People/shadi>), and [Shawn Lawton Henry](https://www.w3.org/People/Shawn) (<https://www.w3.org/People/Shawn>). [Acknowledgements \(https://www.w3.org/WAI/tips/acknowledgements/\).](https://www.w3.org/WAI/tips/acknowledgements/)

Developed by the [Education and Outreach Working Group \(EOWG\)](https://www.w3.org/WAI/EO/) (<https://www.w3.org/WAI/EO/>). Developed with support from the [WAI-DEV project](https://www.w3.org/WAI/DEV/) (<https://www.w3.org/WAI/DEV/>), co-funded by the European Commission [IST Programme](#).

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities. Copyright © 2025 World Wide Web Consortium ([W3C®](#)). See [Permission to Use WAI Material](#).



</> Developing for Web Accessibility

in [Tips for Getting Started](#) (<https://www.w3.org/WAI/tips/>)

Summary

This page introduces some basic considerations to help you get started developing web content that is more accessible to people with disabilities. These tips are good practice to help you meet Web Content Accessibility Guidelines (WCAG) requirements. Follow the links to the related WCAG requirements, detailed background in the “Understanding” document, guidance from Tutorials, user stories, and more.

Page Contents

- [Associate a label with every form control](#)(`\#associate-a-label-with-every-form-control`)
- [Include alternative text for images](#)(`\#include-alternative-text-for-images`)
- [Identify page language and language changes](#)(`\#identify-page-language-and-language-changes`)
- [Use mark-up to convey meaning and structure](#)(`\#use-mark-up-to-convey-meaning-and-structure`)
- [Help users avoid and correct mistakes](#)(`\#help-users-avoid-and-correct-mistakes`)
- [Reflect the reading order in the code order](#)(`\#reflect-the-reading-order-in-the-code-order`)
- [Write code that adapts to the user's technology](#)(`\#write-code-that-adapts-to-the-users-technology`)
- [Provide meaning for non-standard interactive elements](#)(`\#provide-meaning-for-non-standard-interactive-elements`)
- [Ensure that all interactive elements are keyboard accessible](#)(`\#ensure-that-all-interactive-elements-are-keyboard-accessible`)
- [Avoid CAPTCHA where possible](#)(`\#avoid-captcha-where-possible`)

Associate a label with every form control

Use a **for** attribute on the `<label>` element linked to the **id** attribute of the form element, or using WAI-ARIA attributes. In specific situations it may be acceptable to hide `<label>` elements visually, but in most cases labels are needed to help all readers understand the required input.

Example: Using **for** and **id** attributes

Rendered

Username

Code Snippet

```
<label for="username">Username</label>
<input id="username" type="text" name="username">
```

More Information

- WCAG
 - [Labels or Instructions 3.3.2 \(https://www.w3.org/WAI/WCAG21/quickref/#labels-or-instructions\)](https://www.w3.org/WAI/WCAG21/quickref/#labels-or-instructions) ([Understanding 3.3.2 \(https://www.w3.org/WAI/WCAG21/Understanding/labels-or-instructions\)](https://www.w3.org/WAI/WCAG21/Understanding/labels-or-instructions))
- Tutorial
 - [Labeling Controls \(https://www.w3.org/WAI/tutorials/forms/labels/\)](https://www.w3.org/WAI/tutorials/forms/labels/)

Include alternative text for images

Ensure that alternative text for images is added to all informational and functional images. Use empty alternative text, `alt=""` for decorative images, or include them in the CSS instead. Text alternatives are usually provided by those responsible for written content.

More Information

- WCAG
 - [Non-text Content 1.1.1 \(https://www.w3.org/WAI/WCAG21/quickref/#non-text-content\)](https://www.w3.org/WAI/WCAG21/quickref/#non-text-content) ([Understanding 1.1.1 \(https://www.w3.org/WAI/WCAG21/Understanding/non-text-content\)](https://www.w3.org/WAI/WCAG21/Understanding/non-text-content))

- **Tutorial**
 - [Images \(https://www.w3.org/WAI/tutorials/images/\)](https://www.w3.org/WAI/tutorials/images/)
- **User Story**
 - [Stefan, student with attention deficit hyperactivity disorder and dyslexia \(https://www.w3.org/WAI/people-use-web/user-stories/story-eight/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-eight/)

Identify page language and language changes

Indicate the primary language of every page by using the `lang` attribute in the `html` tag, for example `<html lang="en">`. Use the `lang` attribute on specific elements when the language of the element differs from the rest of the page.

More Information

- **WCAG**
 - [Language of Page 3.1.1 \(https://www.w3.org/WAI/WCAG21/quickref/#language-of-page\) \(Understanding 3.1.1 \(https://www.w3.org/WAI/WCAG21/Understanding/language-of-page\)\)](https://www.w3.org/WAI/WCAG21/quickref/#language-of-page)
 - [Language of Parts 3.1.2 \(https://www.w3.org/WAI/WCAG21/quickref/#language-of-parts\) \(Understanding 3.1.2 \(https://www.w3.org/WAI/WCAG21/Understanding/language-of-parts\)\)](https://www.w3.org/WAI/WCAG21/quickref/#language-of-parts)
- **How To**
 - [Declaring language in HTML](#)

Use mark-up to convey meaning and structure

Use appropriate mark-up for headings, lists, tables, etc. HTML5 provides additional elements, such as `<nav>` and `<aside>`, to better structure your content. WAI-ARIA roles can provide additional meaning, for example, using `role="search"` to identify search functionality. Work with designers and content writers to agree on meanings and then use them consistently.

Example: Using HTML to provide structure and meaning

Rendered

Superbear saves the day

7 Aug 2015

The city's favorite bear yet again proves his mettle by rescuing a young cat from a tree. Witnesses say that Superbear's efforts were not appreciated by the feline, who inflicted some minor scratch wounds on his rescuer.

Related Articles

- [Bear receives key to city](#)(§ #)
- [Superbear stands for mayor](#)(§ #)

Code Snippet

```
<section>
  <article>
    <h2>Superbear saves the day</h2>
    <time datetime="2015-08-07">7 Aug 2015</time>
    <p>The city's favorite bear yet again proves his mettle by rescuing a young cat from a tree. Witnesses say that Superbear's efforts were not appreciated by the feline, who inflicted some minor scratch wounds on his rescuer.</p>
    <aside>
      <h3>Related Articles</h3>
      <ul>
        <li><a href="#">Bear receives key to city</a></li>
        <li><a href="#">Superbear stands for mayor</a></li>
      </ul>
    </aside>
  </article>
</section>
```

Example: Search field using WAI-ARIA

Rendered

Search for

Search records by customer id or name

Go

</> Code Snippet

```
<form action="#" method="post">
  <div role="search">
    <label for="search">Search for</label>
    <input type="search" id="search" aria-describedby="search-help">
    <div id="search-help">Search records by customer id or name</div>
    <button type="submit">Go</button>
  </div>
</form>
```

More Information

- **WCAG**
 - [Info and Relationships 1.3.1 \(<https://www.w3.org/WAI/WCAG21/quickref/#info-and-relationships>\)](https://www.w3.org/WAI/WCAG21/quickref/#info-and-relationships) ([Understanding 1.3.1 \(<https://www.w3.org/WAI/WCAG21/Understanding/info-and-relationships>\)](https://www.w3.org/WAI/WCAG21/Understanding/info-and-relationships))
- **Tutorial**
 - [Page Structure \(<https://www.w3.org/WAI/tutorials/page-structure/>\)](https://www.w3.org/WAI/tutorials/page-structure/)
 - [Tables \(<https://www.w3.org/WAI/tutorials/tables/>\)](https://www.w3.org/WAI/tutorials/tables/)

Help users avoid and correct mistakes

Provide clear instructions, error messages, and notifications to help users complete forms on your site. When an error occurs:

- Help users find where the problem is
- Provide specific, understandable explanations
- Suggest corrections

Be as forgiving of format as possible when processing user input. For example, accept phone numbers that include spaces and delete the spaces as needed.

Example: Australian phone number field with forgiving validation

Rendered

Phone

For example, (02) 1234 1234

</> Code Snippet

```
<label for="phone">Phone</label>
<input id="phone" name="phone" type="tel"
       pattern="^(\(?0[1-9]{1}\)?)?[0-9 -]*$"
       aria-describedby="phone-desc">
<p id="phone-desc">For example, (02) 1234 1234</p>
```

More Information

- **WCAG**
 - [Error Identifications 3.3.1 \(https://www.w3.org/WAI/WCAG21/quickref/#error-identification\)](https://www.w3.org/WAI/WCAG21/quickref/#error-identification) ([Understanding 3.3.1 \(https://www.w3.org/WAI/WCAG21/Understanding/error-identification\)](https://www.w3.org/WAI/WCAG21/Understanding/error-identification))
- **Tutorial**
 - [Validating Input \(https://www.w3.org/WAI/tutorials/forms/validation/\)](https://www.w3.org/WAI/tutorials/forms/validation/)
- **User Stories**
 - [Ian, data entry clerk with autism \(https://www.w3.org/WAI/people-use-web/user-stories/story-two/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-two/)
 - [Stefan, student with attention deficit hyperactivity disorder and dyslexia \(https://www.w3.org/WAI/people-use-web/user-stories/story-eight/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-eight/)
 - [Elias, retiree with low vision, hand tremor, and mild short-term memory loss \(https://www.w3.org/WAI/people-use-web/user-stories/story-nine/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-nine/)

Reflect the reading order in the code order

Ensure that the order of elements in the code matches the logical order of the information presented. One way to check this is to remove CSS styling and review that the order of the content makes sense.

Example: Reflecting the logical reading order in the code



Space trainers

Space trainer for a classic and stylish look.

 [Add to cart](#)

Image before heading may be missed

```

<h3>Space trainers</h3>
<p>Space...</p>
<a href="...">Add to cart</a>
```

 [View complete code example](#)

Heading marks the start of the section

```
<h3>Space trainers</h3>

<p>Space...</p>
<a href="...">Add to cart</a>
```

 [View complete code example](#)

More Information

- **WCAG**

- [Meaningful Sequence 1.3.2 \(https://www.w3.org/WAI/WCAG21/quickref/#meaningful-sequence\) \(Understanding 1.3.2 \(https://www.w3.org/WAI/WCAG21/Understanding/meaningful-sequence\)\)](https://www.w3.org/WAI/WCAG21/quickref/#meaningful-sequence)

Write code that adapts to the user's technology

Use responsive design to adapt the display to different zoom states and viewport sizes, such as on mobile devices and tablets. When font size is increased by at least 200%, avoid

horizontal scrolling and prevent any clipping of content. Use progressive enhancement to help ensure that core functionality and content is available regardless of technology being used.

Example: Using media queries to adapt navigation

```
/* On narrow viewports, make the navigation full width */
@media screen and (min-width: 25em) {
    #nav {
        float: none;
        width: auto;
    }
    #main {
        margin-left: 0;
    }
}
/* On wider viewports, put the navigation on the left */
@media screen and (min-width: 43em) {
    #nav {
        float: left;
        width: 24%;
    }
    #main {
        margin-left: 27%;
    }
}
```

More Information

- **WCAG**

- Resize text 1.4.4 (<https://www.w3.org/WAI/WCAG21/quickref/#resize-text>)
(Understanding 1.4.4 (<https://www.w3.org/WAI/WCAG21/Understanding/resize-text>))
- Consistent Identification 3.2.4 (<https://www.w3.org/WAI/WCAG21/quickref/#consistent-identification>)
(Understanding 3.2.4 (<https://www.w3.org/WAI/WCAG21/Understanding/consistent-identification>))

- **Background**

- Small Screen Size

- **User Story**

- Dhruv, older adult student who is deaf (<https://www.w3.org/WAI/people-use-web/user-stories/story-six/>)

Provide meaning for non-standard interactive elements

Use WAI-ARIA to provide information on function and state for custom widgets, such as accordions and custom-made buttons. For example, `role="navigation"` and `aria-expanded="true"`. Additional code is required to implement the behavior of such widgets, such as expanding and collapsing content or how the widget responds to keyboard events.

Example: Menu function and state identified using WAI-ARIA

```
<nav aria-label="Main Navigation" role="navigation">
  <ul>
    <li><a href="...">Home</a></li>
    <li><a href="...">Shop</a></li>
    <li class="has-submenu">
      <a aria-expanded="false" aria-haspopup="true"
 href="...">SpaceBears</a>
      <ul>
        <li><a href="...">SpaceBear 6</a></li>
        <li><a href="...">SpaceBear 6 Plus</a></li>
      </ul>
    </li>
    <li><a href="...">MarsCars</a></li>
    <li><a href="...">Contact</a></li>
  </ul>
</nav>
```

More Information

- **WCAG**

- Name, Role, Value 4.1.2 (<https://www.w3.org/WAI/WCAG21/quickref/#name-role-value>) Understanding 4.1.2 (<https://www.w3.org/WAI/WCAG21/>)

Understanding/name-role-value))

- **Background**

- Notes on Using ARIA in HTML

- **User Stories**

- Sophie, basketball fan with Down syndrome (<https://www.w3.org/WAI/people-use-web/user-stories/story-five/>)
- Elias, retiree with low vision, hand tremor, and mild short-term memory loss (<https://www.w3.org/WAI/people-use-web/user-stories/story-nine/>)

Ensure that all interactive elements are keyboard accessible

Think about keyboard access, especially when developing interactive elements, such as menus, mouseover information, collapsable accordions, or media players. Use `tabindex="0"` to add an element that does not normally receive focus, such as `<div>` or ``, into the navigation order when it is being used for interaction. Use scripting to capture and respond to keyboard events.

Example: Keyboard accessible menu button

Rendered

 Menu

`</> Code Snippet`

```
var buttonExample = document.getElementById('example-button');

buttonExample.addEventListener('keydown', function(e) {
    // Toggle the menu when RETURN is pressed
    if(e.keyCode && e.keyCode == 13) {
        toggleMenu(document.getElementById('example-button-menu'));
    }
});

buttonExample.addEventListener('click', function(e) {
```

```
// Toggle the menu on mouse click
toggleMenu(document.getElementById('example-button-menu'));
});
```

More Information

- **WCAG**

- [Keyboard 2.1.1 \(https://www.w3.org/WAI/WCAG21/quickref/#keyboard\)](https://www.w3.org/WAI/WCAG21/quickref/#keyboard)
[Understanding 2.1.1 \(https://www.w3.org/WAI/WCAG21/Understanding/keyboard\)](https://www.w3.org/WAI/WCAG21/Understanding/keyboard)

- **User Stories**

- [Ade, reporter with limited use of his arms \(https://www.w3.org/WAI/people-use-web/user-stories/story-one/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-one/)
- [Lakshmi, senior accountant who is blind \(https://www.w3.org/WAI/people-use-web/user-stories/story-three/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-three/)
- [Marta, marketing assistant who is deaf and blind \(https://www.w3.org/WAI/people-use-web/user-stories/story-seven/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-seven/)
- [Elias, retiree with low vision, hand tremor, and mild short-term memory loss \(https://www.w3.org/WAI/people-use-web/user-stories/story-nine/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-nine/)

Avoid CAPTCHA where possible

CAPTCHAs create problems for many people. There are other means of verifying that user input was generated by a human that are easier to use, such as automatic detection or interface interactions. If CAPTCHA really needs to be included, ensure that it is simple to understand and includes alternatives for users with disabilities, such as:

- Providing more than two ways to solve the CAPTCHAs
- Providing access to a human representative who can bypass CAPTCHA
- Not requiring CAPTCHAs for authorized users.

More Information

- **WCAG**

- [Non-text Content 1.1.1 \(https://www.w3.org/WAI/WCAG21/quickref/#non-text-content\)](https://www.w3.org/WAI/WCAG21/quickref/#non-text-content) ([Understanding 1.1.1 \(https://www.w3.org/WAI/WCAG21/Understanding/1.1.1\)](https://www.w3.org/WAI/WCAG21/Understanding/1.1.1))

Understanding/non-text-content))

- **Background**

- Inaccessibility of CAPTCHA

- **User Stories**

- Marta, marketing assistant who is deaf and blind (<https://www.w3.org/WAI/people-use-web/user-stories/story-seven/>)
- Elias, retiree with low vision, hand tremor, and mild short-term memory loss (<https://www.w3.org/WAI/people-use-web/user-stories/story-nine/>)

Learn More About Accessibility

These tips are a few of the things you need to consider for web accessibility. The following resources help you learn why accessibility is important, and about guidelines for making the web more accessible to people with disabilities.

- Introduction to Web Accessibility (<https://www.w3.org/WAI/fundamentals/accessibility-intro/>) — Introduces accessibility and provides links to many helpful resources
- Accessibility Principles (<https://www.w3.org/WAI/fundamentals/accessibility-principles/>) — An introduction to the WCAG requirements
- How people with disabilities use the web (<https://www.w3.org/WAI/people-use-web/>) — Real-life examples of the benefits of accessibility for people with disabilities
- Web Accessibility Tutorials (<https://www.w3.org/WAI/tutorials/>) — Shows you how to develop web content that is accessible to people with disabilities
- Before and After Demonstration (<https://www.w3.org/WAI/demos/bad/>) — Example accessible and inaccessible websites that share the same visual design, with annotations that highlight key accessibility barriers and repairs, and evaluation reports for WCAG
- How to Meet WCAG (Quick Reference) (<https://www.w3.org/WAI/WCAG21/quickref/>) — customizable reference of all WCAG requirements and techniques
- Web Accessibility Evaluation Tools List (<https://www.w3.org/WAI/ER/tools/>) — Provides a range of tools to help explore the accessibility of code
- WAI-ARIA Overview (<https://www.w3.org/WAI/standards-guidelines/aria/>) — Introduction to WAI-ARIA with links to all the specifications

← Previous:
Tips for Designing

(<https://www.w3.org/WAI/tips/designing/>)

Updated: 16 July 2024. [Latest changes \(https://www.w3.org/WAI/tips/changelog/\)](https://www.w3.org/WAI/tips/changelog/).

Date: Minor update 16 July 2024. Updated 15 April 2016. First published September 2015.

Editors: [Kevin White](https://www.w3.org/People/kevin) (<https://www.w3.org/People/kevin>), [Shadi Abou-Zahra](https://www.w3.org/People/shadi) (<https://www.w3.org/People/shadi>), and [Shawn Lawton Henry](https://www.w3.org/People/Shawn) (<https://www.w3.org/People/Shawn>). [Acknowledgements \(https://www.w3.org/WAI/tips/acknowledgements/\)](https://www.w3.org/WAI/tips/acknowledgements/) lists contributors and credits.

Developed by the Education and Outreach Working Group ([EOWG \(https://www.w3.org/WAI/EO/\)](https://www.w3.org/WAI/EO/)). Developed as part of the [WAI-DEV project \(https://www.w3.org/WAI/DEV/\)](https://www.w3.org/WAI/DEV/), co-funded by the European Commission.

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities. Copyright © 2025 World Wide Web Consortium ([W3C®](#)). See [Permission to Use WAI Material](#).



[Home](#) / [Accessibility Fundamentals](#) / [Accessibility is About People](#) / [Older Users and Accessibility](#) / [How WCAG 2 Applies](#)

Developing Websites for Older People: How Web Content Accessibility Guidelines (WCAG) 2.0 Applies

in [Older Users and Web Accessibility: Meeting the Needs of Ageing Web Users](#) (<https://www.w3.org/WAI/older-users/>)

Summary

An extensive literature review (<https://www.w3.org/WAI/intro/wai-age-literature>) identified that existing standards from the W3C Web Accessibility Initiative ([WAI](#) (<https://www.w3.org/WAI/>)) address the accessibility needs of older web users. This page introduces how to use Web Content Accessibility Guidelines (WCAG) 2.0 to improve the accessibility and usability of websites and web applications for older people.

Please see **[Web Accessibility and Older People: Meeting the Needs of Ageing Web Users](#)** (<https://www.w3.org/WAI/older-users/>) for additional background and resources on the overlapping needs of older people and people with disabilities.

Page Contents

- [About WCAG 2.0](#)([#wcag](#))
- [How WCAG 2.0 Applies to Older People](#)([#map](#))
 - [Perceivable information and user interface](#)([#p](#))
 - [Text size](#)([#textsize](#))
 - [Text style and text layout](#)([#textstyle](#))
 - [Color and contrast](#)([#color](#))
 - [Multimedia](#)([#multimedia](#))
 - [Text-to-speech \(speech synthesis\)](#)([#speech](#))
 - [CAPTCHA](#)([#captcha](#))

- Operable user interface and navigation(↳ #o)
 - Links(↳ #links)
 - Navigation and location(↳ #navigation)
 - Mouse use(↳ #mouse)
 - Keyboard use and tabbing(↳ #keyboard)
 - Distractions(↳ #distractions)
 - Sufficient time(↳ #timing)
- Understandable information and user interface(↳ #u)
 - Page organization(↳ #pageorg)
 - Understandable language(↳ #language)
 - Consistent navigation and labeling(↳ #consistent)
 - Pop-ups and new windows(↳ #popups)
 - Page refresh and updates(↳ #refresh)
 - Instructions and input assistance(↳ #instructions)
 - Error prevention and recovery for forms(↳ #errors)
- Robust content and reliable interpretation(↳ #r)
 - Older equipment/software(↳ #oldequip)

About WCAG 2.0

The Web Content Accessibility Guidelines (WCAG) 2.0 includes organizing principles and guidelines, and has success criteria at three levels: A, AA, AAA. WAI recommends meeting at least all WCAG 2.0 Level A and AA success criteria.

A related document provides techniques that include specific details on meeting the success criteria, such as code examples. WAI encourages developers to use the How to Meet WCAG 2.0 quick reference (<https://www.w3.org/WAI/WCAG20/quickref/>) to access the guidelines, success criteria, and techniques, along with the additional descriptions, examples, and resources from “Understanding WCAG 2.0”.

To learn more about WCAG, see:

- WCAG Overview (<https://www.w3.org/WAI/standards-guidelines/wcag/>) - provides background, an introduction to WCAG, and links to additional information

- [The WCAG 2.0 Documents](https://www.w3.org/WAI/intro/wcag20) (<https://www.w3.org/WAI/intro/wcag20>) - describes the different WCAG 2.0 technical and supporting documents
- [How to Meet WCAG 2.0](https://www.w3.org/WAI/WCAG20/quickref/) (<https://www.w3.org/WAI/WCAG20/quickref/>) - is a customizable quick reference to Web Content Accessibility Guidelines 2.0 requirements (Success Criteria) and techniques

How WCAG 2.0 Applies to Older People

This section explains how many of the Web Content Accessibility Guidelines (WCAG) 2.0 guidelines and success criteria specifically meet the needs of older web users. Although not all the WCAG 2.0 success criteria are listed here, WAI recommends meeting at least all WCAG 2.0 Level A and AA success criteria. Some of the Level AAA success criteria that are particularly important for older people are listed in this section too.

When implementing WCAG 2.0, developers can use different techniques to meet the success criteria. In some cases, using one technique instead of another can optimize accessibility for certain users. This section lists some of the techniques that can help optimize websites for older people. (Many of the techniques link to additional information; those with “(future link)” will be written up with later edits to the WCAG Techniques.)

This section is organized under the four principles of web accessibility from WCAG 2.0: perceivable, operable, understandable, and robust. Success criteria are listed thematically, to help readability. Sometimes they are drawn from different guidelines to address a particular aspect.

Perceivable information and user interface

Text size

Many older people require large text due to declining vision, including text in form fields and other controls.

WCAG 2.0 success criteria:

- [1.4.4 - Resize text](https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-scale) (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-scale>) (AA) says “text can be resized without assistive technology up to 200 percent without loss of content or functionality”

Example techniques to consider:

- Using relative font-sizes such as percent ([C12 \(https://www.w3.org/TR/WCAG20-TECHS/C12\)](https://www.w3.org/TR/WCAG20-TECHS/C12)) or ems ([C14 \(https://www.w3.org/TR/WCAG20-TECHS/C14\)](https://www.w3.org/TR/WCAG20-TECHS/C14)) and ensuring text containers resize ([C28 \(https://www.w3.org/TR/WCAG20-TECHS/C28\)](https://www.w3.org/TR/WCAG20-TECHS/C28))
- Providing large fonts by default (future link)
- [C17: Scaling form elements which contain text \(https://www.w3.org/TR/WCAG20-TECHS/C17\)](https://www.w3.org/TR/WCAG20-TECHS/C17)
- Avoiding the use of text in raster images (future link)
- [G178: Providing controls on the Web page that allow users to incrementally change the size of all text on the page up to 200 percent \(https://www.w3.org/TR/WCAG20-TECHS/G178\)](https://www.w3.org/TR/WCAG20-TECHS/G178)

More techniques are listed under [1.4.4 - Resize text \(https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-scale\)](https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-scale) in "How to Meet WCAG 2.0".

Literature review reference: [Vision decline with ageing \(https://www.w3.org/TR/wai-age-literature/#whatvision\)](https://www.w3.org/TR/wai-age-literature/#whatvision).

Text style and text layout

Text style and its visual presentation impacts how hard or easy it is for people to read, especially older people with declining vision.

WCAG 2.0 success criteria:

- [1.4.8 - Visual Presentation \(https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-visual-presentation\)](https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-visual-presentation) (AAA) includes requirements on text style, text justification, line spacing, line length, and horizontal scrolling

Example techniques to consider:

- Avoiding fully-aligned text ([C19 \(https://www.w3.org/TR/WCAG20-TECHS/C19\)](https://www.w3.org/TR/WCAG20-TECHS/C19)) or center-aligned text (future link)
- Using readable fonts (future link)
- Using upper and lower case according to the spelling conventions of the text language (future link)
- Avoiding chunks of italic text (future link)

- [G188: Providing a button on the page to increase line spaces and paragraph spaces](https://www.w3.org/TR/WCAG20-TECHS/G188) (<https://www.w3.org/TR/WCAG20-TECHS/G188>)
- Providing sufficient inter-column spacing (future link)
- Avoiding overuse of different styles on individual pages and in sites (future link)

More techniques are listed under [1.4.8 - Visual Presentation](https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-visual-presentation) (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-visual-presentation>) in "How to Meet WCAG 2.0".

Literature review references: [Vision decline with ageing](https://www.w3.org/TR/wai-age-literature/#whatvision) (<https://www.w3.org/TR/wai-age-literature/#whatvision>), [Previous approaches to 'senior friendly' Web guidelines](https://www.w3.org/TR/wai-age-literature/#existguide) (<https://www.w3.org/TR/wai-age-literature/#existguide>), and [Studies of elderly Web users' specific disabilities - Vision](https://www.w3.org/TR/wai-age-literature/#elderlyusersvision) (<https://www.w3.org/TR/wai-age-literature/#elderlyusersvision>).

Color and contrast

Most older people's color perception changes, and they lose contrast sensitivity.

WCAG 2.0 success criteria:

- [1.4.1 - Use of Color](https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-without-color) (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-without-color>) (A) requires that color is not used as the only visual means of conveying information, indicating an action, prompting a response, or distinguishing a visual element
- [1.4.3 - Contrast \(Minimum\)](https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-contrast) (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-contrast>) (AA) requires a contrast ratio of at least 4.5:1 for the visual presentation of text and images
- [1.4.6 - Contrast \(Enhanced\)](https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast7) (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast7>) (AAA) requires a higher contrast ratio of at least 7:1 for the visual presentation of text and images

Example techniques to consider:

- [G18: Ensuring that a contrast ratio of at least 4.5:1 exists between text \(and images of text\) and background behind the text](https://www.w3.org/TR/WCAG20-TECHS/G18) (<https://www.w3.org/TR/WCAG20-TECHS/G18>)
- [G14: Ensuring that information conveyed by color differences is also available in text](https://www.w3.org/TR/WCAG20-TECHS/G14) (<https://www.w3.org/TR/WCAG20-TECHS/G14>)

- Using a light pastel background rather than a white background behind black text to create sufficient but not extreme contrast (future link)
- G183: Using a contrast ratio of 3:1 with surrounding text and providing additional visual cues on focus for links or controls where color alone is used to identify them (<https://www.w3.org/TR/WCAG20-TECHS/G183>)

More techniques are listed under 1.4.1 - Use of Color (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-without-color>), 1.4.3 - Contrast (Minimum) (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-contrast>) and 1.4.6 - Contrast (Enhanced) (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast7>) in "How to Meet WCAG 2.0".

Literature review references: Vision decline with ageing (<https://www.w3.org/TR/wai-age-literature/#whatvision>), and Studies of elderly Web users' specific disabilities: vision (<https://www.w3.org/TR/wai-age-literature/#elderlyusersvision>).

Multimedia

Because many older people's hearing or vision declines, they often need transcripts, captions, and low background sound.

WCAG 2.0 success criteria:

- 1.2.1 - Audio-only and Video-only (Prerecorded) (<https://www.w3.org/WAI/WCAG20/quickref/#media-equiv-av-only-alt>) (A)
- 1.2.2 - Captions (Prerecorded) (<https://www.w3.org/WAI/WCAG20/quickref/#qr-media-equiv-captions>) (A)
- 1.2.3 - Audio Description or Media Alternative (Prerecorded video) (<https://www.w3.org/WAI/WCAG20/quickref/#media-equiv-audio-desc>) (A)
- 1.2.4 - Captions (Live) (<https://www.w3.org/WAI/WCAG20/quickref/#qr-media-equiv-real-time-captions>) (A)
- 1.2.5 - Audio Description (Prerecorded video) (<https://www.w3.org/WAI/WCAG20/quickref/#media-equiv-audio-desc-only>) (AA)
- 1.2.7 - Extended Audio Description (Prerecorded video) (<https://www.w3.org/WAI/WCAG20/quickref/#media-equiv-extended-ad>) (AAA)
- 1.2.8 - Media Alternative (Prerecorded) (<https://www.w3.org/WAI/WCAG20/quickref/#media-equiv-text-doc>) (AAA)
- 1.2.9 - Audio-only (Live) (<https://www.w3.org/WAI/WCAG20/quickref/#qr-media->

equiv-live-audio-only) (AAA)

- 1.4.7 - Low or No Background Audio (Prerecorded) (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-noaudio>) (AAA)

Example techniques to consider:

- All transcript, caption, and audio description techniques from the listed success criteria should be considered as appropriate, for example:
 - G69: Providing an alternative for time based media (<https://www.w3.org/TR/WCAG20-TECHS/G69>)
 - G158: Providing an alternative for time-based media for audio-only content (<https://www.w3.org/TR/WCAG20-TECHS/G158>)
 - G159: Providing an alternative for time-based media for video-only content (<https://www.w3.org/TR/WCAG20-TECHS/G159>)
 - G93: Providing open (always visible) captions (<https://www.w3.org/TR/WCAG20-TECHS/G93>)
 - G173: Providing a version of a movie with audio descriptions (<https://www.w3.org/TR/WCAG20-TECHS/G173>)
- G56: Mixing audio files so that non-speech sounds are at least 20 decibels lower than the speech audio content (<https://www.w3.org/TR/WCAG20-TECHS/G56>)

More techniques are listed under the success criteria for Guideline 1.2 (Time-based Media) (<https://www.w3.org/WAI/WCAG20/quickref/#media-equiv>) and under 1.4.7 - Low or No Background Audio (Prerecorded) (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-noaudio>) in "How to Meet WCAG 2.0".

Literature review references: Vision decline with ageing (<https://www.w3.org/TR/wai-age-literature/#whatvision>) and Hearing loss with age (<https://www.w3.org/TR/wai-age-literature/#whathearing>).

Text-to-speech (speech synthesis)

Some older people use text-to-speech (speech synthesis) software, which is becoming increasingly available in browsers and operating systems.

WCAG 2.0 success criteria:

- 1.1.1 - Non-text Content (<https://www.w3.org/WAI/WCAG20/quickref/#text-equiv-all>) (A) says "a text alternative that serves the equivalent purpose" is required

- [1.3.1 - Info and Relationships \(https://www.w3.org/WAI/WCAG20/quickref/#content-structure-separation-programmatic\)](https://www.w3.org/WAI/WCAG20/quickref/#content-structure-separation-programmatic) (A) says “information, structure, and relationships” to be made available, for example to text-to-speech software

Example techniques to consider:

- [G94: Providing short text alternative for non-text content that serves the same purpose and presents the same information as the non-text content \(https://www.w3.org/TR/WCAG20-TECHS/G94\)](https://www.w3.org/TR/WCAG20-TECHS/G94)
- [H44: Using label elements to associate text labels with form controls \(https://www.w3.org/TR/WCAG20-TECHS/H44\)](https://www.w3.org/TR/WCAG20-TECHS/H44)
- [H42: Using h1-h6 to identify headings \(https://www.w3.org/TR/WCAG20-TECHS/H42\)](https://www.w3.org/TR/WCAG20-TECHS/H42)

More techniques are listed under [1.1.1 - Non-text Content \(https://www.w3.org/WAI/WCAG20/quickref/#text-equiv-all\)](https://www.w3.org/WAI/WCAG20/quickref/#text-equiv-all) and [1.3.1 - Info and Relationships \(https://www.w3.org/WAI/WCAG20/quickref/#content-structure-separation-programmatic\)](https://www.w3.org/WAI/WCAG20/quickref/#content-structure-separation-programmatic) in “How to Meet WCAG 2.0”.

CAPTCHA

Older people with declining eyesight may not be able to discern the characters in a CAPTCHA, especially because CAPTCHAs often have low contrast and do not increase in size when users have text sized larger.

CAPTCHA stands for ‘Completely Automated Public Turing tests to tell Computers and Humans Apart’. An example of CAPTCHA is:



WCAG 2.0 success criteria:

- [1.1.1 - Non-text Content \(https://www.w3.org/WAI/WCAG20/quickref/#text-equiv-all\)](https://www.w3.org/WAI/WCAG20/quickref/#text-equiv-all) (A) includes a requirement for alternative CAPTCHAs

Example techniques to consider:

- [G143: Providing a text alternative that describes the purpose of the CAPTCHA \(https://www.w3.org/TR/WCAG20-TECHS/G143\)](https://www.w3.org/TR/WCAG20-TECHS/G143) AND [G144: Ensuring that the Web Page contains another CAPTCHA serving the same purpose using a different modality \(https://www.w3.org/TR/WCAG20-TECHS/G144\)](https://www.w3.org/TR/WCAG20-TECHS/G144)
- Providing more than two modalities of CAPTCHAs (future link)

- Providing access to a human customer service representative who can bypass CAPTCHA (future link)
- Not requiring CAPTCHAs for authorized users (future link)

Literature review reference: [Vision decline with ageing \(https://www.w3.org/TR/wai-age-literature/#whatvision\)](https://www.w3.org/TR/wai-age-literature/#whatvision).

Operable user interface and navigation

Links

Many older people need links to be particularly clear and identifiable due to declining vision and cognition.

WCAG 2.0 success criteria:

- [2.4.4 - Link Purpose \(In Context\) \(https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-refs\)](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-refs) (A) requires that the purpose of a link can be determined from the link text alone, or from the link text together with its surrounding context
- [2.4.9 - Link Purpose \(Link Only\) \(https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-link\)](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-link) (AAA) says “a mechanism is available to allow the purpose of each link to be identified from link text alone”
- [2.4.7 - Focus Visible \(https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-focus-visible\)](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-focus-visible) (AA) requires a visible keyboard focus indicator that shows what component on the web page has focus

Example techniques to consider:

- [G91: Providing link text that describes the purpose of a link \(https://www.w3.org/TR/WCAG20-TECHS/G91\)](https://www.w3.org/TR/WCAG20-TECHS/G91)
- Limiting the number of links per page (future link)
- Making links visually distinct (future link)
- [G195: Using an author-supplied, highly visible focus indicator \(https://www.w3.org/TR/WCAG20-TECHS/G195\)](https://www.w3.org/TR/WCAG20-TECHS/G195)
- Highlighting a link or control when the mouse hovers over it, or when it receives keyboard focus (future links)

More techniques are listed under [2.4.4 - Link Purpose \(In Context\) \(https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-refs\)](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-refs), [2.4.9 - Link Purpose \(Link Only\) \(https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-link\)](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-link), and [2.4.7 - Focus Visible \(https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-focus-visible\)](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-focus-visible) in “How to Meet WCAG 2.0”.

Literature review reference: [Previous approaches to ‘senior friendly’ Web guidelines \(https://www.w3.org/TR/wai-age-literature/#existguide\)](https://www.w3.org/TR/wai-age-literature/#existguide).

Navigation and location

Many older people need navigation to be particularly clear due to declining cognitive abilities.

WCAG 2.0 success criteria:

- [2.4.5 - Multiple Ways \(https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-mult-loc\)](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-mult-loc) (AA) says “more than one way is available to locate a Web page within a set of Web pages”
- [2.4.8 - Location \(https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-location\)](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-location) (AAA) says “information about the user’s location within a set of Web pages is available”
- [2.4.2 - Page Titled \(https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-title\)](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-title) (A) says “web pages have titles that describe topic or purpose” (this is important for search results as the page title is usually displayed first in the listing)

Example techniques to consider:

- [G63: Providing a site map \(https://www.w3.org/TR/WCAG20-TECHS/G63\)](https://www.w3.org/TR/WCAG20-TECHS/G63)
- [G161: Providing a search function to help users find content \(https://www.w3.org/TR/WCAG20-TECHS/G161\)](https://www.w3.org/TR/WCAG20-TECHS/G161)
- [G88: Providing descriptive titles for Web pages \(https://www.w3.org/TR/WCAG20-TECHS/G88\)](https://www.w3.org/TR/WCAG20-TECHS/G88) to help understand the results from Search based navigation
- [G65: Providing a breadcrumb trail \(https://www.w3.org/TR/WCAG20-TECHS/G65\)](https://www.w3.org/TR/WCAG20-TECHS/G65)
- [G128: Indicating current location within navigation bars \(https://www.w3.org/TR/WCAG20-TECHS/G128\)](https://www.w3.org/TR/WCAG20-TECHS/G128)
- Providing a link to the home page or main page (future link)

More techniques are listed under [2.4.5 - Multiple Ways](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-mult-loc) (<https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-mult-loc>), [2.4.8 - Location](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-location) (<https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-location>) and [2.4.2 - Page Titled](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-title) (<https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-title>) in "How to Meet WCAG 2.0".

Literature review references: [Previous approaches to 'senior friendly' Web guidelines](https://www.w3.org/TR/wai-age-literature/#existguide) (<https://www.w3.org/TR/wai-age-literature/#existguide>) and [Studies of elderly Web users' specific disabilities - Cognition](https://www.w3.org/TR/wai-age-literature/#elderlyuserscog) (<https://www.w3.org/TR/wai-age-literature/#elderlyuserscog>).

Mouse use

It is difficult for some older people to use a mouse due to declining vision or dexterity.

WCAG 2.0 success criteria:

- [2.4.7 - Focus Visible](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-focus-visible) (<https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-focus-visible>) (AA) says that focus indicators should be visible
- [3.3.2 - Labels or Instructions](https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-cues) (<https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-cues>) (A) says that labels should be provided "when content requires user input"
- [1.1.1 - Text Alternatives](https://www.w3.org/WAI/WCAG20/quickref/#text-equiv) (<https://www.w3.org/WAI/WCAG20/quickref/#text-equiv>) (A) says to provide "text alternatives for any non-text content" such as form controls
- [1.4.4 - Resize Text](https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-scale) (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-scale>) (AA) says that text should be resizable up to 200 percent

Example techniques to consider:

- Highlighting a link or control when the mouse hovers over it (future link)
- [G195: Using an author-supplied, highly visible focus indicator](https://www.w3.org/TR/WCAG20-TECHS/G195) (<https://www.w3.org/TR/WCAG20-TECHS/G195>)
- [H44: Using label elements to associate text labels with form controls](https://www.w3.org/TR/WCAG20-TECHS/H44) (<https://www.w3.org/TR/WCAG20-TECHS/H44>) which increases the clickable area for form controls
- Using real text with relative font size (e.g. [C12](https://www.w3.org/TR/WCAG20-TECHS/C12) (<https://www.w3.org/TR/WCAG20-TECHS/C12>), [C14](https://www.w3.org/TR/WCAG20-TECHS/C14) (<https://www.w3.org/TR/WCAG20-TECHS/C14>)) and avoiding the use of text in raster images (future link) as larger text is easy to click

More techniques are listed under [2.4.7 - Focus Visible](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-focus-visible) (<https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-focus-visible>), [3.3.2 - Labels or Instructions](https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-cues) (<https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-cues>), [1.1.1 - Text Alternatives](https://www.w3.org/WAI/WCAG20/quickref/#text-equiv) (<https://www.w3.org/WAI/WCAG20/quickref/#text-equiv>) and [1.4.4 - Resize Text](https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-scale) (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-scale>) in "How to Meet WCAG 2.0".

Literature review references: [Motor skill diminishment](https://www.w3.org/TR/wai-age-literature/#whatmotor) (<https://www.w3.org/TR/wai-age-literature/#whatmotor>) and [Studies of elderly Web users' specific disabilities - Mobility](https://www.w3.org/TR/wai-age-literature/#elderlyusersmobility) (<https://www.w3.org/TR/wai-age-literature/#elderlyusersmobility>).

Keyboard use and tabbing

Some older people cannot use a mouse well or at all and instead use a keyboard.

WCAG 2.0 success criteria:

- [2.1.1 - Keyboard](https://www.w3.org/WAI/WCAG20/quickref/#keyboard-operation-keyboard-operable) (<https://www.w3.org/WAI/WCAG20/quickref/#keyboard-operation-keyboard-operable>) (A) says "the content is operable through a keyboard interface"
- [2.1.2 - No Keyboard trap](https://www.w3.org/WAI/WCAG20/quickref/#keyboard-operation-trapping) (<https://www.w3.org/WAI/WCAG20/quickref/#keyboard-operation-trapping>) (A) makes sure that keyboard focus "can be moved away from that component using only a keyboard"
- [2.1.3 - Keyboard \(No Exception\)](https://www.w3.org/WAI/WCAG20/quickref/#keyboard-operation-all-funcs) (<https://www.w3.org/WAI/WCAG20/quickref/#keyboard-operation-all-funcs>) (AAA) says "all functionality of the content is operable through a keyboard interface"
- [2.4.1 - Bypass Blocks](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-skip) (<https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-skip>) (A) says "a mechanism is available to bypass blocks of content that are repeated"
- [2.4.3 - Focus Order](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-focus-order) (<https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-focus-order>) (A) says "components receive focus in an order that preserves meaning and operability"
- [2.4.7 - Focus Visible](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-focus-visible) (<https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-focus-visible>) (AA) requires an ability for the "keyboard focus indicator (to be) visible"

Example techniques to consider:

- [H91: Using HTML form controls and links](https://www.w3.org/TR/WCAG20-TECHS/H91) (<https://www.w3.org/TR/WCAG20-TECHS/H91>)

TECHS/H91) to ensure that users can use the form without the mouse

- [G90: Providing keyboard-triggered event handlers \(https://www.w3.org/TR/WCAG20-TECHS/G90\)](https://www.w3.org/TR/WCAG20-TECHS/G90)
- [G1: Adding a link at the top of each page that goes directly to the main content area \(https://www.w3.org/TR/WCAG20-TECHS/G1\)](https://www.w3.org/TR/WCAG20-TECHS/G1)
- [G59: Placing the interactive elements in an order that follows sequences and relationships within the content \(https://www.w3.org/TR/WCAG20-TECHS/G59\)](https://www.w3.org/TR/WCAG20-TECHS/G59)
- [G195: Using an author-supplied, highly visible focus indicator \(https://www.w3.org/TR/WCAG20-TECHS/G195\)](https://www.w3.org/TR/WCAG20-TECHS/G195)
- Providing a highly visible highlighting mechanism for links or controls when they receive keyboard focus (future link)

More techniques are listed under various success criteria for [Guideline 2.1 - Keyboard Accessible \(https://www.w3.org/WAI/WCAG20/quickref/#keyboard-operation\)](https://www.w3.org/WAI/WCAG20/quickref/#keyboard-operation) and [Guideline 2.4 - Navigable \(https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms\)](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms) in "How to Meet WCAG 2.0".

Literature review references: [Motor skill diminishment \(https://www.w3.org/TR/wai-age-literature/#whatmotor\)](https://www.w3.org/TR/wai-age-literature/#whatmotor) and [Studies of elderly Web users' specific disabilities - Mobility \(https://www.w3.org/TR/wai-age-literature/#elderlyusersmobility\)](https://www.w3.org/TR/wai-age-literature/#elderlyusersmobility).

Distractions

Some older people are particularly distracted by any movement and sound on web pages.

WCAG 2.0 success criteria:

- [2.2.2 - Pause, Stop, Hide \(https://www.w3.org/WAI/WCAG20/quickref/#time-limits-pause\)](https://www.w3.org/WAI/WCAG20/quickref/#time-limits-pause) (A) says "a mechanism for the user to pause, stop, or hide" moving or blinking content
- [2.2.4 - Interruptions \(https://www.w3.org/WAI/WCAG20/quickref/#time-limits-postponed\)](https://www.w3.org/WAI/WCAG20/quickref/#time-limits-postponed) (AAA) says "interruptions can be postponed or suppressed"
- [1.4.2 - Audio Control \(https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-dis-audio\)](https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-dis-audio) (A) says "a mechanism is available to pause or stop the audio"

Example techniques to consider:

- [G4: Allowing the content to be paused and restarted from where it was paused](#)

(<https://www.w3.org/TR/WCAG20-TECHS/G4>)

- [G11: Creating content that blinks for less than 5 seconds \(<https://www.w3.org/TR/WCAG20-TECHS/G11>\)](https://www.w3.org/TR/WCAG20-TECHS/G11)
- Providing the user with a means to stop moving content even if it stops automatically within 5 seconds (future link)
- [G76: Providing a mechanism to request an update of the content instead of updating automatically \(<https://www.w3.org/TR/WCAG20-TECHS/G76>\)](https://www.w3.org/TR/WCAG20-TECHS/G76)
- [G60: Playing a sound that turns off automatically within three seconds \(<https://www.w3.org/TR/WCAG20-TECHS/G60>\)](https://www.w3.org/TR/WCAG20-TECHS/G60)
- [G171: Playing sounds only on user request \(<https://www.w3.org/TR/WCAG20-TECHS/G171>\)](https://www.w3.org/TR/WCAG20-TECHS/G171)

More techniques are listed under [2.2.2 - Pause, Stop, Hide](https://www.w3.org/WAI/WCAG20/quickref/#time-limits-pause) (<https://www.w3.org/WAI/WCAG20/quickref/#time-limits-pause>), [2.2.4 - Interruptions](https://www.w3.org/WAI/WCAG20/quickref/#time-limits-postponed) (<https://www.w3.org/WAI/WCAG20/quickref/#time-limits-postponed>), and [1.4.2 - Audio Control](https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-dis-audio) (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-dis-audio>) in "How to Meet WCAG 2.0".

Literature review references: [Cognitive decline with age](https://www.w3.org/TR/wai-age-literature/#whatcog) (<https://www.w3.org/TR/wai-age-literature/#whatcog>) and [Studies of elderly Web users' specific disabilities - Cognition](https://www.w3.org/TR/wai-age-literature/#elderlyuserscog). (<https://www.w3.org/TR/wai-age-literature/#elderlyuserscog>)

Sufficient time

It takes some older people longer to read text and complete transactions due to declining vision, dexterity, or cognition.

WCAG 2.0 success criteria:

- [2.2.1 - Timing Adjustment](https://www.w3.org/WAI/WCAG20/quickref/#time-limits-required-behaviors) (<https://www.w3.org/WAI/WCAG20/quickref/#time-limits-required-behaviors>) (A) says that users can turn off, adjust, or extend any time limits
- [2.2.3 - No Timing](https://www.w3.org/WAI/WCAG20/quickref/#time-limits-no-exceptions) (<https://www.w3.org/WAI/WCAG20/quickref/#time-limits-no-exceptions>) (AAA) says "timing is not an essential part of the event or activity presented by the content" (except for multimedia or real-time events)
- [2.2.2 - Pause, Stop, Hide](https://www.w3.org/WAI/WCAG20/quickref/#time-limits-pause) (<https://www.w3.org/WAI/WCAG20/quickref/#time-limits-pause>) (A) says that scrolling content should be able to be paused and that auto updating content can be also paused or controlled

Example techniques to consider:

- G4: Allowing the content to be paused and restarted from where it was paused (<https://www.w3.org/TR/WCAG20-TECHS/G4>)
- G198: Providing a way for the user to turn the time limit off (<https://www.w3.org/TR/WCAG20-TECHS/G198>)
- SCR16: Providing a script that warns the user a time limit is about to expire (<https://www.w3.org/TR/WCAG20-TECHS/SCR16>) and
SCR1: Allowing the user to extend the default time limit (<https://www.w3.org/TR/WCAG20-TECHS/SCR1>)

More techniques are listed under 2.2.1 - Timing Adjustment (<https://www.w3.org/WAI/WCAG20/quickref/#time-limits-required-behaviors>), 2.2.3 - No Timing (<https://www.w3.org/WAI/WCAG20/quickref/#time-limits-no-exceptions>), and 2.2.2 - Pause, Stop, Hide (<https://www.w3.org/WAI/WCAG20/quickref/#time-limits-pause>) in "How to Meet WCAG 2.0".

Literature review references: Cognitive decline with age (<https://www.w3.org/TR/wai-age-literature/#whatcog>) and Studies of elderly Web users' specific disabilities - Cognition. (<https://www.w3.org/TR/wai-age-literature/#elderlyuserscog>)

Understandable information and user interface

Page organization

Many older people are inexperienced web users without advanced browsing habits and therefore read the whole page, so good page organization is important.

WCAG 2.0 success criteria:

- 2.4.6 - Headings and Labels (<https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-descriptive>) (AA) says "headings and labels describe topic or purpose"
- 2.4.10 - Section Headings (<https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-headings>) (AAA) says "section headings are used to organize the content"
- 1.4.8 - Visual (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-visual-presentation>)
Presentation (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-visual-presentation>) (AAA) includes techniques to help with

text organization

Example techniques to consider:

- [G130: Providing descriptive headings](https://www.w3.org/TR/WCAG20-TECHS/G130) (<https://www.w3.org/TR/WCAG20-TECHS/G130>)
- [G131: Providing descriptive labels](https://www.w3.org/TR/WCAG20-TECHS/G131) (<https://www.w3.org/TR/WCAG20-TECHS/G131>)
- [G141: Organizing a page using headings](https://www.w3.org/TR/WCAG20-TECHS/G141) (<https://www.w3.org/TR/WCAG20-TECHS/G141>)
- Using vertical (bulleted or numbered) lists rather than inline lists (future link)

More techniques are listed under [2.4.6 - Headings and Labels](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-descriptive) (<https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-descriptive>), [2.4.10 - Section Headings](https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-headings) (<https://www.w3.org/WAI/WCAG20/quickref/#navigation-mechanisms-headings>) and [1.4.8 - Visual](https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-visual-presentation) (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-visual-presentation>) [Presentation](https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-visual-presentation) (<https://www.w3.org/WAI/WCAG20/quickref/#visual-audio-contrast-visual-presentation>) in "How to Meet WCAG 2.0".

Literature review references: [Cognitive decline with age](https://www.w3.org/TR/wai-age-literature/#whatcog) (<https://www.w3.org/TR/wai-age-literature/#whatcog>) and [Previous approaches to 'senior friendly' Web guidelines](https://www.w3.org/TR/wai-age-literature/#existguide) (<https://www.w3.org/TR/wai-age-literature/#existguide>).

Understandable language

Many older people find it particularly difficult to understand complex sentences, unusual words, and technical jargon.

WCAG 2.0 success criteria:

- [3.1.3 - Unusual Words](https://www.w3.org/WAI/WCAG20/quickref/#meaning-idioms) (<https://www.w3.org/WAI/WCAG20/quickref/#meaning-idioms>) (AAA) says "a mechanism is available for identifying specific definitions of words or phrases used in an unusual or restricted way"
- [3.1.4 - Abbreviations](https://www.w3.org/WAI/WCAG20/quickref/#meaning-located) (<https://www.w3.org/WAI/WCAG20/quickref/#meaning-located>) (AAA) says "a mechanism for identifying the expanded form or meaning of abbreviations is available"
- [3.1.5 - Reading Level](https://www.w3.org/WAI/WCAG20/quickref/#meaning-supplements) (<https://www.w3.org/WAI/WCAG20/quickref/#meaning-supplements>) (AAA) requires providing a version that "does not require reading ability more advanced than the lower secondary education level"

Example techniques to consider:

- Using the clearest and simplest language appropriate for the content (future link)
- [G102: Providing the expansion or explanation of an abbreviation \(https://www.w3.org/TR/WCAG20-TECHS/G102\)](https://www.w3.org/TR/WCAG20-TECHS/G102)
- [G153: Making the text easier to read \(https://www.w3.org/TR/WCAG20-TECHS/G153\)](https://www.w3.org/TR/WCAG20-TECHS/G153)

More techniques are listed under [3.1.3 - Unusual Words](https://www.w3.org/WAI/WCAG20/quickref/#meaning-idioms) (<https://www.w3.org/WAI/WCAG20/quickref/#meaning-idioms>), [3.1.4 - Abbreviations](https://www.w3.org/WAI/WCAG20/quickref/#meaning-located) (<https://www.w3.org/WAI/WCAG20/quickref/#meaning-located>) and [3.1.5 - Reading Level](https://www.w3.org/WAI/WCAG20/quickref/#meaning-supplements) (<https://www.w3.org/WAI/WCAG20/quickref/#meaning-supplements>) in "How to Meet WCAG 2.0".

Literature review references: [Cognitive decline with age](https://www.w3.org/TR/wai-age-literature/#whatcog) (<https://www.w3.org/TR/wai-age-literature/#whatcog>) and [Previous approaches to 'senior friendly' Web guidelines](https://www.w3.org/TR/wai-age-literature/#existguide) (<https://www.w3.org/TR/wai-age-literature/#existguide>).

Consistent navigation and labeling

For people who are new to the web, and older people with some types of cognitive decline, consistent navigation and presentation is particularly important.

WCAG 2.0 success criteria:

- [3.2.3 - Consistent Navigation](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-consistent-locations) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-consistent-locations>) (AA) requires that navigation is presented in the same relative order across a website
- [3.2.4 - Consistent Identification](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-consistent-functionality) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-consistent-functionality>) (AA) requires that components with similar functionality are identified consistently

Example techniques to consider:

- [G61: Presenting repeated components in the same relative order each time they appear](https://www.w3.org/TR/WCAG20-TECHS/G61) (<https://www.w3.org/TR/WCAG20-TECHS/G61>)
- [G197: Using labels, names, and text alternatives consistently for content that has the same functionality](https://www.w3.org/TR/WCAG20-TECHS/G197) (<https://www.w3.org/TR/WCAG20-TECHS/G197>)

More techniques are listed under [3.2.3 - Consistent Navigation](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-consistent-locations) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-consistent-locations>) and [3.2.4 - Consistent Identification](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-consistent-functionality) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-consistent-functionality>) in "How to Meet WCAG 2.0".

Literature review references: [Cognitive decline with age](https://www.w3.org/TR/wai-age-literature/#whatcog) (<https://www.w3.org/TR/wai-age-literature/#whatcog>) and [Previous approaches to ‘senior friendly’ Web guidelines](https://www.w3.org/TR/wai-age-literature/#existguide) (<https://www.w3.org/TR/wai-age-literature/#existguide>).

Pop-ups and new windows

Some older people experiencing cognitive decline can be confused or distracted by pop-ups, new windows, or new tabs.

WCAG 2.0 success criteria:

- [3.2.1 - On Focus](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-receive-focus) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-receive-focus>) (A) says “when any component receives focus, it does not initiate a change of context”
- [3.2.5 - Change on Request](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-no-extreme-changes-context) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-no-extreme-changes-context>) (AAA) says “changes of context are initiated only by user request or a mechanism is available to turn off such changes”

Example techniques to consider:

- Giving users advanced warning when opening a new window (future link)
- [G107: Using “activate” rather than “focus” as a trigger for changes of context](https://www.w3.org/TR/WCAG20-TECHS/G107) (<https://www.w3.org/TR/WCAG20-TECHS/G107>)
- Opening new windows only when best from an accessibility perspective (future link)
- [SCR24: Using progressive enhancement to open new windows on user request](https://www.w3.org/TR/WCAG20-TECHS/SCR24) (<https://www.w3.org/TR/WCAG20-TECHS/SCR24>)
- Opening new windows by providing normal hyperlinks without the target attribute (future link), because many user agents allow users to open links in a new window or tab

More techniques are listed under [3.2.1 - On Focus](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-receive-focus) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-receive-focus>) and [3.2.5 - Change on Request](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-no-extreme-changes-context) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-no-extreme-changes-context>) in “How to Meet WCAG 2.0”.

Literature review reference: [Training the elderly to Use ICT and the Web](https://www.w3.org/TR/wai-age-literature/#training) (<https://www.w3.org/TR/wai-age-literature/#training>)

Page refresh and updates

Some older people with declining vision or cognition can miss content that automatically updates or refreshes in a page.

WCAG 2.0 success criteria:

- [3.2.1 - On Focus](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-receive-focus) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-receive-focus>) (A) says “when any component receives focus, it does not initiate a change of context”
- [3.2.2 - On Input](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-unpredictable-change) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-unpredictable-change>) (A) says that changing a setting does not automatically change the context unless the user has been advised beforehand
- [3.2.5 - Change on Request](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-no-extreme-changes-context) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-no-extreme-changes-context>) (AAA) says “changes of context are initiated only by user request or a mechanism is available to turn off such changes”

Example techniques to consider:

- [G80: Providing a submit button to initiate a change of context](https://www.w3.org/TR/WCAG20-TECHS/G80) (<https://www.w3.org/TR/WCAG20-TECHS/G80>)
- [G13: Describing what will happen before a change to a form control that causes a change of context to occur is made](https://www.w3.org/TR/WCAG20-TECHS/G13) (<https://www.w3.org/TR/WCAG20-TECHS/G13>)
- [G107: Using “activate” rather than “focus” as a trigger for changes of context](https://www.w3.org/TR/WCAG20-TECHS/G107) (<https://www.w3.org/TR/WCAG20-TECHS/G107>)
- Not causing persistent changes of state or value when a component receives focus, or providing an alternate means to reset any changes (future link)
- [G76: Providing a mechanism to request an update of the content instead of updating automatically](https://www.w3.org/TR/WCAG20-TECHS/G76) (<https://www.w3.org/TR/WCAG20-TECHS/G76>)
- [SCR19: Using an onchange event on a select element without causing a change of context](https://www.w3.org/TR/WCAG20-TECHS/SCR19) (<https://www.w3.org/TR/WCAG20-TECHS/SCR19>)

More techniques are listed under [3.2.1 - On Focus](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-receive-focus) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-receive-focus>), [3.2.2 - On Input](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-unpredictable-change) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-unpredictable-change>) and [3.2.5 - Change on Request](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-no-extreme-changes-context) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-no-extreme-changes-context>) in “How to Meet WCAG 2.0”.

Literature review reference: [Studies of elderly Web users' specific disabilities - Cognition](https://www.w3.org/TR/wai-age-literature/#elderlyuserscog) (<https://www.w3.org/TR/wai-age-literature/#elderlyuserscog>)

Instructions and input assistance

It is difficult for some older people to understand the requirements of forms and transactions.

WCAG 2.0 success criteria:

- [3.3.2 - Labels or Instructions](https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-identified) (<https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-identified>) (A) says "labels or instructions are provided when content requires user input"
- [3.3.5 - Help](https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-context-help) (<https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-context-help>) (AAA) says "context-sensitive help is available"
- [3.2.4 - Consistent Identification](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-consistent-functionality) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-consistent-functionality>) (AA) says "components that have the same functionality within a set of Web pages are identified consistently"

Example techniques to consider:

- [G184: Providing text instructions at the beginning of a form or set of fields that describes the necessary input](https://www.w3.org/TR/WCAG20-TECHS/G184) (<https://www.w3.org/TR/WCAG20-TECHS/G184>)
- Providing linear form design and grouping similar items (future link)
- [G194: Providing spell checking and suggestions for text input](https://www.w3.org/TR/WCAG20-TECHS/G194) (<https://www.w3.org/TR/WCAG20-TECHS/G194>)
- [G89: Providing expected data format and example](https://www.w3.org/TR/WCAG20-TECHS/G89) (<https://www.w3.org/TR/WCAG20-TECHS/G89>)
- [G197: Using labels, names, and text alternatives consistently for content that has the same functionality](https://www.w3.org/TR/WCAG20-TECHS/G197) (<https://www.w3.org/TR/WCAG20-TECHS/G197>)

More techniques are listed under the success criteria for [3.3.2 - Labels or Instructions](https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-identified) (<https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-identified>), [3.3.5 - Help](https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-context-help) (<https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-context-help>) and [3.2.4 - Consistent Identification](https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-consistent-functionality) (<https://www.w3.org/WAI/WCAG20/quickref/#consistent-behavior-consistent-functionality>) in "How to Meet WCAG 2.0".

Literature review reference: [Aspects of Web Design affecting the elderly - Forms](https://www.w3.org/TR/wai-age-literature/#designforms) (<https://www.w3.org/TR/wai-age-literature/#designforms>).

Error prevention and recovery for forms

It is difficult for some older people to use forms and complete transactions due to declining cognitive abilities.

WCAG 2.0 success criteria:

- [3.3.4 - Error Prevention \(Legal, Financial, Data\) \(https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-reversible\)](https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-reversible) (AA) says that pages with legal commitments or financial transactions have reversible submissions and can be checked and corrected
- [3.3.6 - Error Prevention \(All\) \(https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-context-help\)](https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-context-help) (AAA) says that users can check and correct any information they submit
- [3.3.1 - Error Identification \(https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-identified\)](https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-identified) (A) says "if an input error is automatically detected, the item that is in error is identified and the error is described to the user"
- [3.3.3 - Error Suggestion \(https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-suggestions\)](https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-suggestions) (AA) says "if an input error is automatically detected and suggestions for correction are known, then the suggestions are provided to the user"

Example techniques to consider:

- [G98: Providing the ability for the user to review and correct answers before submitting \(https://www.w3.org/TR/WCAG20-TECHS/G98\)](https://www.w3.org/TR/WCAG20-TECHS/G98)
- Accepting input data in a variety of formats (future link)
- Informing the user what irreversible action is about to happen (future link)
- Making error messages easy to understand and distinguishable from other text in the Web page (future link)
- [G83: Providing text descriptions to identify required fields that were not completed \(https://www.w3.org/TR/WCAG20-TECHS/G83\)](https://www.w3.org/TR/WCAG20-TECHS/G83)
- [G85: Providing a text description when user input falls outside the required format or values \(https://www.w3.org/TR/WCAG20-TECHS/G85\)](https://www.w3.org/TR/WCAG20-TECHS/G85)
- [G139: Creating a mechanism that allows users to jump to errors \(https://www.w3.org/TR/WCAG20-TECHS/G139\)](https://www.w3.org/TR/WCAG20-TECHS/G139)
- Re-displaying a form with a summary of errors (future link)

- [G177: Providing suggested correction text \(https://www.w3.org/TR/WCAG20-TECHS/G177\)](https://www.w3.org/TR/WCAG20-TECHS/G177)
- Providing a text description that contains information about the number of input errors, suggestions for corrections to each item, and instructions on how to proceed (future link)

More techniques are listed under the success criteria for [Guideline 3.3 - Input Assistance \(https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-reversible\)](https://www.w3.org/WAI/WCAG20/quickref/#minimize-error-reversible) in "How to Meet WCAG 2.0".

Literature review reference: [Aspects of Web Design affecting the elderly - Forms \(https://www.w3.org/TR/wai-age-literature/#designforms\).](https://www.w3.org/TR/wai-age-literature/#designforms)

Robust content and reliable interpretation

Older equipment/software

Some older people will be using older browsers that might not be as capable or fault tolerant as current releases.

WCAG 2.0 success criteria:

- [4.1.1 - Parsing \(https://www.w3.org/WAI/WCAG20/quickref/#ensure-compat-parses\)](https://www.w3.org/WAI/WCAG20/quickref/#ensure-compat-parses) (A) requires that markup is used correctly according to specification

Example techniques to consider:

- [G134: Validating Web pages \(https://www.w3.org/TR/WCAG20-TECHS/G134\)](https://www.w3.org/TR/WCAG20-TECHS/G134)
- [G192: Fully conforming to specifications \(https://www.w3.org/TR/WCAG20-TECHS/G192\)](https://www.w3.org/TR/WCAG20-TECHS/G192)

More techniques are listed under [4.1.1 - Parsing \(https://www.w3.org/WAI/WCAG20/quickref/#ensure-compat-parses\)](https://www.w3.org/WAI/WCAG20/quickref/#ensure-compat-parses) in "How to Meet WCAG 2.0".

This page has been reviewed and is current as of 1 January 2018. [Latest changes \(https://www.w3.org/WAI/older-users/changelog/\).](https://www.w3.org/WAI/older-users/changelog/)

Editors: [Andrew Arch \(https://www.w3.org/People/Andrew/\)](https://www.w3.org/People/Andrew/) and [Shadi Abou-Zahra \(https://www.w3.org/People/shadi/\)](https://www.w3.org/People/shadi/). Contributors: [Shawn Henry \(https://www.w3.org/People/Shawn/\)](https://www.w3.org/People/Shawn/), Suzette Keith, Kate Roberts.

Developed with input from the Education and Outreach Working Group ([EOWG \(https://www.w3.org/WAI/EO/\)](https://www.w3.org/WAI/EO/)), with the [WAI-AGE Task](#)

[Force](https://www.w3.org/WAI/EO/2008/wai-age-tf.html#participants) (<https://www.w3.org/WAI/EO/2008/wai-age-tf.html#participants>). Related to the [WAI-AGE Project](https://www.w3.org/WAI/WAI-AGE/) (<https://www.w3.org/WAI/WAI-AGE/>) funded by the European Commission under the 6th Framework.

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities.

Copyright © 2025 World Wide Web Consortium ([W3C®](#)).
See [Permission to Use WAI Material](#).



Forms Tutorial

in [Tutorials](#) (<https://www.w3.org/WAI/tutorials/>)

Forms are commonly used to provide user interaction on websites and in web applications. For example, login, registering, commenting, and purchasing. This tutorial shows you how to create accessible forms. The same concepts apply to all forms, whether they are processed client or server-side.

Aside from technical considerations, users usually prefer simple and short forms. Only ask users to enter what is required to complete the transaction or process; if irrelevant or excessive data is requested, users are more likely to abandon the form.

- **[Labeling Controls](#)** (<https://www.w3.org/WAI/tutorials/forms/labels/>): Use the `<label>` element, and, in specific cases, other mechanisms (e.g. WAI-ARIA, `title` attribute etc.), to identify each form control.
- **[Grouping Controls](#)** (<https://www.w3.org/WAI/tutorials/forms/grouping/>): Use the `<fieldset>` and `<legend>` elements to group and associate related form controls.
- **[Form Instructions](#)** (<https://www.w3.org/WAI/tutorials/forms/instructions/>): Provide instructions to help users understand how to complete the form and individual form controls.
- **[Validating Input](#)** (<https://www.w3.org/WAI/tutorials/forms/validation/>): Validate input provided by the user and provide options to undo changes and confirm data entry.
- **[User Notifications](#)** (<https://www.w3.org/WAI/tutorials/forms/notifications/>): Notify users about successful task completion, any errors, and provide instructions to help them correct mistakes.
- **[Multi-Page Forms](#)** (<https://www.w3.org/WAI/tutorials/forms/multi-page/>): Divide long forms into multiple smaller forms that constitute a series of logical steps or stages and inform users about their progress.
- **[Custom Controls](#)** (<https://www.w3.org/WAI/tutorials/forms/custom-controls/>):

Use stylized form elements and other progressive enhancement techniques to provide custom controls.

A note on time limits

If possible, forms should not be subject to a time limit to allow users to complete the form at their pace. If a time limit needs to be in place, for example, for security reasons, the user should have the option to turn it off or extend it. This restriction does not apply if the time limit is due to a live event, such as an auction or a game, or if the time to complete the form is essential for a valid submission.

Why is this important?

Forms can be visually and cognitively complex and challenging to use. Accessible forms are easier to use for everyone, including people with disabilities.

- **People with cognitive disabilities** can better understand the form and how to complete it, as making forms accessible improves the layout structure, instructions, and feedback.
- **People using speech input** can use the labels via voice commands to activate controls and move the focus to the fields that they have to complete.
- **People with limited dexterity** benefit from large clickable areas that include the labels, especially for smaller controls, such as radio buttons and checkboxes.
- **People using screen readers** can identify and understand form controls more easily because they are associated with labels, field sets, and other structural elements.

* Related WCAG resources

These tutorials provide best-practice guidance on implementing accessibility in different situations. This page combined the following WCAG success criteria and techniques from different conformance levels:

Success Criteria:

- **1.3.1 Info and Relationships:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-content-structure-separation-programmatic>) Information, structure, and

relationships conveyed through presentation can be programmatically determined or are available in text. (Level A)

- **2.4.6 Headings and Labels:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-navigation-mechanisms-descriptive>) Headings and labels describe topic or purpose. (Level AA)
- **3.3.2 Labels or Instructions:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-minimize-error-cues>) Labels or instructions are provided when content requires user input. (Level A)
- **4.1.2 Name, Role, Value:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-ensure-compat-rsv>) For all user interface components (including but not limited to: form elements, links and components generated by scripts), the name and role can be programmatically determined; states, properties, and values that can be set by the user can be programmatically set; and notification of changes to these items is available to user agents, including assistive technologies. (Level A)

(<https://www.w3.org/WAI/tutorials/forms/labels/>)

Next: [Labeling Controls](#) ➔

Status: Updated 27 July 2019 (first published September 2014)

Editors: [Eric Eggert](https://www.w3.org/People/yatil/) (<https://www.w3.org/People/yatil/>) and [Shadi Abou-Zahra](https://www.w3.org/People/shadi/) (<https://www.w3.org/People/shadi/>). Update Editor: Brian Elton. Contributors: see [Acknowledgements](https://www.w3.org/WAI/tutorials/acknowledgements/) (<https://www.w3.org/WAI/tutorials/acknowledgements/>).

Developed by the Education and Outreach Working Group ([EOWG](https://www.w3.org/groups/wg/eowg) (<https://www.w3.org/groups/wg/eowg>)). Developed with support from the [WAI-ACT project](https://www.w3.org/WAI/ACT/) (<https://www.w3.org/WAI/ACT/>), co-funded by the [European Commission IST Programme](#).

[Latest changes](https://www.w3.org/WAI/tutorials/changelog/) (<https://www.w3.org/WAI/tutorials/changelog/>), [Acknowledgements](https://www.w3.org/WAI/tutorials/acknowledgements/) (<https://www.w3.org/WAI/tutorials/acknowledgements/>)

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities.

Copyright © 2025 World Wide Web Consortium ([W3C](#)®).
See [Permission to Use WAI Material](#).



Guidelines to Make Your No-Code Website Tool Accessible

Summary

This page briefly introduces a standard to help you make your no-code website tool accessible to creators, designers, and end users with disabilities.

Page Contents

- [Why Accessibility Matters](#)(<#why-accessibility-matters>)
- [Example Scenarios](#)(<#example-scenarios>)
- [The Accessibility Standard to Help You](#)(<#the-accessibility-standard-to-help-you>)

Why Accessibility Matters

No-code tools let people create websites without having to code.

People with disabilities need to use your no-code tool. Your tool needs to be accessible to disabled content creators.

The websites and widgets that your tool outputs also need to be accessible to disabled website users.

Accessibility is required by law and is a procurement requirement in many situations.

Improving the accessibility of your tool and its output can be a strong competitive advantage, particularly with the current focus on diversity, equity, and inclusion.

Example Scenarios

These *persona* scenarios show examples of accessibility *problems* that disabled people experience using no-code tools, and what *works well* in tools that are accessible.

Everyone can add blocks of content

Persona: Martin is an independent consultant who has a website to attract clients. He is blind and uses a screen reader that reads aloud the information on his screen.

Problem: “I want to add a new section to my home page. The tool documentation says to drag and drop a content block. I can't do that, because I can't use a mouse. I tried using my screen reader's built-in functionality to drag and drop, but it doesn't work with this tool.”

Works well: “I can use the keyboard to select the content block and put it where I want on the page. Everything is announced properly, including the block type and the location where I place it.”

Users who have auditory, cognitive, neurological, physical, speech, or visual disabilities need to be able to add, customize, and remove blocks of content.

The tool provides accessible components and templates

Persona: Amahle is a hairdresser and is proud that she created her own website. Clients use her website to schedule appointments, and it's a real time saver for her.

Problem: “Some of my older clients say they cannot use the calendar widget to schedule an appointment. One says she uses large text and the buttons are cut off. Others have complained that the light gray text is too hard to read.”

Works well: “The calendar widget that the tool provides works great for all my clients, including the older ones. The tool helped me pick color combinations that are easy to see.”

The templates, components, and pre-authored content that your tool provides need to be accessible. That includes making things like headings, lists, and labels accessible. When your tool offers customization options, such as colors, help content creators make accessible choices.

The tool helps make images and videos accessible

Persona: Dacso runs a car repair shop and does a lot of body work. His daughter created a website for him to showcase his work. Dacso uploads before-and-after photos of his projects. He also posts videos with simple maintenance tips.

Problem: “I usually get lots of great feedback on my website. A couple of Deaf people say they want to learn from my videos, but of course they can't hear what I'm saying. I really want to make it so they can get advice from my videos, but I have no idea what to do.”

Works well: “Whenever I upload a picture, the tool asks me for a text description. It has "more info" that explains how to write a good text description for people who are blind and can't see the image. It also has good support for adding captions to videos. And the "more info" for videos explains how to describe what I'm doing when I make the video, for people who don't see too well.”

Your tool should enable and encourage content creators to provide alternative text for images, audio, and videos. It should include guidance on how to provide useful alternatives for their website consumers who have disabilities.

The Accessibility Standard to Help You

Your no-code tool is sometimes called an “authoring tool” because people use it to author or create websites. There is an international standard that addresses accessibility needs in no-code tools: Authoring Tool Accessibility Guidelines (ATAG).

Use ATAG to help make your tool:

- **accessible to website creators (Part A)**
- **support accessible content for website consumers (Part B)**

Examples

ATAG covers the example scenarios above:

- *Everyone can add blocks of content* — ATAG says: Provide keyboard access to authoring features. Some authors with limited mobility or visual disabilities do not use a mouse and instead require keyboard interface access to all of the functionality of the authoring tool.
- *The tool provides accessible components and templates* — ATAG says: Assist authors with accessible templates. There are accessible template options for a range of template uses. Assist authors with accessible pre-authored content.
- *The tool helps make images and videos accessible* — ATAG says: Assist authors with managing

alternative content for non-text content. Ensure that documentation promotes the production of accessible content.

And ATAG covers much more.

To get started putting ATAG to work for you, see:

- [Authoring Tool Accessibility Guidelines \(ATAG\) Overview \(https://www.w3.org/WAI/standards-guidelines/atag/\)](https://www.w3.org/WAI/standards-guidelines/atag/), with links to the ATAG standard and Implementing ATAG
- [ATAG at a Glance \(https://www.w3.org/WAI/standards-guidelines/atag/glance/\)](https://www.w3.org/WAI/standards-guidelines/atag/glance/), a paraphrased summary to give you an idea of what's covered in ATAG

Updated: 8 December 2022.

Editors: Daniel Montalvo and [Shawn Henry \(https://www.w3.org/People/Shawn/\)](https://www.w3.org/People/Shawn/). Contributors: Kevin White, Michele Williams, and participants of the Education and Outreach Working Group (<https://www.w3.org/groups/wg/eowg/participants>).

Developed by the Education and Outreach Working Group ([EOWG \(https://www.w3.org/WAI/EO\)](https://www.w3.org/WAI/EO)). Developed as part of the [WAI-Guide project \(https://www.w3.org/WAI/about/projects/wai-guide\)](https://www.w3.org/WAI/about/projects/wai-guide), co-funded by the European Commission.

© This work is licensed under a [Creative Commons Attribution 4.0 International License \(https://creativecommons.org/licenses/by/4.0/\)](https://creativecommons.org/licenses/by/4.0/).

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities. Copyright © 2025 World Wide Web Consortium ([W3C®](#)). See [Permission to Use WAI Material](#).



(<https://w3.org/>)

Web Accessibility
Initiative WAI

(<https://w3.org/WAI/>)

How to Meet WCAG (Web Content Accessibility Guidelines) (Quick Reference)

A customizable quick reference to Web Content Accessibility Guidelines (WCAG) 2 requirements (success criteria) and techniques.

➤ Show About & How to Use

Contents

Filter

1. Perceivable

1.1 Text Alternatives

1.1.1 Non-text Content

1.2 Time-based Media

1.2.1 Audio-only and Video-only (Prerecorded)

1.2.2 Captions (Prerecorded)

1.2.3 Audio Description or Media Alternative (Prerecorded)

1.2.4 Captions (Live)

1.2.5 Audio Description (Prerecorded)

1.2.6 Sign Language (Prerecorded)

1.2.7 Extended Audio Description (Prerecorded)

1.2.8 Media Alternative (Prerecorded)

1.2.9 Audio-only (Live)

1.3 Adaptable

1.3.1 Info and Relationships

1.3.2 Meaningful Sequence

1.3.3 Sensory Characteristics

1.3.4 Orientation

1.3.5 Identify Input Purpose

1.3.6 Identify Purpose

1.4 Distinguishable

1.4.1 Use of Color

1.4.2 Audio Control

1.4.3 Contrast (Minimum)

1.4.4 Resize Text

1.4.5 Images of Text

1.4.6 Contrast (Enhanced)

1.4.7 Low or No Background Audio

1.4.8 Visual Presentation

1.4.9 Images of Text (No Exception)

1.4.10 Reflow

1.4.11 Non-text Contrast

1.4.12 Text Spacing

1.4.13 Content on Hover or Focus

2. Operable

2.1 Keyboard Accessible

2.1.1 Keyboard

2.1.2 No Keyboard Trap

2.1.3 Keyboard (No Exception)

2.1.4 Character Key Shortcuts

2.2 Enough Time

2.2.1 Timing Adjustable

2.2.2 Pause, Stop, Hide

2.2.3 No Timing

2.2.4 Interruptions

2.2.5 Re-authenticating

2.2.6 Timeouts

2.3 Seizures and Physical Reactions

2.3.1 Three Flashes or Below Threshold

2.3.2 Three Flashes

2.3.3 Animation from Interactions

2.4 Navigable

2.4.1 Bypass Blocks

2.4.2 Page Titled

2.4.3 Focus Order

2.4.4 Link Purpose (In Context)

2.4.5 Multiple Ways

2.4.6 Headings and Labels

2.4.7 Focus Visible

2.4.8 Location

2.4.9 Link Purpose (Link Only)

2.4.10 Section Headings

2.4.11 Focus Not Obscured (Minimum)

2.4.12 Focus Not Obscured (Enhanced)

2.4.13 Focus Appearance

2.5 Input Modalities

- 2.5.1** Pointer Gestures
- 2.5.2** Pointer Cancellation
- 2.5.3** Label in Name
- 2.5.4** Motion Actuation
- 2.5.5** Target Size (Enhanced)
- 2.5.6** Concurrent Input Mechanisms
- 2.5.7** Dragging Movements
- 2.5.8** Target Size (Minimum)

3. Understandable

3.1 Readable

- 3.1.1** Language of Page
- 3.1.2** Language of Parts
- 3.1.3** Unusual Words
- 3.1.4** Abbreviations
- 3.1.5** Reading Level
- 3.1.6** Pronunciation

3.2 Predictable

- 3.2.1** On Focus
- 3.2.2** On Input
- 3.2.3** Consistent Navigation
- 3.2.4** Consistent Identification
- 3.2.5** Change on Request
- 3.2.6** Consistent Help

3.3 Input Assistance

- 3.3.1** Error Identification
- 3.3.2** Labels or Instructions
- 3.3.3** Error Suggestion
- 3.3.4** Error Prevention (Legal, Financial, Data)
- 3.3.5** Help
- 3.3.6** Error Prevention (All)
- 3.3.7** Redundant Entry
- 3.3.8** Accessible Authentication (Minimum)
- 3.3.9** Accessible Authentication (Enhanced)

4. Robust

4.1 Compatible

- 4.1.2** Name, Role, Value
- 4.1.3** Status Messages

Selected Filters: **WCAG 2.2:** all success criteria and all techniques.[Clear filters](#)[Collapse all sections](#)[Share](#)

Principle 1 – Perceivable

Information and user interface components must be presentable to users in ways they can perceive.

Guideline 1.1 – Text Alternatives

Provide text alternatives for any non-text content so that it can be changed into other forms people need, such as large print, braille, speech, symbols or simpler language.

1.1.1 Non-text Content — Level A

All non-text content that is presented to the user has a text alternative that serves the equivalent purpose, except for the situations listed below. [▼ Hide full description](#)

- **Controls, Input:** If non-text content is a control or accepts user input, then it has a name that describes its purpose. (Refer to Success Criterion 4.1.2 for additional requirements for controls and content that accepts user input.)
- **Time-Based Media:** If non-text content is time-based media, then text alternatives at least provide descriptive identification of the non-text content. (Refer to Guideline 1.2 for additional requirements for media.)
- **Test:** If non-text content is a test or exercise that would be invalid if presented in text, then text alternatives at least provide descriptive identification of the non-text content.
- **Sensory:** If non-text content is primarily intended to create a specific sensory experience, then text alternatives at least provide descriptive identification of the non-text content.
- **CAPTCHA:** If the purpose of non-text content is to confirm that content is being accessed by a person rather than a computer, then text alternatives that identify and describe the purpose of the non-text content are provided, and alternative forms of CAPTCHA using output modes for different types of sensory perception are provided to accommodate different disabilities.
- **Decoration, Formatting, Invisible:** If non-text content is pure decoration, is used only for visual formatting, or is not presented to users, then it is implemented in a

way that it can be ignored by assistive technology.

- ❶ Understanding 1.1.1 (<https://www.w3.org/WAI/WCAG22/Understanding/non-text-content.html>)

▼ Hide techniques and failures for 1.1.1

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

Situation A: If a short description can serve the same purpose and present the same information as the non-text content:

- [G94: Providing short text alternative for non-text content that serves the same purpose and presents the same information as the non-text content](https://www.w3.org/WAI/WCAG22/Techniques/general/G94) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G94>) using one technique from each group outlined below

Short text alternative techniques for Situation A:

- [ARIA6: Using aria-label to provide labels for objects](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA6) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA6>)
- [ARIA10: Using aria-labelledby to provide a text alternative for non-text content](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA10) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA10>)
- [G196: Using a text alternative on one item within a group of images that describes all items in the group](https://www.w3.org/WAI/WCAG22/Techniques/general/G196) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G196>)
- [H2: Combining adjacent image and text links for the same resource](https://www.w3.org/WAI/WCAG22/Techniques/html/H2) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H2>)
- [H37: Using alt attributes on img elements](https://www.w3.org/WAI/WCAG22/Techniques/html/H37) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H37>)
- [H53: Using the body of the object element](https://www.w3.org/WAI/WCAG22/Techniques/html/H53) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H53>)
- [H86: Providing text alternatives for emojis, emoticons, ASCII art, and leetspeak](https://www.w3.org/WAI/WCAG22/Techniques/html/H86) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H86>)
- [PDF1: Applying text alternatives to images with the Alt entry in PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF1) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF1>)

Situation B: If a short description can **not** serve the same purpose and present the same information as the non-text content (e.g., a chart or diagram):

- [G95: Providing short text alternatives that provide a brief description of the non-text content](https://www.w3.org/WAI/WCAG22/Techniques/general/G95) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G95>) using

one technique from each group outlined below

Short text alternative techniques for Situation B:

- [ARIA6: Using aria-label to provide labels for objects](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA6) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA6>)
- [ARIA10: Using aria-labelledby to provide a text alternative for non-text content](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA10) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA10>)
- [G196: Using a text alternative on one item within a group of images that describes all items in the group](https://www.w3.org/WAI/WCAG22/Techniques/general/G196) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G196>)
- [H2: Combining adjacent image and text links for the same resource](https://www.w3.org/WAI/WCAG22/Techniques/html/H2) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H2>)
- [H37: Using alt attributes on img elements](https://www.w3.org/WAI/WCAG22/Techniques/html/H37) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H37>)
- [H53: Using the body of the object element](https://www.w3.org/WAI/WCAG22/Techniques/html/H53) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H53>)
- [H86: Providing text alternatives for emojis, emoticons, ASCII art, and leetspeak](https://www.w3.org/WAI/WCAG22/Techniques/html/H86) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H86>)
- [PDF1: Applying text alternatives to images with the Alt entry in PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF1) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF1>)

Long text alternative techniques for Situation B:

- [ARIA15: Using aria-describedby to provide descriptions of images](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA15) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA15>)
- [G73: Providing a long description in another location with a link to it that is immediately adjacent to the non-text content](https://www.w3.org/WAI/WCAG22/Techniques/general/G73) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G73>)
- [G74: Providing a long description in text near the non-text content, with a reference to the location of the long description in the short description](https://www.w3.org/WAI/WCAG22/Techniques/general/G74) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G74>)
- [G92: Providing long description for non-text content that serves the same purpose and presents the same information](https://www.w3.org/WAI/WCAG22/Techniques/general/G92) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G92>)
- [H53: Using the body of the object element](https://www.w3.org/WAI/WCAG22/Techniques/html/H53) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H53>)

Situation C: If non-text content is a control or accepts user input:

- [G82: Providing a text alternative that identifies the purpose of the non-text content](https://www.w3.org/WAI/WCAG22/Techniques/general/G82) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G82>) using one technique from each group outlined below

Text alternative techniques for controls and input for Situation C:

- ARIA6: Using aria-label to provide labels for objects (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA6>)
- ARIA9: Using aria-labelledby to concatenate a label from several text nodes (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA9>)
- H24: Providing text alternatives for the area elements of image maps (<https://www.w3.org/WAI/WCAG22/Techniques/html/H24>)
- H30: Providing link text that describes the purpose of a link for anchor elements (<https://www.w3.org/WAI/WCAG22/Techniques/html/H30>)
- H36: Using alt attributes on images used as submit buttons (<https://www.w3.org/WAI/WCAG22/Techniques/html/H36>)
- H44: Using label elements to associate text labels with form controls (<https://www.w3.org/WAI/WCAG22/Techniques/html/H44>)
- H65: Using the title attribute to identify form controls when the label element cannot be used (<https://www.w3.org/WAI/WCAG22/Techniques/html/H65>)

Situation D: If non-text content is time-based media (including live video-only and live audio-only); a test or exercise that would be invalid if presented in text; or primarily intended to create a specific sensory experience:

- Providing a descriptive label using one technique from each group outlined below
- G68: Providing a short text alternative that describes the purpose of live audio-only and live video-only content (<https://www.w3.org/WAI/WCAG22/Techniques/general/G68>) using one technique from each group outlined below
- G100: Providing a short text alternative which is the accepted name or a descriptive name of the non-text content (<https://www.w3.org/WAI/WCAG22/Techniques/general/G100>) using one technique from each group outlined below

Short text alternative techniques for Situation D:

- ARIA6: Using aria-label to provide labels for objects (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA6>)
- ARIA10: Using aria-labelledby to provide a text alternative for non-text content (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA10>)
- G196: Using a text alternative on one item within a group of images that describes all items in the group (<https://www.w3.org/WAI/WCAG22/Techniques/general/G196>)
- H2: Combining adjacent image and text links for the same resource (<https://www.w3.org/WAI/WCAG22/Techniques/html/H2>)
- H37: Using alt attributes on img elements (<https://www.w3.org/WAI/WCAG22/Techniques/html/H37>)

- [H53: Using the body of the object element](https://www.w3.org/WAI/WCAG22/Techniques/html/H53) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H53>)
- [H86: Providing text alternatives for emojis, emoticons, ASCII art, and leetspeak](https://www.w3.org/WAI/WCAG22/Techniques/html/H86) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H86>)
- [PDF1: Applying text alternatives to images with the Alt entry in PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF1) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF1>)

Situation E: If non-text content is a CAPTCHA:

- [G143: Providing a text alternative that describes the purpose of the CAPTCHA](https://www.w3.org/WAI/WCAG22/Techniques/general/G143) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G143>) **AND** [G144: Ensuring that the web page contains another CAPTCHA serving the same purpose using a different modality](https://www.w3.org/WAI/WCAG22/Techniques/general/G144) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G144>)

Situation F: If the non-text content should be ignored by assistive technology:

- Implementing or marking the non-text content so that it will be ignored by assistive technology using one technique from each group outlined below

Techniques to indicate that text alternatives are not required for Situation F:

- [C9: Using CSS to include decorative images](https://www.w3.org/WAI/WCAG22/Techniques/css/C9) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C9>)
- [H67: Using null alt text and no title attribute on img elements for images that assistive technology should ignore](https://www.w3.org/WAI/WCAG22/Techniques/html/H67) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H67>)
- [PDF4: Hiding decorative images with the Artifact tag in PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF4) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF4>)

Advisory Techniques

- [C18: Using CSS margin and padding rules instead of spacer images for layout design](https://www.w3.org/WAI/WCAG22/Techniques/css/C18) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C18>)

Failures

- [F3: Failure of Success Criterion 1.1.1 due to using CSS to include images that convey important information](https://www.w3.org/WAI/WCAG22/Techniques/failures/F3) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F3>)
- [F13: Failure of Success Criterion 1.1.1 and 1.4.1 due to having a text alternative that](https://www.w3.org/WAI/WCAG22/Techniques/failures/F13) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F13>)

- does not include information that is conveyed by color differences in the image (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F13>)
- F20: Failure of Success Criterion 1.1.1 and 4.1.2 due to not updating text alternatives when changes to non-text content occur (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F20>)
 - F30: Failure of Success Criterion 1.1.1 and 1.2.1 due to using text alternatives that are not alternatives (e.g., filenames or placeholder text) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F30>)
 - F38: Failure of Success Criterion 1.1.1 due to not marking up decorative images in HTML in a way that allows assistive technology to ignore them (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F38>)
 - F39: Failure of Success Criterion 1.1.1 due to providing a text alternative that is not null (e.g., alt="spacer" or alt="image") for images that should be ignored by assistive technology (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F39>)
 - F65: Failure of Success Criterion 1.1.1 due to omitting the alt attribute or text alternative on img elements, area elements, and input elements of type "image" (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F65>)
 - F67: Failure of Success Criterion 1.1.1 and 1.2.1 due to providing long descriptions for non-text content that does not serve the same purpose or does not present the same information (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F67>)
 - F71: Failure of Success Criterion 1.1.1 due to using text look-alikes to represent text without providing a text alternative (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F71>)
 - F72: Failure of Success Criterion 1.1.1 due to using ASCII art without providing a text alternative (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F72>)

 SHARE |  BACK TO TOP

Guideline 1.2 – Time-based Media

Provide alternatives for time-based media.

1.2.1 Audio-only and Video-only (Prerecorded) — Level A

For prerecorded audio-only and prerecorded video-only media, the following are true, except when the audio or video is a media alternative for text and is clearly labeled as such:

 Hide full description

- **Prerecorded Audio-only:** An alternative for time-based media is provided that presents equivalent information for prerecorded audio-only content.
- **Prerecorded Video-only:** Either an alternative for time-based media or an audio

track is provided that presents equivalent information for prerecorded video-only content.

- ❶ Understanding 1.2.1 (<https://www.w3.org/WAI/WCAG22/Understanding/audio-only-and-video-only-prerecorded.html>)

▼ Hide techniques and failures for 1.2.1

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

Situation A: If the content is prerecorded audio-only:

- [G158: Providing an alternative for time-based media for audio-only content](https://www.w3.org/WAI/WCAG22/Techniques/general/G158) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G158>)

Situation B: If the content is prerecorded video-only:

- [G159: Providing an alternative for time-based media for video-only content](https://www.w3.org/WAI/WCAG22/Techniques/general/G159) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G159>)
- [G166: Providing audio that describes the important video content and describing it as such](https://www.w3.org/WAI/WCAG22/Techniques/general/G166) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G166>)

Advisory Techniques

- [H96: Using the track element to provide audio descriptions](https://www.w3.org/WAI/WCAG22/Techniques/html/H96) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H96>)

Failures

- [F30: Failure of Success Criterion 1.1.1 and 1.2.1 due to using text alternatives that are not alternatives \(e.g., filenames or placeholder text\)](https://www.w3.org/WAI/WCAG22/Techniques/failures/F30) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F30>)
- [F67: Failure of Success Criterion 1.1.1 and 1.2.1 due to providing long descriptions for non-text content that does not serve the same purpose or does not present the same information](https://www.w3.org/WAI/WCAG22/Techniques/failures/F67) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F67>)

1.2.2 Captions (Prerecorded) — Level A

Captions are provided for all prerecorded audio content in synchronized media, except when the media is a media alternative for text and is clearly labeled as such.

 [Understanding 1.2.2 \(https://www.w3.org/WAI/WCAG22/Understanding/captions-prerecorded.html\)](https://www.w3.org/WAI/WCAG22/Understanding/captions-prerecorded.html)

▼ Hide techniques and failures for 1.2.2

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques. \(https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques\)](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques)

- [G93: Providing open \(always visible\) captions \(https://www.w3.org/WAI/WCAG22/Techniques/general/G93\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G93)
- [G87: Providing closed captions \(https://www.w3.org/WAI/WCAG22/Techniques/general/G87\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G87) using any of the following techniques:
 - [SM11: Providing captions through synchronized text streams in SMIL 1.0 \(https://www.w3.org/WAI/WCAG22/Techniques/smil/SM11\)](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM11)
 - [SM12: Providing captions through synchronized text streams in SMIL 2.0 \(https://www.w3.org/WAI/WCAG22/Techniques/smil/SM12\)](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM12)
 - [H95: Using the track element to provide captions \(https://www.w3.org/WAI/WCAG22/Techniques/html/H95\)](https://www.w3.org/WAI/WCAG22/Techniques/html/H95)
 - Using any readily available media format that has a video player that supports closed captioning

Failures

- [F8: Failure of Success Criterion 1.2.2 due to captions omitting some dialogue or important sound effects \(https://www.w3.org/WAI/WCAG22/Techniques/failures/F8\)](https://www.w3.org/WAI/WCAG22/Techniques/failures/F8)
- [F75: Failure of Success Criterion 1.2.2 by providing synchronized media without captions when the synchronized media presents more information than is presented on the page \(https://www.w3.org/WAI/WCAG22/Techniques/failures/F75\)](https://www.w3.org/WAI/WCAG22/Techniques/failures/F75)
- [F74: Failure of Success Criterion 1.2.2 and 1.2.8 due to not labeling a synchronized media alternative to text as an alternative \(https://www.w3.org/WAI/WCAG22/Techniques/failures/F74\)](https://www.w3.org/WAI/WCAG22/Techniques/failures/F74)

1.2.3 Audio Description or Media Alternative (Prerecorded) — Level A

An alternative for time-based media or audio description of the prerecorded video content is provided for synchronized media, except when the media is a media alternative for text and is clearly labeled as such.

 Understanding 1.2.3 (<https://www.w3.org/WAI/WCAG22/Understanding/audio-description-or-media-alternative-prerecorded.html>)

▼ Hide techniques and failures for 1.2.3

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G69: Providing an alternative for time based media](https://www.w3.org/WAI/WCAG22/Techniques/general/G69) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G69>) using one of the following techniques:
 - [G58: Placing a link to the alternative for time-based media immediately next to the non-text content](https://www.w3.org/WAI/WCAG22/Techniques/general/G58) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G58>)
- Linking to the alternative for time-based media using one of the following techniques:
 - [H53: Using the body of the object element](https://www.w3.org/WAI/WCAG22/Techniques/html/H53) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H53>)
- [G78: Providing a second, user-selectable, audio track that includes audio descriptions](https://www.w3.org/WAI/WCAG22/Techniques/general/G78) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G78>)
- [G173: Providing a version of a movie with audio descriptions](https://www.w3.org/WAI/WCAG22/Techniques/general/G173) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G173>) using one of the following techniques:
 - [SM6: Providing audio description in SMIL 1.0](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM6) (<https://www.w3.org/WAI/WCAG22/Techniques/smil/SM6>)
 - [SM7: Providing audio description in SMIL 2.0](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM7) (<https://www.w3.org/WAI/WCAG22/Techniques/smil/SM7>)
 - [G226: Providing audio descriptions by incorporating narration in the soundtrack](https://www.w3.org/WAI/WCAG22/Techniques/general/G226) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G226>)
 - Using any player that supports audio and video
- [G8: Providing a movie with extended audio descriptions](https://www.w3.org/WAI/WCAG22/Techniques/general/G8) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G8>) using one of the following techniques:
 - [SM1: Adding extended audio description in SMIL 1.0](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM1) (<https://www.w3.org/WAI/WCAG22/Techniques/smil/SM1>)

- [SM2: Adding extended audio description in SMIL 2.0](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM2) (<https://www.w3.org/WAI/WCAG22/Techniques/smil/SM2>)
- Using any player that supports audio and video
- [G203: Using a static text alternative to describe a talking head video](https://www.w3.org/WAI/WCAG22/Techniques/general/G203) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G203>)

Advisory Techniques

- [H96: Using the track element to provide audio descriptions](https://www.w3.org/WAI/WCAG22/Techniques/html/H96) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H96>)

 [SHARE](#) |  [BACK TO TOP](#)

1.2.4 Captions (Live) — Level AA

Captions are provided for all live audio content in synchronized media.

 [Understanding 1.2.4](https://www.w3.org/WAI/WCAG22/Understanding/captions-live.html) (<https://www.w3.org/WAI/WCAG22/Understanding/captions-live.html>)

▼ Hide techniques and failures for 1.2.4

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G9: Creating captions for live synchronized media](https://www.w3.org/WAI/WCAG22/Techniques/general/G9) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G9>) **AND** [G93: Providing open \(always visible\) captions](https://www.w3.org/WAI/WCAG22/Techniques/general/G93) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G93>)
- [G9: Creating captions for live synchronized media](https://www.w3.org/WAI/WCAG22/Techniques/general/G9) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G9>) **AND** [G87: Providing closed captions](https://www.w3.org/WAI/WCAG22/Techniques/general/G87) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G87>)
 - [SM11: Providing captions through synchronized text streams in SMIL 1.0](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM11) (<https://www.w3.org/WAI/WCAG22/Techniques/smil/SM11>)
 - [SM12: Providing captions through synchronized text streams in SMIL 2.0](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM12) (<https://www.w3.org/WAI/WCAG22/Techniques/smil/SM12>)
 - Using any readily available media format that has a video player that supports closed captioning

Note: Captions may be generated using real-time text translation service.

1.2.5 Audio Description (Prerecorded) — Level AA

Audio description is provided for all prerecorded video content in synchronized media.

 Understanding 1.2.5 (<https://www.w3.org/WAI/WCAG22/Understanding/audio-description-prerecorded.html>)

▼ Hide techniques and failures for 1.2.5

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G78: Providing a second, user-selectable, audio track that includes audio descriptions](https://www.w3.org/WAI/WCAG22/Techniques/general/G78) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G78>)
- [G173: Providing a version of a movie with audio descriptions](https://www.w3.org/WAI/WCAG22/Techniques/general/G173) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G173>) using one of the following techniques:
 - [SM6: Providing audio description in SMIL 1.0](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM6) (<https://www.w3.org/WAI/WCAG22/Techniques/smil/SM6>)
 - [SM7: Providing audio description in SMIL 2.0](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM7) (<https://www.w3.org/WAI/WCAG22/Techniques/smil/SM7>)
 - [G226: Providing audio descriptions by incorporating narration in the soundtrack](https://www.w3.org/WAI/WCAG22/Techniques/general/G226) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G226>)
 - Using any player that supports audio and video
- [G8: Providing a movie with extended audio descriptions](https://www.w3.org/WAI/WCAG22/Techniques/general/G8) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G8>) using one of the following techniques:
 - [SM1: Adding extended audio description in SMIL 1.0](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM1) (<https://www.w3.org/WAI/WCAG22/Techniques/smil/SM1>)
 - [SM2: Adding extended audio description in SMIL 2.0](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM2) (<https://www.w3.org/WAI/WCAG22/Techniques/smil/SM2>)
 - Using any player that supports audio and video
- [G203: Using a static text alternative to describe a talking head video](https://www.w3.org/WAI/WCAG22/Techniques/general/G203) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G203>)

Advisory Techniques

- [H96: Using the track element to provide audio descriptions](https://www.w3.org/WAI/WCAG22/Techniques/html/H96) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H96>)

Failures

- [F113: Failure of Success Criterion 1.2.5 due to not using available pauses in dialogue to provide audio descriptions of important visual content \(https://www.w3.org/WAI/WCAG22/Techniques/failures/F113\)](https://www.w3.org/WAI/WCAG22/Techniques/failures/F113)

 [SHARE](#) |  [BACK TO TOP](#)

1.2.6 Sign Language (Prerecorded) — Level AAA

Sign language interpretation is provided for all prerecorded audio content in synchronized media.

 [Understanding 1.2.6 \(https://www.w3.org/WAI/WCAG22/Understanding/sign-language-prerecorded.html\)](https://www.w3.org/WAI/WCAG22/Understanding/sign-language-prerecorded.html)

 [Hide techniques and failures for 1.2.6](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques. \(https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques\)](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques)

- [G54: Including a sign language interpreter in the video stream \(https://www.w3.org/WAI/WCAG22/Techniques/general/G54\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G54)
- [G81: Providing a synchronized video of the sign language interpreter that can be displayed in a different viewport or overlaid on the image by the player \(https://www.w3.org/WAI/WCAG22/Techniques/general/G81\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G81) using one of the following techniques:
 - [SM13: Providing sign language interpretation through synchronized video streams in SMIL 1.0 \(https://www.w3.org/WAI/WCAG22/Techniques/smil/SM13\)](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM13)
 - [SM14: Providing sign language interpretation through synchronized video streams in SMIL 2.0 \(https://www.w3.org/WAI/WCAG22/Techniques/smil/SM14\)](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM14)

 [SHARE](#) |  [BACK TO TOP](#)

1.2.7 Extended Audio Description (Prerecorded) — Level AAA

Where pauses in foreground audio are insufficient to allow audio descriptions to convey

the sense of the video, extended audio description is provided for all prerecorded video content in synchronized media.

❶ Understanding 1.2.7 (<https://www.w3.org/WAI/WCAG22/Understanding/extended-audio-description-prerecorded.html>)

▼ Hide techniques and failures for 1.2.7

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G8: Providing a movie with extended audio descriptions](https://www.w3.org/WAI/WCAG22/Techniques/general/G8) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G8>) using one of the following techniques:
 - [SM1: Adding extended audio description in SMIL 1.0](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM1) (<https://www.w3.org/WAI/WCAG22/Techniques/smil/SM1>)
 - [SM2: Adding extended audio description in SMIL 2.0](https://www.w3.org/WAI/WCAG22/Techniques/smil/SM2) (<https://www.w3.org/WAI/WCAG22/Techniques/smil/SM2>)
 - Using any player that supports audio and video

Advisory Techniques

- [H96: Using the track element to provide audio descriptions](https://www.w3.org/WAI/WCAG22/Techniques/html/H96) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H96>)

 [SHARE](#) |  [BACK TO TOP](#)

1.2.8 Media Alternative (Prerecorded) — Level AAA

An alternative for time-based media is provided for all prerecorded synchronized media and for all prerecorded video-only media.

❶ Understanding 1.2.8 (<https://www.w3.org/WAI/WCAG22/Understanding/media-alternative-prerecorded.html>)

▼ Hide techniques and failures for 1.2.8

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

[understanding-techniques](#))

Situation A: If the content is prerecorded synchronized media:

- [G69: Providing an alternative for time based media](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G69>) using one of the following techniques:
 - [G58: Placing a link to the alternative for time-based media immediately next to the non-text content](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G58>)
- Linking to the alternative for time-based media using one of the following techniques:
 - [H53: Using the body of the object element](#) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H53>)

Situation B: If the content is prerecorded video-only:

- [G159: Providing an alternative for time-based media for video-only content](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G159>)

Failures

- [F74: Failure of Success Criterion 1.2.2 and 1.2.8 due to not labeling a synchronized media alternative to text as an alternative](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F74>)

 [SHARE](#) |  [BACK TO TOP](#)

1.2.9 Audio-only (Live) — Level AAA

An alternative for time-based media that presents equivalent information for live audio-only content is provided.

 [Understanding 1.2.9](#) (<https://www.w3.org/WAI/WCAG22/Understanding/audio-only-live.html>)

 [Hide techniques and failures for 1.2.9](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G151: Providing a link to a text transcript of a prepared statement or script if the script is followed](https://www.w3.org/WAI/WCAG22/Techniques/general/G151) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G151>)
- [G150: Providing text based alternatives for live audio-only content](https://www.w3.org/WAI/WCAG22/Techniques/general/G150) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G150>)
- [G157: Incorporating a live audio captioning service into a web page](https://www.w3.org/WAI/WCAG22/Techniques/general/G157) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G157>)

 SHARE |  BACK TO TOP

Guideline 1.3 – Adaptable

Create content that can be presented in different ways (for example simpler layout) without losing information or structure.

1.3.1 Info and Relationships — Level A

Information, structure, and relationships conveyed through presentation can be programmatically determined or are available in text.

 Understanding 1.3.1 (<https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships.html>)

▼ Hide techniques and failures for 1.3.1

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

Situation A: The technology provides semantic structure to make information and relationships conveyed through presentation programmatically determinable:

- [ARIA11: Using ARIA landmarks to identify regions of a page](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA11) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA11>)
- [H101: Using semantic HTML elements to identify regions of a page](https://www.w3.org/WAI/WCAG22/Techniques/html/H101) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H101>)
- [ARIA12: Using role=heading to identify headings](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA12) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA12>)
- [ARIA13: Using aria-labelledby to name regions and landmarks](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA13) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA13>)
- [ARIA16: Using aria-labelledby to provide a name for user interface controls](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA16) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA16>)
- [ARIA17: Using grouping roles to identify related form controls](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA17) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA17>)

[www.w3.org/WAI/WCAG22/Techniques/aria/ARIA17\)](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA17)

- [ARIA20: Using the region role to identify a region of the page \(https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA20\)](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA20)
- [G115: Using semantic elements to mark up structure \(https://www.w3.org/WAI/WCAG22/Techniques/general/G115\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G115) **AND** [H49: Using semantic markup to mark emphasized or special text \(https://www.w3.org/WAI/WCAG22/Techniques/html/H49\)](https://www.w3.org/WAI/WCAG22/Techniques/html/H49)
- [G117: Using text to convey information that is conveyed by variations in presentation of text \(https://www.w3.org/WAI/WCAG22/Techniques/general/G117\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G117)
- [G140: Separating information and structure from presentation to enable different presentations \(https://www.w3.org/WAI/WCAG22/Techniques/general/G140\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G140)
- [ARIA24: Semantically identifying a font icon with role="img" \(https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA24\)](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA24)
- Making information and relationships conveyed through presentation programmatically determinable using the following techniques:
 - [G138: Using semantic markup whenever color cues are used \(https://www.w3.org/WAI/WCAG22/Techniques/general/G138\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G138)
 - [H51: Using table markup to present tabular information \(https://www.w3.org/WAI/WCAG22/Techniques/html/H51\)](https://www.w3.org/WAI/WCAG22/Techniques/html/H51)
 - [PDF6: Using table elements for table markup in PDF Documents \(https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF6\)](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF6)
 - [PDF20: Using Adobe Acrobat Pro's Table Editor to repair mistagged tables \(https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF20\)](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF20)
 - [H39: Using caption elements to associate data table captions with data tables \(https://www.w3.org/WAI/WCAG22/Techniques/html/H39\)](https://www.w3.org/WAI/WCAG22/Techniques/html/H39)
 - [H63: Using the scope attribute to associate header cells with data cells in data tables \(https://www.w3.org/WAI/WCAG22/Techniques/html/H63\)](https://www.w3.org/WAI/WCAG22/Techniques/html/H63)
 - [H43: Using id and headers attributes to associate data cells with header cells in data tables \(https://www.w3.org/WAI/WCAG22/Techniques/html/H43\)](https://www.w3.org/WAI/WCAG22/Techniques/html/H43)
 - [H44: Using label elements to associate text labels with form controls \(https://www.w3.org/WAI/WCAG22/Techniques/html/H44\)](https://www.w3.org/WAI/WCAG22/Techniques/html/H44)
 - [H65: Using the title attribute to identify form controls when the label element cannot be used \(https://www.w3.org/WAI/WCAG22/Techniques/html/H65\)](https://www.w3.org/WAI/WCAG22/Techniques/html/H65)
 - [PDF10: Providing labels for interactive form controls in PDF documents \(https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF10\)](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF10)
 - [PDF12: Providing name, role, value information for form fields in PDF documents \(https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF12\)](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF12)

- [H71: Providing a description for groups of form controls using fieldset and legend elements](https://www.w3.org/WAI/WCAG22/Techniques/html/H71) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H71>)
- [H85: Using optgroup to group option elements inside a select](https://www.w3.org/WAI/WCAG22/Techniques/html/H85) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H85>)
- [H48: Using ol, ul and dl for lists or groups of links](https://www.w3.org/WAI/WCAG22/Techniques/html/H48) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H48>)
- [H42: Using h1-h6 to identify headings](https://www.w3.org/WAI/WCAG22/Techniques/html/H42) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H42>)
- [PDF9: Providing headings by marking content with heading tags in PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF9) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF9>)
- [PDF11: Providing links and link text using the Link annotation and the /Link structure element in PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF11) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF11>)
- [PDF17: Specifying consistent page numbering for PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF17) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF17>)
- [PDF21: Using List tags for lists in PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF21) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF21>)
- [H97: Grouping related links using the nav element](https://www.w3.org/WAI/WCAG22/Techniques/html/H97) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H97>)

Situation B: The technology in use does NOT provide the semantic structure to make the information and relationships conveyed through presentation programmatically determinable:

- [G117: Using text to convey information that is conveyed by variations in presentation of text](https://www.w3.org/WAI/WCAG22/Techniques/general/G117) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G117>)
- Making information and relationships conveyed through presentation programmatically determinable or available in text using the following techniques:
 - [T1: Using standard text formatting conventions for paragraphs](https://www.w3.org/WAI/WCAG22/Techniques/text/T1) (<https://www.w3.org/WAI/WCAG22/Techniques/text/T1>)
 - [T2: Using standard text formatting conventions for lists](https://www.w3.org/WAI/WCAG22/Techniques/text/T2) (<https://www.w3.org/WAI/WCAG22/Techniques/text/T2>)
 - [T3: Using standard text formatting conventions for headings](https://www.w3.org/WAI/WCAG22/Techniques/text/T3) (<https://www.w3.org/WAI/WCAG22/Techniques/text/T3>)

Advisory Techniques

- [C22: Using CSS to control visual presentation of text](https://www.w3.org/WAI/WCAG22/Techniques/css/C22) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C22>)

- [G162: Positioning labels to maximize predictability of relationships](https://www.w3.org/WAI/WCAG22/Techniques/general/G162) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G162>)
- [ARIA1: Using the aria-describedby property to provide a descriptive label for user interface controls](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA1) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA1>)
- [ARIA2: Identifying a required field with the aria-required property](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA2) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA2>)
- [G141: Organizing a page using headings](https://www.w3.org/WAI/WCAG22/Techniques/general/G141) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G141>)

Failures

- [F2: Failure of Success Criterion 1.3.1 due to using changes in text presentation to convey information without using the appropriate markup or text](https://www.w3.org/WAI/WCAG22/Techniques/failures/F2) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F2>)
- [F33: Failure of Success Criterion 1.3.1 and 1.3.2 due to using white space characters to create multiple columns in plain text content](https://www.w3.org/WAI/WCAG22/Techniques/failures/F33) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F33>)
- [F34: Failure of Success Criterion 1.3.1 and 1.3.2 due to using white space characters to format tables in plain text content](https://www.w3.org/WAI/WCAG22/Techniques/failures/F34) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F34>)
- [F42: Failure of Success Criteria 1.3.1, 2.1.1, 2.1.3, or 4.1.2 when emulating links](https://www.w3.org/WAI/WCAG22/Techniques/failures/F42) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F42>)
- [F43: Failure of Success Criterion 1.3.1 due to using structural markup in a way that does not represent relationships in the content](https://www.w3.org/WAI/WCAG22/Techniques/failures/F43) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F43>)
- [F46: Failure of Success Criterion 1.3.1 due to using th elements, caption elements, or non-empty summary attributes in layout tables](https://www.w3.org/WAI/WCAG22/Techniques/failures/F46) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F46>)
- [F48: Failure of Success Criterion 1.3.1 due to using the pre element to markup tabular information](https://www.w3.org/WAI/WCAG22/Techniques/failures/F48) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F48>)
- [F90: Failure of Success Criterion 1.3.1 for incorrectly associating table headers and content via the headers and id attributes](https://www.w3.org/WAI/WCAG22/Techniques/failures/F90) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F90>)
- [F91: Failure of Success Criterion 1.3.1 for not correctly marking up table headers](https://www.w3.org/WAI/WCAG22/Techniques/failures/F91) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F91>)
- [F92: Failure of Success Criterion 1.3.1 due to the use of role presentation on content which conveys semantic information](https://www.w3.org/WAI/WCAG22/Techniques/failures/F92) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F92>)
- [F111: Failure of Success Criteria 1.3.1, 2.5.3, and 4.1.2 due to a control with visible label text but no accessible name](https://www.w3.org/WAI/WCAG22/Techniques/) (<https://www.w3.org/WAI/WCAG22/Techniques/>)

[failures/F111\)](#)

 [SHARE](#) |  [BACK TO TOP](#)

1.3.2 Meaningful Sequence — Level A

When the sequence in which content is presented affects its meaning, a correct reading sequence can be programmatically determined.

 [Understanding 1.3.2](#) (<https://www.w3.org/WAI/WCAG22/Understanding/meaningful-sequence.html>)

 [Hide techniques and failures for 1.3.2](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G57: Ordering the content in a meaningful sequence](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G57>) for all the content in the web page
- Marking sequences in the content as meaningful using one of the following techniques **AND** [G57: Ordering the content in a meaningful sequence](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G57>) for those sequences
 - [H34: Using a Unicode right-to-left mark \(RLM\) or left-to-right mark \(LRM\) to mix text direction inline](#) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H34>)
 - [H56: Using the dir attribute on an inline element to resolve problems with nested directional runs](#) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H56>)
 - [C6: Positioning content based on structural markup](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C6>)
 - [C8: Using CSS letter-spacing to control spacing within a word](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C8>)
- [C27: Making the DOM order match the visual order](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C27>)
- [PDF3: Ensuring correct tab and reading order in PDF documents](#) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF3>)

Failures

- [F34: Failure of Success Criterion 1.3.1 and 1.3.2 due to using white space characters to format tables in plain text content](https://www.w3.org/WAI/WCAG22/Techniques/failures/F34) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F34>)
- [F33: Failure of Success Criterion 1.3.1 and 1.3.2 due to using white space characters to create multiple columns in plain text content](https://www.w3.org/WAI/WCAG22/Techniques/failures/F33) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F33>)
- [F32: Failure of Success Criterion 1.3.2 due to using white space characters to control spacing within a word](https://www.w3.org/WAI/WCAG22/Techniques/failures/F32) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F32>)
- [F49: Failure of Success Criterion 1.3.2 due to using an HTML layout table that does not make sense when linearized](https://www.w3.org/WAI/WCAG22/Techniques/failures/F49) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F49>)
- [F1: Failure of Success Criterion 1.3.2 due to changing the meaning of content by positioning information with CSS](https://www.w3.org/WAI/WCAG22/Techniques/failures/F1) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F1>)

 [SHARE](#) |  [BACK TO TOP](#)

1.3.3 Sensory Characteristics — Level A

Instructions provided for understanding and operating content do not rely solely on sensory characteristics of components such as shape, color, size, visual location, orientation, or sound.

Note: For requirements related to color, refer to Guideline 1.4.

 [Understanding 1.3.3](https://www.w3.org/WAI/WCAG22/Understanding/sensory-characteristics.html) (<https://www.w3.org/WAI/WCAG22/Understanding/sensory-characteristics.html>)

 [Hide techniques and failures for 1.3.3](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G96: Providing textual identification of items that otherwise rely only on sensory information to be understood](https://www.w3.org/WAI/WCAG22/Techniques/general/G96) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G96>)

Failures

- [F14: Failure of Success Criterion 1.3.3 due to identifying content only by its shape or location](https://www.w3.org/WAI/WCAG22/Techniques/failures/F14) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F14>)
- [F26: Failure of Success Criterion 1.3.3 due to using a graphical symbol alone to convey information](https://www.w3.org/WAI/WCAG22/Techniques/failures/F26) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F26>)

 [SHARE](#) |  [BACK TO TOP](#)

1.3.4 Orientation — Level AA (Added in 2.1)

Content does not restrict its view and operation to a single display orientation, such as portrait or landscape, unless a specific display orientation is essential.

Note: Examples where a particular display orientation may be essential are a bank check, a piano application, slides for a projector or television, or virtual reality content where content is not necessarily restricted to landscape or portrait display orientation.

 [Understanding 1.3.4](https://www.w3.org/WAI/WCAG22/Understanding/orientation.html) (<https://www.w3.org/WAI/WCAG22/Understanding/orientation.html>)

 [Hide techniques and failures for 1.3.4](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G214: Using a control to allow access to content in different orientations which is otherwise restricted](https://www.w3.org/WAI/WCAG22/Techniques/general/G214) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G214>)

Failures

- [F97: Failure due to locking the orientation to landscape or portrait view](https://www.w3.org/WAI/WCAG22/Techniques/failures/F97) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F97>)
- [F100: Failure of Success Criterion 1.3.4 due to showing a message asking to reorient device](https://www.w3.org/WAI/WCAG22/Techniques/failures/F100) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F100>)

 [SHARE](#) |  [BACK TO TOP](#)

1.3.5 Identify Input Purpose — Level AA (Added in 2.1)

The purpose of each input field collecting information about the user can be programmatically determined when: ▼ Hide full description

- The input field serves a purpose identified in the Input Purposes for user interface components section; and
- The content is implemented using technologies with support for identifying the expected meaning for form input data.

i Understanding 1.3.5 (<https://www.w3.org/WAI/WCAG22/Understanding/identify-input-purpose.html>)

▼ Hide techniques and failures for 1.3.5

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [H98: Using HTML autocomplete attributes](#) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H98>)

Failures

- [F107: Failure of Success Criterion 1.3.5 due to incorrect autocomplete attribute values](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F107>)

SHARE | ↑ BACK TO TOP

1.3.6 Identify Purpose — Level AAA (Added in 2.1)

In content implemented using markup languages, the purpose of user interface components, icons, and regions can be programmatically determined.

i Understanding 1.3.6 (<https://www.w3.org/WAI/WCAG22/Understanding/identify-purpose.html>)

▼ Hide techniques and failures for 1.3.6

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- Programmatically indicating the purpose of icons, regions and user interface components
- [ARIA11: Using ARIA landmarks to identify regions of a page \(https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA11\)](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA11)
- Using microdata to markup user interface components (future link)

Advisory Techniques

- Enabling user agents to find the version of the content that best fits their needs
- Using semantics to identify important features (e.g., `coga-simplification="simplest"`)
- Using `aria-invalid` and `aria-required`

 [SHARE](#) |  [BACK TO TOP](#)

Guideline 1.4 – Distinguishable

Make it easier for users to see and hear content including separating foreground from background.

1.4.1 Use of Color — Level A

Color is not used as the only visual means of conveying information, indicating an action, prompting a response, or distinguishing a visual element.

Note: This success criterion addresses color perception specifically. Other forms of perception are covered in Guideline 1.3 including programmatic access to color and other visual presentation coding.

 [Understanding 1.4.1 \(https://www.w3.org/WAI/WCAG22/Understanding/use-of-color.html\)](https://www.w3.org/WAI/WCAG22/Understanding/use-of-color.html)

 [Hide techniques and failures for 1.4.1](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques. \(https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques\)](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques)

Situation A: If the color of particular words, backgrounds, or other content is used to indicate information:

- [G14: Ensuring that information conveyed by color differences is also available in text \(https://www.w3.org/WAI/WCAG22/Techniques/general/G14\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G14)
- [G205: Including a text cue for colored form control labels \(https://www.w3.org/WAI/WCAG22/Techniques/general/G205\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G205)

[WAI/WCAG22/Techniques/general/G205](#))

- [G182: Ensuring that additional visual cues are available when text color differences are used to convey information](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G182>)
- [G183: Using a contrast ratio of 3:1 with surrounding text and providing additional visual cues on hover for links or controls where color alone is used to identify them](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G183>)

Situation B: If color is used within an image to convey information:

- [G111: Using color and pattern](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G111>)
- [G14: Ensuring that information conveyed by color differences is also available in text](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G14>)

Advisory Techniques

- [C15: Using CSS to change the presentation of a user interface component when it receives focus](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C15>)

Failures

- [F13: Failure of Success Criterion 1.1.1 and 1.4.1 due to having a text alternative that does not include information that is conveyed by color differences in the image](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F13>)
- [F73: Failure of Success Criterion 1.4.1 due to creating links that are not visually evident without color vision](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F73>)
- [F81: Failure of Success Criterion 1.4.1 due to identifying required or error fields using color differences only](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F81>)

 [SHARE](#) |  [BACK TO TOP](#)

1.4.2 Audio Control — Level A

If any audio on a web page plays automatically for more than 3 seconds, either a mechanism is available to pause or stop the audio, or a mechanism is available to control audio volume independently from the overall system volume level.

Note: Since any content that does not meet this success criterion can interfere with a user's ability to use the whole page, all content on the web page (whether or not it is used to meet other success criteria) must meet this success criterion. See Conformance Requirement 5: Non-Interference.

 Understanding 1.4.2 (<https://www.w3.org/WAI/WCAG22/Understanding/audio-control.html>)

▼ Hide techniques and failures for 1.4.2

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G60: Playing a sound that turns off automatically within three seconds](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G60>)
- [G170: Providing a control near the beginning of the web page that turns off sounds that play automatically](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G170>)
- [G171: Playing sounds only on user request](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G171>)

Failures

- [F23: Failure of 1.4.2 due to playing a sound longer than 3 seconds where there is no mechanism to turn it off](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F23>)
- [F93: Failure of Success Criterion 1.4.2 for absence of a way to pause or stop an HTML5 media element that autoplays](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F93>)

 [SHARE](#) |  [BACK TO TOP](#)

1.4.3 Contrast (Minimum) — Level AA

The visual presentation of text and images of text has a contrast ratio of at least 4.5:1, except for the following: ▼ Hide full description

- **Large Text:** Large-scale text and images of large-scale text have a contrast ratio of at least 3:1;
- **Incidental:** Text or images of text that are part of an inactive user interface component, that are pure decoration, that are not visible to anyone, or that are part

of a picture that contains significant other visual content, have no contrast requirement.

- **Logotypes:** Text that is part of a logo or brand name has no contrast requirement.

 Understanding 1.4.3 (<https://www.w3.org/WAI/WCAG22/Understanding/contrast-minimum.html>)

▼ Hide techniques and failures for 1.4.3

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

Situation A: text is less than 18 point if not bold and less than 14 point if bold

- [G18: Ensuring that a contrast ratio of at least 4.5:1 exists between text \(and images of text\) and background behind the text](https://www.w3.org/WAI/WCAG22/Techniques/general/G18) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G18>)
- [G148: Not specifying background color, not specifying text color, and not using technology features that change those defaults](https://www.w3.org/WAI/WCAG22/Techniques/general/G148) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G148>)
- [G174: Providing a control with a sufficient contrast ratio that allows users to switch to a presentation that uses sufficient contrast](https://www.w3.org/WAI/WCAG22/Techniques/general/G174) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G174>)

Situation B: text is at least 18 point if not bold and at least 14 point if bold

- [G145: Ensuring that a contrast ratio of at least 3:1 exists between text \(and images of text\) and background behind the text](https://www.w3.org/WAI/WCAG22/Techniques/general/G145) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G145>)
- [G148: Not specifying background color, not specifying text color, and not using technology features that change those defaults](https://www.w3.org/WAI/WCAG22/Techniques/general/G148) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G148>)
- [G174: Providing a control with a sufficient contrast ratio that allows users to switch to a presentation that uses sufficient contrast](https://www.w3.org/WAI/WCAG22/Techniques/general/G174) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G174>)

Advisory Techniques

- [G156: Using a technology that has commonly-available user agents that can change the foreground and background of blocks of text](https://www.w3.org/WAI/WCAG22/Techniques/general/G156) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G156>)

[WAI/WCAG22/Techniques/general/G156](#)

Failures

- F24: Failure of Success Criterion 1.4.3, 1.4.6 and 1.4.8 due to specifying foreground colors without specifying background colors or vice versa (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F24>)
- F83: Failure of Success Criterion 1.4.3 and 1.4.6 due to using background images that do not provide sufficient contrast with foreground text (or images of text) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F83>)

 [SHARE](#) |  [BACK TO TOP](#)

1.4.4 Resize Text — Level AA

Except for captions and images of text, text can be resized without assistive technology up to 200 percent without loss of content or functionality.

 [Understanding 1.4.4](#) (<https://www.w3.org/WAI/WCAG22/Understanding/resize-text.html>)

 [Hide techniques and failures for 1.4.4](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G142: Using a technology that has commonly-available user agents that support zoom](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G142>)
- Ensuring that text containers resize when the text resizes **AND** using measurements that are relative to other measurements in the content
 - [C28: Specifying the size of text containers using em units](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C28>)
 - Techniques for relative measurements
 - [C12: Using percent for font sizes](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C12>)
 - [C13: Using named font sizes](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C13>)
 - [C14: Using em units for font sizes](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C14>)

- Techniques for text container resizing
 - [SCR34: Calculating size and position in a way that scales with text size](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR34) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR34>)
 - [G146: Using liquid layout](https://www.w3.org/WAI/WCAG22/Techniques/general/G146) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G146>)
- [G178: Providing controls on the web page that allow users to incrementally change the size of all text on the page up to 200 percent](https://www.w3.org/WAI/WCAG22/Techniques/general/G178) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G178>)
- [G179: Ensuring that there is no loss of content or functionality when the text resizes and text containers do not change their width](https://www.w3.org/WAI/WCAG22/Techniques/general/G179) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G179>)

Advisory Techniques

- [C17: Scaling form elements which contain text](https://www.w3.org/WAI/WCAG22/Techniques/css/C17) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C17>)
- [C20: Using relative measurements to set column widths so that lines can average 80 characters or less when the browser is resized](https://www.w3.org/WAI/WCAG22/Techniques/css/C20) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C20>)
- [C22: Using CSS to control visual presentation of text](https://www.w3.org/WAI/WCAG22/Techniques/css/C22) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C22>)

Failures

- [F69: Failure of Success Criterion 1.4.4 when resizing visually rendered text up to 200 percent causes the text, image or controls to be clipped, truncated or obscured](https://www.w3.org/WAI/WCAG22/Techniques/failures/F69) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F69>)
- [F80: Failure of Success Criterion 1.4.4 when text-based form controls do not resize when visually rendered text is resized up to 200%](https://www.w3.org/WAI/WCAG22/Techniques/failures/F80) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F80>)
- [F94: Failure of Success Criterion 1.4.4 due to incorrect use of viewport units to resize text](https://www.w3.org/WAI/WCAG22/Techniques/failures/F94) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F94>)

 [SHARE](#) |  [BACK TO TOP](#)

1.4.5 Images of Text — Level AA

If the technologies being used can achieve the visual presentation, text is used to convey information rather than images of text except for the following:

 [Hide full description](#)

- **Customizable:** The image of text can be visually customized to the user's requirements;
- **Essential:** A particular presentation of text is essential to the information being conveyed.

Note: Logotypes (text that is part of a logo or brand name) are considered essential.

 Understanding 1.4.5 (<https://www.w3.org/WAI/WCAG22/Understanding/images-of-text.html>)

▼ Hide techniques and failures for 1.4.5

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques. \(<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>\)](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques)

- [C22: Using CSS to control visual presentation of text](https://www.w3.org/WAI/WCAG22/Techniques/css/C22) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C22>)
- [C30: Using CSS to replace text with images of text and providing user interface controls to switch](https://www.w3.org/WAI/WCAG22/Techniques/css/C30) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C30>)
- [G140: Separating information and structure from presentation to enable different presentations](https://www.w3.org/WAI/WCAG22/Techniques/general/G140) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G140>)
- [PDF7: Performing OCR on a scanned PDF document to provide actual text](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF7) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF7>)

Advisory Techniques

- [C12: Using percent for font sizes](https://www.w3.org/WAI/WCAG22/Techniques/css/C12) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C12>)
- [C13: Using named font sizes](https://www.w3.org/WAI/WCAG22/Techniques/css/C13) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C13>)
- [C14: Using em units for font sizes](https://www.w3.org/WAI/WCAG22/Techniques/css/C14) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C14>)
- [C8: Using CSS letter-spacing to control spacing within a word](https://www.w3.org/WAI/WCAG22/Techniques/css/C8) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C8>)
- [C6: Positioning content based on structural markup](https://www.w3.org/WAI/WCAG22/Techniques/css/C6) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C6>)

1.4.6 Contrast (Enhanced) — Level AAA

The visual presentation of text and images of text has a contrast ratio of at least 7:1, except for the following:

Hide full description

- **Large Text:** Large-scale text and images of large-scale text have a contrast ratio of at least 4.5:1;
- **Incidental:** Text or images of text that are part of an inactive user interface component, that are pure decoration, that are not visible to anyone, or that are part of a picture that contains significant other visual content, have no contrast requirement.
- **Logotypes:** Text that is part of a logo or brand name has no contrast requirement.

 Understanding 1.4.6 (<https://www.w3.org/WAI/WCAG22/Understanding/contrast-enhanced.html>)

Hide techniques and failures for 1.4.6

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

Situation A: text is less than 18 point if not bold and less than 14 point if bold

- [G17: Ensuring that a contrast ratio of at least 7:1 exists between text \(and images of text\) and background behind the text](https://www.w3.org/WAI/WCAG22/Techniques/general/G17) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G17>)
- [G148: Not specifying background color, not specifying text color, and not using technology features that change those defaults](https://www.w3.org/WAI/WCAG22/Techniques/general/G148) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G148>)
- [G174: Providing a control with a sufficient contrast ratio that allows users to switch to a presentation that uses sufficient contrast](https://www.w3.org/WAI/WCAG22/Techniques/general/G174) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G174>)

Situation B: text is as least 18 point if not bold and at least 14 point if bold

- [G18: Ensuring that a contrast ratio of at least 4.5:1 exists between text \(and images of text\) and background behind the text](https://www.w3.org/WAI/WCAG22/Techniques/general/G18) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G18>)
- [G148: Not specifying background color, not specifying text color, and not using technology features that change those defaults](https://www.w3.org/WAI/WCAG22/Techniques/general/G148) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G148>)

[WCAG22/Techniques/general/G148](#)

- [G174: Providing a control with a sufficient contrast ratio that allows users to switch to a presentation that uses sufficient contrast](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G174>)

Advisory Techniques

- [G156: Using a technology that has commonly-available user agents that can change the foreground and background of blocks of text](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G156>)

Failures

- [F24: Failure of Success Criterion 1.4.3, 1.4.6 and 1.4.8 due to specifying foreground colors without specifying background colors or vice versa](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F24>)
- [F83: Failure of Success Criterion 1.4.3 and 1.4.6 due to using background images that do not provide sufficient contrast with foreground text \(or images of text\)](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F83>)

 [SHARE](#) |  [BACK TO TOP](#)

1.4.7 Low or No Background Audio — Level AAA

For prerecorded audio-only content that (1) contains primarily speech in the foreground, (2) is not an audio CAPTCHA or audio logo, and (3) is not vocalization intended to be primarily musical expression such as singing or rapping, at least one of the following is true:

 Hide full description

- **No Background:** The audio does not contain background sounds.
- **Turn Off:** The background sounds can be turned off.
- **20 dB:** The background sounds are at least 20 decibels lower than the foreground speech content, with the exception of occasional sounds that last for only one or two seconds.

Note: Per the definition of "decibel," background sound that meets this requirement will be approximately four times quieter than the foreground speech content.

 [Understanding 1.4.7](#) (<https://www.w3.org/WAI/WCAG22/Understanding/low-or-no-background-audio.html>)

▼ Hide techniques and failures for 1.4.7

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G56: Mixing audio files so that non-speech sounds are at least 20 decibels lower than the speech audio content](https://www.w3.org/WAI/WCAG22/Techniques/general/G56) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G56>)

 [SHARE](#) |  [BACK TO TOP](#)

1.4.8 Visual Presentation — Level AAA

For the visual presentation of blocks of text, a mechanism is available to achieve the following:

▼ Hide full description

- Foreground and background colors can be selected by the user.
- Width is no more than 80 characters or glyphs (40 if CJK).
- Text is not justified (aligned to both the left and the right margins).
- Line spacing (leading) is at least space-and-a-half within paragraphs, and paragraph spacing is at least 1.5 times larger than the line spacing.
- Text can be resized without assistive technology up to 200 percent in a way that does not require the user to scroll horizontally to read a line of text on a full-screen window.

Note 1: Content is not required to use these values. The requirement is that a mechanism is available for users to change these presentation aspects. The mechanism can be provided by the browser or other user agent. Content is not required to provide the mechanism.

Note 2: Writing systems for some languages use different presentation aspects to improve readability and legibility. If a presentation aspect in this success criterion is not used in a writing system, content in that writing system does not need to use that presentation setting and can conform without it. Authors are encouraged to follow guidance for improving readability and legibility of text in their writing system.

 [Understanding 1.4.8](https://www.w3.org/WAI/WCAG22/Understanding/visual-presentation.html) (<https://www.w3.org/WAI/WCAG22/Understanding/visual-presentation.html>)

▼ Hide techniques and failures for 1.4.8

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

First Requirement: Techniques to ensure foreground and background colors can be selected by the user

- [C23: Specifying text and background colors of secondary content such as banners, features and navigation in CSS while not specifying text and background colors of the main content](https://www.w3.org/WAI/WCAG22/Techniques/css/C23) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C23>)
- [C25: Specifying borders and layout in CSS to delineate areas of a web page while not specifying text and text-background colors](https://www.w3.org/WAI/WCAG22/Techniques/css/C25) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C25>)
- [G156: Using a technology that has commonly-available user agents that can change the foreground and background of blocks of text](https://www.w3.org/WAI/WCAG22/Techniques/general/G156) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G156>)
- [G148: Not specifying background color, not specifying text color, and not using technology features that change those defaults](https://www.w3.org/WAI/WCAG22/Techniques/general/G148) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G148>)
- [G175: Providing a multi color selection tool on the page for foreground and background colors](https://www.w3.org/WAI/WCAG22/Techniques/general/G175) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G175>)

Second Requirement: Techniques to ensure width is no more than 80 characters or glyphs (40 if CJK)

- [G204: Not interfering with the user agent's reflow of text as the viewing window is narrowed](https://www.w3.org/WAI/WCAG22/Techniques/general/G204) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G204>)
- [C20: Using relative measurements to set column widths so that lines can average 80 characters or less when the browser is resized](https://www.w3.org/WAI/WCAG22/Techniques/css/C20) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C20>)

Third Requirement: Techniques to ensure text is not justified (aligned to both the left and the right margins)

- [C19: Specifying alignment either to the left or right in CSS](https://www.w3.org/WAI/WCAG22/Techniques/css/C19) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C19>)
- [G172: Providing a mechanism to remove full justification of text](https://www.w3.org/WAI/WCAG22/Techniques/general/G172) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G172>)
- [G169: Aligning text on only one side](https://www.w3.org/WAI/WCAG22/Techniques/general/G169) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G169>)

Fourth Requirement: Techniques to ensure line spacing (leading) is at least space-and-a-half within paragraphs, and paragraph spacing is at least 1.5 times larger than the line spacing

- [G188: Providing a button on the page to increase line spaces and paragraph spaces](https://www.w3.org/WAI/WCAG22/Techniques/general/G188) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G188>)
- [C21: Specifying line spacing in CSS](https://www.w3.org/WAI/WCAG22/Techniques/css/C21) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C21>)

Fifth Requirement: Techniques to ensure text can be resized without assistive technology up to 200 percent in a way that does not require the user to scroll horizontally to read a line of text on a full-screen window

- [G204: Not interfering with the user agent's reflow of text as the viewing window is narrowed](https://www.w3.org/WAI/WCAG22/Techniques/general/G204) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G204>)
- [G146: Using liquid layout](https://www.w3.org/WAI/WCAG22/Techniques/general/G146) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G146>) **AND** using measurements that are relative to other measurements in the content
 - [C12: Using percent for font sizes](https://www.w3.org/WAI/WCAG22/Techniques/css/C12) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C12>)
 - [C13: Using named font sizes](https://www.w3.org/WAI/WCAG22/Techniques/css/C13) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C13>)
 - [C14: Using em units for font sizes](https://www.w3.org/WAI/WCAG22/Techniques/css/C14) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C14>)
 - [C24: Using percentage values in CSS for container sizes](https://www.w3.org/WAI/WCAG22/Techniques/css/C24) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C24>)
 - [SCR34: Calculating size and position in a way that scales with text size](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR34) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR34>)
- [G206: Providing options within the content to switch to a layout that does not require the user to scroll horizontally to read a line of text](https://www.w3.org/WAI/WCAG22/Techniques/general/G206) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G206>)

Failures

- [F24: Failure of Success Criterion 1.4.3, 1.4.6 and 1.4.8 due to specifying foreground colors without specifying background colors or vice versa](https://www.w3.org/WAI/WCAG22/Techniques/failures/F24) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F24>)
- [F88: Failure of Success Criterion 1.4.8 due to using text that is justified \(aligned to both the left and the right margins\)](https://www.w3.org/WAI/WCAG22/Techniques/failures/F88) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F88>)

1.4.9 Images of Text (No Exception) — Level AAA

Images of text are only used for pure decoration or where a particular presentation of text is essential to the information being conveyed.

Note: Logotypes (text that is part of a logo or brand name) are considered essential.

 Understanding 1.4.9 (<https://www.w3.org/WAI/WCAG22/Understanding/images-of-text-no-exception.html>)

▼ Hide techniques and failures for 1.4.9

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [C22: Using CSS to control visual presentation of text](https://www.w3.org/WAI/WCAG22/Techniques/css/C22) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C22>)
- [C30: Using CSS to replace text with images of text and providing user interface controls to switch](https://www.w3.org/WAI/WCAG22/Techniques/css/C30) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C30>)
- [G140: Separating information and structure from presentation to enable different presentations](https://www.w3.org/WAI/WCAG22/Techniques/general/G140) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G140>)
- [PDF7: Performing OCR on a scanned PDF document to provide actual text](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF7) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF7>)

Advisory Techniques

- [C12: Using percent for font sizes](https://www.w3.org/WAI/WCAG22/Techniques/css/C12) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C12>)
- [C13: Using named font sizes](https://www.w3.org/WAI/WCAG22/Techniques/css/C13) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C13>)
- [C14: Using em units for font sizes](https://www.w3.org/WAI/WCAG22/Techniques/css/C14) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C14>)
- [C8: Using CSS letter-spacing to control spacing within a word](https://www.w3.org/WAI/WCAG22/Techniques/css/C8) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C8>)
- [C6: Positioning content based on structural markup](https://www.w3.org/WAI/WCAG22/Techniques/css/C6) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C6>)

1.4.10 Reflow — Level AA (Added in 2.1)

Content can be presented without loss of information or functionality, and without requiring scrolling in two dimensions for:  Hide full description

- Vertical scrolling content at a width equivalent to 320 CSS pixels;
- Horizontal scrolling content at a height equivalent to 256 CSS pixels.

Except for parts of the content which require two-dimensional layout for usage or meaning.

Note 1: 320 CSS pixels is equivalent to a starting viewport width of 1280 CSS pixels wide at 400% zoom. For web content which is designed to scroll horizontally (e.g., with vertical text), 256 CSS pixels is equivalent to a starting viewport height of 1024 CSS pixels at 400% zoom.

Note 2: Examples of content which requires two-dimensional layout are images required for understanding (such as maps and diagrams), video, games, presentations, data tables (not individual cells), and interfaces where it is necessary to keep toolbars in view while manipulating content. It is acceptable to provide two-dimensional scrolling for such parts of the content.

 Understanding 1.4.10 (<https://www.w3.org/WAI/WCAG22/Understanding/reflow.html>)

 Hide techniques and failures for 1.4.10

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [C32: Using media queries and grid CSS to reflow columns](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C32>)
- [C31: Using CSS Flexbox to reflow content](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C31>)
- [C33: Allowing for Reflow with Long URLs and Strings of Text](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C33>)
- [C38: Using CSS width, max-width and flexbox to fit labels and inputs](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C38>)
- [SCR34: Calculating size and position in a way that scales with text size](#) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR34>)
- [G206: Providing options within the content to switch to a layout that does not require the user to scroll horizontally to read a line of text](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G206>)

- [G224: Accounting for meaningful text indentation and Reflow](https://www.w3.org/WAI/WCAG22/Techniques/general/G224) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G224>)
- [G225: Section panels that scroll horizontally are designed to fit within a width of 320 CSS pixels on a vertically scrolling page](https://www.w3.org/WAI/WCAG22/Techniques/general/G225) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G225>)
- Using PDF/UA when creating PDFs (Potential future technique)

Advisory Techniques

- [C34: Using media queries to un-fixing sticky headers / footers](https://www.w3.org/WAI/WCAG22/Techniques/css/C34) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C34>)
- [C37: Using CSS max-width and height to fit images](https://www.w3.org/WAI/WCAG22/Techniques/css/C37) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C37>)
- CSS, Reflowing simple data tables (Potential future technique)
- CSS, Fitting data cells within the width of the viewport (Potential future technique)
- Mechanism to allow mobile view at any time (Potential future technique)
- Alternate view supporting Reflow for otherwise excepted content (Potential future technique)

Failures

- [F102: Failure of Success Criterion 1.4.10 due to content disappearing and not being available when content has reflowed](https://www.w3.org/WAI/WCAG22/Techniques/failures/F102) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F102>)

 [SHARE](#) |  [BACK TO TOP](#)

1.4.11 Non-text Contrast — Level AA (Added in 2.1)

The visual presentation of the following have a contrast ratio of at least 3:1 against adjacent color(s):  [Hide full description](#)

- **User Interface Components:** Visual information required to identify user interface components and states, except for inactive components or where the appearance of the component is determined by the user agent and not modified by the author;
- **Graphical Objects:** Parts of graphics required to understand the content, except when a particular presentation of graphics is essential to the information being conveyed.

 [Understanding 1.4.11](https://www.w3.org/WAI/WCAG22/Understanding/non-text-contrast.html) (<https://www.w3.org/WAI/WCAG22/Understanding/non-text-contrast.html>)

▼ Hide techniques and failures for 1.4.11

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

Situation A: Color is used to identify user interface components or used to identify user interface component states

- [G195: Using an author-supplied, visible focus indicator](https://www.w3.org/WAI/WCAG22/Techniques/general/G195) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G195>)
- [G174: Providing a control with a sufficient contrast ratio that allows users to switch to a presentation that uses sufficient contrast](https://www.w3.org/WAI/WCAG22/Techniques/general/G174) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G174>)

Situation B: Color is required to understand graphical content

- [G207: Ensuring that a contrast ratio of 3:1 is provided for icons](https://www.w3.org/WAI/WCAG22/Techniques/general/G207) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G207>)
- [G209: Provide sufficient contrast at the boundaries between adjoining colors](https://www.w3.org/WAI/WCAG22/Techniques/general/G209) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G209>)

Failures

- [F78: Failure of Success Criterion 1.4.11, 2.4.7 and 2.4.13 due to styling element outlines and borders in a way that removes or renders non-visible the visual focus indicator](https://www.w3.org/WAI/WCAG22/Techniques/failures/F78) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F78>)

 [SHARE](#) |  [BACK TO TOP](#)

1.4.12 Text Spacing — Level AA (Added in 2.1)

In content implemented using markup languages that support the following text style properties, no loss of content or functionality occurs by setting all of the following and by changing no other style property:

▼ Hide full description

- Line height (line spacing) to at least 1.5 times the font size;
- Spacing following paragraphs to at least 2 times the font size;

- Letter spacing (tracking) to at least 0.12 times the font size;
- Word spacing to at least 0.16 times the font size.

Exception: Human languages and scripts that do not make use of one or more of these text style properties in written text can conform using only the properties that exist for that combination of language and script.

Note 1: Content is not required to use these text spacing values. The requirement is to ensure that when a user overrides the authored text spacing, content or functionality is not lost.

Note 2: Writing systems for some languages use different text spacing settings, such as paragraph start indent. Authors are encouraged to follow locally available guidance for improving readability and legibility of text in their writing system.

❶ Understanding 1.4.12 (<https://www.w3.org/WAI/WCAG22/Understanding/text-spacing.html>)

▼ Hide techniques and failures for 1.4.12

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [C36: Allowing for text spacing override](https://www.w3.org/WAI/WCAG22/Techniques/css/C36) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C36>)
- [C35: Allowing for text spacing without wrapping](https://www.w3.org/WAI/WCAG22/Techniques/css/C35) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C35>)

Advisory Techniques

- [C8: Using CSS letter-spacing to control spacing within a word](https://www.w3.org/WAI/WCAG22/Techniques/css/C8) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C8>)
- [C21: Specifying line spacing in CSS](https://www.w3.org/WAI/WCAG22/Techniques/css/C21) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C21>)
- [C28: Specifying the size of text containers using em units](https://www.w3.org/WAI/WCAG22/Techniques/css/C28) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C28>)

Failures

- [F104: Failure of Success Criterion 1.4.12 due to clipped or overlapped content when text spacing is adjusted](https://www.w3.org/WAI/WCAG22/Techniques/failures/F104) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F104>)

1.4.13 Content on Hover or Focus — Level AA (Added in 2.1)

Where receiving and then removing pointer hover or keyboard focus triggers additional content to become visible and then hidden, the following are true:

- **Dismissible:** A mechanism is available to dismiss the additional content without moving pointer hover or keyboard focus, unless the additional content communicates an input error or does not obscure or replace other content;
- **Hoverable:** If pointer hover can trigger the additional content, then the pointer can be moved over the additional content without the additional content disappearing;
- **Persistent:** The additional content remains visible until the hover or focus trigger is removed, the user dismisses it, or its information is no longer valid.

Exception: The visual presentation of the additional content is controlled by the user agent and is not modified by the author.

Note 1: Examples of additional content controlled by the user agent include browser tooltips created through use of the [HTML \(HyperText Markup Language\)](#) title attribute [[HTML](#)].

Note 2: Custom tooltips, sub-menus, and other nonmodal popups that display on hover and focus are examples of additional content covered by this criterion.

Note 3: This criterion applies to content that appears in addition to the triggering component itself. Since hidden components that are made visible on keyboard focus (such as links used to skip to another part of a page) do not present additional content they are not covered by this criterion.

 [Understanding 1.4.13](#) (<https://www.w3.org/WAI/WCAG22/Understanding/content-on-hover-or-focus.html>)

 [Hide techniques and failures for 1.4.13](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [SCR39: Making content on focus or hover hoverable, dismissible, and persistent](#) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR39>)
- ARIA: Using role="tooltip" (Potential future technique)
- CSS: Using hover and focus pseudo classes (Potential future technique)

Failures

- [F95: Failure of Success Criterion 1.4.13 due to content shown on hover not being hoverable](https://www.w3.org/WAI/WCAG22/Techniques/failures/F95) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F95>)
- Failure to make content dismissible without moving pointer hover or keyboard focus (Potential future technique)
- Failure to meet by content on hover or focus not remaining visible until dismissed or invalid (Potential future technique)

 [SHARE](#) |  [BACK TO TOP](#)

Principle 2 – Operable

User interface components and navigation must be operable.

Guideline 2.1 – Keyboard Accessible

Make all functionality available from a keyboard.

2.1.1 Keyboard — Level A

All functionality of the content is operable through a keyboard interface without requiring specific timings for individual keystrokes, except where the underlying function requires input that depends on the path of the user's movement and not just the endpoints.

Note 1: This exception relates to the underlying function, not the input technique. For example, if using handwriting to enter text, the input technique (handwriting) requires path-dependent input but the underlying function (text input) does not.

Note 2: This does not forbid and should not discourage providing mouse input or other input methods in addition to keyboard operation.

 [Understanding 2.1.1](https://www.w3.org/WAI/WCAG22/Understanding/keyboard.html) (<https://www.w3.org/WAI/WCAG22/Understanding/keyboard.html>)

 [Hide techniques and failures for 2.1.1](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G202: Ensuring keyboard control for all functionality](https://www.w3.org/WAI/) (<https://www.w3.org/WAI/>)

[WCAG22/Techniques/general/G202](#)

- Ensuring keyboard control using one of the following techniques:
 - [H91: Using HTML form controls and links](#) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H91>)
 - [PDF3: Ensuring correct tab and reading order in PDF documents](#) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF3>)
 - [PDF11: Providing links and link text using the Link annotation and the /Link structure element in PDF documents](#) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF11>)
 - [PDF23: Providing interactive form controls in PDF documents](#) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF23>)
- [G90: Providing keyboard-triggered event handlers](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G90>) using one of the following techniques:
 - [SCR20: Using both keyboard and other device-specific functions](#) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR20>)
 - [SCR35: Making actions keyboard accessible by using the onclick event of anchors and buttons](#) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR35>)
 - [SCR2: Using redundant keyboard and mouse event handlers](#) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR2>)

Advisory Techniques

- Using WAI-ARIA role, state, and value attributes if repurposing static elements as interactive user interface components (future link) **AND** [SCR29: Adding keyboard-accessible actions to static HTML elements](#) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR29>)

Failures

- [F54: Failure of Success Criterion 2.1.1 due to using only pointing-device-specific event handlers \(including gesture\) for a function](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F54>)
- [F55: Failure of Success Criteria 2.1.1, 2.4.7, 2.4.13, and 3.2.1 due to using script to remove focus when focus is received](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F55>)
- [F42: Failure of Success Criteria 1.3.1, 2.1.1, 2.1.3, or 4.1.2 when emulating links](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F42>)

2.1.2 No Keyboard Trap — Level A

If keyboard focus can be moved to a component of the page using a keyboard interface, then focus can be moved away from that component using only a keyboard interface, and, if it requires more than unmodified arrow or tab keys or other standard exit methods, the user is advised of the method for moving focus away.

Note: Since any content that does not meet this success criterion can interfere with a user's ability to use the whole page, all content on the web page (whether it is used to meet other success criteria or not) must meet this success criterion. See Conformance Requirement 5: Non-Interference.

 Understanding 2.1.2 (<https://www.w3.org/WAI/WCAG22/Understanding/no-keyboard-trap.html>)

▼ Hide techniques and failures for 2.1.2

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G21: Ensuring that users are not trapped in content](https://www.w3.org/WAI/WCAG22/Techniques/general/G21) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G21>)

Failures

- [F10: Failure of Success Criterion 2.1.2 and Conformance Requirement 5 due to combining multiple content formats in a way that traps users inside one format type](https://www.w3.org/WAI/WCAG22/Techniques/failures/F10) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F10>)

 SHARE |  BACK TO TOP

2.1.3 Keyboard (No Exception) — Level AAA

All functionality of the content is operable through a keyboard interface without requiring specific timings for individual keystrokes.

 Understanding 2.1.3 (<https://www.w3.org/WAI/WCAG22/Understanding/keyboard-no-exception.html>)

▼ Hide techniques and failures for 2.1.3

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- No additional techniques exist for this success criterion. Follow [techniques for Success Criterion 2.1.1](https://www.w3.org/WAI/WCAG22/Understanding/keyboard#techniques) (<https://www.w3.org/WAI/WCAG22/Understanding/keyboard#techniques>). If that is not possible because there is a requirement for path-dependent input, then it is not possible to meet this Level AAA success criterion.

 [SHARE](#) |  [BACK TO TOP](#)

2.1.4 Character Key Shortcuts — Level A (Added in 2.1)

If a keyboard shortcut is implemented in content using only letter (including upper- and lower-case letters), punctuation, number, or symbol characters, then at least one of the following is true:  [Hide full description](#)

- **Turn off:** A mechanism is available to turn the shortcut off;
- **Remap:** A mechanism is available to remap the shortcut to include one or more non-printable keyboard keys (e.g., Ctrl, Alt);
- **Active only on focus:** The keyboard shortcut for a user interface component is only active when that component has focus.

 [Understanding 2.1.4](https://www.w3.org/WAI/WCAG22/Understanding/character-key-shortcuts.html) (<https://www.w3.org/WAI/WCAG22/Understanding/character-key-shortcuts.html>)

 [Hide techniques and failures for 2.1.4](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G217: Providing a mechanism to allow users to remap or turn off character key shortcuts](https://www.w3.org/WAI/WCAG22/Techniques/general/G217) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G217>)

Failures

- [F99: Failure of Success Criterion 2.1.4 due to implementing character key shortcuts that cannot be turned off or remapped \(<https://www.w3.org/WAI/WCAG22/Techniques/failures/F99>\)](#)

 SHARE |  BACK TO TOP

Guideline 2.2 – Enough Time

Provide users enough time to read and use content.

2.2.1 Timing Adjustable — Level A

For each time limit that is set by the content, at least one of the following is true:

 Hide full description

- **Turn off:** The user is allowed to turn off the time limit before encountering it; or
- **Adjust:** The user is allowed to adjust the time limit before encountering it over a wide range that is at least ten times the length of the default setting; or
- **Extend:** The user is warned before time expires and given at least 20 seconds to extend the time limit with a simple action (for example, "press the space bar"), and the user is allowed to extend the time limit at least ten times; or
- **Real-time Exception:** The time limit is a required part of a real-time event (for example, an auction), and no alternative to the time limit is possible; or
- **Essential Exception:** The time limit is essential and extending it would invalidate the activity; or
- **20 Hour Exception:** The time limit is longer than 20 hours.

Note: This success criterion helps ensure that users can complete tasks without unexpected changes in content or context that are a result of a time limit. This success criterion should be considered in conjunction with Success Criterion 3.2.1, which puts limits on changes of content or context as a result of user action.

 Understanding 2.2.1 (<https://www.w3.org/WAI/WCAG22/Understanding/timing-adjustable.html>)

 Hide techniques and failures for 2.2.1

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

Situation A: If there are session time limits:

- [G133: Providing a checkbox on the first page of a multipart form that allows users to ask for longer session time limit or no session time limit](https://www.w3.org/WAI/WCAG22/Techniques/general/G133) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G133>)
- [G198: Providing a way for the user to turn the time limit off](https://www.w3.org/WAI/WCAG22/Techniques/general/G198) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G198>)

Situation B: If a time limit is controlled by a script on the page:

- [G198: Providing a way for the user to turn the time limit off](https://www.w3.org/WAI/WCAG22/Techniques/general/G198) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G198>)
- [G180: Providing the user with a means to set the time limit to 10 times the default time limit](https://www.w3.org/WAI/WCAG22/Techniques/general/G180) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G180>)
- [SCR16: Providing a script that warns the user a time limit is about to expire](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR16) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR16>) **AND**
[SCR1: Allowing the user to extend the default time limit](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR1) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR1>)

Situation C: If there are time limits on reading:

- [G4: Allowing the content to be paused and restarted from where it was paused](https://www.w3.org/WAI/WCAG22/Techniques/general/G4) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G4>)
- [G198: Providing a way for the user to turn the time limit off](https://www.w3.org/WAI/WCAG22/Techniques/general/G198) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G198>)
- [SCR33: Using script to scroll content, and providing a mechanism to pause it](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR33) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR33>)
- [SCR36: Providing a mechanism to allow users to display moving, scrolling, or auto-updating text in a static window or area](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR36) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR36>)

Failures

- [F40: Failure due to using meta redirect with a time limit](https://www.w3.org/WAI/WCAG22/Techniques/failures/F40) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F40>)
- [F41: Failure of Success Criterion 2.2.1, 2.2.4, and 3.2.5 due to using meta refresh to reload the page](https://www.w3.org/WAI/WCAG22/Techniques/failures/F41) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F41>)
- [F58: Failure of Success Criterion 2.2.1 due to using server-side techniques to automatically redirect pages after a time-out](https://www.w3.org/WAI/WCAG22/Techniques/failures/F58) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F58>)

2.2.2 Pause, Stop, Hide — Level A

For moving, blinking, scrolling, or auto-updating information, all of the following are true:

 Hide full description

- **Moving, blinking, scrolling:** For any moving, blinking or scrolling information that (1) starts automatically, (2) lasts more than five seconds, and (3) is presented in parallel with other content, there is a mechanism for the user to pause, stop, or hide it unless the movement, blinking, or scrolling is part of an activity where it is essential; and
- **Auto-updating:** For any auto-updating information that (1) starts automatically and (2) is presented in parallel with other content, there is a mechanism for the user to pause, stop, or hide it or to control the frequency of the update unless the auto-updating is part of an activity where it is essential.

Note 1: For requirements related to flickering or flashing content, refer to Guideline 2.3.

Note 2: Since any content that does not meet this success criterion can interfere with a user's ability to use the whole page, all content on the web page (whether it is used to meet other success criteria or not) must meet this success criterion. See Conformance Requirement 5: Non-Interference.

Note 3: Content that is updated periodically by software or that is streamed to the user agent is not required to preserve or present information that is generated or received between the initiation of the pause and resuming presentation, as this may not be technically possible, and in many situations could be misleading to do so.

Note 4: An animation that occurs as part of a preload phase or similar situation can be considered essential if interaction cannot occur during that phase for all users and if not indicating progress could confuse users or cause them to think that content was frozen or broken.

 Understanding 2.2.2 (<https://www.w3.org/WAI/WCAG22/Understanding/pause-stop-hide.html>)

 Hide techniques and failures for 2.2.2

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G4: Allowing the content to be paused and restarted from where it was paused](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G4>)
- [SCR33: Using script to scroll content, and providing a mechanism to pause it](#) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR33>)

- [G11: Creating content that blinks for less than 5 seconds](https://www.w3.org/WAI/WCAG22/Techniques/general/G11) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G11>)
- [G152: Setting animated gif images to stop blinking after n cycles \(within 5 seconds\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G152) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G152>)
- [SCR22: Using scripts to control blinking and stop it in five seconds or less](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR22) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR22>)
- [G186: Using a control in the web page that stops moving, blinking, or auto-updating content](https://www.w3.org/WAI/WCAG22/Techniques/general/G186) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G186>)
- [G191: Providing a link, button, or other mechanism that reloads the page without any blinking content](https://www.w3.org/WAI/WCAG22/Techniques/general/G191) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G191>)

Failures

- [F16: Failure of Success Criterion 2.2.2 due to including scrolling content where movement is not essential to the activity without also including a mechanism to pause and restart the content](https://www.w3.org/WAI/WCAG22/Techniques/failures/F16) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F16>)
- [F112: Failure of Success Criterion 2.2.2 due to using blinking content that lasts for more than five seconds without a mechanism to stop it](https://www.w3.org/WAI/WCAG22/Techniques/failures/F112) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F112>)
- [F50: Failure of Success Criterion 2.2.2 due to a script that causes a blink effect without a mechanism to stop the blinking at 5 seconds or less](https://www.w3.org/WAI/WCAG22/Techniques/failures/F50) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F50>)
- [F7: Failure of Success Criterion 2.2.2 due to an object or applet that has blinking content without a mechanism to pause the content that blinks for more than five seconds](https://www.w3.org/WAI/WCAG22/Techniques/failures/F7) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F7>)

 [SHARE](#) |  [BACK TO TOP](#)

2.2.3 No Timing — Level AAA

Timing is not an essential part of the event or activity presented by the content, except for non-interactive synchronized media and real-time events.

 [Understanding 2.2.3](https://www.w3.org/WAI/WCAG22/Understanding/no-timing.html) (<https://www.w3.org/WAI/WCAG22/Understanding/no-timing.html>)

 [Hide techniques and failures for 2.2.3](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See

[Understanding Techniques.](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques) (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G5: Allowing users to complete an activity without any time limit](https://www.w3.org/WAI/WCAG22/Techniques/general/G5) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G5>)

 [SHARE](#) |  [BACK TO TOP](#)

2.2.4 Interruptions — Level AAA

Interruptions can be postponed or suppressed by the user, except interruptions involving an emergency.

 [Understanding 2.2.4](#) (<https://www.w3.org/WAI/WCAG22/Understanding/interruptions.html>)

 [Hide techniques and failures for 2.2.4](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques.](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques) (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G75: Providing a mechanism to postpone any updating of content](https://www.w3.org/WAI/WCAG22/Techniques/general/G75) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G75>)
- [G76: Providing a mechanism to request an update of the content instead of updating automatically](https://www.w3.org/WAI/WCAG22/Techniques/general/G76) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G76>)
- [SCR14: Using scripts to make nonessential alerts optional](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR14) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR14>)

Failures

- [F40: Failure due to using meta redirect with a time limit](https://www.w3.org/WAI/WCAG22/Techniques/failures/F40) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F40>)
- [F41: Failure of Success Criterion 2.2.1, 2.2.4, and 3.2.5 due to using meta refresh to reload the page](https://www.w3.org/WAI/WCAG22/Techniques/failures/F41) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F41>)

 [SHARE](#) |  [BACK TO TOP](#)

2.2.5 Re-authenticating — Level AAA

When an authenticated session expires, the user can continue the activity without loss of data after re-authenticating.

 Understanding 2.2.5 (<https://www.w3.org/WAI/WCAG22/Understanding/re-authenticating.html>)

▼ Hide techniques and failures for 2.2.5

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- Providing options to continue without loss of data using one of the following techniques:
 - [G105: Saving data so that it can be used after a user re-authenticates](https://www.w3.org/WAI/WCAG22/Techniques/general/G105) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G105>)
 - [G181: Encoding user data as hidden or encrypted data in a re-authorization page](https://www.w3.org/WAI/WCAG22/Techniques/general/G181) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G181>)

Note: Refer to Techniques for Addressing Success Criterion 2.2.1 for techniques related to providing notifications about time limits.

Failures

- [F12: Failure of Success Criterion 2.2.5 due to having a session time limit without a mechanism for saving user's input and re-establishing that information upon re-authentication](https://www.w3.org/WAI/WCAG22/Techniques/failures/F12) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F12>)

 [SHARE](#) |  [BACK TO TOP](#)

2.2.6 Timeouts — Level AAA (Added in 2.1)

Users are warned of the duration of any user inactivity that could cause data loss, unless the data is preserved for more than 20 hours when the user does not take any actions.

Note: Privacy regulations may require explicit user consent before user identification has been authenticated and before user data is preserved. In cases where the user is a minor, explicit consent may not be solicited in most jurisdictions, countries or regions. Consultation with privacy professionals and legal counsel is advised when considering data preservation as an approach to satisfy this success criterion.

 Understanding 2.2.6 (<https://www.w3.org/WAI/WCAG22/Understanding/timeouts.html>)

▼ Hide techniques and failures for 2.2.6

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- Setting a session timeout to occur following at least 20 hours of inactivity
- Storing user data for more than 20 hours
- Providing a warning of the duration of user inactivity at the start of a process

 [SHARE](#) |  [BACK TO TOP](#)

Guideline 2.3 – Seizures and Physical Reactions

Do not design content in a way that is known to cause seizures or physical reactions.

2.3.1 Three Flashes or Below Threshold — Level A

Web pages do not contain anything that flashes more than three times in any one second period, or the flash is below the general flash and red flash thresholds.

Note: Since any content that does not meet this success criterion can interfere with a user's ability to use the whole page, all content on the web page (whether it is used to meet other success criteria or not) must meet this success criterion. See Conformance Requirement 5: Non-Interference.

 [Understanding 2.3.1](https://www.w3.org/WAI/WCAG22/Understanding/three-flashes-or-below-threshold.html) (<https://www.w3.org/WAI/WCAG22/Understanding/three-flashes-or-below-threshold.html>)

▼ Hide techniques and failures for 2.3.1

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G19: Ensuring that no component of the content flashes more than three times in any 1-second period](https://www.w3.org/WAI/WCAG22/Techniques/general/G19) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G19>)
- [G176: Keeping the flashing area small enough](https://www.w3.org/WAI/WCAG22/Techniques/general/G176) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G176>)
- [G15: Using a tool to ensure that content does not violate the general flash threshold or red flash threshold](https://www.w3.org/WAI/WCAG22/Techniques/general/G15) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G15>)

 SHARE |  BACK TO TOP

2.3.2 Three Flashes — Level AAA

Web pages do not contain anything that flashes more than three times in any one second period.

 Understanding 2.3.2 (<https://www.w3.org/WAI/WCAG22/Understanding/three-flashes.html>)

▼ Hide techniques and failures for 2.3.2

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G19: Ensuring that no component of the content flashes more than three times in any 1-second period](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G19>)

 SHARE |  BACK TO TOP

2.3.3 Animation from Interactions — Level AAA (Added in 2.1)

Motion animation triggered by interaction can be disabled, unless the animation is essential to the functionality or the information being conveyed.

 Understanding 2.3.3 (<https://www.w3.org/WAI/WCAG22/Understanding/animation-from-interactions.html>)

▼ Hide techniques and failures for 2.3.3

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [C39: Using the CSS prefers-reduced-motion query to prevent motion](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C39>)
- [SCR40: Using the CSS prefers-reduced-motion query in JavaScript to prevent motion](#) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR40>)

- Gx: Allowing users to set a preference that prevents animation

 SHARE |  BACK TO TOP

Guideline 2.4 – Navigable

Provide ways to help users navigate, find content, and determine where they are.

2.4.1 Bypass Blocks — Level A

A mechanism is available to bypass blocks of content that are repeated on multiple web pages.

 Understanding 2.4.1 (<https://www.w3.org/WAI/WCAG22/Understanding/bypass-blocks.html>)

 Hide techniques and failures for 2.4.1

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- Creating links to skip blocks of repeated material using one of the following techniques:
 - [G1: Adding a link at the top of each page that goes directly to the main content area](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G1>)
 - [G123: Adding a link at the beginning of a block of repeated content to go to the end of the block](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G123>)
 - [G124: Adding links at the top of the page to each area of the content](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G124>)
- Grouping blocks of repeated material in a way that can be skipped using one of the following techniques:
 - [ARIA11: Using ARIA landmarks to identify regions of a page](#) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA11>)
 - [H69: Providing heading elements at the beginning of each section of content](#) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H69>)
 - [PDF9: Providing headings by marking content with heading tags in PDF documents](#) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF9>)
 - [H64: Using the title attribute of the iframe element](#) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H64>)
 - [SCR28: Using an expandable and collapsible menu to bypass block of content](#)

(<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR28>)

Advisory Techniques

- [C6: Positioning content based on structural markup](https://www.w3.org/WAI/WCAG22/Techniques/css/C6) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C6>)
- [H97: Grouping related links using the nav element](https://www.w3.org/WAI/WCAG22/Techniques/html/H97) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H97>)

 [SHARE](#) |  [BACK TO TOP](#)

2.4.2 Page Titled — Level A

Web pages have titles that describe topic or purpose.

 [Understanding 2.4.2](#) (<https://www.w3.org/WAI/WCAG22/Understanding/page-titled.html>)

▼ Hide techniques and failures for 2.4.2

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G88: Providing descriptive titles for web pages](https://www.w3.org/WAI/WCAG22/Techniques/general/G88) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G88>) **AND** associating a title with a web page
 - [H25: Providing a title using the title element](https://www.w3.org/WAI/WCAG22/Techniques/html/H25) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H25>)
 - [PDF18: Specifying the document title using the Title entry in the document information dictionary of a PDF document](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF18) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF18>)

Advisory Techniques

- [G127: Identifying a web page's relationship to a larger collection of web pages](https://www.w3.org/WAI/WCAG22/Techniques/general/G127) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G127>)

Failures

- [F25: Failure of Success Criterion 2.4.2 due to the title of a web page not identifying](#)

[the contents \(<https://www.w3.org/WAI/WCAG22/Techniques/failures/F25>\)](https://www.w3.org/WAI/WCAG22/Techniques/failures/F25) [SHARE](#) |  [BACK TO TOP](#)

2.4.3 Focus Order — Level A

If a web page can be navigated sequentially and the navigation sequences affect meaning or operation, focusable components receive focus in an order that preserves meaning and operability.

 [Understanding 2.4.3 \(<https://www.w3.org/WAI/WCAG22/Understanding/focus-order.html>\)](#)

 [Hide techniques and failures for 2.4.3](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G59: Placing the interactive elements in an order that follows sequences and relationships within the content \(<https://www.w3.org/WAI/WCAG22/Techniques/general/G59>\)](#)
- Giving focus to elements in an order that follows sequences and relationships within the content using one of the following techniques:
 - [C27: Making the DOM order match the visual order \(<https://www.w3.org/WAI/WCAG22/Techniques/css/C27>\)](#)
 - [PDF3: Ensuring correct tab and reading order in PDF documents \(<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF3>\)](#)
- Changing a web page dynamically using one of the following techniques:
 - [SCR26: Inserting dynamic content into the Document Object Model immediately following its trigger element \(<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR26>\)](#)
 - [H102: Creating modal dialogs with the HTML dialog element \(<https://www.w3.org/WAI/WCAG22/Techniques/html/H102>\)](#)
 - [SCR27: Reordering page sections using the Document Object Model \(<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR27>\)](#)

Failures

- [F44: Failure of Success Criterion 2.4.3 due to using tabindex to create a tab order](#)

[that does not preserve meaning and operability](https://www.w3.org/WAI/WCAG22/Techniques/failures/F44) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F44>)

- [F85: Failure of Success Criterion 2.4.3 due to using dialogs or menus that are not adjacent to their trigger control in the sequential navigation order](https://www.w3.org/WAI/WCAG22/Techniques/failures/F85) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F85>)

 [SHARE](#) |  [BACK TO TOP](#)

2.4.4 Link Purpose (In Context) — Level A

The purpose of each link can be determined from the link text alone or from the link text together with its programmatically determined link context, except where the purpose of the link would be ambiguous to users in general.

 [Understanding 2.4.4](https://www.w3.org/WAI/WCAG22/Understanding/link-purpose-in-context.html) (<https://www.w3.org/WAI/WCAG22/Understanding/link-purpose-in-context.html>)

 [Hide techniques and failures for 2.4.4](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G91: Providing link text that describes the purpose of a link](https://www.w3.org/WAI/WCAG22/Techniques/general/G91) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G91>)
- [H30: Providing link text that describes the purpose of a link for anchor elements](https://www.w3.org/WAI/WCAG22/Techniques/html/H30) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H30>)
- [H24: Providing text alternatives for the area elements of image maps](https://www.w3.org/WAI/WCAG22/Techniques/html/H24) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H24>)
- Allowing the user to choose short or long link text using one of the following techniques:
 - [G189: Providing a control near the beginning of the web page that changes the link text](https://www.w3.org/WAI/WCAG22/Techniques/general/G189) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G189>)
 - [SCR30: Using scripts to change the link text](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR30) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR30>)
- [G53: Identifying the purpose of a link using link text combined with the text of the enclosing sentence](https://www.w3.org/WAI/WCAG22/Techniques/general/G53) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G53>)
- Providing a supplemental description of the purpose of a link using one of the following techniques:
 - [H33: Supplementing link text with the title attribute](https://www.w3.org/WAI/WCAG22/Techniques/html/H33) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H33>)

[WAI/WCAG22/Techniques/html/H33](#)

- [C7: Using CSS to hide a portion of the link text \(https://www.w3.org/WAI/WCAG22/Techniques/css/C7\)](#)
- Identifying the purpose of a link using link text combined with programmatically determined link context using one of the following techniques:
 - [ARIA7: Using aria-labelledby for link purpose \(https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA7\)](#)
 - [ARIA8: Using aria-label for link purpose \(https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA8\)](#)
 - [H77: Identifying the purpose of a link using link text combined with its enclosing list item \(https://www.w3.org/WAI/WCAG22/Techniques/html/H77\)](#)
 - [H78: Identifying the purpose of a link using link text combined with its enclosing paragraph \(https://www.w3.org/WAI/WCAG22/Techniques/html/H78\)](#)
 - [H79: Identifying the purpose of a link in a data table using the link text combined with its enclosing table cell and associated table header cells \(https://www.w3.org/WAI/WCAG22/Techniques/html/H79\)](#)
 - [H81: Identifying the purpose of a link in a nested list using link text combined with the parent list item under which the list is nested \(https://www.w3.org/WAI/WCAG22/Techniques/html/H81\)](#)
- [G91: Providing link text that describes the purpose of a link \(https://www.w3.org/WAI/WCAG22/Techniques/general/G91\)](#) **AND** semantically indicating links
 - [PDF11: Providing links and link text using the Link annotation and the /Link structure element in PDF documents \(https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF11\)](#)
 - [PDF13: Providing replacement text using the /Alt entry for links in PDF documents \(https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF13\)](#)

Advisory Techniques

- [H2: Combining adjacent image and text links for the same resource \(https://www.w3.org/WAI/WCAG22/Techniques/html/H2\)](#)
- [H80: Identifying the purpose of a link using link text combined with the preceding heading element \(https://www.w3.org/WAI/WCAG22/Techniques/html/H80\)](#)

Failures

- [F63: Failure of Success Criterion 2.4.4 due to providing link context only in content that is not related to the link \(https://www.w3.org/WAI/WCAG22/Techniques/failures/F63\)](#)

- F89: Failure of Success Criteria 2.4.4, 2.4.9 and 4.1.2 due to not providing an accessible name for an image which is the only content in a link (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F89>)

 SHARE |  BACK TO TOP

2.4.5 Multiple Ways — Level AA

More than one way is available to locate a web page within a set of web pages except where the web page is the result of, or a step in, a process.

 [Understanding 2.4.5 \(https://www.w3.org/WAI/WCAG22/Understanding/multiple-ways.html\)](https://www.w3.org/WAI/WCAG22/Understanding/multiple-ways.html)

 Hide techniques and failures for 2.4.5

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques. \(https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques\)](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques)

- Using two or more of the following techniques:
 - [G125: Providing links to navigate to related web pages \(https://www.w3.org/WAI/WCAG22/Techniques/general/G125\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G125)
 - [G64: Providing a Table of Contents \(https://www.w3.org/WAI/WCAG22/Techniques/general/G64\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G64)
 - [G63: Providing a site map \(https://www.w3.org/WAI/WCAG22/Techniques/general/G63\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G63)
 - [G161: Providing a search function to help users find content \(https://www.w3.org/WAI/WCAG22/Techniques/general/G161\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G161)
 - [G126: Providing a list of links to all other web pages \(https://www.w3.org/WAI/WCAG22/Techniques/general/G126\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G126)
 - [G185: Linking to all of the pages on the site from the home page \(https://www.w3.org/WAI/WCAG22/Techniques/general/G185\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G185)

Advisory Techniques

- [PDF2: Creating bookmarks in PDF documents \(https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF2\)](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF2)

 SHARE |  BACK TO TOP

2.4.6 Headings and Labels — Level AA

Headings and labels describe topic or purpose.

❶ Understanding 2.4.6 (<https://www.w3.org/WAI/WCAG22/Understanding/headings-and-labels.html>)

▼ Hide techniques and failures for 2.4.6

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G130: Providing descriptive headings](https://www.w3.org/WAI/WCAG22/Techniques/general/G130) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G130>)
- [G131: Providing descriptive labels](https://www.w3.org/WAI/WCAG22/Techniques/general/G131) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G131>)

Note: Headings and labels must be programmatically determined, per Success Criterion 1.3.1.

 SHARE |  BACK TO TOP

2.4.7 Focus Visible — Level AA

Any keyboard operable user interface has a mode of operation where the keyboard focus indicator is visible.

❶ Understanding 2.4.7 (<https://www.w3.org/WAI/WCAG22/Understanding/focus-visible.html>)

▼ Hide techniques and failures for 2.4.7

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G149: Using user interface components that are highlighted by the user agent when they receive focus](https://www.w3.org/WAI/WCAG22/Techniques/general/G149) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G149>)
- [C15: Using CSS to change the presentation of a user interface component when it receives focus](https://www.w3.org/WAI/WCAG22/Techniques/css/C15) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C15>)
- [G165: Using the default focus indicator for the platform so that high visibility default focus indicators will carry over](https://www.w3.org/WAI/WCAG22/Techniques/general/G165) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G165>)
- [G195: Using an author-supplied, visible focus indicator](https://www.w3.org/WAI/WCAG22/Techniques/general/G195) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G195>)

[WCAG22/Techniques/general/G195](#)

- [C40: Creating a two-color focus indicator to ensure sufficient contrast with all components](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C40>)
- [C45: Using CSS :focus-visible to provide keyboard focus indication](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C45>)
- [SCR31: Using script to change the background color or border of the element with focus](#) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR31>)

Failures

- [F55: Failure of Success Criteria 2.1.1, 2.4.7, 2.4.13, and 3.2.1 due to using script to remove focus when focus is received](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F55>)
- [F78: Failure of Success Criterion 1.4.11, 2.4.7 and 2.4.13 due to styling element outlines and borders in a way that removes or renders non-visible the visual focus indicator](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F78>)

 [SHARE](#) |  [BACK TO TOP](#)

2.4.8 Location — Level AAA

Information about the user's location within a set of web pages is available.

 [Understanding 2.4.8](#) (<https://www.w3.org/WAI/WCAG22/Understanding/location.html>)

 [Hide techniques and failures for 2.4.8](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G65: Providing a breadcrumb trail](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G65>)
- [G63: Providing a site map](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G63>)
- [G128: Indicating current location within navigation bars](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G128>)
- [G127: Identifying a web page's relationship to a larger collection of web pages](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G127>)

Advisory Techniques

- [PDF14: Providing running headers and footers in PDF documents \(https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF14\)](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF14)
- [PDF17: Specifying consistent page numbering for PDF documents \(https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF17\)](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF17)

 [SHARE](#) |  [BACK TO TOP](#)

2.4.9 Link Purpose (Link Only) — Level AAA

A mechanism is available to allow the purpose of each link to be identified from link text alone, except where the purpose of the link would be ambiguous to users in general.

 [Understanding 2.4.9 \(https://www.w3.org/WAI/WCAG22/Understanding/link-purpose-link-only.html\)](https://www.w3.org/WAI/WCAG22/Understanding/link-purpose-link-only.html)

 [Hide techniques and failures for 2.4.9](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques. \(https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques\)](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques)

- [ARIA8: Using aria-label for link purpose \(https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA8\)](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA8)
- [G91: Providing link text that describes the purpose of a link \(https://www.w3.org/WAI/WCAG22/Techniques/general/G91\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G91)
- [H30: Providing link text that describes the purpose of a link for anchor elements \(https://www.w3.org/WAI/WCAG22/Techniques/html/H30\)](https://www.w3.org/WAI/WCAG22/Techniques/html/H30)
- [H24: Providing text alternatives for the area elements of image maps \(https://www.w3.org/WAI/WCAG22/Techniques/html/H24\)](https://www.w3.org/WAI/WCAG22/Techniques/html/H24)
- Allowing the user to choose short or long link text using one of the following techniques:
 - [G189: Providing a control near the beginning of the web page that changes the link text \(https://www.w3.org/WAI/WCAG22/Techniques/general/G189\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G189)
 - [SCR30: Using scripts to change the link text \(https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR30\)](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR30)
- Providing a supplemental description of the purpose of a link using one of the following techniques:
 - [C7: Using CSS to hide a portion of the link text \(https://www.w3.org/WAI/WCAG22/Techniques/css/C7\)](https://www.w3.org/WAI/WCAG22/Techniques/css/C7)

WCAG22/Techniques/css/C7)

- Semantically indicating links using one of the following techniques:
 - [PDF11: Providing links and link text using the /Link annotation and the /Link structure element in PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF11) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF11>)
 - [PDF13: Providing replacement text using the /Alt entry for links in PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF13) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF13>)

Advisory Techniques

- [H2: Combining adjacent image and text links for the same resource](https://www.w3.org/WAI/WCAG22/Techniques/html/H2) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H2>)
- [H33: Supplementing link text with the title attribute](https://www.w3.org/WAI/WCAG22/Techniques/html/H33) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H33>)

Failures

- [F84: Failure of Success Criterion 2.4.9 due to using a non-specific link such as "click here" or "more" without a mechanism to change the link text to specific text.](https://www.w3.org/WAI/WCAG22/Techniques/failures/F84) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F84>)
- [F89: Failure of Success Criteria 2.4.4, 2.4.9 and 4.1.2 due to not providing an accessible name for an image which is the only content in a link](https://www.w3.org/WAI/WCAG22/Techniques/failures/F89) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F89>)

 [SHARE](#) |  [BACK TO TOP](#)

2.4.10 Section Headings — Level AAA

Section headings are used to organize the content.

Note 1: "Heading" is used in its general sense and includes titles and other ways to add a heading to different types of content.

Note 2: This success criterion covers sections within writing, not user interface components. User interface components are covered under Success Criterion 4.1.2.

 [Understanding 2.4.10](https://www.w3.org/WAI/WCAG22/Understanding/section-headings.html) (<https://www.w3.org/WAI/WCAG22/Understanding/section-headings.html>)

 [Hide techniques and failures for 2.4.10](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G141: Organizing a page using headings](https://www.w3.org/WAI/WCAG22/Techniques/general/G141) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G141>)
- [H69: Providing heading elements at the beginning of each section of content](https://www.w3.org/WAI/WCAG22/Techniques/html/H69) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H69>)

 [SHARE](#) |  [BACK TO TOP](#)

2.4.11 Focus Not Obscured (Minimum) — Level AA (Added in 2.2)

When a user interface component receives keyboard focus, the component is not entirely hidden due to author-created content.

Note 1: Where content in a configurable interface can be repositioned by the user, then only the initial positions of user-movable content are considered for testing and conformance of this success criterion.

Note 2: Content opened by the *user* may obscure the component receiving focus. If the user can reveal the focused component without advancing the keyboard focus, the component with focus is not considered visually hidden due to author-created content.

 Understanding 2.4.11 (<https://www.w3.org/WAI/WCAG22/Understanding/focus-not-obsured-minimum.html>)

 Hide techniques and failures for 2.4.11

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [C43: Using CSS scroll-padding to un-obscure content](https://www.w3.org/WAI/WCAG22/Techniques/css/C43) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C43>)

Failures

- [F110: Failure of Success Criterion 2.4.11 Focus Not Obscured \(Minimum\) due to a sticky footer or header completely hiding focused elements](https://www.w3.org/WAI/WCAG22/Techniques/failures/F110) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F110>)

2.4.12 Focus Not Obscured (Enhanced) — Level AAA (Added in 2.2)

When a user interface component receives keyboard focus, no part of the component is hidden by author-created content.

Understanding 2.4.12 (<https://www.w3.org/WAI/WCAG22/Understanding/focus-not-obscured-enhanced.html>)

Hide techniques and failures for 2.4.12

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [C43: Using CSS scroll-padding to un-obscure content](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C43>)

Failures

- An interaction that causes content to appear over the component with keyboard focus, visually covering part of the focus indicator. This behavior might be encountered with advertising or promotional material meant to provide more information about a product as the user navigates through a catalogue.
- A page has a sticky footer (attached to the bottom of the viewport). When tabbing down the page, a focused item is partially obscured by the footer because content in the viewport scrolls without sufficient scroll padding (<https://www.w3.org/TR/css-scroll-snap/#propdef-scroll-padding>).

2.4.13 Focus Appearance — Level AAA (Added in 2.2)

When the keyboard focus indicator is visible, an area of the focus indicator meets all the following:

Hide full description

- is at least as large as the area of a 2 CSS pixel thick perimeter of the unfocused component or sub-component, and
- has a contrast ratio of at least 3:1 between the same pixels in the focused and

unfocused states.

Exceptions:

- The focus indicator is determined by the user agent and cannot be adjusted by the author, or
- The focus indicator and the indicator's background color are not modified by the author.

Note 1: What is perceived as the user interface component or sub-component (to determine the perimeter) depends on its visual presentation. The visual presentation includes the component's visible content, border, and component-specific background. It does not include shadow and glow effects outside the component's content, background, or border.

Note 2: Examples of sub-components that may receive a focus indicator are menu items in an opened drop-down menu, or focusable cells in a grid.

Note 3: Contrast calculations can be based on colors defined within the technology (such as [HTML \(HyperText Markup Language\)](#), CSS, and SVG). Pixels modified by user agent resolution enhancements and anti-aliasing can be ignored.

 Understanding 2.4.13 (<https://www.w3.org/WAI/WCAG22/Understanding/focus-appearance.html>)

▼ Hide techniques and failures for 2.4.13

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G195: Using an author-supplied, visible focus indicator](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G195>)
- [C40: Creating a two-color focus indicator to ensure sufficient contrast with all components](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C40>)
- [C41: Creating a strong focus indicator within the component](#) (<https://www.w3.org/WAI/WCAG22/Techniques/css/C41>)

Failures

- [F55: Failure of Success Criteria 2.1.1, 2.4.7, 2.4.13, and 3.2.1 due to using script to remove focus when focus is received](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F55>)
- [F78: Failure of Success Criterion 1.4.11, 2.4.7 and 2.4.13 due to styling element](#)

outlines and borders in a way that removes or renders non-visible the visual focus indicator (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F78>)

 [SHARE](#) |  [BACK TO TOP](#)

Guideline 2.5 – Input Modalities

Make it easier for users to operate functionality through various inputs beyond keyboard.

2.5.1 Pointer Gestures — Level A (Added in 2.1)

All functionality that uses multipoint or path-based gestures for operation can be operated with a single pointer without a path-based gesture, unless a multipoint or path-based gesture is essential.

Note: This requirement applies to web content that interprets pointer actions (i.e., this does not apply to actions that are required to operate the user agent or assistive technology).

 [Understanding 2.5.1](#) (<https://www.w3.org/WAI/WCAG22/Understanding/pointer-gestures.html>)

 [Hide techniques and failures for 2.5.1](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G215: Providing controls to achieve the same result as path based or multipoint gestures](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G215>)
- [G216: Providing single point activation for a control slider](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G216>)

Failures

- [F105: Failure of Success Criterion 2.5.1 due to providing functionality via a path-based gesture without simple pointer alternative](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F105>)

 [SHARE](#) |  [BACK TO TOP](#)

2.5.2 Pointer Cancellation — Level A (Added in 2.1)

For functionality that can be operated using a single pointer, at least one of the following is true: ▼ Hide full description

- **No Down-Event:** The down-event of the pointer is not used to execute any part of the function;
- **Abort or Undo:** Completion of the function is on the up-event, and a mechanism is available to abort the function before completion or to undo the function after completion;
- **Up Reversal:** The up-event reverses any outcome of the preceding down-event;
- **Essential:** Completing the function on the down-event is essential.

Note 1: Functions that emulate a keyboard or numeric keypad key press are considered essential.

Note 2: This requirement applies to web content that interprets pointer actions (i.e., this does not apply to actions that are required to operate the user agent or assistive technology).

i Understanding 2.5.2 (<https://www.w3.org/WAI/WCAG22/Understanding/pointer-cancellation.html>)

▼ Hide techniques and failures for 2.5.2

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G210: Ensuring that drag-and-drop actions can be cancelled](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G210>)
- [G212: Using native controls to ensure functionality is triggered on the up-event.](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G212>)
- Touch events are only triggered when touch is removed from a control (Potential future technique)

Failures

- [F101: Failure of Success Criterion 2.5.2 due to activating a control on the down-event](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F101>)

 [SHARE](#) |  [BACK TO TOP](#)

2.5.3 Label in Name — Level A (Added in 2.1)

For user interface components with labels that include text or images of text, the name contains the text that is presented visually.

Note: A best practice is to have the text of the label at the start of the name.

 [Understanding 2.5.3](https://www.w3.org/WAI/WCAG22/Understanding/label-in-name.html) (<https://www.w3.org/WAI/WCAG22/Understanding/label-in-name.html>)

▼ Hide techniques and failures for 2.5.3

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G208: Including the text of the visible label as part of the accessible name](https://www.w3.org/WAI/WCAG22/Techniques/general/G208) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G208>)
- [G211: Matching the accessible name to the visible label](https://www.w3.org/WAI/WCAG22/Techniques/general/G211) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G211>)

Advisory Techniques

- [G162: Positioning labels to maximize predictability of relationships](https://www.w3.org/WAI/WCAG22/Techniques/general/G162) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G162>)
- If an icon has no accompanying text, consider using its hover text as its accessible name (Potential future technique)

Failures

- [F96: Failure due to the accessible name not containing the visible label text](https://www.w3.org/WAI/WCAG22/Techniques/failures/F96) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F96>)
- [F111: Failure of Success Criteria 1.3.1, 2.5.3, and 4.1.2 due to a control with visible label text but no accessible name](https://www.w3.org/WAI/WCAG22/Techniques/failures/F111) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F111>)
- Accessible name contains the visible label text, but the words of the visible label are not in the same order as they are in the visible label text (Potential future technique)
- Accessible name contains the visible label text, but one or more other words are interspersed in the label (Potential future technique)

2.5.4 Motion Actuation — Level A (Added in 2.1)

Functionality that can be operated by device motion or user motion can also be operated by user interface components and responding to the motion can be disabled to prevent accidental actuation, except when:

 Hide full description

- **Supported Interface:** The motion is used to operate functionality through an accessibility supported interface;
- **Essential:** The motion is essential for the function and doing so would invalidate the activity.

 Understanding 2.5.4 (<https://www.w3.org/WAI/WCAG22/Understanding/motion-actuation.html>)

 Hide techniques and failures for 2.5.4

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G213: Provide conventional controls and an application setting for motion activated input](#) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G213>)
- GXXX: Supporting system level features which allow the user to disable motion actuation

Failures

- [F106: Failure due to inability to deactivate motion actuation](#) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F106>)
- FXXX: Failure of Success Criterion 2.5.4 due to disrupting or disabling system level features which allow the user to disable motion actuation

 SHARE |  BACK TO TOP

2.5.5 Target Size (Enhanced) — Level AAA (Added in 2.1)

The size of the target for pointer inputs is at least 44 by 44 CSS pixels except when:

 Hide full description

- **Equivalent:** The target is available through an equivalent link or control on the same

page that is at least 44 by 44 CSS pixels;

- **Inline:** The target is in a sentence or block of text;
- **User Agent Control:** The size of the target is determined by the user agent and is not modified by the author;
- **Essential:** A particular presentation of the target is essential to the information being conveyed.

 Understanding 2.5.5 (<https://www.w3.org/WAI/WCAG22/Understanding/target-size-enhanced.html>)

▼ Hide techniques and failures for 2.5.5

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- Ensuring that targets are at least 44 by 44 CSS pixels

Advisory Techniques

- Ensuring inline links provide sufficiently large activation target

Failures

- Failure of Success Criterion 2.5.5 due to target being less than 44 by 44 CSS pixels

 [SHARE](#) |  [BACK TO TOP](#)

2.5.6 Concurrent Input Mechanisms — Level AAA (Added in 2.1)

Web content does not restrict use of input modalities available on a platform except where the restriction is essential, required to ensure the security of the content, or required to respect user settings.

 Understanding 2.5.6 (<https://www.w3.org/WAI/WCAG22/Understanding/concurrent-input-mechanisms.html>)

▼ Hide techniques and failures for 2.5.6

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- Only using high-level, input-agnostic event handlers, such as `focus`, `blur`, `click`, in Javascript (Potential future technique)
- Registering event handlers for keyboard/keyboard-like and pointer inputs simultaneously in Javascript; see [Example 1 in Pointer Events Level 2](https://www.w3.org/TR/pointerevents2/#example_1) (https://www.w3.org/TR/pointerevents2/#example_1) (Potential future technique)

Failures

- [F98: Failure due to interactions being limited to touch-only on touchscreen devices](https://www.w3.org/WAI/WCAG22/Techniques/failures/F98) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F98>)

 [SHARE](#) |  [BACK TO TOP](#)

2.5.7 Dragging Movements — Level AA (Added in 2.2)

All functionality that uses a dragging movement for operation can be achieved by a single pointer without dragging, unless dragging is essential or the functionality is determined by the user agent and not modified by the author.

Note: This requirement applies to web content that interprets pointer actions (i.e., this does not apply to actions that are required to operate the user agent or assistive technology).

 [Understanding 2.5.7](https://www.w3.org/WAI/WCAG22/Understanding/dragging-movements.html) (<https://www.w3.org/WAI/WCAG22/Understanding/dragging-movements.html>)

 [Hide techniques and failures for 2.5.7](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G219: Ensuring that an alternative is available for dragging movements that operate on content](https://www.w3.org/WAI/WCAG22/Techniques/general/G219) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G219>)

Failures

- [F108: Failure of Success Criterion 2.5.7 Dragging Movements due to not providing a single pointer method that does not require a dragging movement \(https://www.w3.org/WAI/WCAG22/Techniques/failures/F108\)](https://www.w3.org/WAI/WCAG22/Techniques/failures/F108)

 [SHARE](#) |  [BACK TO TOP](#)

2.5.8 Target Size (Minimum) — Level AA (Added in 2.2)

The size of the target for pointer inputs is at least 24 by 24 CSS pixels, except when:

 [Hide full description](#)

- **Spacing:** Undersized targets (those less than 24 by 24 CSS pixels) are positioned so that if a 24 CSS pixel diameter circle is centered on the bounding box of each, the circles do not intersect another target or the circle for another undersized target;
- **Equivalent:** The function can be achieved through a different control on the same page that meets this criterion;
- **Inline:** The target is in a sentence or its size is otherwise constrained by the line-height of non-target text;
- **User Agent Control:** The size of the target is determined by the user agent and is not modified by the author;
- **Essential:** A particular presentation of the target is essential or is legally required for the information being conveyed.

Note 1: Targets that allow for values to be selected spatially based on position within the target are considered one target for the purpose of the success criterion. Examples include sliders, color pickers displaying a gradient of colors, or editable areas where you position the cursor.

Note 2: For inline targets the line-height should be interpreted as perpendicular to the flow of text. For example, in a language displayed vertically, the line-height would be horizontal.

 [Understanding 2.5.8 \(https://www.w3.org/WAI/WCAG22/Understanding/target-size-minimum.html\)](https://www.w3.org/WAI/WCAG22/Understanding/target-size-minimum.html)

 [Hide techniques and failures for 2.5.8](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques. \(https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques\)](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques)

- C42: Using min-height and min-width to ensure sufficient target spacing (<https://www.w3.org/WAI/WCAG22/Techniques/css/C42>)

[!\[\]\(6e5bbb627821c955d737473165ee8f9b_img.jpg\) SHARE](#) | [!\[\]\(656d06ff0082b164bc7dcfa171299a17_img.jpg\) BACK TO TOP](#)

Principle 3 – Understandable

Information and the operation of the user interface must be understandable.

Guideline 3.1 – Readable

Make text content readable and understandable.

3.1.1 Language of Page — Level A

The default human language of each web page can be programmatically determined.

 Understanding 3.1.1 (<https://www.w3.org/WAI/WCAG22/Understanding/language-of-page.html>)

▼ Hide techniques and failures for 3.1.1

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [H57: Using the language attribute on the HTML element](#) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H57>)
- [PDF16: Setting the default language using the /Lang entry in the document catalog of a PDF document](#) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF16>)
- [PDF19: Specifying the language for a passage or phrase with the Lang entry in PDF documents](#) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF19>)

Advisory Techniques

- [SVR5: Specifying the default language in the HTTP header](#) (<https://www.w3.org/WAI/WCAG22/Techniques/server-side-script/SVR5>)

 SHARE |  BACK TO TOP

3.1.2 Language of Parts — Level AA

The human language of each passage or phrase in the content can be programmatically determined except for proper names, technical terms, words of indeterminate language, and words or phrases that have become part of the vernacular of the immediately surrounding text.

 Understanding 3.1.2 (<https://www.w3.org/WAI/WCAG22/Understanding/language-of-parts.html>)

▼ Hide techniques and failures for 3.1.2

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See

[Understanding Techniques. \(https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques\)](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques)

- [H58: Using language attributes to identify changes in the human language \(https://www.w3.org/WAI/WCAG22/Techniques/html/H58\)](https://www.w3.org/WAI/WCAG22/Techniques/html/H58)
- [PDF19: Specifying the language for a passage or phrase with the Lang entry in PDF documents \(https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF19\)](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF19)

 [SHARE](#) |  [BACK TO TOP](#)

3.1.3 Unusual Words — Level AAA

A mechanism is available for identifying specific definitions of words or phrases used in an unusual or restricted way, including idioms and jargon.

 [Understanding 3.1.3 \(https://www.w3.org/WAI/WCAG22/Understanding/unusual-words.html\)](#)

 [Hide techniques and failures for 3.1.3](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques. \(https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques\)](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques)

Situation A: If the word or phrase has a unique meaning within the web page:

- [G101: Providing the definition of a word or phrase used in an unusual or restricted way \(https://www.w3.org/WAI/WCAG22/Techniques/general/G101\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G101) for the first occurrence of the word or phrase in a web page using one of the following techniques:
 - [G55: Linking to definitions \(https://www.w3.org/WAI/WCAG22/Techniques/general/G55\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G55)
 - [H40: Using description lists \(https://www.w3.org/WAI/WCAG22/Techniques/html/H40\)](https://www.w3.org/WAI/WCAG22/Techniques/html/H40)
 - [G112: Using inline definitions \(https://www.w3.org/WAI/WCAG22/Techniques/general/G112\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G112)
 - [H54: Using the dfn element to identify the defining instance of a word \(https://www.w3.org/WAI/WCAG22/Techniques/html/H54\)](https://www.w3.org/WAI/WCAG22/Techniques/html/H54)
- [G101: Providing the definition of a word or phrase used in an unusual or restricted way \(https://www.w3.org/WAI/WCAG22/Techniques/general/G101\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G101) for each occurrence of the word or phrase in a web page using one of the following techniques:

- [G55: Linking to definitions](https://www.w3.org/WAI/WCAG22/Techniques/general/G55) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G55>)
 - [H40: Using description lists](https://www.w3.org/WAI/WCAG22/Techniques/html/H40) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H40>)
- [G62: Providing a glossary](https://www.w3.org/WAI/WCAG22/Techniques/general/G62) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G62>)
- [G70: Providing a function to search an online dictionary](https://www.w3.org/WAI/WCAG22/Techniques/general/G70) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G70>)

Situation B: If the word or phrase means different things within the same web page:

- [G101: Providing the definition of a word or phrase used in an unusual or restricted way](https://www.w3.org/WAI/WCAG22/Techniques/general/G101) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G101>) for each occurrence of the word or phrase in a web page using one of the following techniques:
 - [G55: Linking to definitions](https://www.w3.org/WAI/WCAG22/Techniques/general/G55) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G55>)
 - [H40: Using description lists](https://www.w3.org/WAI/WCAG22/Techniques/html/H40) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H40>)
 - [G112: Using inline definitions](https://www.w3.org/WAI/WCAG22/Techniques/general/G112) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G112>)
 - [H54: Using the dfn element to identify the defining instance of a word](https://www.w3.org/WAI/WCAG22/Techniques/html/H54) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H54>)

 [SHARE](#) |  [BACK TO TOP](#)

3.1.4 Abbreviations — Level AAA

A mechanism for identifying the expanded form or meaning of abbreviations is available.

 [Understanding 3.1.4](https://www.w3.org/WAI/WCAG22/Understanding/abbreviations.html) (<https://www.w3.org/WAI/WCAG22/Understanding/abbreviations.html>)

 [Hide techniques and failures for 3.1.4](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

Situation A: If the abbreviation has only one meaning within the web page:

- [G102: Providing the expansion or explanation of an abbreviation](https://www.w3.org/WAI/WCAG22/Techniques/general/G102) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G102>) for the first occurrence of the abbreviation in a web page using one of the following techniques:
 - [G97: Providing the first use of an abbreviation immediately before or after the expanded form](https://www.w3.org/WAI/WCAG22/Techniques/general/G97) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G97>)
 - [G55: Linking to definitions](https://www.w3.org/WAI/WCAG22/Techniques/general/G55) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G55>)
 - [PDF8: Providing definitions for abbreviations via an E entry for a structure element](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF8) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF8>)
- [G102: Providing the expansion or explanation of an abbreviation](https://www.w3.org/WAI/WCAG22/Techniques/general/G102) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G102>) for all occurrences of the abbreviation in a web page using one of the following techniques:
 - [G55: Linking to definitions](https://www.w3.org/WAI/WCAG22/Techniques/general/G55) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G55>)
 - [G62: Providing a glossary](https://www.w3.org/WAI/WCAG22/Techniques/general/G62) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G62>)
 - [G70: Providing a function to search an online dictionary](https://www.w3.org/WAI/WCAG22/Techniques/general/G70) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G70>)
 - [PDF8: Providing definitions for abbreviations via an E entry for a structure element](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF8) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF8>)

Situation B: If the abbreviation means different things within the same web page:

- [G102: Providing the expansion or explanation of an abbreviation](https://www.w3.org/WAI/WCAG22/Techniques/general/G102) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G102>) for all occurrences of abbreviations in a web page using one of the following techniques:
 - [G55: Linking to definitions](https://www.w3.org/WAI/WCAG22/Techniques/general/G55) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G55>)
 - [PDF8: Providing definitions for abbreviations via an E entry for a structure element](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF8) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF8>)

Advisory Techniques

- [H28: Providing definitions for abbreviations by using the abbr element](https://www.w3.org/WAI/WCAG22/Techniques/html/H28) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H28>)

3.1.5 Reading Level — Level AAA

When text requires reading ability more advanced than the lower secondary education level after removal of proper names and titles, supplemental content, or a version that does not require reading ability more advanced than the lower secondary education level, is available.

 [Understanding 3.1.5 \(https://www.w3.org/WAI/WCAG22/Understanding/reading-level.html\)](https://www.w3.org/WAI/WCAG22/Understanding/reading-level.html)

 Hide techniques and failures for 3.1.5

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques. \(https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques\)](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques)

- [G86: Providing a text summary that can be understood by people with lower secondary education level reading ability \(https://www.w3.org/WAI/WCAG22/Techniques/general/G86\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G86)
- [G103: Providing visual illustrations, pictures, and symbols to help explain ideas, events, and processes \(https://www.w3.org/WAI/WCAG22/Techniques/general/G103\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G103)
- [G79: Providing a spoken version of the text \(https://www.w3.org/WAI/WCAG22/Techniques/general/G79\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G79)
- [G153: Making the text easier to read \(https://www.w3.org/WAI/WCAG22/Techniques/general/G153\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G153)
- [G160: Providing sign language versions of information, ideas, and processes that must be understood in order to use the content \(https://www.w3.org/WAI/WCAG22/Techniques/general/G160\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G160)

Note: Different sites may address this success criterion in different ways. An audio version of the content may be helpful to some users. For some people who are deaf, a sign language version of the page may be easier to understand than a written language version since sign language may be their first language. Some sites may decide to do both or other combinations. No technique will help all users who have difficulty. So different techniques are provided as sufficient techniques here for authors trying to make their sites more accessible. Any numbered technique or combination above can be used by a particular site and it is considered sufficient by the Working Group.

 [SHARE](#) |  [BACK TO TOP](#)

3.1.6 Pronunciation — Level AAA

A mechanism is available for identifying specific pronunciation of words where meaning of the words, in context, is ambiguous without knowing the pronunciation.

ⓘ Understanding 3.1.6 (<https://www.w3.org/WAI/WCAG22/Understanding/pronunciation.html>)

▼ Hide techniques and failures for 3.1.6

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See Understanding Techniques. (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- G120: Providing the pronunciation immediately following the word (<https://www.w3.org/WAI/WCAG22/Techniques/general/G120>)
- G121: Linking to pronunciations (<https://www.w3.org/WAI/WCAG22/Techniques/general/G121>)
- G62: Providing a glossary (<https://www.w3.org/WAI/WCAG22/Techniques/general/G62>) that includes pronunciation information for words that have a unique pronunciation in the content and have meaning that depends on pronunciation
- G163: Using standard diacritical marks that can be turned off (<https://www.w3.org/WAI/WCAG22/Techniques/general/G163>)
- H62: Using the ruby element (<https://www.w3.org/WAI/WCAG22/Techniques/html/H62>)

◀ SHARE | ↑ BACK TO TOP

Guideline 3.2 – Predictable

Make web pages appear and operate in predictable ways.

3.2.1 On Focus — Level A

When any user interface component receives focus, it does not initiate a change of context.

ⓘ Understanding 3.2.1 (<https://www.w3.org/WAI/WCAG22/Understanding/on-focus.html>)

▼ Hide techniques and failures for 3.2.1

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See Understanding Techniques. (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

understanding-techniques)

- G107: Using "activate" rather than "focus" as a trigger for changes of context (<https://www.w3.org/WAI/WCAG22/Techniques/general/G107>)

Note: A change of content is not always a change of context. This success criterion is automatically met if changes in content are not also changes of context.

Advisory Techniques

- G200: Opening new windows and tabs from a link only when necessary (<https://www.w3.org/WAI/WCAG22/Techniques/general/G200>)
- G201: Giving users advanced warning when opening a new window (<https://www.w3.org/WAI/WCAG22/Techniques/general/G201>)

Failures

- F55: Failure of Success Criteria 2.1.1, 2.4.7, 2.4.13, and 3.2.1 due to using script to remove focus when focus is received (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F55>)

 [SHARE](#) |  [BACK TO TOP](#)

3.2.2 On Input — Level A

Changing the setting of any user interface component does not automatically cause a change of context unless the user has been advised of the behavior before using the component.

 [Understanding 3.2.2](#) (<https://www.w3.org/WAI/WCAG22/Understanding/on-input.html>)

 [Hide techniques and failures for 3.2.2](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See Understanding Techniques. (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- G80: Providing a submit button to initiate a change of context (<https://www.w3.org/WAI/WCAG22/Techniques/general/G80>) using one of the following techniques:

- [H32: Providing submit buttons](https://www.w3.org/WAI/WCAG22/Techniques/html/H32) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H32>)
- [H84: Using a button with a select element to perform an action](https://www.w3.org/WAI/WCAG22/Techniques/html/H84) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H84>)
- [PDF15: Providing submit buttons with the submit-form action in PDF forms](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF15) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF15>)
- [G13: Describing what will happen before a change to a form control that causes a change of context to occur is made](https://www.w3.org/WAI/WCAG22/Techniques/general/G13) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G13>)
- [SCR19: Using an onchange event on a select element without causing a change of context](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR19) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR19>)

Note: A change of content is not always a change of context. This success criterion is automatically met if changes in content are not also changes of context.

Advisory Techniques

- [G201: Giving users advanced warning when opening a new window](https://www.w3.org/WAI/WCAG22/Techniques/general/G201) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G201>)

Failures

- [F36: Failure of Success Criterion 3.2.2 due to automatically submitting a form and presenting new content without prior warning when the last field in the form is given a value](https://www.w3.org/WAI/WCAG22/Techniques/failures/F36) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F36>)
- [F37: Failure of Success Criterion 3.2.2 due to launching a new window without prior warning when the selection of a radio button, check box or select list is changed](https://www.w3.org/WAI/WCAG22/Techniques/failures/F37) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F37>)

 [SHARE](#) |  [BACK TO TOP](#)

3.2.3 Consistent Navigation — Level AA

Navigational mechanisms that are repeated on multiple web pages within a set of web pages occur in the same relative order each time they are repeated, unless a change is initiated by the user.

 [Understanding 3.2.3](https://www.w3.org/WAI/WCAG22/Understanding/consistent-navigation.html) (<https://www.w3.org/WAI/WCAG22/Understanding/consistent-navigation.html>)

 [Hide techniques and failures for 3.2.3](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G61: Presenting repeated components in the same relative order each time they appear](https://www.w3.org/WAI/WCAG22/Techniques/general/G61) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G61>)

Advisory Techniques

- [PDF14: Providing running headers and footers in PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF14) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF14>)
- [PDF17: Specifying consistent page numbering for PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF17) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF17>)

Failures

- [F66: Failure of Success Criterion 3.2.3 due to presenting navigation links in a different relative order on different pages](https://www.w3.org/WAI/WCAG22/Techniques/failures/F66) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F66>)

 [SHARE](#) |  [BACK TO TOP](#)

3.2.4 Consistent Identification — Level AA

Components that have the same functionality within a set of web pages are identified consistently.

 [Understanding 3.2.4](https://www.w3.org/WAI/WCAG22/Understanding/consistent-identification.html) (<https://www.w3.org/WAI/WCAG22/Understanding/consistent-identification.html>)

 [Hide techniques and failures for 3.2.4](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G197: Using labels, names, and text alternatives consistently for content that has the same functionality](https://www.w3.org/WAI/WCAG22/Techniques/general/G197) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G197>)

G197) AND following the sufficient techniques for Success Criterion 1.1.1 (<https://www.w3.org/WAI/WCAG22/Understanding/non-text-content#techniques>) and sufficient techniques for Success Criterion 4.1.2 (<https://www.w3.org/WAI/WCAG22/Understanding/name-role-value#techniques>) for providing labels, names, and text alternatives

Note:

Text alternatives that are "consistent" are not always "identical." For instance, you may have a graphical arrow at the bottom of a web page that links to the next web page. The text alternative may say "Go to page 4." Naturally, it would not be appropriate to repeat this exact text alternative on the next web page. It would be more appropriate to say "Go to page 5". Although these text alternatives would not be identical, they would be consistent, and therefore would satisfy this success criterion.

A single non-text-content-item may be used to serve different functions. In such cases, different text alternatives are necessary and should be used. Examples can be commonly found with the use of icons such as check marks, cross marks, and traffic signs. Their functions can be different depending on the context of the web page. A check mark icon may function as "approved", "completed", or "included", to name a few, depending on the situation. Using "check mark" as text alternative across all web pages does not help users understand the function of the icon. Different text alternatives can be used when the same non-text content serves multiple functions.

Failures

- F31: Failure of Success Criterion 3.2.4 due to using two different labels for the same function on different web pages within a set of web pages (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F31>)

 [SHARE](#) |  [BACK TO TOP](#)

3.2.5 Change on Request — Level AAA

Changes of context are initiated only by user request or a mechanism is available to turn off such changes.

 Understanding 3.2.5 (<https://www.w3.org/WAI/WCAG22/Understanding/change-on-request.html>)

 Hide techniques and failures for 3.2.5

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See Understanding Techniques. (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

Situation A: If the web page allows automatic updates:

- [G76: Providing a mechanism to request an update of the content instead of updating automatically](https://www.w3.org/WAI/WCAG22/Techniques/general/G76) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G76>)

Situation B: If automatic redirects are possible:

- [SVR1: Implementing automatic redirects on the server side instead of on the client side](https://www.w3.org/WAI/WCAG22/Techniques/server-side-script/SVR1) (<https://www.w3.org/WAI/WCAG22/Techniques/server-side-script/SVR1>)
- [G110: Using an instant client-side redirect](https://www.w3.org/WAI/WCAG22/Techniques/general/G110) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G110>) using one of the following techniques:
 - [H76: Using meta refresh to create an instant client-side redirect](https://www.w3.org/WAI/WCAG22/Techniques/html/H76) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H76>)

Situation C: If the web page uses pop-up windows:

- Including pop-up windows using one of the following techniques:
 - [H83: Using the target attribute to open a new window on user request and indicating this in link text](https://www.w3.org/WAI/WCAG22/Techniques/html/H83) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H83>)
 - [SCR24: Using progressive enhancement to open new windows on user request](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR24) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR24>)

Situation D: If using an onchange event on a select element:

- [SCR19: Using an onchange event on a select element without causing a change of context](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR19) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR19>)

Advisory Techniques

- [G200: Opening new windows and tabs from a link only when necessary](https://www.w3.org/WAI/WCAG22/Techniques/general/G200) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G200>)

Failures

- [F60: Failure of Success Criterion 3.2.5 due to launching a new window when a user](#)

[enters text into an input field](https://www.w3.org/WAI/WCAG22/Techniques/failures/F60) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F60>)

- [F61: Failure of Success Criterion 3.2.5 due to complete change of main content through an automatic update that the user cannot disable from within the content](https://www.w3.org/WAI/WCAG22/Techniques/failures/F61) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F61>)
- [F9: Failure of Success Criterion 3.2.5 due to changing the context when the user removes focus from a form element](https://www.w3.org/WAI/WCAG22/Techniques/failures/F9) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F9>)
- [F22: Failure of Success Criterion 3.2.5 due to opening windows that are not requested by the user](https://www.w3.org/WAI/WCAG22/Techniques/failures/F22) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F22>)
- [F52: Failure of Success Criterion 3.2.5 due to opening a new window as soon as a new page is loaded](https://www.w3.org/WAI/WCAG22/Techniques/failures/F52) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F52>)
- [F40: Failure due to using meta redirect with a time limit](https://www.w3.org/WAI/WCAG22/Techniques/failures/F40) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F40>)
- [F41: Failure of Success Criterion 2.2.1, 2.2.4, and 3.2.5 due to using meta refresh to reload the page](https://www.w3.org/WAI/WCAG22/Techniques/failures/F41) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F41>)

 [SHARE](#) |  [BACK TO TOP](#)

3.2.6 Consistent Help — Level A (Added in 2.2)

If a web page contains any of the following help mechanisms, and those mechanisms are repeated on multiple web pages within a set of web pages, they occur in the same order relative to other page content, unless a change is initiated by the user:

 [Hide full description](#)

- Human contact details;
- Human contact mechanism;
- Self-help option;
- A fully automated contact mechanism.

Note 1: Help mechanisms may be provided directly on the page, or may be provided via a direct link to a different page containing the information.

Note 2: For this success criterion, "the same order relative to other page content" can be thought of as how the content is ordered when the page is serialized. The visual position of a help mechanism is likely to be consistent across pages for the same page variation (e.g., CSS break-point). The user can initiate a change, such as changing the page's zoom or orientation, which may trigger a different page variation. This criterion is concerned with relative order across pages displayed in the same page variation (e.g., same zoom level and orientation).

ⓘ Understanding 3.2.6 (<https://www.w3.org/WAI/WCAG22/Understanding/consistent-help.html>)

▼ Hide techniques and failures for 3.2.6

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G220: Provide a contact-us link in a consistent location](https://www.w3.org/WAI/WCAG22/Techniques/general/G220) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G220>)

Failures

- Inconsistent Help Location

 SHARE |  BACK TO TOP

Guideline 3.3 – Input Assistance

Help users avoid and correct mistakes.

3.3.1 Error Identification — Level A

If an input error is automatically detected, the item that is in error is identified and the error is described to the user in text.

ⓘ Understanding 3.3.1 (<https://www.w3.org/WAI/WCAG22/Understanding/error-identification.html>)

▼ Hide techniques and failures for 3.3.1

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

Situation A: If a form contains fields for which information from the user is mandatory.

- [G83: Providing text descriptions to identify required fields that were not completed](https://www.w3.org/WAI/WCAG22/Techniques/general/G83) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G83>)
- [ARIA2: Identifying a required field with the aria-required property](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA2) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA2>)

- [ARIA21: Using aria-invalid to Indicate An Error Field](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA21) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA21>)
- [SCR18: Providing client-side validation and alert](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR18) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR18>)
- [PDF5: Indicating required form controls in PDF forms](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF5) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF5>)

Situation B: If information provided by the user is required to be in a specific data format or of certain values.

- [ARIA18: Using aria-alertdialog to Identify Errors](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA18) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA18>)
- [ARIA19: Using ARIA role=alert or Live Regions to Identify Errors](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA19) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA19>)
- [ARIA21: Using aria-invalid to Indicate An Error Field](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA21) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA21>)
- [G84: Providing a text description when the user provides information that is not in the list of allowed values](https://www.w3.org/WAI/WCAG22/Techniques/general/G84) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G84>)
- [G85: Providing a text description when user input falls outside the required format or values](https://www.w3.org/WAI/WCAG22/Techniques/general/G85) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G85>)
- [SCR18: Providing client-side validation and alert](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR18) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR18>)
- [SCR32: Providing client-side validation and adding error text via the DOM](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR32) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR32>)
- [PDF22: Indicating when user input falls outside the required format or values in PDF forms](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF22) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF22>)

Advisory Techniques

- [G139: Creating a mechanism that allows users to jump to errors](https://www.w3.org/WAI/WCAG22/Techniques/general/G139) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G139>)
- [G199: Providing success feedback when data is submitted successfully](https://www.w3.org/WAI/WCAG22/Techniques/general/G199) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G199>)

 SHARE |  BACK TO TOP

3.3.2 Labels or Instructions — Level A

Labels or instructions are provided when content requires user input.

❶ Understanding 3.3.2 (<https://www.w3.org/WAI/WCAG22/Understanding/labels-or-instructions.html>)

▼ Hide techniques and failures for 3.3.2

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G131: Providing descriptive labels](https://www.w3.org/WAI/WCAG22/Techniques/general/G131) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G131>) **AND** one of the following techniques:
 - [ARIA1: Using the aria-describedby property to provide a descriptive label for user interface controls](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA1) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA1>)
 - [ARIA9: Using aria-labelledby to concatenate a label from several text nodes](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA9) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA9>)
 - [ARIA17: Using grouping roles to identify related form controls](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA17) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA17>)
 - [G89: Providing expected data format and example](https://www.w3.org/WAI/WCAG22/Techniques/general/G89) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G89>)
 - [G184: Providing text instructions at the beginning of a form or set of fields that describes the necessary input](https://www.w3.org/WAI/WCAG22/Techniques/general/G184) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G184>)
 - [G162: Positioning labels to maximize predictability of relationships](https://www.w3.org/WAI/WCAG22/Techniques/general/G162) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G162>)
 - [G83: Providing text descriptions to identify required fields that were not completed](https://www.w3.org/WAI/WCAG22/Techniques/general/G83) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G83>)
 - [H90: Indicating required form controls using label or legend](https://www.w3.org/WAI/WCAG22/Techniques/html/H90) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H90>)
 - [PDF5: Indicating required form controls in PDF forms](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF5) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF5>)
- [H44: Using label elements to associate text labels with form controls](https://www.w3.org/WAI/WCAG22/Techniques/html/H44) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H44>)
- [PDF10: Providing labels for interactive form controls in PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF10) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF10>)
- [H71: Providing a description for groups of form controls using fieldset and legend elements](https://www.w3.org/WAI/WCAG22/Techniques/html/H71) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H71>)
- [G167: Using an adjacent button to label the purpose of a field](https://www.w3.org/WAI/WCAG22/Techniques/general/G167) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G167>)

Note: The techniques at the end of the above list should be considered "last resort" and only used when

the other techniques cannot be applied to the page. The earlier techniques are preferred because they increase accessibility to a wider user group.

Advisory Techniques

- [G13: Describing what will happen before a change to a form control that causes a change of context to occur is made](https://www.w3.org/WAI/WCAG22/Techniques/general/G13) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G13>)

Failures

- [F82: Failure of Success Criterion 3.3.2 by visually formatting a set of phone number fields but not including a text label](https://www.w3.org/WAI/WCAG22/Techniques/failures/F82) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F82>)

 [SHARE](#) |  [BACK TO TOP](#)

3.3.3 Error Suggestion — Level AA

If an input error is automatically detected and suggestions for correction are known, then the suggestions are provided to the user, unless it would jeopardize the security or purpose of the content.

 [Understanding 3.3.3](https://www.w3.org/WAI/WCAG22/Understanding/error-suggestion.html) (<https://www.w3.org/WAI/WCAG22/Understanding/error-suggestion.html>)

 [Hide techniques and failures for 3.3.3](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

Situation A: If information for a field is required to be in a specific data format:

- [ARIA18: Using aria-alertdialog to Identify Errors](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA18) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA18>)
- [G85: Providing a text description when user input falls outside the required format or values](https://www.w3.org/WAI/WCAG22/Techniques/general/G85) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G85>)
- [G177: Providing suggested correction text](https://www.w3.org/WAI/WCAG22/Techniques/general/G177) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G177>)
- [PDF22: Indicating when user input falls outside the required format or values in](https://www.w3.org/WAI/WCAG22/Techniques/general/PDF22) (<https://www.w3.org/WAI/WCAG22/Techniques/general/PDF22>)

[PDF forms \(https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF22\)](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF22)

Situation B: Information provided by the user is required to be one of a limited set of values:

- [ARIA18: Using aria-alertdialog to Identify Errors \(https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA18\)](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA18)
- [G84: Providing a text description when the user provides information that is not in the list of allowed values \(https://www.w3.org/WAI/WCAG22/Techniques/general/G84\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G84)
- [G177: Providing suggested correction text \(https://www.w3.org/WAI/WCAG22/Techniques/general/G177\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G177)
- [PDF22: Indicating when user input falls outside the required format or values in PDF forms \(https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF22\)](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF22)

Note: In some cases, more than one of these situations may apply. For example, when a mandatory field also requires the data to be in a specific format.

Advisory Techniques

- [G139: Creating a mechanism that allows users to jump to errors \(https://www.w3.org/WAI/WCAG22/Techniques/general/G139\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G139)
- [G199: Providing success feedback when data is submitted successfully \(https://www.w3.org/WAI/WCAG22/Techniques/general/G199\)](https://www.w3.org/WAI/WCAG22/Techniques/general/G199)
- [SCR18: Providing client-side validation and alert \(https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR18\)](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR18)
- [SCR32: Providing client-side validation and adding error text via the DOM \(https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR32\)](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR32)

 [SHARE](#) |  [BACK TO TOP](#)

3.3.4 Error Prevention (Legal, Financial, Data) — Level AA

For web pages that cause legal commitments or financial transactions for the user to occur, that modify or delete user-controllable data in data storage systems, or that submit user test responses, at least one of the following is true: [▼ Hide full description](#)

- **Reversible:** Submissions are reversible.
- **Checked:** Data entered by the user is checked for input errors and the user is

provided an opportunity to correct them.

- **Confirmed:** A mechanism is available for reviewing, confirming, and correcting information before finalizing the submission.

❶ Understanding 3.3.4 (<https://www.w3.org/WAI/WCAG22/Understanding/error-prevention-legal-financial-data.html>)

▼ Hide techniques and failures for 3.3.4

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

Situation A: If an application causes a legal transaction to occur, such as making a purchase or submitting an income tax return:

- [G164: Providing a stated time within which an online request \(or transaction\) may be amended or canceled by the user after making the request](https://www.w3.org/WAI/WCAG22/Techniques/general/G164) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G164>)
- [G98: Providing the ability for the user to review and correct answers before submitting](https://www.w3.org/WAI/WCAG22/Techniques/general/G98) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G98>)
- [G155: Providing a checkbox in addition to a submit button](https://www.w3.org/WAI/WCAG22/Techniques/general/G155) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G155>)

Situation B: If an action causes information to be deleted:

- [G99: Providing the ability to recover deleted information](https://www.w3.org/WAI/WCAG22/Techniques/general/G99) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G99>)
- [G168: Requesting confirmation to continue with selected action](https://www.w3.org/WAI/WCAG22/Techniques/general/G168) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G168>)
- [G155: Providing a checkbox in addition to a submit button](https://www.w3.org/WAI/WCAG22/Techniques/general/G155) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G155>)

Situation C: If the web page includes a testing application

- [G98: Providing the ability for the user to review and correct answers before submitting](https://www.w3.org/WAI/WCAG22/Techniques/general/G98) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G98>)
- [G168: Requesting confirmation to continue with selected action](https://www.w3.org/WAI/WCAG22/Techniques/general/G168) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G168>)

Advisory Techniques

- [SCR18: Providing client-side validation and alert](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR18) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR18>)
- [G199: Providing success feedback when data is submitted successfully](https://www.w3.org/WAI/WCAG22/Techniques/general/G199) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G199>)

 [SHARE](#) |  [BACK TO TOP](#)

3.3.5 Help — Level AAA

Context-sensitive help is available.

 [Understanding 3.3.5](#) (<https://www.w3.org/WAI/WCAG22/Understanding/help.html>)

 [Hide techniques and failures for 3.3.5](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

Situation A: If a form requires text input:

- [G71: Providing a help link on every web page](https://www.w3.org/WAI/WCAG22/Techniques/general/G71) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G71>)
- [G193: Providing help by an assistant in the web page](https://www.w3.org/WAI/WCAG22/Techniques/general/G193) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G193>)
- [G194: Providing spell checking and suggestions for text input](https://www.w3.org/WAI/WCAG22/Techniques/general/G194) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G194>)
- [G184: Providing text instructions at the beginning of a form or set of fields that describes the necessary input](https://www.w3.org/WAI/WCAG22/Techniques/general/G184) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G184>)

Situation B: If a form requires text input in an expected data format:

- [G89: Providing expected data format and example](https://www.w3.org/WAI/WCAG22/Techniques/general/G89) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G89>)
- [G184: Providing text instructions at the beginning of a form or set of fields that describes the necessary input](https://www.w3.org/WAI/WCAG22/Techniques/general/G184) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G184>)

Advisory Techniques

- [H89: Using the title attribute to provide context-sensitive help \(<https://www.w3.org/WAI/WCAG22/Techniques/html/H89>\)](https://www.w3.org/WAI/WCAG22/Techniques/html/H89)

 [SHARE](#) |  [BACK TO TOP](#)

3.3.6 Error Prevention (All) — Level AAA

For web pages that require the user to submit information, at least one of the following is true: 

- **Reversible:** Submissions are reversible.
- **Checked:** Data entered by the user is checked for input errors and the user is provided an opportunity to correct them.
- **Confirmed:** A mechanism is available for reviewing, confirming, and correcting information before finalizing the submission.

 [Understanding 3.3.6 \(<https://www.w3.org/WAI/WCAG22/Understanding/error-prevention-all.html>\)](https://www.w3.org/WAI/WCAG22/Understanding/error-prevention-all.html)

 [Hide techniques and failures for 3.3.6](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques. \(<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>\)](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques)

- Following the [sufficient techniques for Success Criterion 3.3.4 \(<https://www.w3.org/WAI/WCAG22/Understanding/error-prevention-legal-financial-data#techniques>\)](https://www.w3.org/WAI/WCAG22/Understanding/error-prevention-legal-financial-data#techniques) for all forms that require the user to submit information

 [SHARE](#) |  [BACK TO TOP](#)

3.3.7 Redundant Entry — Level A (Added in 2.2)

Information previously entered by or provided to the user that is required to be entered again in the same process is either: 

- auto-populated, or
- available for the user to select.

Except when:

- re-entering the information is essential,
- the information is required to ensure the security of the content, or
- previously entered information is no longer valid.

 Understanding 3.3.7 (<https://www.w3.org/WAI/WCAG22/Understanding/redundant-entry.html>)

 Hide techniques and failures for 3.3.7

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G221: Provide data from a previous step in a process](https://www.w3.org/WAI/WCAG22/Techniques/general/G221) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G221>)
- Not requesting the same information twice (Potential future technique)

 SHARE |  BACK TO TOP

3.3.8 Accessible Authentication (Minimum) — Level AA (Added in 2.2)

A cognitive function test (such as remembering a password or solving a puzzle) is not required for any step in an authentication process unless that step provides at least one of the following:  Hide full description

- **Alternative:** Another authentication method that does not rely on a cognitive function test.
- **Mechanism:** A mechanism is available to assist the user in completing the cognitive function test.
- **Object Recognition:** The cognitive function test is to recognize objects.
- **Personal Content:** The cognitive function test is to identify non-text content the user provided to the website.

Note 1: "Object recognition" and "Personal content" may be represented by images, video, or audio.

Note 2: Examples of mechanisms that satisfy this criterion include:

- support for password entry by password managers to reduce memory need, and

- copy and paste to reduce the cognitive burden of re-typing.

❶ Understanding 3.3.8 (<https://www.w3.org/WAI/WCAG22/Understanding/accessible-authentication-minimum.html>)

▼ Hide techniques and failures for 3.3.8

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G218: Email link authentication](https://www.w3.org/WAI/WCAG22/Techniques/general/G218) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G218>)
- [H100: Providing properly marked up email and password inputs](https://www.w3.org/WAI/WCAG22/Techniques/html/H100) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H100>)
- Providing WebAuthn as an alternative to username/password (Potential future technique)
- Providing a third-party login using OAuth (Potential future technique)
- Using two techniques to provide two-factor authentication (Potential future technique)

Failures

- [F109: Failure of Success Criterion 3.3.8 and 3.3.9 due to preventing password or code re-entry in the same format](https://www.w3.org/WAI/WCAG22/Techniques/failures/F109) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F109>)

 SHARE |  BACK TO TOP

3.3.9 Accessible Authentication (Enhanced) — Level AAA (Added in 2.2)

A cognitive function test (such as remembering a password or solving a puzzle) is not required for any step in an authentication process unless that step provides at least one of the following:

▼ Hide full description

- **Alternative:** Another authentication method that does not rely on a cognitive function test.
- **Mechanism:** A mechanism is available to assist the user in completing the cognitive function test.

❶ Understanding 3.3.9 (<https://www.w3.org/WAI/WCAG22/Understanding/accessible-authentication-enhanced.html>)

▼ Hide techniques and failures for 3.3.9

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques). (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

- [G218: Email link authentication](https://www.w3.org/WAI/WCAG22/Techniques/general/G218) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G218>)
- [H100: Providing properly marked up email and password inputs](https://www.w3.org/WAI/WCAG22/Techniques/html/H100) (<https://www.w3.org/WAI/WCAG22/Techniques/html/H100>)
- Providing WebAuthn as an alternative to username/password (Potential future technique)
- Providing a third-party login using OAuth (Potential future technique)
- Using two techniques to provide two-factor authentication (Potential future technique)

Failures

- [F109: Failure of Success Criterion 3.3.8 and 3.3.9 due to preventing password or code re-entry in the same format](https://www.w3.org/WAI/WCAG22/Techniques/failures/F109) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F109>)

 SHARE |  BACK TO TOP

Principle 4 – Robust

Content must be robust enough that it can be interpreted by a wide variety of user agents, including assistive technologies.

Guideline 4.1 – Compatible

Maximize compatibility with current and future user agents, including assistive technologies.

4.1.2 Name, Role, Value — Level A

For all user interface components (including but not limited to: form elements, links and components generated by scripts), the name and role can be programmatically determined; states, properties, and values that can be set by the user can be programmatically set; and notification of changes to these items is available to user agents, including assistive technologies.

Note: This success criterion is primarily for web authors who develop or script their own user interface components. For example, standard HTML (HyperText Markup Language) controls already meet this success criterion when used according to specification.

 Understanding 4.1.2 (<https://www.w3.org/WAI/WCAG22/Understanding/name-role-value.html>)

▼ Hide techniques and failures for 4.1.2

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See Understanding Techniques. (<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>)

Situation A: If using a standard user interface component in a markup language (e.g., HTML):

- ARIA14: Using aria-label to provide an invisible label where a visible label cannot be used (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA14>)
- ARIA16: Using aria-labelledby to provide a name for user interface controls (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA16>)
- G108: Using markup features to expose the name and role, allow user-settable properties to be directly set, and provide notification of changes (<https://www.w3.org/WAI/WCAG22/Techniques/general/G108>) using one or more of the following techniques:
 - H91: Using HTML form controls and links (<https://www.w3.org/WAI/WCAG22/Techniques/html/H91>)
 - H44: Using label elements to associate text labels with form controls (<https://www.w3.org/WAI/WCAG22/Techniques/html/H44>)
 - H64: Using the title attribute of the iframe element (<https://www.w3.org/WAI/WCAG22/Techniques/html/H64>)
 - H65: Using the title attribute to identify form controls when the label element cannot be used (<https://www.w3.org/WAI/WCAG22/Techniques/html/H65>)
 - H88: Using HTML according to spec (<https://www.w3.org/WAI/WCAG22/Techniques/html/H88>)

Situation B: If using script or code to re-purpose a standard user interface component

in a markup language:

- Exposing the names and roles, allowing user-settable properties to be directly set, and providing notification of changes using one of the following techniques:
 - [ARIA16: Using aria-labelledby to provide a name for user interface controls](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA16) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA16>)

Situation C: If using a standard user interface component in a programming technology:

- [G135: Using the accessibility API features of a technology to expose names and roles, to allow user-settable properties to be directly set, and to provide notification of changes](https://www.w3.org/WAI/WCAG22/Techniques/general/G135) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G135>) using one or more of the following techniques:
 - [PDF10: Providing labels for interactive form controls in PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF10) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF10>)
 - [PDF12: Providing name, role, value information for form fields in PDF documents](https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF12) (<https://www.w3.org/WAI/WCAG22/Techniques/pdf/PDF12>)

Situation D: If creating your own user interface component in a programming language:

- [G10: Creating components using a technology that supports the accessibility API features of the platforms on which the user agents will be run to expose the names and roles, allow user-settable properties to be directly set, and provide notification of changes](https://www.w3.org/WAI/WCAG22/Techniques/general/G10) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G10>) using one or more of the following techniques:
 - [ARIA4: Using a WAI-ARIA role to expose the role of a user interface component](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA4) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA4>)
 - [ARIA5: Using WAI-ARIA state and property attributes to expose the state of a user interface component](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA5) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA5>)
 - [ARIA16: Using aria-labelledby to provide a name for user interface controls](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA16) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA16>)

Failures

- [F59: Failure of Success Criterion 4.1.2 due to using script to make div or span a user interface control in HTML without providing a role for the control](https://www.w3.org/WAI/WCAG22/Techniques/failures/F59) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F59>)
- [F15: Failure of Success Criterion 4.1.2 due to implementing custom controls that do](https://www.w3.org/WAI/WCAG22/Techniques/failures/F15)

- not use an accessibility API for the technology, or do so incompletely (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F15>)
- F20: Failure of Success Criterion 1.1.1 and 4.1.2 due to not updating text alternatives when changes to non-text content occur (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F20>)
 - F42: Failure of Success Criteria 1.3.1, 2.1.1, 2.1.3, or 4.1.2 when emulating links (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F42>)
 - F68: Failure of Success Criterion 4.1.2 due to a user interface control not having a programmatically determined name (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F68>)
 - F79: Failure of Success Criterion 4.1.2 due to the focus state of a user interface component not being programmatically determinable or no notification of change of focus state available (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F79>)
 - F86: Failure of Success Criterion 4.1.2 due to not providing names for each part of a multi-part form field, such as a US telephone number (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F86>)
 - F89: Failure of Success Criteria 2.4.4, 2.4.9 and 4.1.2 due to not providing an accessible name for an image which is the only content in a link (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F89>)
 - F111: Failure of Success Criteria 1.3.1, 2.5.3, and 4.1.2 due to a control with visible label text but no accessible name (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F111>)

 [SHARE](#) |  [BACK TO TOP](#)

4.1.3 Status Messages — Level AA (Added in 2.1)

In content implemented using markup languages, status messages can be programmatically determined through role or properties such that they can be presented to the user by assistive technologies without receiving focus.

 [Understanding 4.1.3 \(<https://www.w3.org/WAI/WCAG22/Understanding/status-messages.html>\)](#)

 [Hide techniques and failures for 4.1.3](#)

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques. \(<https://www.w3.org/WAI/WCAG22/Understanding/understanding-techniques>\)](#)

Situation A: If a status message advises on the success or results of an action, or the state of an application:

- [ARIA22: Using role=status to present status messages](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA22) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA22>) in combination with any of the following techniques:
 - [G199: Providing success feedback when data is submitted successfully](https://www.w3.org/WAI/WCAG22/Techniques/general/G199) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G199>)

Situation B: If a status message conveys a suggestion, or a warning on the existence of an error:

- [ARIA19: Using ARIA role=alert or Live Regions to Identify Errors](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA19) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA19>) in combination with any of the following techniques:
 - [G83: Providing text descriptions to identify required fields that were not completed](https://www.w3.org/WAI/WCAG22/Techniques/general/G83) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G83>)
 - [G84: Providing a text description when the user provides information that is not in the list of allowed values](https://www.w3.org/WAI/WCAG22/Techniques/general/G84) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G84>)
 - [G85: Providing a text description when user input falls outside the required format or values](https://www.w3.org/WAI/WCAG22/Techniques/general/G85) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G85>)
 - [G177: Providing suggested correction text](https://www.w3.org/WAI/WCAG22/Techniques/general/G177) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G177>)
 - [G194: Providing spell checking and suggestions for text input](https://www.w3.org/WAI/WCAG22/Techniques/general/G194) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G194>)

Note: Not all examples in the preceding general techniques use status messages to convey warnings or errors to users. A role of "alert" is only necessary where a change of context does *not* take place.

Situation C: If a status message conveys information on the progress of a process:

- [ARIA23: Using role=log to identify sequential information updates](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA23) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA23>)
- Using role="progressbar" (future link)
- [ARIA22: Using role=status to present status messages](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA22) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA22>) **AND** [G193: Providing help by an assistant in the web page](https://www.w3.org/WAI/WCAG22/Techniques/general/G193) (<https://www.w3.org/WAI/WCAG22/Techniques/general/G193>)

Advisory Techniques

- Using aria-live regions with chat clients (future link)
- Using aria-live regions to support [1.4.13 Content on Hover or Focus](https://www.w3.org/WAI/WCAG22/Understanding/content-on-hover-or-focus) (<https://www.w3.org/WAI/WCAG22/Understanding/content-on-hover-or-focus>) (future link)
- Using role="marquee" (future link)
- Using role="timer" (future link)
- Where appropriate, moving focus to new content with [ARIA18: Using aria-alertdialog to Identify Errors](https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA18) (<https://www.w3.org/WAI/WCAG22/Techniques/aria/ARIA18>)
- Supporting personalization with [SCR14: Using scripts to make nonessential alerts optional](https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR14) (<https://www.w3.org/WAI/WCAG22/Techniques/client-side-script/SCR14>)

Failures

- [F103: Failure of Success Criterion 4.1.3 due to providing status messages that cannot be programmatically determined through role or properties](https://www.w3.org/WAI/WCAG22/Techniques/failures/F103) (<https://www.w3.org/WAI/WCAG22/Techniques/failures/F103>)
- Using role="alert" or aria-live="assertive" on content which is not important and time-sensitive (future link)

 [SHARE](#) |  [BACK TO TOP](#)

Contribute

We welcome feedback and suggestions:

- This resource — [report bugs](https://github.com/w3c/wai-wcag-quickref/issues/) (<https://github.com/w3c/wai-wcag-quickref/issues/>) and contribute directly to the [Github repository](https://github.com/w3c/wai-wcag-quickref) (<https://github.com/w3c/wai-wcag-quickref>)
 - [Instructions for Commenting on WCAG 2 Documents](https://www.w3.org/WAI/standards-guidelines/wcag/commenting/) (<https://www.w3.org/WAI/standards-guidelines/wcag/commenting/>)
-

Status: Updated 22 Sep 2025.

Previous editors and developers: [Shadi Abou-Zahra](https://www.w3.org/People/shadi/) (<https://www.w3.org/People/shadi/>) (Project Lead), [Eric Eggert](https://www.w3.org/People/yatil/) (<https://www.w3.org/People/yatil/>), Gregg Vanderheiden, Loretta Guarino Reid, Ben Caldwell, [Shawn Lawton Henry](https://www.w3.org/People/Shawn/) (<https://www.w3.org/People/Shawn/>), Gez Lemon.

The 2016 redesign was developed by the Education and Outreach Working Group ([EOWG](https://www.w3.org/WAI/EO/) (<https://www.w3.org/WAI/EO/>)) and the Web Content Accessibility Guidelines Working Group ([WCAG WG](https://www.w3.org/WAI/GL/) (<https://www.w3.org/WAI/GL/>)), with support from the [WAI-DEV project](https://www.w3.org/WAI/DEV/) (<https://www.w3.org/WAI/DEV/>), a project of the European Commission IST Programme.

[WAI Site Map (<https://www.w3.org/WAI/sitemap.html>)] [Help with WAI Website (<https://www.w3.org/WAI/sitehelp.html>)] [Search (<https://www.w3.org/WAI/search.php>)] [Contacting WAI (/WAI/contacts)]

Feedback welcome to wai@w3.org (mailto:wai@w3.org) (a WAI staff-only list).

Copyright © 2025 W3C (<https://w3.org>) ®. See Permission to Use WAI Material (<https://www.w3.org/WAI/about/using-wai-material/>).

```
var _paq = _paq || []; _paq.push(["setDomains", ["*.www.w3.org/WAI"]]);  
_paq.push(['trackPageView']); _paq.push(['enableLinkTracking']); (function() { var u="//www.w3.org/analytics/piwik/"; _paq.push(['setTrackerUrl', u+'piwik.php']); _paq.push(['setSiteId', 328]); var d=document, g=d.createElement('script'), s=d.getElementsByTagName('script')[0]; g.type='text/javascript'; g.async=true; g.defer=true; g.src=u+'piwik.js'; s.parentNode.insertBefore(g,s); })();
```



Images Tutorial

in [Tutorials](https://www.w3.org/WAI/tutorials/) (<https://www.w3.org/WAI/tutorials/>)

Images must have text alternatives that describe the information or function represented by them. This ensures that images can be used by [people with various disabilities](#) (see [#why-is-this-important](#)). This tutorial demonstrates how to provide appropriate text alternatives based on the purpose of the image:

- **[Informative images](https://www.w3.org/WAI/tutorials/images/informative/)** (<https://www.w3.org/WAI/tutorials/images/informative/>): Images that graphically represent concepts and information, typically pictures, photos, and illustrations. The text alternative should be at least a short description conveying the essential information presented by the image.
- **[Decorative images](https://www.w3.org/WAI/tutorials/images/decorative/)** (<https://www.w3.org/WAI/tutorials/images/decorative/>): Provide a null text alternative (`alt=""`) when the only purpose of an image is to add visual decoration to the page, rather than to convey information that is important to understanding the page.
- **[Functional images](https://www.w3.org/WAI/tutorials/images/functional/)** (<https://www.w3.org/WAI/tutorials/images/functional/>): The text alternative of an image used as a link or as a button should describe the functionality of the link or button rather than the visual image. Examples of such images are a printer icon to represent the print function or a button to submit a form.
- **[Images of text](https://www.w3.org/WAI/tutorials/images/textual/)** (<https://www.w3.org/WAI/tutorials/images/textual/>): Readable text is sometimes presented within an image. If the image is not a logo, avoid text in images. However, if images of text are used, the text alternative should contain the same words as in the image.
- **[Complex images](https://www.w3.org/WAI/tutorials/images/complex/)** (<https://www.w3.org/WAI/tutorials/images/complex/>) such as graphs and diagrams: To convey data or detailed information, provide a complete text equivalent of the data or information provided in the image as the text alternative.
- **[Groups of images](https://www.w3.org/WAI/tutorials/images/groups/)** (<https://www.w3.org/WAI/tutorials/images/groups/>): If multiple images convey a single piece of information, the text alternative for one image should convey the information for the entire group.

- **Image maps (<https://www.w3.org/WAI/tutorials/images/imagemap/>):** The text alternative for an image that contains multiple clickable areas should provide an overall context for the set of links. Also, each individually clickable area should have alternative text that describes the purpose or destination of the link.

For a quick overview on deciding which category a particular image fits into, see the [alt Decision Tree \(<https://www.w3.org/WAI/tutorials/images/decision-tree/>\)](https://www.w3.org/WAI/tutorials/images/decision-tree/). The text alternative needs to be determined by the author, depending on the usage, context, and content of an image. For example, the exact type and look of a bird in an image might be less relevant and described only briefly on a website about parks, but may be appropriate on a website specifically about birds.

Why is this important?

Images and graphics make content more pleasant and easier to understand for many people, and in particular for those with cognitive and learning disabilities. They serve as cues that are used by people with visual impairments, including people with low vision, to orient themselves in the content.

However, images are used extensively on websites and can create major barriers when they are not accessible. Accessible images are beneficial in many situations, such as:

- **People using screen readers:** The text alternative can be read aloud or rendered as Braille
- **People using speech input software:** Users can put the focus onto a button or linked image with a single voice command
- **People browsing speech-enabled websites:** The text alternative can be read aloud
- **Mobile web users:** Images can be turned off, especially for data-roaming
- **Search engine optimization:** Images become indexable by search engines

Note

Removing images from websites (so-called “text-only versions”) make them less accessible and functional for these users and situations.

* Related WCAG resources

These tutorials provide best-practice guidance on implementing accessibility in

different situations. This page combined the following WCAG success criteria and techniques from different conformance levels:

Success Criteria:

- **1.1.1 Non-text Content:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-text-equiv-all>) All non-text content that is presented to the user has a text alternative that serves the equivalent purpose, except for the situations listed[...]. (Level A)
- **1.4.5 Images of Text:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-visual-audio-contrast-text-presentation>) If the technologies being used can achieve the visual presentation, text is used to convey information rather than images of text except [for customizable and essential images] (Level AA)
- **1.4.9 Images of Text (No Exception):** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-visual-audio-contrast-text-images>) Images of text are only used for pure decoration or where a particular presentation of text is essential to the information being conveyed (Level AAA)

(<https://www.w3.org/WAI/tutorials/images/informative/>)

Next: [Informative Images](#) ➔

Status: Updated 08 February 2022

Editors: [Eric Eggert](https://www.w3.org/People/yatil/) (<https://www.w3.org/People/yatil/>) and [Shadi Abou-Zahra](https://www.w3.org/People/shadi/) (<https://www.w3.org/People/shadi/>). Update Editor: Brian Elton. Contributors: see [Acknowledgements](https://www.w3.org/WAI/tutorials/acknowledgements/) (<https://www.w3.org/WAI/tutorials/acknowledgements/>).

Developed by the Education and Outreach Working Group ([EOWG](https://www.w3.org/groups/wg/eowg) (<https://www.w3.org/groups/wg/eowg>)). Developed with support from the WAI-ACT project (<https://www.w3.org/WAI/ACT/>), co-funded by the [European Commission IST Programme](#).

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities.

Copyright © 2025 World Wide Web Consortium ([W3C®](#)).
See [Permission to Use WAI Material](#).



[Home](#) / [Teach & Advocate](#) / [Making Events Accessible Checklist](#)

Making Events Accessible:

Checklist for meetings, conferences, training, and presentations that are remote/virtual, in-person, or hybrid

Summary

This page helps presenters, participants, and organizers make events inclusive, particularly to people with disabilities. This benefits everyone, particularly international participants.

Page Contents

- [Everyone: Understanding the Basics](#)([#everyone-understanding-the-basics](#))
- [Organizers: Planning the Event](#)([#organizers-planning-the-event](#))
- [Speakers: Planning Your Session](#)([#speakers-planning-your-session](#))
- [Participants and Speakers: During the Meeting or Presentation](#)([#participants-and-speakers-during-the-meeting-or-presentation](#))
- [For More Information](#)([#for-more-information](#))

User Experience

Do you remember a time when people around you broke out in laughter, but you didn't hear the joke? Be careful not to leave out information for some people in your audience. For example, if you say "you can read it on the slide", you are probably excluding people who cannot see the slide.

[+ Expand All Sections](#)

[- Collapse All Sections](#)

[Everyone: Understanding the Basics](#)

Be open to diversity in your audience and any accessibility issues. Basically, be aware that some of your audience might not be able to:

- **see** well or at all,
- **hear** well or at all,
- **move** well or at all,
- **speak** well or at all, or
- **understand** information presented in some ways well or at all.

For example, to be inclusive:

- organizers ensure the remote platform and the in-person venue is accessible
- speakers describe pertinent visual information
- participants speak clearly into a good quality microphone
- and other considerations listed on this page

Most aspects that are general good practice are particularly important for people with cognitive disabilities. For example:

- For meetings and presentations, start with an overview and end with a review of the most important points
- Use consistent design in slide presentations to limit cognitive load
- Use clear and understandable content ↗ (<https://www.w3.org/WAI/WCAG2/supplemental/objectives/o3-clear-content/>)

Respect participant's needs and be open for other accessibility issues. People might have accessibility needs that you didn't think of. For example: Someone might need to take breaks at set times for insulin injections. Someone with Tourette syndrome might randomly shout out during a session. Someone with a physical disability who cannot take notes might need to record the session.

Often speakers won't know if participants have disabilities. For example, at a large conference where organizers didn't ask registrants. In some cases, you might know the accessibility needs of participants ahead of time. Even then something could change. For example, a new participant could join the training at the last minute. Or someone could develop accessibility needs before the training.

Make your event and your presentations accessible so that you are prepared for such situations.



Media Players

in [Making Audio and Video Media Accessible](#) (<https://www.w3.org/WAI/media/av/>)



Summary

Some media players are not accessible to people with disabilities. There are players developed specifically for accessibility. Usually it's best to use one of these existing players, rather than code your own.

Page Contents

- [Introduction](#)(`↳ #introduction`)
- [Skills Needed](#)(`↳ #skills`)
- [Player Accessibility Functionality](#)(`↳ #player-accessibility-functionality`)
- [Existing Players](#)(`↳ #existing-players`)
 - [Support for Description Methods](#)(`↳ #support-for-description-methods`)

Introduction

Modern browsers provide a default media player. Most have limited functionality to support accessibility.

Rather than coding all the things needed to make a player support accessibility, most organizations choose to use an existing player with good accessibility support.

There are players developed specifically for accessibility. Some are free, open source and some are commercial.

Skills Needed

Using an existing media player developed for accessibility requires moderate HTML skills.

Developing your own accessible media player requires advanced HTML and JavaScript skills.

Player Accessibility Functionality

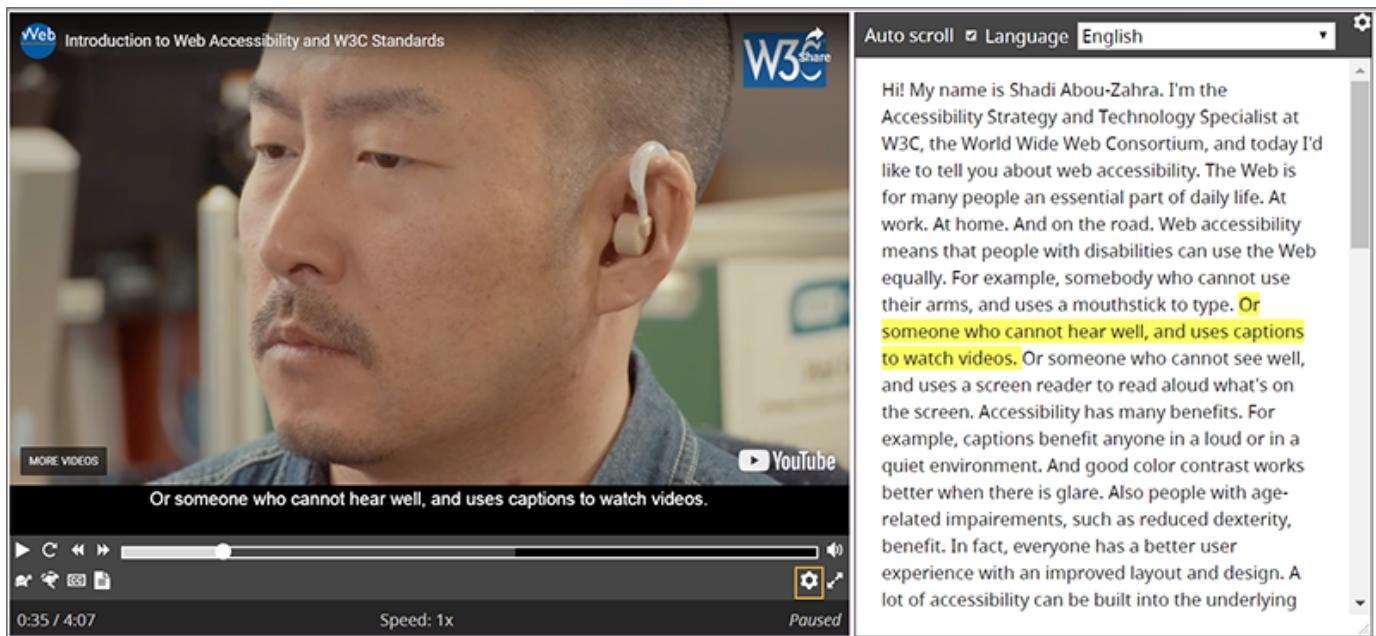
Accessible media players provide a user interface that works without a mouse, through speech interface, when the page is zoomed larger, and with screen readers. For example, media players need to:

- Provide keyboard support ([in Understanding WCAG: Keyboard Accessible \(https://www.w3.org/WAI/WCAG22/Understanding/keyboard-accessible\)](https://www.w3.org/WAI/WCAG22/Understanding/keyboard-accessible))
- Make the keyboard focus indicator visible ([in Understanding WCAG: Focus Visible \(https://www.w3.org/WAI/WCAG22/Understanding/focus-visible\)](https://www.w3.org/WAI/WCAG22/Understanding/focus-visible))
- Provide clear labels ([in Understanding WCAG: Labels or Instructions \(https://www.w3.org/WAI/WCAG22/Understanding/labels-or-instructions\)](https://www.w3.org/WAI/WCAG22/Understanding/labels-or-instructions), [Info and Relationships \(https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships\)](https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships))
- Have sufficient contrast between colors for text, controls, and backgrounds ([in Understanding WCAG: Contrast \(Minimum\) \(https://www.w3.org/WAI/WCAG22/Understanding/contrast-minimum\)](https://www.w3.org/WAI/WCAG22/Understanding/contrast-minimum), [Contrast \(Enhanced\) \(https://www.w3.org/WAI/WCAG22/Understanding/contrast-enhanced\)](https://www.w3.org/WAI/WCAG22/Understanding/contrast-enhanced), [Non-text Contrast \(https://www.w3.org/WAI/WCAG22/Understanding/non-text-contrast.html\)](https://www.w3.org/WAI/WCAG22/Understanding/non-text-contrast.html))

Some media players provide additional accessibility functionality to users such as:

- Changing the speed of the video
- Setting how captions are displayed (e.g., text style, text size, colors, and position of the captions)
- Reading the captions with a screen reader and braille device
- Interactive transcripts

Interactive transcripts use the captions file. Interactive transcripts highlight text phrases as they are spoken. Users can select text in the transcript and go to that point in the video.



More details on player accessibility functionality are in a separate document: [Media Accessibility User Requirements](https://www.w3.org/TR/media-accessibility-reqs/) (<https://www.w3.org/TR/media-accessibility-reqs/>).

Existing Players

There is information online about the accessibility of media players. For example, [Web-Based Media Player Accessibility Comparison Table \(last updated July 2016\)](https://kensgists.github.io/apt/) (<https://kensgists.github.io/apt/>).

Each media player provides documentation of the steps to set it up in a web page. For example, [AblePlayer Setup Steps](https://ableplayer.github.io/ableplayer/#setup-step-1-use-html5-doctype) (<https://ableplayer.github.io/ableplayer/#setup-step-1-use-html5-doctype>).

Support for Description Methods

Media player functionality is required for some methods of providing description of visual information (*called audio description, video description, or described video*), as explained in the [Description page](https://www.w3.org/WAI/media/av/description/) (<https://www.w3.org/WAI/media/av/description/>). To the best of our knowledge, the following media players provide such functionality:

- Supports description in text file (VTT format):
 - AblePlayer — supports description when there is space in the audio, and when the video needs to pause ("extended description")

- video.js — supports description when there is space in the audio; does **not** support description when the video needs to pause (“extended description”)
- Supports separate audio file with description:
 - AblePlayer
 - OzPlayer
 - video.js — with plug-in

(If you know of other players that provide that functionality, please let us know via GitHub or e-mail with the links in [Help improve this page below](#)( #helpimprove). Thanks!)

← Previous:
[Sign Languages](https://www.w3.org/WAI/media/av/sign-languages/) (https://www.w3.org/WAI/media/av/sign-languages/)

Next: → [Acknowledgements](https://www.w3.org/WAI/media/av/acknowledgements/) (https://www.w3.org/WAI/media/av/acknowledgements/)

Updated: 17 September 2024. [Latest changes](https://www.w3.org/WAI/media/av/changelog/) (https://www.w3.org/WAI/media/av/changelog/).
First published September 2019.

Editor: [Shawn Lawton Henry](https://www.w3.org/People/Shawn) (https://www.w3.org/People/Shawn). [Acknowledgements](https://www.w3.org/WAI/media/av/acknowledgements/) (https://www.w3.org/WAI/media/av/acknowledgements/) lists contributors and credits.

Developed by the Education and Outreach Working Group ([EOWG](https://www.w3.org/WAI/EO/) (https://www.w3.org/WAI/EO/)). Originally drafted as part of the [WCAG TA Project](https://www.w3.org/WAI/WCAGTA/) (https://www.w3.org/WAI/WCAGTA/) funded by the U.S. Access Board. Revised as part of the [WAI Expanding Access project](https://www.w3.org/WAI/expand-access/) (https://www.w3.org/WAI/expand-access/) funded by the Ford Foundation.

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities. Copyright © 2025 World Wide Web Consortium (W3C®). See [Permission to Use WAI Material](#).



Menus Tutorial

in [Tutorials](https://www.w3.org/WAI/tutorials/) (<https://www.w3.org/WAI/tutorials/>)

Menus are used for navigation and to provide functionality which are critical parts of web page operability.

- **Structure** (<https://www.w3.org/WAI/tutorials/menus/structure/>): Mark up menus in a way that reflects their structure and appropriately labels them.
- **Styling** (<https://www.w3.org/WAI/tutorials/menus/styling/>): Use commonly recognized design patterns to distinguish menus and the state of menu items.
- **Fly-out Menus** (<https://www.w3.org/WAI/tutorials/menus/flyout/>): Ensure fly-out (drop-down) submenus can be used appropriately by mouse and keyboard.
- **Application Menus** (<https://www.w3.org/WAI/tutorials/menus/application-menus/>): Add specific markup and keyboard behavior to resemble desktop application menus.

Why is this important?

Navigation menus reflect the underlying structure of websites. Application menus provide access to the essential functionality of an application. Thus menus are critical parts of web pages and applications and require particular attention during design and development.

- **Screen reader and keyboard users** benefit from keyboard interoperability and markup that allows them to operate menus in different ways.
- **Users with fine motor difficulties and touch screen users** require larger targets to click or tap on. In fly-out menus, submenus should not disappear immediately after the mouse has left the clickable area.
- **People with limited attention or short-term memory** benefit from clear and distinct menus with easily identifiable states, such as the current page.

* Related WCAG resources

These tutorials provide best-practice guidance on implementing accessibility in different situations. This page combined the following WCAG success criteria and techniques from different conformance levels:

Success Criteria:

- **2.4.1 Bypass Blocks:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-navigation-mechanisms-skip>) A mechanism is available to bypass blocks of content that are repeated on multiple Web pages. (Level A)
- **2.4.3 Focus Order:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-navigation-mechanisms-focus-order>) If a Web page can be navigated sequentially and the navigation sequences affect meaning or operation, focusable components receive focus in an order that preserves meaning and operability. (Level A)
- **2.4.5 Multiple Ways:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-navigation-mechanisms-mult-loc>) More than one way is available to locate a Web page within a set of Web pages except where the Web Page is the result of, or a step in, a process. (Level AA)
- **2.4.7 Focus Visible:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-navigation-mechanisms-focus-visible>) Any keyboard operable user interface has a mode of operation where the keyboard focus indicator is visible. (Level AA)
- **2.4.8 Location:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-navigation-mechanisms-location>) Information about the user's location within a set of Web pages is available. (Level AAA)
- **4.1.2 Name, Role, Value:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-ensure-compat-rsv>) For all user interface components (including but not limited to: form elements, links and components generated by scripts), the name and role can be programmatically determined; states, properties, and values that can be set by the user can be programmatically set; and notification of changes to these items is available to user agents, including assistive technologies. (Level A)

Techniques:

- **ARIA6: Using aria-label to provide labels for objects** (<https://www.w3.org/TR/WCAG20-TECHS/ARIA6>)
- **ARIA11: Using ARIA landmarks to identify regions of a page** (<https://www.w3.org/TR/WCAG20-TECHS/ARIA11>)

- [G14: Ensuring that information conveyed by color differences is also available in text](https://www.w3.org/TR/WCAG20-TECHS/G14) (<https://www.w3.org/TR/WCAG20-TECHS/G14>)
- [G63: Providing a site map](https://www.w3.org/TR/WCAG20-TECHS/G63) (<https://www.w3.org/TR/WCAG20-TECHS/G63>)
- [G65: Providing a breadcrumb trail](https://www.w3.org/TR/WCAG20-TECHS/G65) (<https://www.w3.org/TR/WCAG20-TECHS/G65>)
- [G127: Identifying a Web page's relationship to a larger collection of Web pages](https://www.w3.org/TR/WCAG20-TECHS/G127) (<https://www.w3.org/TR/WCAG20-TECHS/G127>)
- [G128: Indicating current location within navigation bars](https://www.w3.org/TR/WCAG20-TECHS/G128) (<https://www.w3.org/TR/WCAG20-TECHS/G128>)
- [G161: Providing a search function to help users find content](https://www.w3.org/TR/WCAG20-TECHS/G161) (<https://www.w3.org/TR/WCAG20-TECHS/G161>)
- [G182: Ensuring that additional visual cues are available when text color differences are used to convey information](https://www.w3.org/TR/WCAG20-TECHS/G182) (<https://www.w3.org/TR/WCAG20-TECHS/G182>)
- [G183: Using a contrast ratio of 3:1 with surrounding text and providing additional visual cues on focus for links or controls where color alone is used to identify them](https://www.w3.org/TR/WCAG20-TECHS/G183) (<https://www.w3.org/TR/WCAG20-TECHS/G183>)
- [H4: Creating a logical tab order through links, form controls, and objects](https://www.w3.org/TR/WCAG20-TECHS/H4) (<https://www.w3.org/TR/WCAG20-TECHS/H4>)
- [SCR26: Inserting dynamic content into the Document Object Model immediately following its trigger element](https://www.w3.org/TR/WCAG20-TECHS/SCR26) (<https://www.w3.org/TR/WCAG20-TECHS/SCR26>)

(<https://www.w3.org/WAI/tutorials/menus/structure/>)

Next: [Menu Structure](#) ➔

Status: Updated 13 April 2017 (first published May 2015)

Editors: [Eric Eggert](https://www.w3.org/People/yatil/) (<https://www.w3.org/People/yatil/>) and [Shadi Abou-Zahra](https://www.w3.org/People/shadi/) (<https://www.w3.org/People/shadi/>). Update Editor: Brian Elton. Contributors: see [Acknowledgements](https://www.w3.org/WAI/tutorials/acknowledgements/) (<https://www.w3.org/WAI/tutorials/acknowledgements/>).

Developed by the Education and Outreach Working Group ([EOWG](https://www.w3.org/groups/wg/eowg) (<https://www.w3.org/groups/wg/eowg>)). Developed with support from the [WAI-ACT project](https://www.w3.org/WAI/ACT/) (<https://www.w3.org/WAI/ACT/>), co-funded by the [European Commission IST Programme](#).

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities.

Copyright © 2025 World Wide Web Consortium ([W3C®](#)).
See [Permission to Use WAI Material](#).



Page Structure Tutorial

Well-structured content allows more efficient navigation and processing. Use HTML and WAI-ARIA to improve navigation and orientation on web pages and in applications.

- **Page Regions** (<https://www.w3.org/WAI/tutorials/page-structure/regions/>): Identify and mark up regions on web pages using HTML5 and WAI-ARIA roles.
- **Labeling Regions** (<https://www.w3.org/WAI/tutorials/page-structure/labels/>): Label regions to allow users to distinguish and access them.
- **Headings** (<https://www.w3.org/WAI/tutorials/page-structure/headings/>): Add headings and nest them logically to label sections of web pages according to their relationships and importance.
- **Content Structure** (<https://www.w3.org/WAI/tutorials/page-structure/content/>): Mark up the content on a page in a way that uses appropriate and meaningful elements.

Why is this important?

Pages with well-structured content are essential for many web users, for example:

- **People with cognitive and learning disabilities** can more easily find and prioritize content on the page.
- **People using screen readers** can skip to the main content directly and navigate to sections that are important to them.
- **Keyboard users** can browse pages and their sections more efficiently. Otherwise, users have to press the tab key multiple times to navigate through all links in each section.
- **People using software that only shows the main content** of a web page, such as people with cognitive disabilities, will receive better results if the page structure is correctly marked up.

- **People with visual impairments**, including people with low vision, have cues that provide orientation on the page and in the content.
- **Mobile web users** often have access to a so-called “reader” or “reading” mode that will only show the main content of the page if it is correctly marked up.
- **People using certain browser plugins** can use landmark roles to jump to specific sections on a page.
- There are additional benefits to a good, accessible page structure, beyond those experienced by people with disabilities. As an example, search engines can use the data to better index the content of a page.

* Related WCAG resources

These tutorials provide best-practice guidance on implementing accessibility in different situations. This page combined the following WCAG success criteria and techniques from different conformance levels:

Success Criteria:

- **1.3.1 Info and Relationships:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-content-structure-separation-programmatic>) Information, structure, and relationships conveyed through presentation can be programmatically determined or are available in text. (Level A)
- **2.4.1 Bypass Blocks:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-navigation-mechanisms-skip>) A mechanism is available to bypass blocks of content that are repeated on multiple Web pages. (Level A)
- **2.4.6 Headings and Labels:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-navigation-mechanisms-descriptive>) Headings and labels describe topic or purpose. (Level AA)
- **2.4.10 Section Headings:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-navigation-mechanisms-headings>) Section headings are used to organize the content. (Level AAA)

(<https://www.w3.org/WAI/tutorials/page-structure/regions/>)

Next: [Page Regions](#) ➔

[shadi/\). Update Editor: Brian Elton. Contributors: see \[Acknowledgements\]\(#\)](#) (<https://www.w3.org/WAI/tutorials/acknowledgements/>).

Developed by the Education and Outreach Working Group ([EOWG](#) (<https://www.w3.org/groups/wg/eowg>)). Developed with support from the WAI-ACT project (<https://www.w3.org/WAI/ACT/>), co-funded by the [European Commission IST Programme](#).

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities.

Copyright © 2025 World Wide Web Consortium ([W3C®](#)).
See [Permission to Use WAI Material](#).



Planning Audio and Video Media

in [Making Audio and Video Media Accessible](#) (<https://www.w3.org/WAI/media/av/>)



Summary

This page helps you:

- **figure out which accessibility aspects your specific audio or video needs** (captions, description, a transcript, etc.)
- manage projects and plan what to develop in-house and what to outsource
- understand the **standards** for audio and video media in Web Content Accessibility Guidelines (WCAG)

Page Contents

- [Influences](#)(↳ #influences)
- [Checklists for Audio and Video](#)(↳ #checklist)
- [Project Management](#)(↳ #project-management)
- [WCAG Standard](#)(↳ #wcag-standard)

Influences

What accessibility features you provide with your media will likely be influenced by:

- [user needs and business benefits](#) (<https://www.w3.org/WAI/media/av/users-orgs/>)
- governmental regulations and other policy requirements
- budget and time constraints

This multi-page resource endeavors to help you know the requirements and encourages you to meet all user needs.

Checklists for Audio and Video

The checklists below cover audio-only content and video content, and pre-recorded and live. They include:

- What is required in the Web Content Accessibility Guidelines (WCAG) standard at Level A, AA, and AAA. ([WCAG\(↴ #wcag-standard\)](#) is explained below.)
- What is needed to meet user needs, beyond WCAG. (If it doesn't have any 'A', then it is not required in WCAG.)

The links below go to a web page in this resource with details on understanding and implementing each thing.

All Audio and Video Media

- [**Audio content** \(<https://www.w3.org/WAI/media/av/av-content/#audio>\) \(A\)](#) is accessible (for example, what is said and how it's recorded)
- [**Video content** \(<https://www.w3.org/WAI/media/av/av-content/#video>\) \(A\)](#) is accessible (for example, doesn't cause seizures)
- [**Media player** \(<https://www.w3.org/WAI/media/av/player/>\) \(A\)](#) supports accessibility

Audio-only Checklists

This section covers audio-only media, like podcasts that don't have video.

Pre-Recorded Audio-only

- [**Transcript** \(<https://www.w3.org/WAI/media/av/transcripts/>\) \(A\)](#) separate from the audio
- [**Captions** \(<https://www.w3.org/WAI/media/av/captions/>\)](#) synchronized with the audio
- [**Sign language\(s\)** \(<https://www.w3.org/WAI/media/av/sign-languages/>\)](#)

Live Audio-only

- [**Transcript** \(<https://www.w3.org/WAI/media/av/transcripts/>\) \(AAA\)](#) — live stream or transcript when live
- [**Captions** \(<https://www.w3.org/WAI/media/av/captions/>\)](#)
- [**Sign language\(s\)** \(<https://www.w3.org/WAI/media/av/sign-languages/>\)](#)

Video Checklists

 **Pre-Recorded Video**

Does the video have speech or other audio that is needed to understand the content?

- If yes,
 - [Captions \(https://www.w3.org/WAI/media/av/captions/\) \(A\)](https://www.w3.org/WAI/media/av/captions/)
 - [Transcript of audio information \(https://www.w3.org/WAI/media/av/transcripts/\) \(AAA\)](https://www.w3.org/WAI/media/av/transcripts/)
 - [Sign language\(s\) \(https://www.w3.org/WAI/media/av/sign-languages/\) \(AAA\)](https://www.w3.org/WAI/media/av/sign-languages/)
- If no, inform users(§ #none).

Does the video have visual information that is needed to understand the content?

- If yes,
 - [Audio description of the visual information \(https://www.w3.org/WAI/media/av/description/\) or descriptive transcript \(https://www.w3.org/WAI/media/av/transcripts/\) \(A\)](https://www.w3.org/WAI/media/av/description/)
 - [Audio description of the visual information \(https://www.w3.org/WAI/media/av/description/\) \(AA\)](https://www.w3.org/WAI/media/av/description/)
 - [Descriptive transcript \(https://www.w3.org/WAI/media/av/transcripts/\) \(AAA\)](https://www.w3.org/WAI/media/av/transcripts/) (*If you have a descriptive transcript, you do not need an additional transcript of only audio information from the previous question.*)
- If no, inform users(§ #none).

 **Live Video**

Is there speech or other audio that is needed to understand the content?

- If yes,
 - [Captions \(https://www.w3.org/WAI/media/av/captions/\) \(AA\)](https://www.w3.org/WAI/media/av/captions/)
 - [Sign language\(s\) \(https://www.w3.org/WAI/media/av/sign-languages/\)](https://www.w3.org/WAI/media/av/sign-languages/)
 - Text stream available to screen readers (and braille devices)
- If no, inform users(§ #none).

Is there visual information that is needed to understand the content?

- If yes,
 - Description of important visual information in a text stream available to screen readers (and braille devices)
- If no, inform users(§ #none).

Inform Users When Not Needed

If your video does not need captions (because there is no substantive audio content) or does not need description (because there is no substantive visual content), it's good to let users know that. Otherwise, they might think that you accidentally forgot to provide it.

Users who need captions will look there, so you can provide a captions file with only the appropriate indication, such as "[background music]". Or you can provide the information in text with the video, such as:

Captions not needed: The only sound in this video is background music.

Description not needed: The visuals in this video only support what is spoken; the visuals do not provide additional information.

Provide Both Captions and a Transcript

It is best to provide captions and a separate transcript.

For videos, captions enable people who are Deaf or hard of hearing to see the visual content and read the captions at the same time.

For audio-only, captions enable people who are hard of hearing to get the richness of listening to the audio and fill in what they don't hear well by reading the captions.

Transcripts are needed to provide access to people who are Deaf-blind and use braille. Also, transcripts are used by people without disabilities, as listed in the intro page of this resource under [Benefits to Organizations and Individuals](#) (<https://www.w3.org/WAI/media/av/users-orgs/#benefits>).

Descriptive Transcripts

Descriptive transcripts for videos:

- are needed for most videos to be accessible to people who are "Deaf-blind"
- meet a wide range of accessibility needs, including for people who have difficulty processing auditory information and people who cannot focus and comprehend auditory or visual information when there is changing visuals
- are used by people *without* disabilities, and benefit your organization (examples are in the intro page under [Benefits to Organizations and Individuals](#) (<https://www.w3.org/WAI/media/av/users-orgs/#benefits>))
- **are easy and inexpensive to develop** using captions and description that you already have to

meet Level AA

Captions and transcripts use the same text. Once you have one, it's fairly easy to develop the other.

Other Languages

Translation of the audio into other languages can be provided:

- as text, using captions format (called subtitles or interlingual subtitles)
- as spoken audio, usually as a separate audio stream (for people who cannot read captions)
- as sign language

Project Management

Include specific accessibility requirements in your:

- Project requirements - internal and external
- Requests for Proposal (RFP) or Requests for Tender (RFT)
- Contracts

Here is an example workflow for developing an accessible video, with notes on who develops the material. Links go to other pages in this resource.

Example Workflow and Responsibilities

1. Address accessibility in [video content](https://www.w3.org/WAI/media/av/av-content/) (<https://www.w3.org/WAI/media/av/av-content/>) as the video is planned and produced.
By: Script writers, videographers, producers, and others.
2. Develop a [described version of the video](https://www.w3.org/WAI/media/av/description/) (<https://www.w3.org/WAI/media/av/description/>) at the same time as the main video, if needed.
By: Usually the same people who produce the main video also produce the described version.
3. Develop [captions](https://www.w3.org/WAI/media/av/captions/) (<https://www.w3.org/WAI/media/av/captions/>) for the main video and for the described version.
By: Usually if the video is professionally produced, the producers provide captions. Sometimes when informal videos are developed in-house, captions are outsourced.
4. Develop a [descriptive transcript](https://www.w3.org/WAI/media/av/transcripts/) (<https://www.w3.org/WAI/media/av/transcripts/>)

using the text from the caption files.

By: Often transcripts are developed in-house from caption files.

5. Implement it in an **accessible media player** (<https://www.w3.org/WAI/media/av/player/>) (usually an existing player with good accessibility support).

By: Usually in-house web developers.

Resourcing Accessibility

To help you **plan in-house and outsourced work**, the pages of this resource include considerations, skills, and tools needed for creating accessible media in these sections:

- [Description Considerations, Skills, and Tools](https://www.w3.org/WAI/media/av/description/#description-considerations-skills-and-tools) (<https://www.w3.org/WAI/media/av/description/#description-considerations-skills-and-tools>)
- [Captions, Skills and Tools](https://www.w3.org/WAI/media/av/captions/#skills-and-tools) (<https://www.w3.org/WAI/media/av/captions/#skills-and-tools>) and [Automatic Captions are Not Sufficient](https://www.w3.org/WAI/media/av/captions/#automatic-captions-are-not-sufficient) (<https://www.w3.org/WAI/media/av/captions/#automatic-captions-are-not-sufficient>)
- [Transcripts, Process - Skills and Tools](https://www.w3.org/WAI/media/av/transcripts/#process-skills-and-tools) (<https://www.w3.org/WAI/media/av/transcripts/#process-skills-and-tools>)
- [Media Players, Skills and Tools](https://www.w3.org/WAI/media/av/player/#skills) (<https://www.w3.org/WAI/media/av/player/#skills>)
- [Sign Languages, Skills and Tools](https://www.w3.org/WAI/media/av/sign-languages/#skills-and-tools) (<https://www.w3.org/WAI/media/av/sign-languages/#skills-and-tools>)

When planning and budgeting for accessible media, it is often helpful to communicate the **benefits to organizations**, such as search engine optimization (SEO), better user experience for all, improved customer satisfaction, and more listed in the intro page under [Benefits to Organizations and Individuals](https://www.w3.org/WAI/media/av/users-orgs/#benefits) (<https://www.w3.org/WAI/media/av/users-orgs/#benefits>).

WCAG Standard

Web Content Accessibility Guidelines (WCAG) is introduced in a separate resource: [WCAG Overview](https://www.w3.org/WAI/standards-guidelines/wcag/) (<https://www.w3.org/WAI/standards-guidelines/wcag/>).

This resource uses most WCAG terminology, with a few differences:

- “time-based media” in WCAG = “audio and video media” in this resource
- “[alternative for time based media in WCAG](https://www.w3.org/TR/WCAG#alt-time-based-mediadef) (<https://www.w3.org/TR/WCAG#alt-time-based-mediadef>)” = “transcript” for audio-only and “descriptive transcript” for video in this resource

WCAG includes requirements for audio and video media at Level A, AA, and AAA. (More info in a separate resource: [Understanding Levels of Conformance \(https://www.w3.org/WAI/WCAG22/Understanding/conformance.html#levels\)](https://www.w3.org/WAI/WCAG22/Understanding/conformance.html#levels).) Most media is required by governing policies to meet Level AA — which includes both A and AA listed in the tables below.

Accessibility requirements for video and audio are different based on if they are:

- pre-recorded or live
- video with audio, video without audio (video-only), or audio-only

The links in the tables below go to a page in a separate resource: Understanding WCAG 2.1.

Pre-Recorded

	Transcript (including auditory and visual content)	Captions	Audio Description (if visual content not in audio)	Sign Language
Audio-only	A 1.2.1 (https://www.w3.org/WAI/WCAG22/Understanding/audio-only-and-video-only-prerecorded)			
Video-only	A 1.2.1 (https://www.w3.org/WAI/WCAG22/Understanding/audio-only-and-video-only-prerecorded) (transcript or audio track) AAA 1.2.8 (https://www.w3.org/WAI/WCAG22/Understanding/media-alternative)		A 1.2.1 (https://www.w3.org/WAI/WCAG22/Understanding/audio-only-and-video-only-prerecorded) (audio track or transcript)	

	<u>prerecorded)</u>			
Video with Audio	<u>AAA 1.2.8</u> <u>(https://</u> <u>www.w3.org/</u> <u>WAI/WCAG22/</u> <u>Understanding/</u> <u>media-</u> <u>alternative-</u> <u>prerecorded)</u>	A 1.2.2 (https:// www.w3.org/ WAI/WCAG22/ Understanding/ captions- prerecorded)	A 1.2.3 (https:// www.w3.org/WAI/ WCAG22/ Understanding/ audio-description- or-media- alternative- prerecorded) (audio description <i>or</i> transcript) AA 1.2.5 (https:// www.w3.org/WAI/ WCAG22/ Understanding/ audio-description- prerecorded) <u>AAA 1.2.7 (https://</u> <u>www.w3.org/WAI/</u> <u>WCAG22/</u> <u>Understanding/</u> <u>extended-audio-</u> <u>description-</u> <u>prerecorded)</u>	<u>AAA 1.2.6</u> <u>(https://</u> <u>www.w3.org/</u> <u>WAI/WCAG22/</u> <u>Understanding/</u> <u>sign-language-</u> <u>prerecorded)</u>

Live

	Transcript	Captions	Audio Description	Sign Language
Audio- only	<u>AAA 1.2.9 (https://</u> <u>www.w3.org/WAI/</u> <u>WCAG22/</u> <u>Understanding/audio-</u> <u>only-live)</u> (live stream <i>or</i> accurate transcript when live)			
Video- only				

Video with Audio	<u>AA 1.2.4 (https://www.w3.org/WAI/WCAG22/Understanding/captions-live)</u>	
---------------------------------	--	--

More about Standards

To learn more about WCAG requirements for media, see Understanding Guideline 1.2: Time-based Media (<https://www.w3.org/WAI/WCAG22/Understanding/time-based-media>).

Other WCAG requirements related to audio and video include:

- In this resource:
 - Accessible Audio and Video Content (<https://www.w3.org/WAI/media/av/av-content/>)
 - Media Player Functionality (<https://www.w3.org/WAI/media/av/player/#player-accessibility-functionality>)
- In Understanding WCAG:
 - 2.2.2 Pause, Stop, Hide (<https://www.w3.org/WAI/WCAG22/Understanding/pause-stop-hide>) (Level A) For moving, blinking, scrolling, or auto-updating information, all of the following are true:...
 - 1.4.2 Audio Control (<https://www.w3.org/WAI/WCAG22/Understanding/audio-control>) (Level A) If any audio on a Web page plays automatically for more than 3 seconds, either a mechanism is available to pause or stop the audio, or a mechanism is available to control audio volume...

Your audio and video may be subject to additional requirements, for example under governmental regulations. Some of these are listed in Web Accessibility Laws & Policies (<https://www.w3.org/WAI/policies/>).

 Previous: User Needs	(https://www.w3.org/WAI/media/av/users-orgs/)	(https://www.w3.org/WAI/media/av/av-content/)	Next: Audio Content & Video Content 
--	---	---	---

Updated: 17 September 2024. Latest changes (<https://www.w3.org/WAI/media/av/changelog/>).
First published September 2019.

Editor: Shawn Lawton Henry (<https://www.w3.org/People/Shawn>). Acknowledgements (<https://www.w3.org/WAI/media/av/acknowledgements>) lists contributors and credits.

Developed by the Education and Outreach Working Group (EOWG (<https://www.w3.org/WAI/EO/>)). Originally drafted as part of the WCAG TA Project (<https://www.w3.org/WAI/WCAGTA/>) funded by the U.S. Access Board. Revised as part of the WAI Expanding Access project (<https://www.w3.org/WAI/expand-access/>) funded by the Ford Foundation.

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web
accessible to people with disabilities.

Copyright © 2025 World Wide Web Consortium ([W3C®](#)). See
[Permission to Use WAI Material](#).



Sign Languages

in [Making Audio and Video Media Accessible](#) (<https://www.w3.org/WAI/media/av/>)



Summary

Sign languages use hand and arm movements, facial expressions, and body positions to convey meaning. For many people who are Deaf, sign language is their native language, and some do not understand written language well.

This page helps you understand and create sign language interpretation for audio and video. (Sign language is not required by most policies.)

Page Contents

- [Introduction](#)(<#introduction>)
- [Standards Requirements](#)(<#standards-requirements>)
- [Skills and Tools](#)(<#skills-and-tools>)
- [Example](#)(<#example>)
- [Creating Sign Language Alternatives](#)(<#creating-sign-language-alternatives>)

Introduction

Sign languages use hand and arm movements, facial expressions, and body positions to convey meaning.

Sign language is the native language of many people who are Deaf. Some do not read or understand written language well — especially at the speed of most captions.

Some people will want to have sign language and



captions at the same time.



Not everyone who is Deaf knows sign language, especially if they became Deaf or hard-of-hearing later in life. Some people use "lip reading" to help understand speech, though that cannot be relied upon for accessibility.

Sign languages are different across regions and countries. For example, American Sign Language (ASL), Black American Sign Language (BASL), British Sign Language (BSL), and Auslan (Australian Sign Language) are different.

There are some efforts to provide automatic sign language from text; however, avatars that simulate sign language interpretation are not robust enough to be adequate.

Standards Requirements

Sign language is not required in most web accessibility policies.

Sign language is in the WCAG standard at Level AAA: [Understanding 1.2.6 Sign Language \(https://www.w3.org/WAI/WCAG22/Understanding/sign-language-prerecorded.html\)](https://www.w3.org/WAI/WCAG22/Understanding/sign-language-prerecorded.html). (*The Planning page of this resource introduces the WCAG Standard (https://www.w3.org/WAI/media/av/planning/#wcag-standard).*)

Skills and Tools

To include sign language alternatives, you will need people, skills, and tools to:

- do the sign language interpretation
- record it
- edit it with the audio or video file

Example

Example sign language in video: [NHS 111 British Sign Language \(BSL\) Advert \(YouTube\)](#)



(<https://www.youtube.com/watch?v=TCq3ru9HQSc>)

Creating Sign Language Alternatives

Use colors that are easy to distinguish - *planning, recording*

Usually it is best if the background and the signer's clothing are solid colors that contrast with their skin tone. That way their hands and face are easier to see.

Use good lighting - *planning, recording*

Ensure good lighting to help make the signer clearly visible.

Capture the full signing space - *recording*

For most sign languages the signing space extends from well below the waist to above the head and at least an elbow width to each side.

Ensure the signer is large enough - *post-production*

Viewers need to be able to clearly see all movements and facial expressions.

Avoid obscuring important content - *post-production*

Position the signer to avoid obscuring important information in the video. The signer is usually at the bottom right. If your video has information such as a news ticker at the bottom, position the signer above that.

Ideally when the video was made, the position of the signer was planned for, as noted in another page of this resource: [Plan for sign language – storyboarding, recording](#) (<https://www.w3.org/WAI/media/av/av-content/#plan-for-sign-language>).

Make sign language video(s) easy to discover and use

Make it easy to get to the video with sign language and to the video without sign

language.

For example, directly underneath the media player, include a large visible toggle button and/or labelled image for the sign language video(s).



There are resources on the web that provide additional guidance on creating sign language alternatives. For example:

- [Sign Language Interpretation in HBBTV \(PDF\)](http://pagines.uab.cat/hbb4all/sites/pagines.uab.cat.hbb4all/files/sign_language_interpreting_in_hbbtv.pdf) (http://pagines.uab.cat/hbb4all/sites/pagines.uab.cat.hbb4all/files/sign_language_interpreting_in_hbbtv.pdf) includes specific guidance on aspects such as working with sign language interpreters and types of onscreen presentation
- Guidelines for the Production of Signing Books includes [Sign language presentation](https://www.sign-lang.uni-hamburg.de/signingbooks/deliver/d31/deliv_31_part3-2.html#3.2.2.6) (https://www.sign-lang.uni-hamburg.de/signingbooks/deliver/d31/deliv_31_part3-2.html#3.2.2.6) and [Editing](https://www.sign-lang.uni-hamburg.de/signingbooks/sbrc/grid/d71/guide13.htm) (<https://www.sign-lang.uni-hamburg.de/signingbooks/sbrc/grid/d71/guide13.htm>)

Previous:

← [Transcribing Audio to Text](#)

(<https://www.w3.org/WAI/media/av/transcribing/>)

(<https://www.w3.org/WAI/media/av/player/>)

Next:

[Media Player](#) →

Updated: 17 September 2024. [Latest changes](https://www.w3.org/WAI/media/av/changelog/) (<https://www.w3.org/WAI/media/av/changelog/>).

First published September 2019.

Editor: [Shawn Lawton Henry](https://www.w3.org/People/Shawn) (<https://www.w3.org/People/Shawn>). [Acknowledgements](https://www.w3.org/WAI/media/av/acknowledgements/) (<https://www.w3.org/WAI/media/av/acknowledgements/>) lists contributors and credits.

Developed by the Education and Outreach Working Group ([EOWG](https://www.w3.org/WAI/EO/) (<https://www.w3.org/WAI/EO/>)). Originally drafted as part of the [WCAG TA Project](#) (<https://www.w3.org/WAI/WCAGTA/>) funded by the U.S. Access Board. Revised as part of the [WAI Expanding Access project](#) (<https://www.w3.org/WAI/expand-access/>) funded by the Ford Foundation.

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities.

Copyright © 2025 World Wide Web Consortium ([W3C](#)®).
See [Permission to Use WAI Material](#).



Standards to Make Your LMS Accessible

Summary

This page briefly introduces a standard to help make your learning management system (LMS) and other education tools accessible to people with disabilities.

Page Contents

- [Why Accessibility Matters](#)(↳ #why-accessibility-matters)
- [Example Scenarios](#)(↳ #example-scenarios)
- [The Accessibility Standard to Help You](#)(↳ #the-accessibility-standard-to-help-you)

Why Accessibility Matters

Your current customers and potential new customers need learning management systems (LMS) that:

- are accessible to instructors and other LMS users with disabilities
- produce accessible content for students

Accessibility for instructors and students is an essential aspect of diversity, equity, and inclusion.

Accessibility is required by law and is a procurement requirement in many situations.

Example Scenarios

These *persona* scenarios show examples of accessibility *problems* that disabled people experience using LMSs, and what *works well* in tools that are accessible.

Everyone can use previews

Persona: Zola is a physics professor at a large university. She is blind and uses a screen reader that reads aloud the information on her screen.

Problem: “I can't preview uploaded content for my students using my screen reader. I can't navigate the preview like I can a regular web page.”

Works well: “The content preview works just like a web page in a browser. I can navigate it easily using my screen reader's functionality.”

Users who have auditory, cognitive, neurological, physical, speech, or visual disabilities need to be able to preview the course content before posting it for students.

Everyone can edit content

Persona: Aroon is a school administrator. He can't move his arms due to a spinal cord injury and he uses speech recognition to navigate through applications and websites.

Problem: “I can't sort the course attendees data table. There's no way to activate the column sort with voice.”

Works well: “I can interact with the course attendees table by voice. I can select the column I want to sort by.”

Some people cannot use a mouse. They need to be able to use your LMS with their tools, such as speech recognition, keyboard, and switches.

The LMS helps make course content accessible

Persona: Irina is the director of an online professional training center. She is an expert in her job; however, she doesn't know much about accessibility.

Problem: “I'm adding charts to our courses. I have no idea what I need to do to make them accessible to students with disabilities.”

Works well: “Whenever I add a chart to a course, the tool prompts me to add a short description and a full text alternative for the chart data. There's also “Learn More” that explains why it's important for accessibility, and how to do it well.”

LMSs need to produce accessible content. Part of that is up to your tool and part is up to

the user. Your tool can provide prompts and information to help users know what they need to do to provide accessible course content.

The Accessibility Standard to Help You

Your learning management system (LMS) is sometimes called an “authoring tool” because people use it to author or create course content. There is an international standard that addresses accessibility needs in LMSs: Authoring Tool Accessibility Guidelines (ATAG).

Use ATAG to help make your tool:

- **accessible to instructors and other users with disabilities (Part A)**
- **support accessible content for students (Part B)**

Examples

ATAG covers the example scenarios above:

- *Everyone can use previews* — ATAG says: Ensure that previews are accessible. Course authors often periodically check how browsers will display the content to students. Authors with disabilities need the same opportunity to check their work.
- *Everyone can edit content* — ATAG says: Provide keyboard access to authoring features. Some authors with limited mobility or visual disabilities do not use a mouse. Instead, they require keyboard interface access to all of the functionality of the authoring tool.
- *The LMS helps make course content accessible* — ATAG says: Guide authors to produce accessible content. Assist authors with managing alternative content for non-text content. Ensure that documentation promotes the production of accessible content.

And ATAG covers much more.

To get started putting ATAG to work for you, see:

- **Authoring Tool Accessibility Guidelines (ATAG) Overview (<https://www.w3.org/WAI/standards-guidelines/atag/>)**, with links to the ATAG standard and Implementing ATAG
- **ATAG at a Glance (<https://www.w3.org/WAI/standards-guidelines/atag/glance/>)**, a paraphrased summary to give you an idea of what's covered in ATAG

Updated: 8 December 2022.

Editors: Daniel Montalvo and Shawn Henry (<https://www.w3.org/People/Shawn/>). Contributors: Kevin White, Jade Matos Carew, and participants of the Education and Outreach Working Group (<https://www.w3.org/groups/wg/eowg/participants>).

Developed by the Education and Outreach Working Group (EOWG (<https://www.w3.org/WAI/EO>)). Developed as part of the WAI-Guide

[project](https://www.w3.org/WAI/about/projects/wai-guide) (<https://www.w3.org/WAI/about/projects/wai-guide>), co-funded by the European Commission.

© This work is licensed under a [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/) (<https://creativecommons.org/licenses/by/4.0/>).

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities.

Copyright © 2025 World Wide Web Consortium ([W3C](#)®).
See [Permission to Use WAI Material](#).



Tables Tutorial

in [Tutorials](https://www.w3.org/WAI/tutorials/) (<https://www.w3.org/WAI/tutorials/>)

Data tables are used to organize data with a logical relationship in grids. Accessible tables need HTML markup that indicates header cells and data cells and defines their relationship. Assistive technologies use this information to provide context to users.

Header cells must be marked up with `<th>`, and data cells with `<td>` to make tables accessible. For more complex tables, explicit associations may be needed using `scope`, `id`, and `headers` attributes.

This tutorial shows you how to apply appropriate structural markup to tables. It includes the following pages:

Tables with one header (<https://www.w3.org/WAI/tutorials/tables/one-header/>) for rows or columns: For tables with content that is easy to distinguish, mark up header cells with `<th>` and data cells with `<td>` elements.

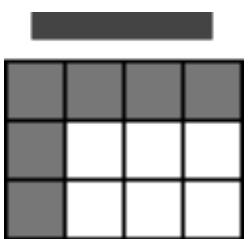
Tables with two headers (<https://www.w3.org/WAI/tutorials/tables/two-headers/>) have a simple row header and a simple column header: For tables with unclear header directions, define the direction of each header by setting the `scope` attribute to `col` or `row`.

Tables with irregular headers (<https://www.w3.org/WAI/tutorials/tables/irregular/>) have header cells that span multiple columns and/or rows: For these tables, define column and row groups and set the range of the header cells using the `colgroup` and `rowgroup` values of the `scope` attribute.

Tables with multi-level headers (<https://www.w3.org/WAI/tutorials/tables/multi-level/>) have multiple header cells associated per data cell:



For tables that are so complex that header cells can't be associated in a strictly horizontal or vertical way, use `id` and `headers` attributes to associate header and data cells explicitly.



Caption & Summary (<https://www.w3.org/WAI/tutorials/tables/caption-summary/>): A caption identifies the overall topic of a table and is useful in most situations. A summary provides orientation or navigation hints in complex tables.

Some document formats other than HTML, such as PDF, provide similar mechanisms to markup table structures. Word processing applications may also provide mechanisms to markup tables. Tables markup is often lost when converting from one format to another, though some programs may provide functionality to assist converting table markup.

Many web authoring tools and content management systems (CMS) provide functions to define header cells during table creation without having to edit the code manually.

Note

This tutorial provides guidance for creating tables used to display data in a grid. This tutorial does not apply to tables used for layout. As a general rule, tables aren't meant to be used for layout purposes. Instead, a best practice is to use Cascading Style Sheets (CSS) for visual presentation.

Why is this important?

Tables without structural markup to differentiate and properly link between header and data cells, create accessibility barriers. Relying on visual cues alone is not sufficient to create an accessible table. With structural markup, headers and data cells can be programmatically determined by software, which means that:

- **People using screen readers** can have the row and column headers read aloud as they navigate through the table. Screen readers speak one cell at a time and reference the associated header cells, so the reader doesn't lose context.
- **Some people use alternative ways to render the data**, for example by using custom stylesheets to display header cells more prominently. Techniques like this enable them to change text size and colors and display the information as lists rather than grids. The table code needs to be properly structured to allow alternative renderings.

* Related WCAG resources

These tutorials provide best-practice guidance on implementing accessibility in different situations. This page combined the following WCAG success criteria and techniques from different conformance levels:

Success Criteria:

- **1.3.1 Info and Relationships:** (<https://www.w3.org/WAI/WCAG21/quickref/#qr-content-structure-separation-programmatic>) Information, structure, and relationships conveyed through presentation can be programmatically determined or are available in text. (Level A)

(<https://www.w3.org/WAI/tutorials/tables/one-header/>)

Next:
One Header ➔

Status: Updated 16 February 2023

Editors: [Eric Eggert](https://www.w3.org/People/yatil/) (<https://www.w3.org/People/yatil/>) and [Shadi Abou-Zahra](https://www.w3.org/People/shadi/) (<https://www.w3.org/People/shadi/>). Update Editor: Brian Elton. Contributors: see [Acknowledgements](https://www.w3.org/WAI/tutorials/acknowledgements/) (<https://www.w3.org/WAI/tutorials/acknowledgements/>).

Developed by the Education and Outreach Working Group ([EOWG](https://www.w3.org/groups/wg/eowg) (<https://www.w3.org/groups/wg/eowg>)). Developed with support from the [WAI-ACT project](https://www.w3.org/WAI/ACT/) (<https://www.w3.org/WAI/ACT/>), co-funded by the [European Commission IST Programme](#).

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities.

Copyright © 2025 World Wide Web Consortium ([W3C®](#)).
See [Permission to Use WAI Material](#).



[Home](#) / [Teach & Advocate](#) / [Business Case](#)

The Business Case for Digital Accessibility



Summary

This article examines the rationale for organizations to address accessibility. It includes tangible and intangible benefits, and the risks of not addressing accessibility adequately. It explores how accessibility can:

- **Drive Innovation:** Accessibility features in products and services often solve unanticipated problems.
- **Enhance Your Brand:** Diversity and inclusion efforts so important to business success are accelerated with a clear, well-integrated accessibility commitment.
- **Extend Market Reach:** The global market of people with disabilities is over 1 billion people with a spending power of more than \$6 trillion. Accessibility often improves the online experience for all users.
- **Minimize Legal Risk:** Many countries have laws requiring digital accessibility, and the issue is of increased legal concern.

Note that “web accessibility” and “websites” throughout this article include web and mobile applications and other digital technologies.

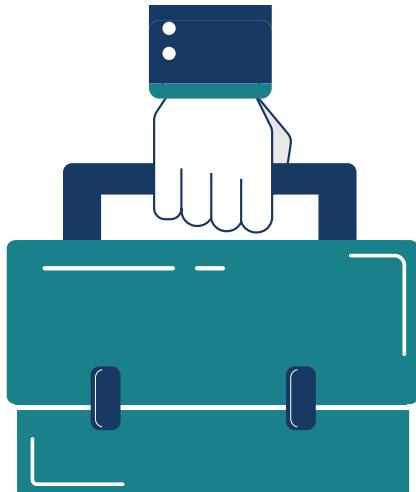
Page Contents

- [Is there a Business Case for Accessibility? \(#is-there-a-business-case-for-accessibility\)](#)
- [Accessibility is good for business \(#accessibility-is-good-for-business\)](#)
 - [Drive Innovation \(#drive-innovation\)](#)

- [Enhance Your Brand](#)([#enhance-your-brand](#))
- [Increase Market Reach](#)([#increase-market-reach](#))
- [Minimize Legal Risk](#)([#minimize-legal-risk](#))
- [Share your experience](#)([#share-your-experience](#))
- [Conclusion](#)([#conclusion](#))
- [Resources](#)([#resources](#))

Is there a Business Case for Accessibility?

A business case is a necessary tool for organizations when planning for various initiatives. Whether an organization is commercial, educational, non-profit, or governmental, most require justification for dedicating resources such as money or effort in support of a specific organizational policy or goal. “Business” in this article refers to all types of organizations with the understanding that different aspects will be relevant depending on the organizational focus and purpose. For example, government agencies may be strongly motivated by legal and equity aspects. Commercial businesses may be more persuaded by innovation and market expansion opportunities. Educational and nonprofit businesses may be especially drawn to brand enhancement.



To create a compelling business case, it is important to highlight the most relevant accessibility benefits within your specific operational landscape. There are both tangible and intangible benefits to businesses that dedicate resources to digital accessibility. A frequent argument against the accessibility business case is that the direct return on investment (ROI) is too difficult to measure. ROI is important of course, but not by any means the only way to measure how an accessibility commitment benefits organizations of all kinds. A useful business case also presents the cost and risk of inaction. It is most likely your business will respond to a mix of motivating factors as you consider implementing an integrated accessibility program. This article provides research and examples to inspire confidence among leaders and decision makers that continued investment in accessibility is good for your business.

Businesses that integrate accessibility are more likely to be innovative, inclusive

enterprises that reach more people with positive brand messaging that meets emerging global legal requirements.

A research study of Fortune 100 companies indicates that disability inclusion, as part of an overall diversity strategy, is common practice among high performing businesses.^{1(✉ #fn:1)}

When accessibility is part of strategic planning, businesses are better equipped for success in our connected world of commerce, academia, and civic engagement.

Let's look at some examples and research outcomes that will help you make the business case that is most effective in your business environment.

Accessibility is good for business



Many organisations are waking up to the fact that embracing accessibility leads to multiple benefits – reducing legal risks, strengthening brand presence, improving customer experience and colleague productivity.”

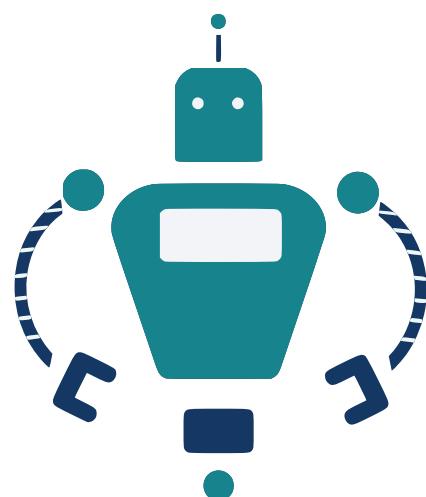
— Paul Smyth, Head of Digital Accessibility, Barclays

The sections below explore key advantages of web accessibility to businesses of all types. Real world examples are presented to show how benefits are realized in the global marketplace.

Drive Innovation

Integrating accessibility removes architectural, digital, and social barriers that can get in the way of innovation^{2(✉ #fn:2)}, for example:

- Accessible design thinking provides varied and flexible ways for users to interact with websites and applications, options that are useful for people with and without disabilities.
- Design of user interaction considers experiences other than screens when accessibility is a



consideration. The result is interaction that is more human-centered, natural, and contextual.

- Accessibility is closely related to general usability – both aim to define and deliver a more intuitive user experience.
- Innovations like the typewriter, telephone, punch cards, text to speech, email, and voice controls were initially meant to include those with a disability, and all have found a much broader application.^{3(↳ #fn:4)}
- Driverless cars, so promising for the independence of blind people, are projected to also help solve traffic fatalities and congestion.
- Research and development of the artificial retina project to help restore sight for participants who are blind may also help future robots with real-time image-processing systems, effectively enabling them to “see.” ^{4(↳ #fn:3)}

Accessible design is by its nature flexible, allowing content to faithfully render across a broad spectrum of devices, platforms, assistive technologies, and operating systems. In physical environments, everyone takes advantage of lower curbs, automatic door openers, ramps, and other features provided for disability access. On the web, accessibility features become options that are also often used more widely.

A compelling example comes from the early 2000's, when people increasingly used mobile devices to browse the web. Accessible and standards-compliant websites were in many cases more mobile-ready as they did not rely on mouse input. Imagine the delight of those who were already committed to and had designed for accessibility! This revelation led to the responsive-design trend that has accessibility at its core.^{5(↳ #fn:5)}

The following case studies from two large technology companies provide useful examples for companies of all sizes.

Case Study: Apple

Apple engineers have been innovators in the accessibility space since the inception of the company, both willing to listen and work with outside constituents. ^{6(↳ #fn:6)} The company also anticipates market direction by integrating disability



Accessibility is a core value at Apple and something we view as a basic human right.

— Sarah Herrlinger, Director of Global Accessibility Policy and

needs into product development.

Initiatives, Apple

Examples include:

- **iTunes U:** In the early 2000's, the 23-campus California State University system was unable to take advantage of Apple's iTunes U educational program because the application was not fully accessible to blind students. Teachers within the CSU system were prohibited from using it. This was resolved by innovation, not litigation, as Apple listened to CSU's concerns and worked to make iTunes accessible on both the Mac and Windows platforms. CSU System was soon able to use the program widely.
- **VoiceOver on iPhone:** Early in the evolution of the iPhone, Apple began considering the implications that a touchscreen device would have on their blind customers. Iterating over several years behind the scenes, the company invested the resources to develop the voice technology that led to VoiceOver, the world's first gesture-based screen reader. Within weeks of launch, Apple received a special commendation from the National Federation of the Blind "For designing the first fully accessible touchscreen interface." ^{7(§#fn:7)}

Screen readers on other touch screen devices have now become more prevalent in the industry and Apple has shown important leadership in the effort to ensure inclusion for all.

Innovation from voice interactions at Apple and elsewhere have contributed to the abundance of personal digital assistants now found in many homes and offices. Today millions of people use these devices, regardless of ability.

Case Study: Google

A 2016 article in the FastCompany online magazine^{8(§#fn:8)} highlights how Google's investment in accessibility provides the company with an innovation edge in a broad array of products and services. Eve Andersson, the lead engineer, featured in the article, says "I'm passionate about accessibility, not just because I believe in a level playing field, but because (it) makes



The accessibility problems of today are the mainstream breakthroughs of tomorrow.

— Eve Andersson, Director, Accessibility Engineering, Google

life more livable for everyone." Among the innovations cited as examples are these:

- contrast minimums, required for people with low vision, help all people see in bright light glare situations
- auto-complete, initially provided for people with disabilities, is now widely used by all
- voice control, implemented for users with physical impairments, has been more widely adopted as a great convenience by millions of others
- artificial intelligence advances are based on research originally done to provide visual context to users with visual impairments
- auto-captioning using machine learning has been problematic for the main target population of deaf users and many feel it is still inadequate to meet that need. However, work continues and machine learning itself is steadily improving and has found broader applications due to this effort.

Enhance Your Brand

Businesses need to protect and enhance their brands. A clear commitment to accessibility can demonstrate that a business has a genuine sense of Corporate Social Responsibility (CSR). As businesses understand and act on the diverse needs of their stakeholders and make the commitment to sustainable, inclusive marketing and employment practices, they can achieve a range of benefits.

Potential outcomes for CSR programs include enhanced brand image and reputation, increased sales and customer loyalty, improved workforce diversity and many other benefits.⁹ Further studies¹⁰ emphasize the benefits to the brand of companies that institute policies of broad diversity.



Employing people with disabilities is an essential aspect of creating a diverse workforce. To be successful, the technology that employees use, including websites and applications, must be accessible.

Barclays shared their approach to diversity and inclusion through the accessibility initiatives in their company and how that approach has made them a better company.

Case Study: Barclays

Establishing an organisation-wide accessibility strategy for identifying, anticipating and addressing the additional needs of customers and colleagues with impairments contributes in several ways to our brand identity – through tailored services, fostering an inclusive culture, creating new ways to communicate and consult with existing and potential customers.

We want to leverage inclusive technology to enable and empower all people to bank, work and reach their full potential. The Barclays Accessibility team does this by supporting digital teams to embed accessibility into our services and culture through effective governance, partnering, training, and tools. Establishing an enterprise-wide accessibility strategy, standards and programmes coupled with senior sponsorship helps support our publicly stated ambition of becoming the most accessible and inclusive Financial Times Stock Exchange (FTSE) company.



At Barclays, accessibility is about more than just disability. It's about helping everyone to work, bank and live their lives regardless of their age, situation, abilities or circumstances.

— Paul Smyth, Head of Digital Accessibility, Barclays

When we shift our thinking away from the minimum legal compliance to focus instead on the commercial opportunity and the creative challenge of building better experiences for everyone, we create a more sustainable, customer orientated approach to digital information and services.

To help everyone understand Barclays accessibility-focused mindset, we've created a range of animations which help our colleagues understand what accessibility is all about, who benefits, and what the different types of impairments are. We've also shared these animations on our [Accessible Banking YouTube playlist](https://www.youtube.com/playlist?list=PLecqH2uhOR0Zb31X7hh5BzWJv4KGLnuUy) (https://www.youtube.com/playlist?list=PLecqH2uhOR0Zb31X7hh5BzWJv4KGLnuUy).

Barclays demonstrates how a strong commitment to accessibility results in distributed responsibility and shared understanding. Accessibility awareness permeates the company culture. The company is perceived as open and fair. People are proud to work there and to do business with Barclays. Read the [full Barclays case study](https://www.w3.org/community/wai-engage/wiki/Barclays_Bank_Case_Study) (https://www.w3.org/community/wai-engage/wiki/Barclays_Bank_Case_Study) to learn more.

Another well-recognized example of how a brand can be affected is Microsoft. After long advocacy and some criticism by accessibility advocates [11\(§ #fn:11\)](#), Microsoft made a real and appreciable commitment to accessibility demonstrated by genuine engagement with stakeholders of all abilities.[12\(§ #fn:12\)](#) The resulting improvements to products and services have strengthened its overall brand image [13\(§ #fn:13\)](#), and accessibility efforts are now lauded in the community of people with disabilities.[14\(§ #fn:14\)](#)

Increase Market Reach

The market of people with disabilities is large and growing as the global population ages. In the UK, where the large disability market is known as the Purple Pound, people with disabilities and their families spend at least £249 billion every year. In the US, the annual discretionary spending of people with disabilities is over \$200 billion. The global estimate of the disability market is nearly \$7 trillion.

Consider these facts when estimating market size:



- At least one billion people – 15% of the world's population – have a recognized disability[15\(§ #fn:15\)](#)
- As the population ages, many more acquire disability and yet do not identify as a "person with a disability"[16\(§ #fn:16\)](#)
- In countries with life expectancies of over 70 years of age, people spend 11.5 percent of their lifespan living with a disability.[17\(§ #fn:17\)](#)
- Globally, the extended market is estimated at 2.3 billion people who control an incremental \$6.9 trillion in annual disposable income.[18\(§ #fn:18\)](#)

A Forrester Research Economic Impact Study commissioned by Microsoft concluded that accessibility could contribute to cost savings when it is integrated into existing and ongoing development cycles. [19\(§ #fn:19\)](#) Technology updates and redesigns that include accessibility along with other best practices have demonstrated reduced costs for maintenance and service. Moreover, according to Microsoft, as accessibility features are included, overall customer satisfaction improves.



Designing inclusive software results in improved usability and customer satisfaction.

— Microsoft's app developer guide

Accessible design considerations often lead to improvements in general customer experience and loyalty. For customers with disabilities, such improvements are essential for equal access. However, accessibility provides options that are useful to all customers in various situations. For example, web accessibility also benefits:

- people using mobile phones, smart watches, smart TVs, and other devices with small screens, different input modes, etc.,
- older people with changing abilities due to ageing,
- people with “temporary disabilities” such as a broken arm or lost glasses,
- challenging situations like bright, glaring sunlight or noisy environments where audio can't be heard,
- those with a slow internet connection, limited or expensive bandwidth, which is common in rural areas and some geographical regions.

Case Study: NPR Weekly Broadcast

This American Life is a broadcast heard on more than 500 National Public Radio (NPR) stations by about 2.1 million listeners each week in the United States. In 2011, in response to new regulations around broadcast media from the US Federal Communications Commission (FCC), the broadcaster committed to creating transcripts for their entire archive of recorded programs. A study by their media partner, conducted over several months, concluded that the provision of transcripts not only met legal obligations but returned significant benefits including:

- search traffic increased 6.86%,
- better comprehension for visitors who use English as a second language,
- visitors were able to use transcripts in noisy or sound-sensitive environments,
- ability to more easily translate, and
- ability to search text to reference a specific section of audio.

The study, conducted over more than one year, used Google Analytics to capture the following data:

- 7.23% of visitors viewed at least one transcript,
- unique visitors increased 4.18%, and
- new inbound links to transcript accounted for an increase of 3.89%

Read more detail and the full report of the [This American Life Case Study](https://www.3playmedia.com/customers/case-studies/this-american-life/) (https://www.3playmedia.com/customers/case-studies/this-american-life/) and how it was conducted.

Minimize Legal Risk



Consideration of the cost and risk of inaction is a critical aspect of any business case. As web use is woven into modern life all over the world, governments and regulators began to mandate laws and policies that strengthen the rights of people with disabilities to participate in online digital information and services.

One of the earliest examples of legal consequences to web accessibility was a complaint put to the Human Rights and Equal Opportunities Commission (HREOC) in 2000 about the inaccessibility of the website of the Sydney Olympics. The plaintiff, who was blind, claimed the site was a violation of the Commonwealth Disability Discrimination Act 1992.²¹ The 2001 decision in the Sydney Olympics suit²² raised awareness of the need to address accessibility in the emerging practice of web communication.

Over time, the legal risk increased with the adoption of more specific laws and policies in countries throughout the world.

- The Convention on the Rights of People with Disabilities (CRPD) is a comprehensive human rights document that includes a direct reference to the rights of all people to have equal access to communications technology. Passed by the General Assembly of the United Nations, more than 175 countries ratified it by 2018.
- The European Union adopted the European Accessibility Act, requiring ATMs and banking services, PCs, telephones and TV equipment, telephony and audiovisual services, transport, e-books, and e-commerce meet accessibility requirements.

A significant demonstration of the risk of ignoring accessibility requirements was the 2008 settlement by the National Federation of the Blind with Target retailers:²⁰

- class damages of \$6 million
- plaintiff legal fees over \$3 million
- undisclosed defense legal fees
- court oversight of website for several years

- In the US, the number of legal actions continues to rise and courts increasingly decide in favor of equal access^{23(§ #fn:23)}, often citing the Americans with Disabilities Act (ADA). Structured Negotiation is another way that legal pressure is effective, encouraging companies to meet accessibility requirements while avoiding litigation.^{24(§ #fn:24)}

In Norway where it is now illegal for commercial websites to fail to provide equivalent access for people with disabilities, the government fines commercial companies that do not comply.^{25(§ #fn:25)} Austria has had customer protection regulation in place since 2006 requiring most public websites to meet accessibility standards. Customers that believe they have been discriminated against can take legal action. Parties meet in mediation before they are permitted to go to court. In the United States, by contrast, the regulation is less clear but legal action continues to accelerate.^{26(§ #fn:26)}

Between government oversight and regulation on the one hand, and increased legal action on the other, the legal landscape is rapidly changing in favor of equal access.

With legal risks increasing, smart businesses – particularly those with global activities – are creating accessibility policies and programs to mitigate risk to protect both their assets and their reputations.

Case Study: ADA and Website Accessibility

The blind plaintiff shopped at the local Winn-Dixie grocery store and pharmacy in person but was unable to access the website for information such as the store locator, coupons, store events, and specials. The historic suit was the first trial in the history of the ADA about the accessibility of a public accommodation's website.

Winn-Dixie asked the court to dismiss the suit based on their argument that a website is not a public accommodation covered by Title III of the ADA. The court decided otherwise, allowing the plaintiff to recover their attorney fees. The decision included the following conclusions by the court:

- The link between the website and Winn-Dixie Stores was a circumstance that made Title III of the ADA, applicable to "Public Accommodations," relevant in this situation.
- The website must be made accessible to "*individuals with disabilities who use computers, laptops, tablets, and smartphones.*"

The court required the chain of grocery stores to

- adopt and post an explicit Accessibility Policy "*to ensure the persons with disabilities*

have full and equal enjoyment of its website and shall accompany the public policy statement with an accessible means of submitting accessibility questions and problems."

- conduct annual accessibility training for IT and web staff, so they learn to create and maintain content that meets WCAG criteria.
- make sure that any third-party applications or content posted to the Winn-Dixie site must also meet WCAG requirements.

Share your experience

User experience research and case studies confirm the many ways that accessible design supports an organization's ability to innovate, enhance their brand, increase market reach, and minimize legal risk – among many other benefits. The WAI is a global community of practice, and we encourage you to share your examples. If you have a story of how your commitment to accessibility improved your online business model, please submit it via email to wai-eo-editors@w3.org or post it to the [WAI-Engage wiki](https://www.w3.org/community/wai-engage/wiki/Case_studies) (https://www.w3.org/community/wai-engage/wiki/Case_studies).

Conclusion

Public use of the web is more than 25 years old. It is no longer a novelty but an integrated, critical tool of modern life. As smart businesses integrate accessible design into their development and procurement processes, they understand the need for equal access by all people. The legal risks of ignoring accessibility are significant, and the benefits have also been demonstrated by leaders like Apple, Barclays, NPR, IBM, Microsoft and hundreds more. Business leaders and the advocates who influence them can have tremendous social impact and a healthy return on investment as they follow a roadmap that leads to equal access. More than one billion people with disabilities in the world are eager to engage with you as customers, clients, partners, employees, and equal participants in civic and social activities. By developing a long-term commitment to accessibility and by using WAI resources to develop policy and implement a strategy to bring that commitment to life, your business will reach this market and is likely to thrive in unexpected and self-sustaining ways.

Resources

This article was written after reading and exploring many external resources that shaped our understanding of the current landscape. We have provided links and notes about our research in this [Annotated Bibliography](https://www.w3.org/WAI/business-case/bibliography/) (<https://www.w3.org/WAI/business-case/bibliography/>).

Also, the WAI website has an extensive library of useful support to help companies realize the benefits outlined in this article for integrating accessibility into their development, procurement, and general business practice. Listed below are a few of what we consider especially useful as you start your accessibility program.

- [Developing Organizational Policies on Web Accessibility](https://www.w3.org/WAI/planning/org-policies/) (<https://www.w3.org/WAI/planning/org-policies/>): Start with a clear policy.
- [Planning and Managing Web Accessibility](https://www.w3.org/WAI/planning-and-managing/) (<https://www.w3.org/WAI/planning-and-managing/>): A guide for implementation.
- [Web Accessibility Perspectives Videos: Explore the Impact and Benefits for Everyone](https://www.w3.org/WAI/perspective-videos/) (<https://www.w3.org/WAI/perspective-videos/>): Watch how accessibility impacts users.

These are suggested merely as a way to get started. We hope you will explore throughout the WAI site as you dive deeper into accessibility and begin to realize the related benefits for you and your organization.

Resources

1. [Disability as diversity in fortune 100 companies](https://onlinelibrary.wiley.com/doi/abs/10.1002/bls.629) (<https://onlinelibrary.wiley.com/doi/abs/10.1002/bls.629>). Ball, P., Monaco, G., Schmeling, J., Schatz, H., and Blanck, P.; Law, Health Policy and Disability Center (2005). ↵(§ #fnref:1)
2. Pullin, Graham. *Design Meets Disability*. MIT Press, 2011. ↵(§ #fnref:2)
3. [People with Disabilities Drive Innovation](https://habengirma.com/2017/09/13/people-with-disabilities-drive-innovation/) (<https://habengirma.com/2017/09/13/people-with-disabilities-drive-innovation/>). Germa, H. (2017) ↵(§ #fnref:4)
4. [Professor's Research Helps Restore Sight to the Blind](https://medicalxpress.com/news/2012-01-professor-sight.html) (<https://medicalxpress.com/news/2012-01-professor-sight.html>). Brown, P.; University of Arizona (2012) ↵(§ #fnref:3)
5. [What Does Responsive Design Have To Do With Accessibility?](https://www.levelaccess.com/what-does-responsive-web-design-have-to-do-with-accessibility/) (<https://www.levelaccess.com/what-does-responsive-web-design-have-to-do-with-accessibility/>) Avila, J.; Level Access (2013). ↵(§ #fnref:5)
6. [When it comes to accessibility, Apple continues to lead in awareness and innovation](https://techcrunch.com/2016/05/19/when-it-comes-to-accessibility-apple-continues-to-lead-in-awareness/) (<https://techcrunch.com/2016/05/19/when-it-comes-to-accessibility-apple-continues-to-lead-in-awareness->)

- and-innovation/). Aquino, S. (2016) ↵(§ #fnref:6)
7. *National Federation of the Blind Commands Apple for Including VoiceOver on iPad* (<https://nfb.org/node/1083>). Danielsen, C. NFB (2010) ↵(§ #fnref:7)
8. *How Designing For Disabled People Is Giving Google An Edge* (<https://www.fastcompany.com/3060090/how-designing-for-the-disabled-is-giving-google-an-edge>). Brownlee, J.; Fast Company (2018). ↵(§ #fnref:8)
9. *Corporate Social Responsibility (CSR)* (<https://www.iisd.org/business/issues/sr.aspx>). IISD's Business and Sustainable Development Guide (2013). ↵(§ #fnref:9)
10. *Reaping The Benefits Of Diversity For Modern Business Innovation* (<https://www.forbes.com/sites/ekaterinawalter/2014/01/14/reaping-the-benefits-of-diversity-for-modern-business-innovation/#155c39652a8f>). Walter, E.; Forbes Magazine (2014). ↵(§ #fnref:10)
11. *Accessibility at Microsoft* (<https://nfb.org/images/nfb/publications/bm/bm15/bm1504/bm150403.htm>). Chong, C.; Braille Monitor (2015). ↵(§ #fnref:11)
12. *The Moment That Forever Changed Our Lives* (<https://blogs.msdn.microsoft.com/accessibility/2017/10/21/satya-nadella-the-moment-that-forever-changed-our-lives/>). Nadella, S.; PowerShell Team Blog, LinkedIn, (2017). ↵(§ #fnref:12)
13. *How Fathering a Son with Disabilities Helped Microsoft's CEO Transform the Company* (<https://forums.windowscentral.com/windows-central-news-discussion/465496-how-fathering-son-disabilities-helped-microsofts-ceo-transform-company.html>). Ward, J.; Windows Central (2017). ↵(§ #fnref:13)
14. *Microsoft Adding New Accessibility Improvements in Windows 10* (<https://coolblindtech.com/microsoft-adding-new-accessibility-improvements-in-windows-10/>). Rego, N.; Cool Blind Tech (2018). ↵(§ #fnref:14)
15. *Disability Inclusion Overview* (<http://www.worldbank.org/en/topic/disability>). The World Bank (2018). ↵(§ #fnref:15)
16. *The Business Case for Accessibility and Inclusive Design* (<https://digileaders.com/the-business-case-for-accessibility-and-inclusive-design/>). Walker, M.; Digital Leaders (2018). ↵(§ #fnref:16)
17. *World Population Chart for Countries and Continents* (<https://www.disabled-world.com/calculators-charts/wpc.php>). Disabled World (2017). ↵(§ #fnref:17)
18. *Richard Branson Supports People With Disabilities – Here Are Six Ways You Can Do It, Too* (<https://www.forbes.com/sites/gaudianohunt/2016/10/31/richard-branson-supports-disabilities/#4da9aa36788e>). Gaudiano, P. and Hunt, E.; Forbes.com (2016). ↵(§ #fnref:18)
19. *Assessing the Value Of Accessible Technologies for Organizations* (<https://web.archive.org/web/20170710171528/https://mscorpmedia.azureedge.net/mscorpmedia/2016/07/Microsoft-TEI-Accessibility-Study Edited FINAL-v2.pdf>). Parks, S., and Sedov V.; Forrester Research, Inc. (2016) ↵(§ #fnref:19)
20. *National Federation of the Blind v. Target Corp.* (<https://en.wikipedia.org/wiki/>

- National Federation of the Blind v. Target Corp.) Wikipedia. (2008) ↵(§ #fnref:20)
21. *Sydney Olympics 2000 Website Accessibility Decision* (<https://www.independentliving.org/docs5/sydney-olympics-blind-accessibility-decision.html>). Independent Living Institute (2000). ↵(§ #fnref:21)
22. *Web Accessibility and the DDA* (https://warwick.ac.uk/fac/soc/law/elj/jilt/2001_2/sloan/). Sloan, M.; Journal of Information Law & Technology, University of Warwick (2005). ↵(§ #fnref:22)
23. *List of Web Accessibility-Related Litigation and Settlements* (<http://www.karlgroves.com/2011/11/15/list-of-web-accessibility-related-litigation-and-settlements/>). Groves, K. (2017). ↵(§ #fnref:23)
24. *Settlements in Structured Negotiation* (<http://www.lflegal.com/negotiations/>). Feingold, L. (2018) ↵(§ #fnref:24)
25. *It's Illegal to Have an Inaccessible Website in Norway - and That's Good News for All of Us* (<https://medium.com/confrere/its-illegal-to-have-an-inaccessible-website-in-norway-and-that-s-good-news-for-all-of-us-b59a9e929d54>). Aalen, I.; Confrere/Medium (2018). ↵(§ #fnref:25)
26. *"Absence of U.S. Regulation Leads to Web Accessibility Lawsuits"* (<https://www.practicalecommerce.com/Absence-of-U-S-Regulation-Leads-to-Web-Accessibility-Lawsuits>). Roggio, A.; Practical Ecommerce (2015). ↵(§ #fnref:26)

Updated: 15 July 2024.

Published 9 Nov 2018.

Editor: Sharron Rush. Contributors: Shawn Lawton Henry, Eric Eggert, Brent Bakken, Vicki Menezes Miller, Laura Keen. Acknowledgements (<https://www.w3.org/WAI/business-case/acknowledgements/>) lists additional contributors.

Developed by the Education and Outreach Working Group ([EOWG \(<https://www.w3.org/WAI/EO/>\)](https://www.w3.org/WAI/EO/)).

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities. Copyright © 2025 World Wide Web Consortium ([W3C®](#)). See [Permission to Use WAI Material](#).



Transcripts

in [Making Audio and Video Media Accessible](#) (<https://www.w3.org/WAI/media/av/>)



Summary

Basic transcripts are a text version of the speech and non-speech audio information needed to understand the content. Descriptive transcripts also include text description of the visual information needed to understand the content. Descriptive transcripts are required to provide video content to people who are both Deaf and blind.

This page helps you understand and create transcripts.

Page Contents

- [Introduction](#)(<#introduction>)
- [Does My Media Need a Transcript?](#)(<#checklist>)
- [Process – Skills and Tools](#)(<#process-skills-and-tools>)
 - [If You Start with Captions](#)(<#if-you-start-with-captions>)
 - [If You Start with Transcribing](#)(<#if-you-start-with-transcribing>)
- [Creating Transcripts](#)(<#creating-transcripts>)
 - [Transcript File Format](#)(<#transcript-file-format>)
 - [Making Transcripts More Useful](#)(<#making-transcripts-more-useful>)
- [Where to Put Transcripts](#)(<#where-to-put-transcripts>)
- [Example Descriptive Transcript from Caption Files](#)(<#example-descriptive-transcript-from-caption-files>)

Introduction

Basic transcripts are a text version of the speech and non-speech audio information needed

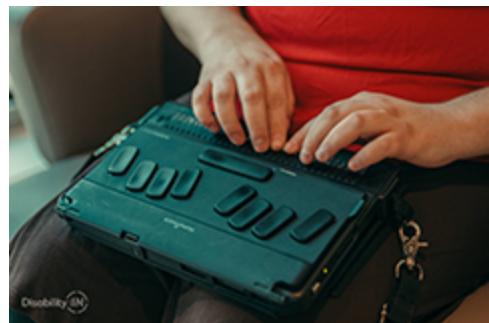
to understand the content.

Who: Basic transcripts are used by people who are Deaf, are hard of hearing, have difficulty processing auditory information, and others.

Descriptive transcripts for videos also include visual information needed to understand the content.

Who: Descriptive transcripts are needed to provide audio and video content to people who are both Deaf and blind. They are also used by people who process text information better than audio and visual/pictorial information.

Ideally you provide a descriptive transcript, and then you do not need a separate basic transcript.



Interactive transcripts highlight text phrases as they are spoken. Users can select text in the transcript and go to that point in the video. This is a feature of the media player. It uses the captions file.

The image shows a split-screen view of a video player interface. On the left, a video frame displays a man's face in profile, wearing a white hearing aid in his ear. Below the video frame, a subtitle reads: "Or someone who cannot hear well, and uses captions to watch videos." On the right, a transcript window is open, showing a speech-to-text transcription of the video content. The transcript includes several highlighted text segments in yellow, such as "Hi! My name is Shadi Abou-Zahra. I'm the Accessibility Strategy and Technology Specialist at W3C, the World Wide Web Consortium, and today I'd like to tell you about web accessibility. The Web is for many people an essential part of daily life. At work. At home. And on the road. Web accessibility means that people with disabilities can use the Web equally. For example, somebody who cannot use their arms, and uses a mouthstick to type. Or someone who cannot hear well, and uses captions to watch videos. Or someone who cannot see well, and uses a screen reader to read aloud what's on the screen. Accessibility has many benefits. For example, captions benefit anyone in a loud or in a quiet environment. And good color contrast works better when there is glare. Also people with age-related impairments, such as reduced dexterity, benefit. In fact, everyone has a better user experience with an improved layout and design. A lot of accessibility can be built into the underlying". The transcript window also includes settings for "Auto scroll" and "Language English".

Does My Media Need a Transcript?

Short answer: Yes, descriptive transcripts are needed to meet the wide range of user needs.

In some cases, transcripts are not required to meet WCAG standards. (*The Planning page of this resource introduces the WCAG Standard (<https://www.w3.org/WAI/media/av/planning/#wcag-standard>).*)

Audio-only (e.g., podcast):

- For pre-recorded:
 - Transcripts are **required** at WCAG Level A.
- For live:
 - Transcripts are at WCAG Level AAA. Usually this needs to be a live text stream. If the audio follows a script, you can provide the text script.

Video-only (no audio content):

- For pre-recorded:
 - Descriptive transcript **or** audio description is **required** at WCAG Level A.
- For live:
 - A description of the visual information is needed. Usually it is a live text stream, rather than a transcript. It is not required in WCAG.

Video with audio content:

- For pre-recorded:
 - Transcripts are at WCAG Level AAA. (Captions are A.)
- For live:
 - A live stream separate from the media player is needed for people who cannot access the captions. It is not required in WCAG. (Captions are AA.)

WCAG excerpts with emphasis added, additions in [brackets], and links to more information in “Understanding WCAG”:

- A 1.2.1 Audio-only and Video-only (<https://www.w3.org/WAI/WCAG22/Understanding/audio-only-and-video-only-prerecorded.html>) (Prerecorded): For prerecorded audio-only and prerecorded video-only media, the following are true...
 - Prerecorded Audio-only: An alternative for time-based media [transcript] is provided that presents equivalent information for prerecorded audio-only

content.

- Prerecorded Video-only: Either an alternative for time-based media [descriptive transcript] **or** an audio track [of description] is provided that presents equivalent information for prerecorded video-only content.
- [AAA 1.2.8 Media Alternative](https://www.w3.org/WAI/WCAG22/Understanding/media-alternative-prerecorded.html) (<https://www.w3.org/WAI/WCAG22/Understanding/media-alternative-prerecorded.html>) (Prerecorded): An alternative for time-based media [transcript] is provided for all prerecorded synchronized media and for all prerecorded video-only media.
- [AAA 1.2.9 Audio-only](https://www.w3.org/WAI/WCAG22/Understanding/audio-only-live.html) (<https://www.w3.org/WAI/WCAG22/Understanding/audio-only-live.html>) (Live): An alternative for time-based media [live stream text or transcript] that presents equivalent information for live audio-only content is provided.

Provide a Descriptive Transcript for Your Videos

Descriptive transcripts are needed by people who are Deaf-blind and others. (A bit more justification is in the Planning page: [Descriptive Transcripts](https://www.w3.org/WAI/media/av/planning/#descriptive-transcripts) (<https://www.w3.org/WAI/media/av/planning/#descriptive-transcripts>)). And **descriptive transcripts are easy and inexpensive to make** using captions and audio description that you already have to meet Level AA, as explained on this page.

Process – Skills and Tools

The process for providing transcripts is basically:

1. Get a text version of the audio, called “transcribing”.
2. Format the transcript.
3. Put the transcript online, and make it easy for users to find the transcript from the audio or video file.

If You Start with Captions

For videos, often transcribing the audio to text is done to create captions, as described in the [captions page](https://www.w3.org/WAI/media/av/captions/) (<https://www.w3.org/WAI/media/av/captions/>). Then the captions file is used to create the transcript.

Creating transcripts from caption files is easy with basic web skills and tools.

If You Start with Transcribing

Transcribing an audio file takes quite a bit of time for people who don't have the software and skill for it. Many organizations choose to outsource the transcribing. Guidance for doing it yourself (DIY) is in another page of this resource: [Transcribing Audio to Text](https://www.w3.org/WAI/media/av/transcribing/) (<https://www.w3.org/WAI/media/av/transcribing/>).

Once you have the transcription, creating the transcript is easy with basic web skills and tools.

Creating Transcripts

If you already have captions, you can use that file to create the transcript. Most caption-editing tools provide an option to export a plain text transcript. Otherwise, you will need to delete the timestamps, or edit them per below.

If you don't have captions, you'll need transcription of the audio information. That's addressed in another page of this resource: [Transcribing Audio to Text](https://www.w3.org/WAI/media/av/transcribing/) (<https://www.w3.org/WAI/media/av/transcribing/>).

Captions are generally written to be viewed along with the visual video. Transcripts should include important visual information for those not seeing the video. When you use captions to create transcripts, **usually you will need to add visual information to the transcript**, such as text that is in the video and speaker identification.

Transcript File Format

Most transcripts on the web are provided in HTML. There is not a set design for transcripts. Different options and examples are described throughout the guidance below.

A transcript of a podcast can be simple text paragraphs with the speakers identified.

Example transcript of a podcast interview with two speakers (excerpt)

Rajwinder Kaur: Welcome to the podcast.

Shawn Henry: Thank you for this opportunity to share information about accessibility.

Rajwinder: Would you start by telling us a little about your role at W3C?

Shawn: I work within the Web Accessibility Initiative, W-A-I, pronounced "way".

A descriptive transcript can be in a table so that readers can easily read only the audio information down a column if they choose. A [descriptive transcript example is below](#)([#descriptive](#)).

Making Transcripts More Useful

Keep in mind that the main purpose of a transcript is to provide the information to people who cannot get it from the audio and/or video. That will help you know what to include and how to design it. Add information to make the transcript more useful. For example, add headings, links, a summary, and maybe time stamps, as described below. The following are optional, not requirements.

- **Put the information in logical paragraphs, lists, and sections.** If you're starting with a captions file, you will probably combine several lines into paragraphs. For example, in the [example excerpts below](#)([#example-descriptive-transcript-from-caption-files](#)), 6 lines of captions are grouped into 2 paragraphs of text (in table cells).
- **Add navigation and clarifications:**
 - Add headings and links where it will make the transcript more usable. (This also helps with SEO, search engine optimization.) Here's an [example with added links in short podcast transcript](#) (<https://www.w3.org/WAI/highlights/200606wcag2interview.html>).
 - It is generally acceptable to add clarifying information, *as long as it is clear that it is not part of the actual audio* — for example, words added to a paragraph put in [brackets], or separate sections with headings "Introduction", "Transcript", "Resources". Here's an [example with added headings in long presentation transcript](#) (<https://www.w3.org/WAI/highlights/200706wcag2pres>).
- **Indicate the speakers based on the type of content.** For example:
 - When there are multiple speakers, you could use hanging indents to make it easy to skim for a particular speaker.
 - When you want the focus on the interviewee's answers and not the

interviewer, you could bold the interviewee's name so it stands out more clearly.

- **Include timestamps only when useful.** In many cases, including timestamps would be unnecessary clutter. If you do include them, they usually don't need to be as granular as the captions, and do not need to include end times.
- **If starting with captions for video:** The video might have text information that was not included in the captions, for example, the title of the video or the name and title of people speaking. If you also have the description of visual information, it should already be in there. If not, you'll need to review the video and see if there is text visually that wasn't repeated in the captions, and add that to your transcript.

Where to Put Transcripts

Make it is easy for users to know that a transcript is available and to get to the transcript. For example, put the transcript itself or a link to the transcript right under the video.

For media on your website, usually it's best to include the transcript on the same page. Here's an example descriptive transcript at the bottom of same page with a video (<https://www.w3.org/WAI/perspective-videos/captions/#transcript>).

When your media is hosted elsewhere, you might have the transcript on a separate web page. Here's an example podcast transcript on separate page (<https://www.w3.org/WAI/highlights/200606wcag2interview.html>).

Example Descriptive Transcript from Caption Files

← Previous:
Captions/Subtitles

(<https://www.w3.org/WAI/media/av/captions/>)

(<https://www.w3.org/WAI/media/av/transcribing/>) Next:
Transcribing Audio to Text →

First published September 2019.

Editor: [Shawn Lawton Henry](https://www.w3.org/People/Shawn) (<https://www.w3.org/People/Shawn>). [Acknowledgements](https://www.w3.org/WAI/media/av/acknowledgements/) (<https://www.w3.org/WAI/media/av/acknowledgements/>) lists contributors and credits.

Developed by the Education and Outreach Working Group ([EOWG](https://www.w3.org/WAI/EO/) (<https://www.w3.org/WAI/EO/>)). Originally drafted as part of the [WCAG TA Project](https://www.w3.org/WAI/WCAGTA/) (<https://www.w3.org/WAI/WCAGTA/>) funded by the U.S. Access Board. Revised as part of the [WAI Expanding Access project](https://www.w3.org/WAI/expand-access/) (<https://www.w3.org/WAI/expand-access/>) funded by the Ford Foundation.

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities.

Copyright © 2025 World Wide Web Consortium ([W3C®](#)).

See [Permission to Use WAI Material](#).

Web Content Accessibility Guidelines (WCAG) 2.2

W3C Recommendation 12 December 2024



▼ More details about this document

This version:

<https://www.w3.org/TR/2024/REC-WCAG22-20241212/>

Latest published version:

<https://www.w3.org/TR/WCAG22/>

Latest editor's draft:

<https://w3c.github.io/wcag/guidelines/22/>

History:

<https://www.w3.org/standards/history/WCAG22/>

[Commit history](#)

Implementation report:

<https://www.w3.org/WAI/WCAG22/implementation-report/>

Previous Recommendation:

<https://www.w3.org/TR/WCAG21/>

Editors:

[Alastair Campbell](#) (Nomensa)

[Chuck Adams](#) (Oracle)

[Rachael Bradley Montgomery](#) (Library of Congress)

[Michael Cooper](#) (W3C)

Andrew Kirkpatrick (Adobe)

Feedback:

[GitHub w3c/wcag](#) ([pull requests](#), [new issue](#), [open issues](#))

Errata:

[Errata exists.](#)

See also [translations](#).

Copyright © 2020-2024 [World Wide Web Consortium](#). W3C® [liability](#), [trademark](#) and [document use](#) rules apply.

Abstract

Web Content Accessibility Guidelines (WCAG) 2.2 covers a wide range of recommendations for making web content more accessible. Following these guidelines will make content more accessible to a wider range of people with disabilities, including accommodations for blindness and low vision, deafness and hearing loss, limited movement, speech disabilities, photosensitivity, and combinations of these, and some accommodation for learning disabilities and cognitive limitations; but will not address every user need for people with these disabilities. These guidelines address accessibility of web content on any kind of device (including desktops, laptops, kiosks, and mobile devices). Following these guidelines will also often make web content more usable to users in general.

WCAG 2.2 success criteria are written as testable statements that are not technology-specific. Guidance about satisfying the success criteria in specific technologies, as well as general information about interpreting the success criteria, is provided in separate documents. See [Web Content Accessibility Guidelines \(WCAG\) Overview](#) for an introduction and links to WCAG technical and educational material.

WCAG 2.2 extends [Web Content Accessibility Guidelines 2.1 \[WCAG21\]](#), which was published as a W3C Recommendation June 2018. Content that conforms to WCAG 2.2 also conforms to WCAG 2.0 and WCAG 2.1. The WG intends that for policies requiring conformance to WCAG 2.0 or WCAG 2.1, WCAG 2.2 can provide an alternate means of conformance. The publication of WCAG 2.2 does not deprecate or supersede WCAG 2.0 or WCAG 2.1. While WCAG 2.0 and WCAG 2.1 remain W3C Recommendations, the W3C advises the use of WCAG 2.2 to maximize future applicability of accessibility efforts. The W3C also encourages use of the most current version of WCAG when developing or updating web accessibility policies.

Status of This Document

This section describes the status of this document at the time of its publication. A list of current W3C publications and the latest revision of this technical report can be found in the [W3C technical reports index](#) at <https://www.w3.org/TR/>.

To comment, [file an issue in the W3C WCAG GitHub repository](#). Although the proposed success criteria in this document reference issues tracking discussion, the Working Group requests that public comments be filed as new issues, one issue per discrete comment. It is free to create a GitHub account to file issues. If filing issues in GitHub is not feasible, send email to public-agwg-comments@w3.org ([comment archive](#)).

This document was published by the [Accessibility Guidelines Working Group](#) as a Recommendation

using the [Recommendation track](#).

W3C recommends the wide deployment of this specification as a standard for the Web.

A W3C Recommendation is a specification that, after extensive consensus-building, is endorsed by W3C and its Members, and has commitments from Working Group members to [royalty-free licensing](#) for implementations.

This document was produced by a group operating under the [W3C Patent Policy](#). W3C maintains a [public list of any patent disclosures](#) made in connection with the deliverables of the group; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual believes contains [Essential Claim\(s\)](#) must disclose the information in accordance with [section 6 of the W3C Patent Policy](#).

This document is governed by the [03 November 2023 W3C Process Document](#).

Table of Contents

Abstract

Status of This Document

Introduction

Background on WCAG 2

WCAG 2 Layers of Guidance

WCAG 2.2 Supporting Documents

Requirements for WCAG 2.2

Comparison with WCAG 2.1

 New Features in WCAG 2.2

 Numbering in WCAG 2.2

 Conformance to WCAG 2.2

Later Versions of Accessibility Guidelines

1. Perceivable

1.1 Text Alternatives

 1.1.1 Non-text Content

1.2 Time-based Media

 1.2.1 Audio-only and Video-only (Prerecorded)

 1.2.2 Captions (Prerecorded)

- 1.2.3 Audio Description or Media Alternative (Prerecorded)
- 1.2.4 Captions (Live)
- 1.2.5 Audio Description (Prerecorded)
- 1.2.6 Sign Language (Prerecorded)
- 1.2.7 Extended Audio Description (Prerecorded)
- 1.2.8 Media Alternative (Prerecorded)
- 1.2.9 Audio-only (Live)
- 1.3 Adaptable
 - 1.3.1 Info and Relationships
 - 1.3.2 Meaningful Sequence
 - 1.3.3 Sensory Characteristics
 - 1.3.4 Orientation
 - 1.3.5 Identify Input Purpose
 - 1.3.6 Identify Purpose
- 1.4 Distinguishable
 - 1.4.1 Use of Color
 - 1.4.2 Audio Control
 - 1.4.3 Contrast (Minimum)
 - 1.4.4 Resize Text
 - 1.4.5 Images of Text
 - 1.4.6 Contrast (Enhanced)
 - 1.4.7 Low or No Background Audio
 - 1.4.8 Visual Presentation
 - 1.4.9 Images of Text (No Exception)
 - 1.4.10 Reflow
 - 1.4.11 Non-text Contrast
 - 1.4.12 Text Spacing
 - 1.4.13 Content on Hover or Focus

2. Operable

- 2.1 Keyboard Accessible
 - 2.1.1 Keyboard
 - 2.1.2 No Keyboard Trap
 - 2.1.3 Keyboard (No Exception)
 - 2.1.4 Character Key Shortcuts
- 2.2 Enough Time
 - 2.2.1 Timing Adjustable
 - 2.2.2 Pause, Stop, Hide

- 2.2.3 No Timing
- 2.2.4 Interruptions
- 2.2.5 Re-authenticating
- 2.2.6 Timeouts
- 2.3 Seizures and Physical Reactions
 - 2.3.1 Three Flashes or Below Threshold
 - 2.3.2 Three Flashes
 - 2.3.3 Animation from Interactions
- 2.4 Navigable
 - 2.4.1 Bypass Blocks
 - 2.4.2 Page Titled
 - 2.4.3 Focus Order
 - 2.4.4 Link Purpose (In Context)
 - 2.4.5 Multiple Ways
 - 2.4.6 Headings and Labels
 - 2.4.7 Focus Visible
 - 2.4.8 Location
 - 2.4.9 Link Purpose (Link Only)
 - 2.4.10 Section Headings
 - 2.4.11 Focus Not Obscured (Minimum)
 - 2.4.12 Focus Not Obscured (Enhanced)
 - 2.4.13 Focus Appearance
- 2.5 Input Modalities
 - 2.5.1 Pointer Gestures
 - 2.5.2 Pointer Cancellation
 - 2.5.3 Label in Name
 - 2.5.4 Motion Actuation
 - 2.5.5 Target Size (Enhanced)
 - 2.5.6 Concurrent Input Mechanisms
 - 2.5.7 Dragging Movements
 - 2.5.8 Target Size (Minimum)

3. Understandable

- 3.1 Readable
 - 3.1.1 Language of Page
 - 3.1.2 Language of Parts
 - 3.1.3 Unusual Words
 - 3.1.4 Abbreviations

- 3.1.5 Reading Level
- 3.1.6 Pronunciation
- 3.2 Predictable
 - 3.2.1 On Focus
 - 3.2.2 On Input
 - 3.2.3 Consistent Navigation
 - 3.2.4 Consistent Identification
 - 3.2.5 Change on Request
 - 3.2.6 Consistent Help
- 3.3 Input Assistance
 - 3.3.1 Error Identification
 - 3.3.2 Labels or Instructions
 - 3.3.3 Error Suggestion
 - 3.3.4 Error Prevention (Legal, Financial, Data)
 - 3.3.5 Help
 - 3.3.6 Error Prevention (All)
 - 3.3.7 Redundant Entry
 - 3.3.8 Accessible Authentication (Minimum)
 - 3.3.9 Accessible Authentication (Enhanced)

4. Robust

- 4.1 Compatible
 - 4.1.1 Parsing (Obsolete and removed)
 - 4.1.2 Name, Role, Value
 - 4.1.3 Status Messages

5. Conformance

- 5.1 Interpreting Normative Requirements
- 5.2 Conformance Requirements
 - 5.2.1 Conformance Level
 - 5.2.2 Full pages
 - 5.2.3 Complete processes
 - 5.2.4 Only Accessibility-Supported Ways of Using Technologies
 - 5.2.5 Non-Interference
- 5.3 Conformance Claims (Optional)
 - 5.3.1 Required Components of a Conformance Claim
 - 5.3.2 Optional Components of a Conformance Claim
- 5.4 Statement of Partial Conformance - Third Party Content

- 5.5 Statement of Partial Conformance - Language
- 5.6 Privacy Considerations
- 5.7 Security Considerations

6. Glossary

7. Input Purposes for User Interface Components

A. Change Log

B. Acknowledgments

- B.1 Participants of the AG WG active in the development of this document:
- B.2 Other previously active WCAG WG participants and other contributors to WCAG 2.0, WCAG 2.1, or supporting resources
- B.3 Enabling funders

C. References

- C.1 Informative references

§ Introduction

This section is non-normative.

§ Background on WCAG 2

Web Content Accessibility Guidelines (WCAG) 2.2 defines how to make web content more accessible to people with disabilities. Accessibility involves a wide range of disabilities, including visual, auditory, physical, speech, cognitive, language, learning, and neurological disabilities. Although these guidelines cover a wide range of issues, they are not able to address the needs of people with all types, degrees, and combinations of disability. These guidelines also make web content more usable by older individuals with changing abilities due to aging and often improve usability for users in general.

WCAG 2.2 is developed through the [W3C process](#) in cooperation with individuals and organizations around the world, with a goal of providing a shared standard for web content accessibility that meets the needs of individuals, organizations, and governments internationally. WCAG 2.2 builds on WCAG 2.0 [[WCAG20](#)] and WCAG 2.1 [[WCAG21](#)], which in turn built on WCAG 1.0 [[WAI](#)-

[**WEBCONTENT**] and is designed to apply broadly to different web technologies now and in the future, and to be testable with a combination of automated testing and human evaluation. For an introduction to WCAG, see the [Web Content Accessibility Guidelines \(WCAG\) Overview](#).

Significant challenges were encountered in defining additional criteria to address cognitive, language, and learning disabilities, including a short timeline for development as well as challenges in reaching consensus on testability, implementability, and international considerations of proposals. Work will carry on in this area in future versions of WCAG. We encourage authors to refer to our supplemental guidance on [improving inclusion for people with disabilities, including learning and cognitive disabilities, people with low-vision, and more](#).

Web accessibility depends not only on accessible content but also on accessible web browsers and other user agents. Authoring tools also have an important role in web accessibility. For an overview of how these components of web development and interaction work together, see:

- [Essential Components of Web Accessibility](#)
- [User Agent Accessibility Guidelines \(UAAG\) Overview](#)
- [Authoring Tool Accessibility Guidelines \(ATAG\) Overview](#)

Where this document refers to “WCAG 2” it is intended to mean any and all versions of WCAG that start with 2.

WCAG 2 Layers of Guidance

The individuals and organizations that use WCAG vary widely and include web designers and developers, policy makers, purchasing agents, teachers, and students. In order to meet the varying needs of this audience, several layers of guidance are provided including overall *principles*, general *guidelines*, testable *success criteria* and a rich collection of *sufficient techniques*, *advisory techniques*, and *documented common failures* with examples, resource links and code.

- **Principles** - At the top are four principles that provide the foundation for web accessibility: *perceivable, operable, understandable, and robust*. See also [Understanding the Four Principles of Accessibility](#).
- **Guidelines** - Under the principles are guidelines. The 13 guidelines provide the basic goals that authors should work toward in order to make content more accessible to users with different disabilities. The guidelines are not testable, but provide the framework and overall objectives to help authors understand the success criteria and better implement the techniques.

- **Success Criteria** - For each guideline, testable success criteria are provided to allow WCAG 2.2 to be used where requirements and conformance testing are necessary such as in design specification, purchasing, regulation, and contractual agreements. In order to meet the needs of different groups and different situations, three levels of conformance are defined: A (lowest), AA, and AAA (highest). Additional information on WCAG levels can be found in [Understanding Levels of Conformance](#).
- **Sufficient and Advisory Techniques** - For each of the *guidelines* and *success criteria* in the WCAG 2.2 document itself, the working group has also documented a wide variety of *techniques*. The techniques are informative and fall into two categories: those that are *sufficient* for meeting the success criteria and those that are *advisory*. The advisory techniques go beyond what is required by the individual success criteria and allow authors to better address the guidelines. Some advisory techniques address accessibility barriers that are not covered by the testable success criteria. Where common failures are known, these are also documented. See also [Sufficient and Advisory Techniques in Understanding WCAG 2.2](#).

All of these layers of guidance (principles, guidelines, success criteria, and sufficient and advisory techniques) work together to provide guidance on how to make content more accessible. Authors are encouraged to view and apply all layers that they are able to, including the advisory techniques, in order to best address the needs of the widest possible range of users.

Note that even content that conforms at the highest level (AAA) will not be accessible to individuals with all types, degrees, or combinations of disability, particularly in the cognitive, language, and learning areas. Authors are encouraged to consider the full range of techniques, including the advisory techniques, [Making Content Usable for People with Cognitive and Learning Disabilities](#), as well as to seek relevant advice about current best practice to ensure that web content is accessible, as far as possible, to this community. [Metadata](#) may assist users in finding content most suitable for their needs.

§ WCAG 2.2 Supporting Documents

The WCAG 2.2 document is designed to meet the needs of those who need a stable, referenceable technical standard. Other documents, called supporting documents, are based on the WCAG 2.2 document and address other important purposes, including the ability to be updated to describe how WCAG would be applied with new technologies. Supporting documents include:

1. [**How to Meet WCAG 2.2**](#) - A customizable quick reference to WCAG 2.2 that includes all of the guidelines, success criteria, and techniques for authors to use as they are developing and evaluating web content. This includes content from WCAG 2.0, 2.1, and 2.2, and can be filtered

in many ways to help authors focus on relevant content.

2. **Understanding WCAG 2.2** - A guide to understanding and implementing WCAG 2.2. There is a short "Understanding" document for each guideline and success criterion in WCAG 2.2 as well as key topics.
3. **Techniques for WCAG 2.2** - A collection of techniques and common failures, each in a separate document that includes a description, examples, code and tests.
4. **The WCAG 2 Documents** - A brief introduction to the WCAG 2 supporting documents and supplemental guidance.
5. **What's New in WCAG 2.2** introduces the new success criteria with persona quotes that illustrate the accessibility issues.

See [Web Content Accessibility Guidelines \(WCAG\) Overview](#) for a description of the WCAG 2.2 supporting material, including education resources related to WCAG 2. Additional resources covering topics such as the business case for web accessibility, planning implementation to improve the accessibility of websites, and accessibility policies are listed in [WAI Resources](#).

Requirements for WCAG 2.2

WCAG 2.2 meets a set of [requirements for WCAG 2.2](#) which, in turn, inherit requirements from previous WCAG 2 versions. Requirements structure the overall framework of guidelines and ensure backwards compatibility. The Working Group also used a less formal set of acceptance criteria for success criteria, to help ensure success criteria are similar in style and quality to those in WCAG 2.0. These requirements constrained what could be included in WCAG 2.2. This constraint was important to preserve its nature as a dot-release of WCAG 2.

Comparison with WCAG 2.1

WCAG 2.2 was initiated with the goal to continue the work of WCAG 2.1: Improving accessibility guidance for three major groups: users with cognitive or learning disabilities, users with low vision, and users with disabilities on mobile devices. Many ways to meet these needs were proposed and evaluated, and a set of these were refined by the Working Group. Structural requirements inherited from WCAG 2.0, clarity and impact of proposals, and timeline led to the final set of success criteria

included in this version. The Working Group considers that WCAG 2.2 incrementally advances web content accessibility guidance for all these areas, but underscores that not all user needs are met by these guidelines.

WCAG 2.2 builds on and is backwards compatible with WCAG 2.1, meaning web pages that conform to WCAG 2.2 are at least as accessible as pages that conform to WCAG 2.1. Requirements have been added that build on 2.1 and 2.0. WCAG 2.2 has removed one success criterion, [4.1.1 Parsing](#). Authors that are required by policy to conform with WCAG 2.0 or 2.1 will be able to update content to WCAG 2.2, but may need to continue to test and report 4.1.1. Authors following more than one version of the guidelines should be aware of the following additions.

New Features in WCAG 2.2

WCAG 2.2 extends WCAG 2.1 by adding new success criteria, definitions to support them, and guidelines to organize the additions. This additive approach helps to make it clear that sites which conform to WCAG 2.2 also conform to WCAG 2.1. The Accessibility Guidelines Working Group recommends that sites adopt WCAG 2.2 as their new conformance target, even if formal obligations mention previous versions, to provide improved accessibility and to anticipate future policy changes.

The following success criteria are new in WCAG 2.2:

- 2.4.11 [Focus Not Obscured \(Minimum\)](#) (AA)
- 2.4.12 [Focus Not Obscured \(Enhanced\)](#) (AAA)
- 2.4.13 [Focus Appearance](#) (AAA)
- 2.5.7 [Dragging Movements](#) (AA)
- 2.5.8 [Target Size \(Minimum\)](#) (AA)
- 3.2.6 [Consistent Help](#) (A)
- 3.3.7 [Redundant Entry](#) (A)
- 3.3.8 [Accessible Authentication \(Minimum\)](#) (AA)
- 3.3.9 [Accessible Authentication \(Enhanced\)](#) (AAA)

The new success criteria may reference new terms that have also been added to the glossary and form part of the normative requirements of the success criteria.

WCAG 2.2 also introduces new sections detailing aspects of the specification which may impact

[privacy](#) and [security](#).

§ Numbering in WCAG 2.2

In order to avoid confusion for implementers for whom backwards compatibility to WCAG 2 versions is important, new success criteria in WCAG 2.2 have been appended to the end of the set of success criteria within their guideline. This avoids the need to change the section number of success criteria from WCAG 2, which would be caused by inserting new success criteria between existing success criteria in the guideline, but it means success criteria in each guideline are no longer grouped by conformance level. The order of success criteria within each guideline does not imply information about conformance level; only the conformance level indicator (A / AA / AAA) on the success criterion itself indicates this. The [WCAG 2.2 Quick Reference](#) will provide a way to view success criteria grouped by conformance level, along with many other filter and sort options.

§ Conformance to WCAG 2.2

WCAG 2.2 uses the same conformance model as WCAG 2.0. It is intended that sites that conform to WCAG 2.2 also conform to WCAG 2.0 and WCAG 2.1, which means they meet the requirements of any policies that reference WCAG 2.0 or WCAG 2.1, while also better meeting the needs of users on the current Web.

§ Later Versions of Accessibility Guidelines

In parallel with WCAG 2.2, the Accessibility Guidelines Working Group is developing another major version of accessibility guidelines. The result of this work is expected to be a more substantial restructuring of web accessibility guidance than would be realistic for dot-releases of WCAG 2. The work follows a research-focused, user-centered design methodology to produce the most effective and flexible outcome, including the roles of content authoring, user agent support, and authoring tool support. This is a multi-year effort, so WCAG 2.2 is needed as an interim measure to provide updated web accessibility guidance to reflect changes on the web since the publication of WCAG 2.0. The Working Group might also develop additional interim versions, continuing with WCAG 2.2, on a similar short timeline to provide additional support while the major version is completed.

§ 1. Perceivable

Information and user interface components must be presentable to users in ways they can perceive.

§ Guideline 1.1 Text Alternatives

Provide text alternatives for any non-text content so that it can be changed into other forms people need, such as large print, braille, speech, symbols or simpler language.

[Understanding Text Alternatives](#)

[How to Meet Text Alternatives](#)

§ Success Criterion 1.1.1 Non-text Content

(Level A)

All [non-text content](#) that is presented to the user has a [text alternative](#) that serves the equivalent purpose, except for the situations listed below.

[Understanding Non-text Content](#)

[How to Meet Non-text Content](#)

Controls, Input

If non-text content is a control or accepts user input, then it has a [name](#) that describes its purpose.

(Refer to [Success Criterion 4.1.2](#) for additional requirements for controls and content that accepts user input.)

Time-Based Media

If non-text content is time-based media, then text alternatives at least provide descriptive identification of the non-text content. (Refer to [Guideline 1.2](#) for additional requirements for media.)

Test

If non-text content is a test or exercise that would be invalid if presented in [text](#), then text alternatives at least provide descriptive identification of the non-text content.

Sensory

If non-text content is primarily intended to create a [specific sensory experience](#), then text alternatives at least provide descriptive identification of the non-text content.

CAPTCHA

If the purpose of non-text content is to confirm that content is being accessed by a person rather than a computer, then text alternatives that identify and describe the purpose of the non-text content are provided, and alternative forms of CAPTCHA using output modes for different types of sensory perception are provided to accommodate different disabilities.

Decoration, Formatting, Invisible

If non-text content is [pure decoration](#), is used only for visual formatting, or is not presented to users, then it is implemented in a way that it can be ignored by [assistive technology](#).

§ Guideline 1.2 Time-based Media

Provide alternatives for time-based media.

[Understanding Time-based Media](#)

[How to Meet Time-based Media](#)

§ Success Criterion 1.2.1 Audio-only and Video-only (Prerecorded)

(Level A)

For [prerecorded audio-only](#) and [prerecorded video-only](#) media, the following are true, except when the audio or video is a [media alternative for text](#) and is clearly labeled as such:

[Understanding Audio-only and Video-only \(Prerecorded\)](#)

[How to Meet Audio-only and Video-only \(Prerecorded\)](#)

Prerecorded Audio-only

An [alternative for time-based media](#) is provided that presents equivalent information for prerecorded audio-only content.

Prerecorded Video-only

Either an alternative for time-based media or an audio track is provided that presents equivalent information for prerecorded video-only content.

§ Success Criterion 1.2.2 Captions (Prerecorded)

(Level A)

[Captions](#) are provided for all [prerecorded audio](#) content in [synchronized media](#), except when the media is a [media alternative for text](#) and is clearly labeled as such.

[Understanding Captions \(Prerecorded\)](#)

[How to Meet Captions \(Prerecorded\)](#)

§ Success Criterion 1.2.3 Audio Description or Media Alternative (Prerecorded)

(Level A)

An alternative for time-based media or audio description of the prerecorded video content is provided for synchronized media, except when the media is a media alternative for text and is clearly labeled as such.

[Understanding Audio Description or Media Alternative \(Prerecorded\)](#)

[How to Meet Audio Description or Media Alternative \(Prerecorded\)](#)

§ Success Criterion 1.2.4 Captions (Live)

(Level AA)

Captions are provided for all live audio content in synchronized media.

[Understanding Captions \(Live\)](#)

[How to Meet Captions \(Live\)](#)

§ Success Criterion 1.2.5 Audio Description (Prerecorded)

(Level AA)

Audio description is provided for all prerecorded video content in synchronized media.

[Understanding Audio Description \(Prerecorded\)](#)

[How to Meet Audio Description \(Prerecorded\)](#)

§ Success Criterion 1.2.6 Sign Language (Prerecorded)

(Level AAA)

Sign language interpretation is provided for all prerecorded audio content in synchronized media.

[Understanding Sign Language \(Prerecorded\)](#)

[How to Meet Sign Language \(Prerecorded\)](#)

§ Success Criterion 1.2.7 Extended Audio Description (Prerecorded)

(Level AAA)

[Understanding Extended Audio](#)

Where pauses in foreground audio are insufficient to allow [audio descriptions](#) to convey the sense of the video, [extended audio description](#) is provided for all [prerecorded video](#) content in [synchronized media](#).

[Description \(Prerecorded\)](#)

[How to Meet Extended Audio Description \(Prerecorded\)](#)

§ Success Criterion 1.2.8 Media Alternative (Prerecorded)

(Level AAA)

An [alternative for time-based media](#) is provided for all [prerecorded synchronized media](#) and for all [prerecorded video-only](#) media.

[Understanding Media Alternative \(Prerecorded\)](#)

[How to Meet Media Alternative \(Prerecorded\)](#)

§ Success Criterion 1.2.9 Audio-only (Live)

(Level AAA)

An [alternative for time-based media](#) that presents equivalent information for [live audio-only](#) content is provided.

[Understanding Audio-only \(Live\)](#)

[How to Meet Audio-only \(Live\)](#)

§ Guideline 1.3 Adaptable

Create content that can be presented in different ways (for example simpler layout) without losing information or structure.

[Understanding Adaptable](#)

[How to Meet Adaptable](#)

§ Success Criterion 1.3.1 Info and Relationships

(Level A)

Information, [structure](#), and [relationships](#) conveyed through [presentation](#) can be [programmatically determined](#) or are available in text.

[Understanding Info and Relationships](#)

[How to Meet Info and Relationships](#)

§ Success Criterion 1.3.2 Meaningful Sequence

(Level A)

When the sequence in which content is presented affects its meaning, a correct reading sequence can be programmatically determined.

[Understanding Meaningful Sequence](#)

[How to Meet Meaningful Sequence](#)

§ Success Criterion 1.3.3 Sensory Characteristics

(Level A)

Instructions provided for understanding and operating content do not rely solely on sensory characteristics of components such as shape, color, size, visual location, orientation, or sound.

[Understanding Sensory Characteristics](#)

[How to Meet Sensory Characteristics](#)

NOTE

For requirements related to color, refer to [Guideline 1.4](#).

§ Success Criterion 1.3.4 Orientation

(Level AA)

Content does not restrict its view and operation to a single display orientation, such as portrait or landscape, unless a specific display orientation is essential.

[Understanding Orientation](#)

[How to Meet Orientation](#)

NOTE

Examples where a particular display orientation may be essential are a bank check, a piano application, slides for a projector or television, or virtual reality content where content is not necessarily restricted to landscape or portrait display orientation.

§ Success Criterion 1.3.5 Identify Input Purpose

(Level AA)

The purpose of each input field collecting information about the user can be [programmatically determined](#) when:

[Understanding Identify Input Purpose](#)[How to Meet Identify Input Purpose](#)

- The input field serves a purpose identified in the [Input Purposes for user interface components section](#); and
- The content is implemented using technologies with support for identifying the expected meaning for form input data.

§ Success Criterion 1.3.6 Identify Purpose

(Level AAA)

In content implemented using markup languages, the purpose of [user interface components](#), icons, and [regions](#) can be [programmatically determined](#).

[Understanding Identify Purpose](#)[How to Meet Identify Purpose](#)

§ Guideline 1.4 Distinguishable

Make it easier for users to see and hear content including separating foreground from background.

[Understanding Distinguishable](#)[How to Meet Distinguishable](#)

§ Success Criterion 1.4.1 Use of Color

(Level A)

Color is not used as the only visual means of conveying information, indicating an action, prompting a response, or distinguishing a visual element.

[Understanding Use of Color](#)[How to Meet Use of Color](#)

NOTE

This success criterion addresses color perception specifically. Other forms of perception are covered in [Guideline 1.3](#) including programmatic access to color and other visual presentation coding.

§ Success Criterion 1.4.2 Audio Control

(Level A)

If any audio on a web page plays automatically for more than 3 seconds, either a mechanism is available to pause or stop the audio, or a mechanism is available to control audio volume independently from the overall system volume level.

[Understanding Audio Control](#)

[How to Meet Audio Control](#)

NOTE

Since any content that does not meet this success criterion can interfere with a user's ability to use the whole page, all content on the web page (whether or not it is used to meet other success criteria) must meet this success criterion. See [Conformance Requirement 5: Non-Interference](#).

§ Success Criterion 1.4.3 Contrast (Minimum)

(Level AA)

The visual presentation of text and images of text has a contrast ratio of at least 4.5:1, except for the following:

[Understanding Contrast \(Minimum\)](#)

[How to Meet Contrast \(Minimum\)](#)

Large Text

Large-scale text and images of large-scale text have a contrast ratio of at least 3:1;

Incidental

Text or images of text that are part of an inactive user interface component, that are pure decoration, that are not visible to anyone, or that are part of a picture that contains significant other visual content, have no contrast requirement.

Logotypes

Text that is part of a logo or brand name has no contrast requirement.

§ Success Criterion 1.4.4 Resize Text

(Level AA)

Except for captions and images of text, text can be resized without

[Understanding Resize Text](#)

[How to Meet Resize Text](#)

assistive technology up to 200 percent without loss of content or functionality.

§ Success Criterion 1.4.5 Images of Text

(Level AA)

If the technologies being used can achieve the visual presentation, text

[Understanding Images of Text](#)

is used to convey information rather than images of text except for the following:

[How to Meet Images of Text](#)

Customizable

The image of text can be visually customized to the user's requirements;

Essential

A particular presentation of text is essential to the information being conveyed.

NOTE

Logotypes (text that is part of a logo or brand name) are considered essential.

§ Success Criterion 1.4.6 Contrast (Enhanced)

(Level AAA)

The visual presentation of text and images of text has a contrast ratio of at least 7:1, except for the following:

[Understanding Contrast \(Enhanced\)](#)

[How to Meet Contrast \(Enhanced\)](#)

Large Text

Large-scale text and images of large-scale text have a contrast ratio of at least 4.5:1;

Incidental

Text or images of text that are part of an inactive user interface component, that are pure decoration, that are not visible to anyone, or that are part of a picture that contains significant other visual content, have no contrast requirement.

Logotypes

Text that is part of a logo or brand name has no contrast requirement.

§ Success Criterion 1.4.7 Low or No Background Audio

(Level AAA)

For prerecorded audio-only content that (1) contains primarily speech in the foreground, (2) is not an audio CAPTCHA or audio logo, and (3) is not vocalization intended to be primarily musical expression such as singing or rapping, at least one of the following is true:

[Understanding Low or No Background Audio](#)

[How to Meet Low or No Background Audio](#)

No Background

The audio does not contain background sounds.

Turn Off

The background sounds can be turned off.

20 dB

The background sounds are at least 20 decibels lower than the foreground speech content, with the exception of occasional sounds that last for only one or two seconds.

NOTE

Per the definition of "decibel," background sound that meets this requirement will be approximately four times quieter than the foreground speech content.

§ Success Criterion 1.4.8 Visual Presentation

(Level AAA)

For the visual presentation of blocks of text, a mechanism is available to achieve the following:

[Understanding Visual Presentation](#)

[How to Meet Visual Presentation](#)

- Foreground and background colors can be selected by the user.
- Width is no more than 80 characters or glyphs (40 if CJK).
- Text is not justified (aligned to both the left and the right margins).
- Line spacing (leading) is at least space-and-a-half within paragraphs, and paragraph spacing is at least 1.5 times larger than the line spacing.

- Text can be resized without [assistive technology](#) up to 200 percent in a way that does not require the user to scroll horizontally to read a line of text [on a full-screen window](#).

NOTE 1

Content is not required to use these values. The requirement is that a mechanism is available for users to change these presentation aspects. The mechanism can be provided by the browser or other user agent. Content is not required to provide the mechanism.

NOTE 2

Writing systems for some languages use different presentation aspects to improve readability and legibility. If a presentation aspect in this success criterion is not used in a writing system, content in that writing system does not need to use that presentation setting and can conform without it. Authors are encouraged to follow guidance for improving readability and legibility of text in their writing system.

§ Success Criterion 1.4.9 Images of Text (No Exception)

(Level AAA)

[Images of text](#) are only used for [pure decoration](#) or where a particular presentation of [text](#) is [essential](#) to the information being conveyed.

[Understanding Images of Text \(No Exception\)](#)

[How to Meet Images of Text \(No Exception\)](#)

NOTE

Logotypes (text that is part of a logo or brand name) are considered essential.

§ Success Criterion 1.4.10 Reflow

(Level AA)

Content can be presented without loss of information or functionality, and without requiring scrolling in two dimensions for:

[Understanding Reflow](#)

[How to Meet Reflow](#)

- Vertical scrolling content at a width equivalent to 320 [CSS pixels](#);

- Horizontal scrolling content at a height equivalent to 256 [CSS pixels](#).

Except for parts of the content which require two-dimensional layout for usage or meaning.

NOTE 1

320 CSS pixels is equivalent to a starting [viewport](#) width of 1280 CSS pixels wide at 400% zoom. For web content which is designed to scroll horizontally (e.g., with vertical text), 256 CSS pixels is equivalent to a starting viewport height of 1024 CSS pixels at 400% zoom.

NOTE 2

Examples of content which requires two-dimensional layout are images required for understanding (such as maps and diagrams), video, games, presentations, data tables (not individual cells), and interfaces where it is necessary to keep toolbars in view while manipulating content. It is acceptable to provide two-dimensional scrolling for such parts of the content.

§ Success Criterion 1.4.11 Non-text Contrast

(Level AA)

The visual [presentation](#) of the following have a [contrast ratio](#) of at least 3:1 against adjacent color(s):

[Understanding Non-text Contrast](#)

[How to Meet Non-text Contrast](#)

User Interface Components

Visual information required to identify [user interface components](#) and [states](#), except for inactive components or where the appearance of the component is determined by the [user agent](#) and not modified by the author;

Graphical Objects

Parts of graphics required to understand the content, except when a particular presentation of graphics is [essential](#) to the information being conveyed.

§ Success Criterion 1.4.12 Text Spacing

(Level AA)

In content implemented using markup languages that support the

[Understanding Text Spacing](#)

[How to Meet Text Spacing](#)

following [text style properties](#), no loss of content or functionality occurs by setting all of the following and by changing no other style property:

- Line height (line spacing) to at least 1.5 times the font size;
- Spacing following paragraphs to at least 2 times the font size;
- Letter spacing (tracking) to at least 0.12 times the font size;
- Word spacing to at least 0.16 times the font size.

Exception: [Human languages](#) and scripts that do not make use of one or more of these text style properties in written text can conform using only the properties that exist for that combination of language and script.

NOTE 1

Content is not required to use these text spacing values. The requirement is to ensure that when a user overrides the authored text spacing, content or functionality is not lost.

NOTE 2

Writing systems for some languages use different text spacing settings, such as paragraph start indent. Authors are encouraged to follow locally available guidance for improving readability and legibility of text in their writing system.

§ Success Criterion 1.4.13 Content on Hover or Focus

(Level AA)

Where receiving and then removing pointer hover or keyboard focus triggers additional content to become visible and then hidden, the following are true:

[Understanding Content on Hover or Focus](#)

[How to Meet Content on Hover or Focus](#)

Dismissible

A [mechanism](#) is available to dismiss the additional content without moving pointer hover or keyboard focus, unless the additional content communicates an [input error](#) or does not obscure or replace other content;

Hoverable

If pointer hover can trigger the additional content, then the pointer can be moved over the

additional content without the additional content disappearing;

Persistent

The additional content remains visible until the hover or focus trigger is removed, the user dismisses it, or its information is no longer valid.

Exception: The visual presentation of the additional content is controlled by the [user agent](#) and is not modified by the author.

NOTE 1

Examples of additional content controlled by the user agent include browser tooltips created through use of the [HTML title attribute \[HTML\]](#).

NOTE 2

Custom tooltips, sub-menus, and other nonmodal popups that display on hover and focus are examples of additional content covered by this criterion.

NOTE 3

This criterion applies to content that appears in addition to the triggering component itself. Since hidden components that are made visible on keyboard focus (such as links used to skip to another part of a page) do not present additional content they are not covered by this criterion.

§ 2. Operable

User interface components and navigation must be operable.

§ Guideline 2.1 Keyboard Accessible

Make all functionality available from a keyboard.

[Understanding Keyboard Accessible](#)

[How to Meet Keyboard Accessible](#)

§ Success Criterion 2.1.1 Keyboard

(Level A)

All functionality of the content is operable through a keyboard interface without requiring specific timings for individual keystrokes, except where the underlying function requires input that depends on the path of the user's movement and not just the endpoints.

[Understanding Keyboard](#)

[How to Meet Keyboard](#)

NOTE 1

This exception relates to the underlying function, not the input technique. For example, if using handwriting to enter text, the input technique (handwriting) requires path-dependent input but the underlying function (text input) does not.

NOTE 2

This does not forbid and should not discourage providing mouse input or other input methods in addition to keyboard operation.

§ Success Criterion 2.1.2 No Keyboard Trap

(Level A)

If keyboard focus can be moved to a component of the page using a keyboard interface, then focus can be moved away from that component using only a keyboard interface, and, if it requires more than unmodified arrow or tab keys or other standard exit methods, the user is advised of the method for moving focus away.

[Understanding No Keyboard Trap](#)

[How to Meet No Keyboard Trap](#)

NOTE

Since any content that does not meet this success criterion can interfere with a user's ability to use the whole page, all content on the web page (whether it is used to meet other success criteria or not) must meet this success criterion. See [Conformance Requirement 5: Non-Interference](#).

§ Success Criterion 2.1.3 Keyboard (No Exception)

(Level AAA)

All functionality of the content is operable through a keyboard interface without requiring specific timings for individual keystrokes.

[Understanding Keyboard \(No Exception\)](#)

[How to Meet Keyboard \(No Exception\)](#)

§ Success Criterion 2.1.4 Character Key Shortcuts

(Level A)

If a keyboard shortcut is implemented in content using only letter (including upper- and lower-case letters), punctuation, number, or symbol characters, then at least one of the following is true:

Turn off

A mechanism is available to turn the shortcut off;

Remap

A mechanism is available to remap the shortcut to include one or more non-printable keyboard keys (e.g., Ctrl, Alt);

Active only on focus

The keyboard shortcut for a user interface component is only active when that component has focus.

[Understanding Character Key Shortcuts](#)

[How to Meet Character Key Shortcuts](#)

§ Guideline 2.2 Enough Time

Provide users enough time to read and use content.

[Understanding Enough Time](#)

[How to Meet Enough Time](#)

§ Success Criterion 2.2.1 Timing Adjustable

(Level A)

For each time limit that is set by the content, at least one of the following is true:

[Understanding Timing Adjustable](#)

[How to Meet Timing Adjustable](#)

Turn off

The user is allowed to turn off the time limit before encountering it; or

Adjust

The user is allowed to adjust the time limit before encountering it over a wide range that is at least ten times the length of the default setting; or

Extend

The user is warned before time expires and given at least 20 seconds to extend the time limit with a simple action (for example, "press the space bar"), and the user is allowed to extend the time limit at least ten times; or

Real-time Exception

The time limit is a required part of a [real-time event](#) (for example, an auction), and no alternative to the time limit is possible; or

Essential Exception

The time limit is [essential](#) and extending it would invalidate the activity; or

20 Hour Exception

The time limit is longer than 20 hours.

NOTE

This success criterion helps ensure that users can complete tasks without unexpected changes in content or context that are a result of a time limit. This success criterion should be considered in conjunction with [Success Criterion 3.2.1](#), which puts limits on changes of content or context as a result of user action.

§ Success Criterion 2.2.2 Pause, Stop, Hide

(Level A)

For moving, [blinking](#), scrolling, or auto-updating information, all of the following are true:

[Understanding Pause, Stop, Hide](#)

[How to Meet Pause, Stop, Hide](#)

Moving, [blinking](#), scrolling

For any moving, blinking or scrolling information that (1) starts automatically, (2) lasts more than five seconds, and (3) is presented in parallel with other content, there is a [mechanism](#) for the user to [pause](#), stop, or hide it unless the movement, blinking, or scrolling is part of an activity where it is [essential](#); and

Auto-updating

For any auto-updating information that (1) starts automatically and (2) is presented in parallel with other content, there is a mechanism for the user to pause, stop, or hide it or to control the frequency of the update unless the auto-updating is part of an activity where it is essential.

NOTE 1

For requirements related to flickering or flashing content, refer to [Guideline 2.3](#).

NOTE 2

Since any content that does not meet this success criterion can interfere with a user's ability to use the whole page, all content on the web page (whether it is used to meet other success criteria or not) must meet this success criterion. See [Conformance Requirement 5: Non-Interference](#).

NOTE 3

Content that is updated periodically by software or that is streamed to the user agent is not required to preserve or present information that is generated or received between the initiation of the pause and resuming presentation, as this may not be technically possible, and in many situations could be misleading to do so.

NOTE 4

An animation that occurs as part of a preload phase or similar situation can be considered essential if interaction cannot occur during that phase for all users and if not indicating progress could confuse users or cause them to think that content was frozen or broken.

§ Success Criterion 2.2.3 No Timing

(Level AAA)

Timing is not an [essential](#) part of the event or activity presented by the content, except for non-interactive [synchronized media](#) and [real-time events](#).

[Understanding No Timing](#)

[How to Meet No Timing](#)

§ Success Criterion 2.2.4 Interruptions

(Level AAA)

Interruptions can be postponed or suppressed by the user, except interruptions involving an [emergency](#).

[Understanding Interruptions](#)

[How to Meet Interruptions](#)

§ Success Criterion 2.2.5 Re-authenticating

(Level AAA)

When an authenticated session expires, the user can continue the activity without loss of data after re-authenticating.

[Understanding Re-authenticating](#)

[How to Meet Re-authenticating](#)

§ Success Criterion 2.2.6 Timeouts

(Level AAA)

Users are warned of the duration of any [user inactivity](#) that could cause data loss, unless the data is preserved for more than 20 hours when the user does not take any actions.

[Understanding Timeouts](#)

[How to Meet Timeouts](#)

NOTE

Privacy regulations may require explicit user consent before user identification has been authenticated and before user data is preserved. In cases where the user is a minor, explicit consent may not be solicited in most jurisdictions, countries or regions. Consultation with privacy professionals and legal counsel is advised when considering data preservation as an approach to satisfy this success criterion.

§ Guideline 2.3 Seizures and Physical Reactions

Do not design content in a way that is known to cause seizures or

[Understanding Seizures and Physical Reactions](#)

physical reactions.

[How to Meet Seizures and Physical Reactions](#)

§ Success Criterion 2.3.1 Three Flashes or Below Threshold

(Level A)

Web pages do not contain anything that flashes more than three times in any one second period, or the flash is below the general flash and red flash thresholds.

[Understanding Three Flashes or Below Threshold](#)

[How to Meet Three Flashes or Below Threshold](#)

NOTE

Since any content that does not meet this success criterion can interfere with a user's ability to use the whole page, all content on the web page (whether it is used to meet other success criteria or not) must meet this success criterion. See [Conformance Requirement 5: Non-Interference](#).

§ Success Criterion 2.3.2 Three Flashes

(Level AAA)

Web pages do not contain anything that flashes more than three times in any one second period.

[Understanding Three Flashes](#)

[How to Meet Three Flashes](#)

§ Success Criterion 2.3.3 Animation from Interactions

(Level AAA)

Motion animation triggered by interaction can be disabled, unless the animation is essential to the functionality or the information being conveyed.

[Understanding Animation from Interactions](#)

[How to Meet Animation from Interactions](#)

§ Guideline 2.4 Navigable

Provide ways to help users navigate, find content, and determine

[Understanding Navigable](#)

where they are.

[How to Meet Navigable](#)

§ Success Criterion 2.4.1 Bypass Blocks

(Level A)

A [mechanism](#) is available to bypass blocks of content that are repeated on multiple [web pages](#).

[Understanding Bypass Blocks](#)

[How to Meet Bypass Blocks](#)

§ Success Criterion 2.4.2 Page Titled

(Level A)

[Web pages](#) have titles that describe topic or purpose.

[Understanding Page Titled](#)

[How to Meet Page Titled](#)

§ Success Criterion 2.4.3 Focus Order

(Level A)

If a [web page](#) can be [navigated sequentially](#) and the navigation sequences affect meaning or operation, focusable components receive focus in an order that preserves meaning and operability.

[Understanding Focus Order](#)

[How to Meet Focus Order](#)

§ Success Criterion 2.4.4 Link Purpose (In Context)

(Level A)

The [purpose of each link](#) can be determined from the link text alone or from the link text together with its [programmatically determined link context](#), except where the purpose of the link would be [ambiguous to users in general](#).

[Understanding Link Purpose \(In Context\)](#)

[How to Meet Link Purpose \(In Context\)](#)

§ Success Criterion 2.4.5 Multiple Ways

(Level AA)

More than one way is available to locate a web page within a set of web pages except where the web page is the result of, or a step in, a process.

[Understanding Multiple Ways](#)[How to Meet Multiple Ways](#)

§ Success Criterion 2.4.6 Headings and Labels

(Level AA)

Headings and labels describe topic or purpose.

[Understanding Headings and Labels](#)[How to Meet Headings and Labels](#)

§ Success Criterion 2.4.7 Focus Visible

(Level AA)

Any keyboard operable user interface has a mode of operation where the keyboard focus indicator is visible.

[Understanding Focus Visible](#)[How to Meet Focus Visible](#)

§ Success Criterion 2.4.8 Location

(Level AAA)

Information about the user's location within a set of web pages is available.

[Understanding Location](#)[How to Meet Location](#)

§ Success Criterion 2.4.9 Link Purpose (Link Only)

(Level AAA)

A mechanism is available to allow the purpose of each link to be identified from link text alone, except where the purpose of the link would be ambiguous to users in general.

[Understanding Link Purpose \(Link Only\)](#)[How to Meet Link Purpose \(Link Only\)](#)

Success Criterion 2.4.10 Section Headings



(Level AAA)

Section headings are used to organize the content.

[Understanding Section Headings](#)[How to Meet Section Headings](#)

NOTE 1

"Heading" is used in its general sense and includes titles and other ways to add a heading to different types of content.

NOTE 2

This success criterion covers sections within writing, not user interface components. User interface components are covered under Success Criterion 4.1.2.



Success Criterion 2.4.11 Focus Not Obscured (Minimum)

(Level AA) **New**

When a user interface component receives keyboard focus, the component is not entirely hidden due to author-created content.

[Understanding Focus Not Obscured \(Minimum\)](#)[How to Meet Focus Not Obscured \(Minimum\)](#)

NOTE 1

Where content in a configurable interface can be repositioned by the user, then only the initial positions of user-movable content are considered for testing and conformance of this success criterion.

NOTE 2

Content opened by the *user* may obscure the component receiving focus. If the user can reveal the focused component without advancing the keyboard focus, the component with focus is not considered visually hidden due to author-created content.



Success Criterion 2.4.12 Focus Not Obscured (Enhanced)

(Level AAA) New

When a [user interface component](#) receives keyboard focus, no part of the component is hidden by author-created content.

[Understanding Focus Not Obscured \(Enhanced\)](#)

[How to Meet Focus Not Obscured \(Enhanced\)](#)

Success Criterion 2.4.13 Focus Appearance

(Level AAA) New

When the keyboard [focus indicator](#) is visible, an area of the focus indicator meets all the following:

- is at least as large as the area of a 2 [CSS pixel](#) thick [perimeter](#) of the unfocused component or sub-component, and
- has a contrast ratio of at least 3:1 between the same pixels in the focused and unfocused states.

Exceptions:

- The focus indicator is determined by the [user agent](#) and cannot be adjusted by the author, or
- The focus indicator and the indicator's background color are not modified by the author.

NOTE 1

What is perceived as the user interface component or sub-component (to determine the perimeter) depends on its visual [presentation](#). The visual presentation includes the component's visible [content](#), border, and component-specific background. It does not include shadow and glow effects outside the component's content, background, or border.

NOTE 2

Examples of sub-components that may receive a focus indicator are menu items in an opened drop-down menu, or focusable cells in a grid.

NOTE 3

Contrast calculations can be based on colors defined within the [technology](#) (such as [HTML](#), [CSS](#), and [SVG](#)). Pixels modified by user agent resolution enhancements and anti-aliasing can be ignored.

§ Guideline 2.5 Input Modalities

Make it easier for users to operate functionality through various inputs beyond keyboard.

[Understanding Input Modalities](#)

[How to Meet Input Modalities](#)

§ Success Criterion 2.5.1 Pointer Gestures

(Level A)

All [functionality](#) that uses multipoint or path-based gestures for operation can be operated with a [single pointer](#) without a path-based gesture, unless a multipoint or path-based gesture is [essential](#).

[Understanding Pointer Gestures](#)

[How to Meet Pointer Gestures](#)

NOTE

This requirement applies to web content that interprets pointer actions (i.e., this does not apply to actions that are required to operate the user agent or assistive technology).

§ Success Criterion 2.5.2 Pointer Cancellation

(Level A)

For [functionality](#) that can be operated using a [single pointer](#), at least one of the following is true:

[Understanding Pointer Cancellation](#)

[How to Meet Pointer Cancellation](#)

No Down-Event

The [down-event](#) of the pointer is not used to execute any part of the function;

Abort or Undo

Completion of the function is on the [up-event](#), and a [mechanism](#) is available to abort the function before completion or to undo the function after completion;

Up Reversal

The up-event reverses any outcome of the preceding down-event;

Essential

Completing the function on the down-event is [essential](#).

NOTE 1

Functions that emulate a keyboard or numeric keypad key press are considered essential.

NOTE 2

This requirement applies to web content that interprets pointer actions (i.e., this does not apply to actions that are required to operate the user agent or assistive technology).

§ Success Criterion 2.5.3 Label in Name

(Level A)

For user interface components with labels that include text or images of text, the name contains the text that is presented visually.

[Understanding Label in Name](#)

[How to Meet Label in Name](#)

NOTE

A best practice is to have the text of the label at the start of the name.

§ Success Criterion 2.5.4 Motion Actuation

(Level A)

Functionality that can be operated by device motion or user motion can also be operated by user interface components and responding to the motion can be disabled to prevent accidental actuation, except when:

[Understanding Motion Actuation](#)

[How to Meet Motion Actuation](#)

Supported Interface

The motion is used to operate functionality through an accessibility supported interface;

Essential

The motion is essential for the function and doing so would invalidate the activity.

§ Success Criterion 2.5.5 Target Size (Enhanced)

(Level AAA)

The size of the target for pointer inputs is at least 44 by 44 CSS pixels except when:

[Understanding Target Size \(Enhanced\)](#)

[How to Meet Target Size \(Enhanced\)](#)

Equivalent

The target is available through an equivalent link or control on the same page that is at least 44 by 44 CSS pixels;

Inline

The target is in a sentence or block of text;

User Agent Control

The size of the target is determined by the user agent and is not modified by the author;

Essential

A particular presentation of the target is essential to the information being conveyed.

§ Success Criterion 2.5.6 Concurrent Input Mechanisms

(Level AAA)

Web content does not restrict use of input modalities available on a platform except where the restriction is essential, required to ensure the security of the content, or required to respect user settings.

[Understanding Concurrent Input Mechanisms](#)

[How to Meet Concurrent Input Mechanisms](#)

§ Success Criterion 2.5.7 Dragging Movements

(Level AA) **New**

All functionality that uses a dragging movement for operation can be achieved by a single pointer without dragging, unless dragging is essential or the functionality is determined by the user agent and not modified by the author.

[Understanding Dragging Movements](#)

[How to Meet Dragging Movements](#)

NOTE

This requirement applies to web content that interprets pointer actions (i.e., this does not apply to actions that are required to operate the user agent or assistive technology).

§ Success Criterion 2.5.8 Target Size (Minimum)

(Level AA) **New**

The size of the target for pointer inputs is at least 24 by 24 CSS pixels, except when:

[Understanding Target Size \(Minimum\)](#)

[How to Meet Target Size \(Minimum\)](#)

Spacing

Undersized targets (those less than 24 by 24 CSS pixels) are positioned so that if a 24 CSS pixel diameter circle is centered on the bounding box of each, the circles do not intersect another target or the circle for another undersized target;

Equivalent

The function can be achieved through a different control on the same page that meets this criterion;

Inline

The target is in a sentence or its size is otherwise constrained by the line-height of non-target text;

User Agent Control

The size of the target is determined by the user agent and is not modified by the author;

Essential

A particular presentation of the target is essential or is legally required for the information being conveyed.

NOTE 1

Targets that allow for values to be selected spatially based on position within the target are considered one target for the purpose of the success criterion. Examples include sliders, color pickers displaying a gradient of colors, or editable areas where you position the cursor.

NOTE 2

For inline targets the line-height should be interpreted as perpendicular to the flow of text. For example, in a language displayed vertically, the line-height would be horizontal.

§ 3. Understandable

Information and the operation of the user interface must be understandable.

§ Guideline 3.1 Readable

Make text content readable and understandable.

[Understanding Readable](#)

[How to Meet Readable](#)

§ Success Criterion 3.1.1 Language of Page

(Level A)

The default human language of each web page can be programmatically determined.

[Understanding Language of Page](#)

[How to Meet Language of Page](#)

§ Success Criterion 3.1.2 Language of Parts

(Level AA)

The human language of each passage or phrase in the content can be programmatically determined except for proper names, technical terms, words of indeterminate language, and words or phrases that have become part of the vernacular of the immediately surrounding text.

[Understanding Language of Parts](#)

[How to Meet Language of Parts](#)

§ Success Criterion 3.1.3 Unusual Words

(Level AAA)

A mechanism is available for identifying specific definitions of words or phrases used in an unusual or restricted way, including idioms and jargon.

[Understanding Unusual Words](#)

[How to Meet Unusual Words](#)

§ Success Criterion 3.1.4 Abbreviations

(Level AAA)

A mechanism for identifying the expanded form or meaning of abbreviations is available.

[Understanding Abbreviations](#)

[How to Meet Abbreviations](#)

§ Success Criterion 3.1.5 Reading Level

(Level AAA)

When text requires reading ability more advanced than the lower secondary education level after removal of proper names and titles, supplemental content, or a version that does not require reading ability more advanced than the lower secondary education level, is available.

[Understanding Reading Level](#)

[How to Meet Reading Level](#)

§ Success Criterion 3.1.6 Pronunciation

(Level AAA)

A mechanism is available for identifying specific pronunciation of words where meaning of the words, in context, is ambiguous without knowing the pronunciation.

[Understanding Pronunciation](#)

[How to Meet Pronunciation](#)

§ Guideline 3.2 Predictable

Make web pages appear and operate in predictable ways.

[Understanding Predictable](#)

[How to Meet Predictable](#)

§ Success Criterion 3.2.1 On Focus

(Level A)

When any user interface component receives focus, it does not initiate a change of context.

[Understanding On Focus](#)

[How to Meet On Focus](#)

§ Success Criterion 3.2.2 On Input

(Level A)

Changing the setting of any user interface component does not

[Understanding On Input](#)

[How to Meet On Input](#)

automatically cause a [change of context](#) unless the user has been advised of the behavior before using the component.

§ Success Criterion 3.2.3 Consistent Navigation

(Level AA)

Navigational mechanisms that are repeated on multiple [web pages](#) within a [set of web pages](#) occur in the [same relative order](#) each time they are repeated, unless a change is initiated by the user.

[Understanding Consistent Navigation](#)

[How to Meet Consistent Navigation](#)

§ Success Criterion 3.2.4 Consistent Identification

(Level AA)

Components that have the [same functionality](#) within a [set of web pages](#) are identified consistently.

[Understanding Consistent Identification](#)

[How to Meet Consistent Identification](#)

§ Success Criterion 3.2.5 Change on Request

(Level AAA)

[Changes of context](#) are initiated only by user request or a [mechanism](#) is available to turn off such changes.

[Understanding Change on Request](#)

[How to Meet Change on Request](#)

§ Success Criterion 3.2.6 Consistent Help

(Level A) **New**

If a [web page](#) contains any of the following help [mechanisms](#), and those mechanisms are repeated on multiple web pages within a [set of web pages](#), they occur in the same order relative to other page content, unless a change is initiated by the user:

[Understanding Consistent Help](#)

[How to Meet Consistent Help](#)

- Human contact details;
- Human contact mechanism;

- Self-help option;
- A fully automated contact mechanism.

NOTE 1

Help mechanisms may be provided directly on the page, or may be provided via a direct link to a different page containing the information.

NOTE 2

For this success criterion, "the same order relative to other page content" can be thought of as how the content is ordered when the page is serialized. The visual position of a help mechanism is likely to be consistent across pages for the same page variation (e.g., CSS break-point). The user can initiate a change, such as changing the page's zoom or orientation, which may trigger a different page variation. This criterion is concerned with relative order across pages displayed in the same page variation (e.g., same zoom level and orientation).

§ Guideline 3.3 Input Assistance

Help users avoid and correct mistakes.

[Understanding Input Assistance](#)

[How to Meet Input Assistance](#)

§ Success Criterion 3.3.1 Error Identification

(Level A)

If an [input error](#) is automatically detected, the item that is in error is identified and the error is described to the user in text.

[Understanding Error Identification](#)

[How to Meet Error Identification](#)

§ Success Criterion 3.3.2 Labels or Instructions

(Level A)

[Labels](#) or instructions are provided when content requires user input.

[Understanding Labels or Instructions](#)

[How to Meet Labels or Instructions](#)

§ Success Criterion 3.3.3 Error Suggestion

(Level AA)

If an [input error](#) is automatically detected and suggestions for correction are known, then the suggestions are provided to the user, unless it would jeopardize the security or purpose of the content.

[Understanding Error Suggestion](#)

[How to Meet Error Suggestion](#)

§ Success Criterion 3.3.4 Error Prevention (Legal, Financial, Data)

(Level AA)

For [web pages](#) that cause [legal commitments](#) or financial transactions for the user to occur, that modify or delete [user-controllable](#) data in data storage systems, or that submit user test responses, at least one of the following is true:

Reversible

Submissions are reversible.

Checked

Data entered by the user is checked for [input errors](#) and the user is provided an opportunity to correct them.

Confirmed

A [mechanism](#) is available for reviewing, confirming, and correcting information before finalizing the submission.

[Understanding Error Prevention \(Legal, Financial, Data\)](#)

[How to Meet Error Prevention \(Legal, Financial, Data\)](#)

§ Success Criterion 3.3.5 Help

(Level AAA)

[Context-sensitive help](#) is available.

[Understanding Help](#)

[How to Meet Help](#)

§ Success Criterion 3.3.6 Error Prevention (All)

(Level AAA)

[Understanding Error Prevention \(All\)](#)

For [web pages](#) that require the user to submit information, at least one of the following is true:

[How to Meet Error Prevention \(All\)](#)

Reversible

Submissions are reversible.

Checked

Data entered by the user is checked for [input errors](#) and the user is provided an opportunity to correct them.

Confirmed

A [mechanism](#) is available for reviewing, confirming, and correcting information before finalizing the submission.

§ Success Criterion 3.3.7 Redundant Entry

(Level A) New

[Understanding Redundant Entry](#)

[How to Meet Redundant Entry](#)

Information previously entered by or provided to the user that is required to be entered again in the same [process](#) is either:

- auto-populated, or
- available for the user to select.

Except when:

- re-entering the information is [essential](#),
- the information is required to ensure the security of the content, or
- previously entered information is no longer valid.

§ Success Criterion 3.3.8 Accessible Authentication (Minimum)

(Level AA) New

[Understanding Accessible Authentication \(Minimum\)](#)

[How to Meet Accessible Authentication \(Minimum\)](#)

A [cognitive function test](#) (such as remembering a password or solving a puzzle) is not required for any step in an authentication [process](#) unless that step provides at least one of the following:

Alternative

Another authentication method that does not rely on a cognitive function test.

Mechanism

A mechanism is available to assist the user in completing the cognitive function test.

Object Recognition

The cognitive function test is to recognize objects.

Personal Content

The cognitive function test is to identify non-text content the user provided to the website.

NOTE 1

"Object recognition" and "Personal content" may be represented by images, video, or audio.

NOTE 2

Examples of mechanisms that satisfy this criterion include:

- support for password entry by password managers to reduce memory need, and
- copy and paste to reduce the cognitive burden of re-typing.

§ Success Criterion 3.3.9 Accessible Authentication (Enhanced)

(Level AAA) New

A cognitive function test (such as remembering a password or solving a puzzle) is not required for any step in an authentication process unless that step provides at least one of the following:

[Understanding Accessible Authentication \(Enhanced\)](#)

[How to Meet Accessible Authentication \(Enhanced\)](#)

Alternative

Another authentication method that does not rely on a cognitive function test.

Mechanism

A mechanism is available to assist the user in completing the cognitive function test.

§ 4. Robust

Content must be robust enough that it can be interpreted by a wide variety of user agents, including

assistive technologies.

§ Guideline 4.1 Compatible

Maximize compatibility with current and future user agents, including assistive technologies.

[Understanding Compatible](#)

[How to Meet Compatible](#)

§ Success Criterion 4.1.1 Parsing (Obsolete and removed)

NOTE

This criterion was originally adopted to address problems that assistive technology had directly parsing HTML. Assistive technology no longer has any need to directly parse HTML. Consequently, these problems either no longer exist or are addressed by other criteria. This criterion no longer has utility and is removed.

[Understanding Parsing \(Obsolete and removed\)](#)

[How to Meet Parsing \(Obsolete and removed\)](#)

§ Success Criterion 4.1.2 Name, Role, Value

(Level A)

For all user interface components (including but not limited to: form elements, links and components generated by scripts), the name and role can be programmatically determined; states, properties, and values that can be set by the user can be programmatically set; and notification of changes to these items is available to user agents, including assistive technologies.

[Understanding Name, Role, Value](#)

[How to Meet Name, Role, Value](#)

NOTE

This success criterion is primarily for web authors who develop or script their own user interface components. For example, standard HTML controls already meet this success criterion when used according to specification.

§ Success Criterion 4.1.3 Status Messages

(Level AA)

In content implemented using markup languages, [status messages](#) can be [programmatically determined](#) through [role](#) or properties such that they can be presented to the user by [assistive technologies](#) without receiving focus.

[Understanding Status Messages](#)

[How to Meet Status Messages](#)

§ 5. Conformance

This section lists requirements for [conformance](#) to WCAG 2.2. It also gives information about how to make conformance claims, which are optional. Finally, it describes what it means to be [accessibility supported](#), since only accessibility-supported ways of using technologies can be [relied upon](#) for conformance. [Understanding Conformance](#) includes further explanation of the accessibility-supported concept.

§ 5.1 Interpreting Normative Requirements

The main content of WCAG 2.2 is [normative](#) and defines requirements that impact conformance claims. Introductory material, appendices, sections marked as "non-normative", diagrams, examples, and notes are [informative](#) (non-normative). Non-normative material provides advisory information to help interpret the guidelines but does not create requirements that impact a conformance claim.

The key words *MAY*, *MUST*, *MUST NOT*, *NOT RECOMMENDED*, *RECOMMENDED*, *SHOULD*, and *SHOULD NOT* are to be interpreted as described in [RFC2119].

§ 5.2 Conformance Requirements

In order for a web page to conform to WCAG 2.2, all of the following conformance requirements must be satisfied:

5.2.1 Conformance Level

One of the following levels of conformance is met in full.

- For Level A conformance (the minimum level of conformance), the [web page satisfies](#) all the Level A success criteria, or a [conforming alternate version](#) is provided.
- For Level AA conformance, the web page satisfies all the Level A and Level AA success criteria, or a Level AA conforming alternate version is provided.
- For Level AAA conformance, the web page satisfies all the Level A, Level AA and Level AAA success criteria, or a Level AAA conforming alternate version is provided.

NOTE 1

Although conformance can only be achieved at the stated levels, authors are encouraged to report (in their claim) any progress toward meeting success criteria from all levels beyond the achieved level of conformance.

NOTE 2

It is not recommended that Level AAA conformance be required as a general policy for entire sites because it is not possible to satisfy all Level AAA success criteria for some content.

5.2.2 Full pages

[Conformance](#) (and conformance level) is for full [web page\(s\)](#) only, and cannot be achieved if part of a web page is excluded.

NOTE 1

For the purpose of determining conformance, alternatives to part of a page's content are considered part of the page when the alternatives can be obtained directly from the page, e.g., a long description or an alternative presentation of a video.

NOTE 2

Authors of web pages that cannot conform due to content outside of the author's control may consider a [Statement of Partial Conformance](#).

NOTE 3

A full page includes each variation of the page that is automatically presented by the page for various screen sizes (e.g. variations in a responsive web page). Each of these variations needs to conform (or needs to have a conforming alternate version) in order for the entire page to conform.

§ 5.2.3 Complete processes

When a [web page](#) is one of a series of web pages presenting a [process](#) (i.e., a sequence of steps that need to be completed in order to accomplish an activity), all web pages in the process conform at the specified level or better. (Conformance is not possible at a particular level if any page in the process does not conform at that level or better.)

EXAMPLE

An online store has a series of pages that are used to select and purchase products. All pages in the series from start to finish (checkout) conform in order for any page that is part of the process to conform.

§ 5.2.4 Only Accessibility-Supported Ways of Using Technologies

Only [accessibility-supported](#) ways of using [technologies](#) are [relied upon](#) to satisfy the success criteria. Any information or functionality that is provided in a way that is not accessibility supported is also available in a way that is accessibility supported. (See [Understanding accessibility support](#).)

§ 5.2.5 Non-Interference

If technologies are used in a way that is not accessibility supported, or if they are used in a non-conforming way, then they do not block the ability of users to access the rest of the page. In addition, the web page as a whole continues to meet the conformance requirements under each of the following conditions:

1. when any technology that is not relied upon is turned on in a user agent,
2. when any technology that is not relied upon is turned off in a user agent, and
3. when any technology that is not relied upon is not supported by a user agent

In addition, the following success criteria apply to all content on the page, including content that is not otherwise relied upon to meet conformance, because failure to meet them could interfere with any use of the page:

- **1.4.2 - Audio Control**,
- **2.1.2 - No Keyboard Trap**,
- **2.3.1 - Three Flashes or Below Threshold**, and
- **2.2.2 - Pause, Stop, Hide**.

NOTE

If a page cannot conform (for example, a conformance test page or an example page), it cannot be included in the scope of conformance or in a conformance claim.

For more information, including examples, see [Understanding Conformance Requirements](#).

§ 5.3 Conformance Claims (Optional)

Conformance is defined only for web pages. However, a conformance claim may be made to cover one page, a series of pages, or multiple related web pages.

§ 5.3.1 Required Components of a Conformance Claim

Conformance claims are **not required**. Authors can conform to WCAG 2.2 without making a claim. However, if a conformance claim is made, then the conformance claim **must** include the following

information:

1. **Date** of the claim
2. **Guidelines title, version and URI** "Web Content Accessibility Guidelines 2.2 at <https://www.w3.org/TR/WCAG22/>"
3. **Conformance level** satisfied: (Level A, AA or AAA)
4. **A concise description of the web pages**, such as a list of URIs for which the claim is made, including whether subdomains are included in the claim.

NOTE 1

The web pages may be described by list or by an expression that describes all of the URIs included in the claim.

NOTE 2

Web-based products that do not have a URI prior to installation on the customer's website may have a statement that the product would conform when installed.

5. A list of the web content technologies relied upon.

NOTE 3

If a conformance logo is used, it would constitute a claim and must be accompanied by the required components of a conformance claim listed above.

§ 5.3.2 Optional Components of a Conformance Claim

In addition to the required components of a conformance claim above, consider providing additional information to assist users. Recommended additional information includes:

- A list of success criteria beyond the level of conformance claimed that have been met. This information should be provided in a form that users can use, preferably machine-readable metadata.
- A list of the specific technologies that are "*used but not relied upon*."
- A list of user agents, including assistive technologies that were used to test the content.

- A list of specific accessibility characteristics of the content, provided in machine-readable metadata.
- Information about any additional steps taken that go beyond the success criteria to enhance accessibility.
- A machine-readable metadata version of the list of specific technologies that are relied upon.
- A machine-readable metadata version of the conformance claim.

NOTE 1

Refer to [Understanding Conformance Claims](#) for more information and example conformance claims.

NOTE 2

Refer to [Understanding Metadata](#) for more information about the use of metadata in conformance claims.

5.4 Statement of Partial Conformance - Third Party Content

Web pages that will later have additional content added can use a 'statement of partial conformance'. For example, an email program, a blog, an article that allows users to add comments, or applications supporting user-contributed content. Another example would be a page, such as a portal or news site, composed of content aggregated from multiple contributors, or sites that automatically insert content from other sources over time, such as when advertisements are inserted dynamically.

In these cases, it is not possible to know at the time of original posting what the uncontrolled content of the pages will be. It is important to note that the uncontrolled content can affect the accessibility of the controlled content as well. Two options are available:

1. A determination of conformance can be made based on best knowledge. If a page of this type is monitored and repaired (non-conforming content is removed or brought into conformance) within two business days, then a determination or claim of conformance can be made since, except for errors in externally contributed content which are corrected or removed when encountered, the page conforms. No conformance claim can be made if it is not possible to monitor or correct non-conforming content;

OR

2. A "statement of partial conformance" may be made that the page does not conform, but could conform if certain parts were removed. The form of that statement would be, "This page does not conform, but would conform to WCAG 2.2 at level X if the following parts from uncontrolled sources were removed." In addition, the following would also be true of uncontrolled content that is described in the statement of partial conformance:

1. It is not content that is under the author's control.
2. It is described in a way that users can identify (e.g., they cannot be described as "all parts that we do not control" unless they are clearly marked as such.)

5.5 Statement of Partial Conformance - Language

A "statement of partial conformance due to language" may be made when the page does not conform, but would conform if accessibility support existed for (all of) the language(s) used on the page. The form of that statement would be, "This page does not conform, but would conform to WCAG 2.2 at level X if accessibility support existed for the following language(s):"

5.6 Privacy Considerations

This section is non-normative.

Success criteria within this specification which the Working Group has identified possible implications for privacy, either by providing protections for end users or which are important for website providers to take into consideration when implementing features designed to protect user privacy, are listed below. This list reflects the current understanding of the Working Group but other Success criteria may have privacy implications that the Working Group is not aware of at the time of publishing.

Success criteria within this specification that may relate to privacy are:

- [2.2.6 Timeouts \(AAA\)](#)
- [3.3.7 Redundant Entry \(A\)](#)

§ 5.7 Security Considerations

This section is non-normative.

Success criteria within this specification which the Working Group has identified possible implications for security, either by providing protections for end users or which are important for website providers to take into consideration when implementing features designed to protect user security, are listed below. This list reflects the current understanding of the Working Group but other Success criteria may have security implications that the Working Group is not aware of at the time of publishing.

Success criteria within this specification that may relate to security are:

- [1.1.1 Non-text Content \(A\)](#)
- [1.3.5 Identify Input Purpose \(AA\)](#)
- [1.4.7 Low or No Background Audio \(AAA\)](#)
- [2.2.1 Timing Adjustable \(A\)](#)
- [2.2.5 Re-authenticating \(AAA\)](#)
- [2.2.6 Timeouts \(AAA\)](#)
- [2.5.6 Concurrent Input Mechanisms \(AAA\)](#)
- [3.3.3 Error Suggestion \(AA\)](#)
- [3.3.7 Redundant Entry \(A\)](#)
- [3.3.8 Accessible Authentication \(Minimum\) \(AA\)](#)
- [3.3.9 Accessible Authentication \(Enhanced\) \(AAA\)](#)

§ 6. Glossary

abbreviation

shortened form of a word, phrase, or name where the abbreviation has not become part of the language

NOTE 1

This includes initialisms and acronyms where:

1. **initialisms** are shortened forms of a name or phrase made from the initial letters of words or syllables contained in that name or phrase

NOTE 2

Not defined in all languages.

EXAMPLE 1

SNCF is a French initialism that contains the initial letters of the Société Nationale des Chemins de Fer, the French national railroad.

EXAMPLE 2

ESP is an initialism for extrasensory perception.

2. **acronyms** are abbreviated forms made from the initial letters or parts of other words (in a name or phrase) which may be pronounced as a word

EXAMPLE 3

NOAA is an acronym made from the initial letters of the National Oceanic and Atmospheric Administration in the United States.

NOTE 3

Some companies have adopted what used to be an initialism as their company name. In these cases, the new name of the company is the letters (for example, Ecma) and the word is no longer considered an abbreviation.

accessibility supported

supported by users' assistive technologies as well as the accessibility features in browsers and other user agents

To qualify as an accessibility-supported use of a web content technology (or feature of a technology), both 1 and 2 must be satisfied for a web content technology (or feature):

1. The way that the web content technology is used must be supported by users' assistive

technology (AT). This means that the way that the technology is used has been tested for interoperability with users' assistive technology in the human language(s) of the content,

AND

2. The web content technology must have accessibility-supported user agents that are available to users. This means that at least one of the following four statements is true:

1. The technology is supported natively in widely-distributed user agents that are also accessibility supported (such as HTML and CSS);

OR

2. The technology is supported in a widely-distributed plug-in that is also accessibility supported;

OR

3. The content is available in a closed environment, such as a university or corporate network, where the user agent required by the technology and used by the organization is also accessibility supported;

OR

4. The user agent(s) that support the technology are accessibility supported and are available for download or purchase in a way that:

- does not cost a person with a disability any more than a person without a disability **and**
- is as easy to find and obtain for a person with a disability as it is for a person without disabilities.

NOTE 1

The Accessibility Guidelines Working Group and the W3C do not specify which or how much support by assistive technologies there must be for a particular use of a web technology in order for it to be classified as accessibility supported. (See [Level of Assistive Technology Support Needed for "Accessibility Support"](#).)

NOTE 2

Web technologies can be used in ways that are not accessibility supported as long as they are not relied upon and the page as a whole meets the conformance requirements, including Conformance Requirement 4 and Conformance Requirement 5.

NOTE 3

When a web technology is used in a way that is "accessibility supported," it does not imply that the entire technology or all uses of the technology are supported. Most technologies, including HTML, lack support for at least one feature or use. Pages conform to WCAG only if the uses of the technology that are accessibility supported can be relied upon to meet WCAG requirements.

NOTE 4

When citing web content technologies that have multiple versions, the version(s) supported should be specified.

NOTE 5

One way for authors to locate uses of a technology that are accessibility supported would be to consult compilations of uses that are documented to be accessibility supported. (See Understanding Accessibility-Supported Web Technology Uses.) Authors, companies, technology vendors, or others may document accessibility-supported ways of using web content technologies. However, all ways of using technologies in the documentation would need to meet the definition of accessibility-supported Web content technologies above.

alternative for time-based media

document including correctly sequenced text descriptions of time-based visual and auditory information and providing a means for achieving the outcomes of any time-based interaction

NOTE

A screenplay used to create the synchronized media content would meet this definition only if it was corrected to accurately represent the final synchronized media after editing.

ambiguous to users in general

the purpose cannot be determined from the link and all information of the web page presented to

the user simultaneously with the link (i.e., readers without disabilities would not know what a link would do until they activated it)

EXAMPLE

The word guava in the following sentence "One of the notable exports is guava" is a link. The link could lead to a definition of guava, a chart listing the quantity of guava exported or a photograph of people harvesting guava. Until the link is activated, all readers are unsure and the person with a disability is not at any disadvantage.

ASCII art

picture created by a spatial arrangement of characters or glyphs (typically from the 95 printable characters defined by ASCII)

assistive technology (as used in this document)

hardware and/or software that acts as a [user agent](#), or along with a mainstream user agent, to provide functionality to meet the requirements of users with disabilities that go beyond those offered by mainstream user agents

NOTE 1

Functionality provided by assistive technology includes alternative presentations (e.g., as synthesized speech or magnified content), alternative input methods (e.g., voice), additional navigation or orientation mechanisms, and content transformations (e.g., to make tables more accessible).

NOTE 2

Assistive technologies often communicate data and messages with mainstream user agents by using and monitoring APIs.

NOTE 3

The distinction between mainstream user agents and assistive technologies is not absolute. Many mainstream user agents provide some features to assist individuals with disabilities. The basic difference is that mainstream user agents target broad and diverse audiences that usually include people with and without disabilities. Assistive technologies target narrowly defined populations of users with specific disabilities. The assistance provided by an assistive technology is more specific and appropriate to the needs of its target users. The mainstream user agent may provide important functionality to assistive technologies like retrieving web content from program objects or parsing markup into identifiable bundles.

EXAMPLE

Assistive technologies that are important in the context of this document include the following:

- screen magnifiers, and other visual reading assistants, which are used by people with visual, perceptual and physical print disabilities to change text font, size, spacing, color, synchronization with speech, etc. in order to improve the visual readability of rendered text and images;
- screen readers, which are used by people who are blind to read textual information through synthesized speech or braille;
- text-to-speech software, which is used by some people with cognitive, language, and learning disabilities to convert text into synthetic speech;
- speech recognition software, which may be used by people who have some physical disabilities;
- alternative keyboards, which are used by people with certain physical disabilities to simulate the keyboard (including alternate keyboards that use head pointers, single switches, sip/puff and other special input devices.);
- alternative pointing devices, which are used by people with certain physical disabilities to simulate mouse pointing and button activations.

audio

the technology of sound reproduction

NOTE

Audio can be created synthetically (including speech synthesis), recorded from real world sounds, or both.

audio description

narration added to the soundtrack to describe important visual details that cannot be understood from the main soundtrack alone

NOTE 1

Audio description of [video](#) provides information about actions, characters, scene changes, on-screen text, and other visual content.

NOTE 2

In standard audio description, narration is added during existing pauses in dialogue. (See also [extended audio description](#).)

NOTE 3

Where all of the [video](#) information is already provided in existing [audio](#), no additional audio description is necessary.

NOTE 4

Also called "video description" and "descriptive narration."

audio-only

a time-based presentation that contains only [audio](#) (no [video](#) and no interaction)

blinking

switch back and forth between two visual states in a way that is meant to draw attention

NOTE

See also [flash](#). It is possible for something to be large enough and blink brightly enough at the right frequency to be also classified as a flash.

blocks of text

more than one sentence of text

CAPTCHA

initialism for "Completely Automated Public Turing test to tell Computers and Humans Apart"

NOTE 1

CAPTCHA tests often involve asking the user to type in text that is displayed in an obscured image or audio file.

NOTE 2

A Turing test is any system of tests designed to differentiate a human from a computer. It is named after famed computer scientist Alan Turing. The term was coined by researchers at Carnegie Mellon University.

captions

synchronized visual and/or [text alternative](#) for both speech and non-speech audio information needed to understand the media content

NOTE 1

Captions are similar to dialogue-only subtitles except captions convey not only the content of spoken dialogue, but also equivalents for non-dialogue audio information needed to understand the program content, including sound effects, music, laughter, speaker identification and location.

NOTE 2

Closed Captions are equivalents that can be turned on and off with some players.

NOTE 3

Open Captions are any captions that cannot be turned off. For example, if the captions are visual equivalent [images of text](#) embedded in [video](#).

NOTE 4

Captions should not obscure or obstruct relevant information in the video.

NOTE 5

In some countries, captions are called subtitles.

NOTE 6

Audio descriptions can be, but do not need to be, captioned since they are descriptions of information that is already presented visually.

changes of context

major changes that, if made without user awareness, can disorient users who are not able to view the entire page simultaneously

Changes in context include changes of:

- user agent;
- viewport;
- focus;
- content that changes the meaning of the web page

NOTE

A change of content is not always a change of context. Changes in content, such as an expanding outline, dynamic menu, or a tab control do not necessarily change the context, unless they also change one of the above (e.g., focus).

EXAMPLE

Opening a new window, moving focus to a different component, going to a new page (including anything that would look to a user as if they had moved to a new page) or significantly re-arranging the content of a page are examples of changes of context.

cognitive function test**New**

A task that requires the user to remember, manipulate, or transcribe information. Examples include, but are not limited to:

- memorization, such as remembering a username, password, set of characters, images, or patterns. The common identifiers name, e-mail, and phone number are not considered cognitive function tests as they are personal to the user and consistent across websites;
- transcription, such as typing in characters;
- use of correct spelling;
- performance of calculations;
- solving of puzzles.

conformance

satisfying all the requirements of a given standard, guideline or specification

conforming alternate version

version that

1. conforms at the designated level, and
2. provides all of the same information and functionality in the same human language, and
3. is as up to date as the non-conforming content, and
4. for which at least one of the following is true:
 1. the conforming version can be reached from the non-conforming page via an accessibility-supported mechanism, or
 2. the non-conforming version can only be reached from the conforming version, or
 3. the non-conforming version can only be reached from a conforming page that also provides a mechanism to reach the conforming version

NOTE 1

In this definition, "can only be reached" means that there is some mechanism, such as a conditional redirect, that prevents a user from "reaching" (loading) the non-conforming page unless the user had just come from the conforming version.

NOTE 2

The alternate version does not need to be matched page for page with the original (e.g., the conforming alternate version may consist of multiple pages).

NOTE 3

If multiple language versions are available, then conforming alternate versions are required for each language offered.

NOTE 4

Alternate versions may be provided to accommodate different technology environments or user groups. Each version should be as conformant as possible. One version would need to be fully conformant in order to meet [conformance requirement 1](#).

NOTE 5

The conforming alternative version does not need to reside within the scope of conformance, or even on the same website, as long as it is as freely available as the non-conforming version.

NOTE 6

Alternate versions should not be confused with [supplementary content](#), which support the original page and enhance comprehension.

NOTE 7

Setting user preferences within the content to produce a conforming version is an acceptable mechanism for reaching another version as long as the method used to set the preferences is accessibility supported.

See [Understanding Conforming Alternate Versions](#)

content (web content)

information and sensory experience to be communicated to the user by means of a [user agent](#), including code or markup that defines the content's [structure](#), [presentation](#), and interactions

context-sensitive help

help text that provides information related to the function currently being performed

NOTE

Clear labels can act as context-sensitive help.

contrast ratio

$(L1 + 0.05) / (L2 + 0.05)$, where

- L1 is the relative luminance of the lighter of the colors, and
- L2 is the relative luminance of the darker of the colors.

NOTE 1

Contrast ratios can range from 1 to 21 (commonly written 1:1 to 21:1).

NOTE 2

Because authors do not have control over user settings as to how text is rendered (for example font smoothing or anti-aliasing), the contrast ratio for text can be evaluated with anti-aliasing turned off.

NOTE 3

For the purpose of Success Criteria 1.4.3 and 1.4.6, contrast is measured with respect to the specified background over which the text is rendered in normal usage. If no background color is specified, then white is assumed.

NOTE 4

Background color is the specified color of content over which the text is to be rendered in normal usage. It is a failure if no background color is specified when the text color is specified, because the user's default background color is unknown and cannot be evaluated for sufficient contrast. For the same reason, it is a failure if no text color is specified when a background color is specified.

NOTE 5

When there is a border around the letter, the border can add contrast and would be used in calculating the contrast between the letter and its background. A narrow border around the letter would be used as the letter. A wide border around the letter that fills in the inner details of the letters acts as a halo and would be considered background.

NOTE 6

WCAG conformance should be evaluated for color pairs specified in the content that an author would expect to appear adjacent in typical presentation. Authors need not consider unusual presentations, such as color changes made by the user agent, except where caused by authors' code.

correct reading sequence

any sequence where words and paragraphs are presented in an order that does not change the meaning of the content

CSS pixel

visual angle of about 0.0213 degrees

A CSS pixel is the canonical unit of measure for all lengths and measurements in CSS. This unit is density-independent, and distinct from actual hardware pixels present in a display. User agents and operating systems should ensure that a CSS pixel is set as closely as possible to the [CSS Values and Units Module Level 3 reference pixel \[css3-values\]](#), which takes into account the physical dimensions of the display and the assumed viewing distance (factors that cannot be determined by content authors).

down-event

platform event that occurs when the trigger stimulus of a pointer is depressed

The down-event may have different names on different platforms, such as "touchstart" or "mousedown".

dragging movement**New**

an operation where the pointer engages with an element on the [down-event](#) and the element (or a representation of its position) follows the pointer until an [up-event](#)

NOTE

Examples of draggable elements include list items, text elements, and images.

emergency

a sudden, unexpected situation or occurrence that requires immediate action to preserve health, safety, or property

essential

if removed, would fundamentally change the information or functionality of the content, **and** information and functionality cannot be achieved in another way that would conform

extended audio description

audio description that is added to an audiovisual presentation by pausing the [video](#) so that there is time to add additional description

NOTE

This technique is only used when the sense of the [video](#) would be lost without the additional [audio description](#) and the pauses between dialogue/narration are too short.

flash

a pair of opposing changes in [relative luminance](#) that can cause seizures in some people if it is large enough and in the right frequency range

NOTE 1

See [general flash and red flash thresholds](#) for information about types of flash that are not allowed.

NOTE 2

See also [blinking](#).

focus indicator

New

pixels that are changed to visually indicate when a [user interface component](#) is in a focused [state](#)

functionality

[processes](#) and outcomes achievable through user action

general flash and red flash thresholds

a [flash](#) or rapidly changing image sequence is below the threshold (i.e., content **passes**) if any of the following are true:

- there are no more than three **general flashes** and / or no more than three **red flashes** within any one-second period; or
- the combined area of flashes occurring concurrently occupies no more than a total of .006 steradians within any 10 degree visual field on the screen (25% of any 10 degree visual field)

on the screen) at typical viewing distance

where:

- A **general flash** is defined as a pair of opposing changes in relative luminance of 10% or more of the maximum relative luminance (1.0) where the relative luminance of the darker image is below 0.80; and where "a pair of opposing changes" is an increase followed by a decrease, or a decrease followed by an increase, and
- A **red flash** is defined as any pair of opposing transitions involving a saturated red

Exception: Flashing that is a fine, balanced, pattern such as white noise or an alternating checkerboard pattern with "squares" smaller than 0.1 degree (of visual field at typical viewing distance) on a side does not violate the thresholds.

NOTE 1

For general software or web content, using a 341 x 256 pixel rectangle anywhere on the displayed screen area when the content is viewed at 1024 x 768 pixels will provide a good estimate of a 10 degree visual field for standard screen sizes and viewing distances (e.g., 15-17 inch screen at 22-26 inches). This resolution of 75 - 85 ppi is known to be lower, and thus more conservative than the nominal CSS pixel resolution of 96 ppi in CSS specifications. Higher resolutions displays showing the same rendering of the content yield smaller and safer images so it is lower resolutions that are used to define the thresholds.

NOTE 2

A transition is the change in relative luminance (or relative luminance/color for red flashing) between adjacent peaks and valleys in a plot of relative luminance (or relative luminance/color for red flashing) measurement against time. A flash consists of two opposing transitions.

NOTE 3

The new working definition in the field for "**pair of opposing transitions involving a saturated red**" (from WCAG 2.2) is a pair of opposing transitions where, one transition is either to or from a state with a value $R/(R + G + B)$ that is greater than or equal to 0.8, and the difference between states is more than 0.2 (unitless) in the CIE 1976 UCS chromaticity diagram. [\[ISO_9241-391\]](#)

NOTE 4

Tools are available that will carry out analysis from video screen capture. However, no tool is necessary to evaluate for this condition if flashing is less than or equal to 3 flashes in any one second. Content automatically passes (see #1 and #2 above).

human language

language that is spoken, written or signed (through visual or tactile means) to communicate with humans

NOTE

See also [sign language](#).

idiom

phrase whose meaning cannot be deduced from the meaning of the individual words and the specific words cannot be changed without losing the meaning

NOTE

Idioms cannot be translated directly, word for word, without losing their (cultural or language-dependent) meaning.

EXAMPLE 1

In English, "spilling the beans" means "revealing a secret." However, "knocking over the beans" or "spilling the vegetables" does not mean the same thing.

EXAMPLE 2

In Japanese, the phrase "さじを投げる" literally translates into "he throws a spoon," but it means that there is nothing he can do and finally he gives up.

EXAMPLE 3

In Dutch, "Hij ging met de kippen op stok" literally translates into "He went to roost with the chickens," but it means that he went to bed early.

image of text

text that has been rendered in a non-text form (e.g., an image) in order to achieve a particular visual effect

NOTE

This does not include text that is part of a picture that contains significant other visual content.

EXAMPLE

A person's name on a nametag in a photograph.

informative

for information purposes and not required for conformance

NOTE

Content required for conformance is referred to as "normative."

input error

information provided by the user that is not accepted

NOTE

This includes:

1. Information that is required by the web page but omitted by the user
2. Information that is provided by the user but that falls outside the required data format or values

jargon

words used in a particular way by people in a particular field

EXAMPLE

The word StickyKeys is jargon from the field of assistive technology/accessibility.

keyboard interface

interface used by software to obtain keystroke input

NOTE 1

A keyboard interface allows users to provide keystroke input to programs even if the native technology does not contain a keyboard.

EXAMPLE

A touchscreen PDA has a keyboard interface built into its operating system as well as a connector for external keyboards. Applications on the PDA can use the interface to obtain keyboard input either from an external keyboard or from other applications that provide simulated keyboard output, such as handwriting interpreters or speech-to-text applications with "keyboard emulation" functionality.

NOTE 2

Operation of the application (or parts of the application) through a keyboard-operated mouse emulator, such as MouseKeys, does not qualify as operation through a keyboard interface because operation of the program is through its pointing device interface, not through its keyboard interface.

keyboard shortcut

alternative means of triggering an action by the pressing of one or more keys

label

text or other component with a text alternative that is presented to a user to identify a component within web content

NOTE 1

A label is presented to all users whereas the name may be hidden and only exposed by assistive technology. In many (but not all) cases the name and the label are the same.

NOTE 2

The term label is not limited to the label element in HTML.

large scale (text)

with at least 18 point or 14 point bold or font size that would yield equivalent size for Chinese, Japanese and Korean (CJK) fonts

NOTE 1

Fonts with extraordinarily thin strokes or unusual features and characteristics that reduce the familiarity of their letter forms are harder to read, especially at lower contrast levels.

NOTE 2

Font size is the size when the content is delivered. It does not include resizing that may be done by a user.

NOTE 3

The actual size of the character that a user sees is dependent both on the author-defined size and the user's display or user agent settings. For many mainstream body text fonts, 14 and 18 point is roughly equivalent to 1.2 and 1.5 em or to 120% or 150% of the default size for body text (assuming that the body font is 100%), but authors would need to check this for the particular fonts in use. When fonts are defined in relative units, the actual point size is calculated by the user agent for display. The point size should be obtained from the user agent, or calculated based on font metrics as the user agent does, when evaluating this success criterion. Users who have low vision would be responsible for choosing appropriate settings.

NOTE 4

When using text without specifying the font size, the smallest font size used on major browsers for unspecified text would be a reasonable size to assume for the font. If a level 1 heading is rendered in 14pt bold or higher on major browsers, then it would be reasonable to assume it is large text. Relative scaling can be calculated from the default sizes in a similar fashion.

NOTE 5

The 18 and 14 point sizes for roman texts are taken from the minimum size for large print (14pt) and the larger standard font size (18pt). For other fonts such as CJK languages, the "equivalent" sizes would be the minimum large print size used for those languages and the next larger standard large print size.

legal commitments

transactions where the person incurs a legally binding obligation or benefit

EXAMPLE

A marriage license, a stock trade (financial and legal), a will, a loan, adoption, signing up for the army, a contract of any type, etc.

link purpose

nature of the result obtained by activating a hyperlink

live

information captured from a real-world event and transmitted to the receiver with no more than a broadcast delay

NOTE 1

A broadcast delay is a short (usually automated) delay, for example used in order to give the broadcaster time to cue or censor the audio (or video) feed, but not sufficient to allow significant editing.

NOTE 2

If information is completely computer generated, it is not live.

lower secondary education level

the two or three year period of education that begins after completion of six years of school and ends nine years after the beginning of [primary education](#)

NOTE

This definition is based on the International Standard Classification of Education [UNESCO].

mechanism

[process](#) or technique for achieving a result

NOTE 1

The mechanism may be explicitly provided in the content, or may be [relied upon](#) to be provided by either the platform or by [user agents](#), including [assistive technologies](#).

NOTE 2

The mechanism needs to meet all success criteria for the conformance level claimed.

media alternative for text

media that presents no more information than is already presented in text (directly or via text alternatives)

NOTE

A media alternative for text is provided for those who benefit from alternate representations of text. Media alternatives for text may be audio-only, video-only (including sign-language video), or audio-video.

motion animation

addition of steps between conditions to create the illusion of movement or to give a sense of a smooth transition

EXAMPLE

For example, an element which moves into place or changes size while appearing is considered to be animated. An element which appears instantly without transitioning is not using animation. Motion animation does not include changes of color, blurring, or opacity which do not change the perceived size, shape, or position of the element.

minimum bounding box**New**

the smallest enclosing rectangle aligned to the horizontal axis within which all the points of a shape lie. For components which wrap onto multiple lines as part of a sentence or [block of text](#) (such as hypertext links), the bounding box is based on how the component would appear on a single line.

name

text by which software can identify a component within web content to the user

NOTE 1

The name may be hidden and only exposed by assistive technology, whereas a [label](#) is presented to all users. In many (but not all) cases, the label and the name are the same.

NOTE 2

This is unrelated to the name attribute in [HTML](#).

navigated sequentially

navigated in the order defined for advancing focus (from one element to the next) using a [keyboard interface](#)

non-text content

any content that is not a sequence of characters that can be [programmatically determined](#) or where the sequence is not expressing something in [human language](#)

NOTE

This includes [ASCII art](#) (which is a pattern of characters), emoticons, leetspeak (which uses character substitution), and images representing text

normative

required for conformance

NOTE 1

One may conform in a variety of well-defined ways to this document.

NOTE 2

Content identified as "[informative](#)" or "non-normative" is never required for [conformance](#).

on a full-screen window

on the most common sized desktop/laptop display with the [viewport](#) maximized

NOTE

Since people generally keep their computers for several years, it is best not to rely on the latest desktop/laptop display resolutions but to consider the common desktop/laptop display resolutions over the course of several years when making this evaluation.

paused

stopped by user request and not resumed until requested by user

perimeter**New**

continuous line forming the boundary of a shape not including shared pixels, or the [minimum bounding box](#), whichever is shortest.

EXAMPLE

The perimeter calculation for a 2 CSS pixel perimeter around a rectangle is $4h+4w$, where h is the height and w is the width. For a 2 CSS pixel perimeter around a circle it is $4\pi r$.

pointer input

input from a device that can target a specific coordinate (or set of coordinates) on a screen, such as a mouse, pen, or touch contact

NOTE

See the [Pointer Events definition for "pointer" \[pointerevents\]](#).

prerecorded

information that is not [live](#)

presentation

rendering of the [content](#) in a form to be perceived by users

primary education level

six year time period that begins between the ages of five and seven, possibly without any previous education

NOTE

This definition is based on the International Standard Classification of Education [[UNESCO](#)].

process

series of user actions where each action is required in order to complete an activity

EXAMPLE 1

Successful use of a series of web pages on a shopping site requires users to view alternative products, prices and offers, select products, submit an order, provide shipping information and provide payment information.

EXAMPLE 2

An account registration page requires successful completion of a [Turing test](#) before the registration form can be accessed.

programmatically determined (programmatically determinable)

determined by software from author-supplied data provided in a way that different [user agents](#), including [assistive technologies](#), can extract and present this information to users in different modalities

EXAMPLE 1

Determined in a markup language from elements and attributes that are accessed directly by commonly available assistive technology.

EXAMPLE 2

Determined from technology-specific data structures in a non-markup language and exposed to assistive technology via an accessibility API that is supported by commonly available assistive technology.

programmatically determined link context

additional information that can be [programmatically determined](#) from [relationships](#) with a link, combined with the link text, and presented to users in different modalities

EXAMPLE

In HTML, information that is programmatically determinable from a link in English includes text that is in the same paragraph, list item, or table cell as the link or in a table header cell that is associated with the table cell that contains the link.

NOTE

Since screen readers interpret punctuation, they can also provide the context from the current sentence, when the focus is on a link in that sentence.

programmatically set

set by software using methods that are supported by user agents, including assistive technologies

pure decoration

serving only an aesthetic purpose, providing no information, and having no functionality

NOTE

Text is only purely decorative if the words can be rearranged or substituted without changing their purpose.

EXAMPLE

The cover page of a dictionary has random words in very light text in the background.

real-time event

event that a) occurs at the same time as the viewing and b) is not completely generated by the content

EXAMPLE 1

A Webcast of a live performance (occurs at the same time as the viewing and is not prerecorded).

EXAMPLE 2

An on-line auction with people bidding (occurs at the same time as the viewing).

EXAMPLE 3

Live humans interacting in a virtual world using avatars (is not completely generated by the content and occurs at the same time as the viewing).

region

perceivable, programmatically determined section of content

NOTE

In HTML, any area designated with a landmark role would be a region.

relationships

meaningful associations between distinct pieces of content

relative luminance

the relative brightness of any point in a colorspace, normalized to 0 for darkest black and 1 for lightest white

NOTE 1

For the sRGB colorspace, the relative luminance of a color is defined as $L = 0.2126 * R + 0.7152 * G + 0.0722 * B$ where **R**, **G** and **B** are defined as:

- if $RsRGB \leq 0.04045$ then **R** = $RsRGB/12.92$ else **R** = $((RsRGB+0.055)/1.055)^{2.4}$
- if $GsRGB \leq 0.04045$ then **G** = $GsRGB/12.92$ else **G** = $((GsRGB+0.055)/1.055)^{2.4}$
- if $BsRGB \leq 0.04045$ then **B** = $BsRGB/12.92$ else **B** = $((BsRGB+0.055)/1.055)^{2.4}$

and $RsRGB$, $GsRGB$, and $BsRGB$ are defined as:

- $RsRGB = R8bit/255$
- $GsRGB = G8bit/255$
- $BsRGB = B8bit/255$

The " \wedge " character is the exponentiation operator. (Formula taken from [SRGB].)

NOTE 2

Before May 2021 the value of 0.04045 in the definition was different (0.03928). It was taken from an older version of the specification and has been updated. It has no practical effect on the calculations in the context of these guidelines.

NOTE 3

Almost all systems used today to view web content assume sRGB encoding. Unless it is known that another color space will be used to process and display the content, authors should evaluate using sRGB colorspace. If using other color spaces, see [Understanding Success Criterion 1.4.3.](#)

NOTE 4

If dithering occurs after delivery, then the source color value is used. For colors that are dithered at the source, the average values of the colors that are dithered should be used (average R, average G, and average B).

NOTE 5

Tools are available that automatically do the calculations when testing contrast and flash.

NOTE 6

A [separate page giving the relative luminance definition using MathML](#) to display the formulas is available.

relied upon (technologies that are)

the content would not [conform](#) if that [technology](#) is turned off or is not supported

role

text or number by which software can identify the function of a component within Web content

EXAMPLE

A number that indicates whether an image functions as a hyperlink, command button, or check box.

same functionality

same result when used

EXAMPLE

A submit "search" button on one web page and a "find" button on another web page may both have a field to enter a term and list topics in the website related to the term submitted. In this case, they would have the same functionality but would not be labeled consistently.

same relative order

same position relative to other items

NOTE

Items are considered to be in the same relative order even if other items are inserted or removed from the original order. For example, expanding navigation menus may insert an additional level of detail or a secondary navigation section may be inserted into the reading order.

satisfies a success criterion

the success criterion does not evaluate to 'false' when applied to the page

section

a self-contained portion of written content that deals with one or more related topics or thoughts

NOTE

A section may consist of one or more paragraphs and include graphics, tables, lists and sub-sections.

set of web pages

collection of [web pages](#) that share a common purpose and that are created by the same author, group or organization

EXAMPLE

Examples include:

- a publication which is split across multiple web pages, where each page contains one chapter or other significant section of the work. The publication is logically a single contiguous unit, and contains navigation features that enable access to the full set of pages.
- an e-commerce website shows products in a set of web pages that all share the same navigation and identification. However, when progressing to the checkout process, the template changes; the navigation and other elements are removed, so the pages in that process are functionally and visually different. The checkout pages are not part of the set of product pages.
- a blog on a sub-domain (e.g. blog.example.com) which has a different navigation and is authored by a distinct set of people from the pages on the primary domain (example.com).

NOTE

Different language versions would be considered different sets of web pages.

sign language

a language using combinations of movements of the hands and arms, facial expressions, or body positions to convey meaning

sign language interpretation

translation of one language, generally a spoken language, into a [sign language](#)

NOTE

True sign languages are independent languages that are unrelated to the spoken language(s) of the same country or region.

single pointer

an input modality that only targets a single point on the page/screen at a time – such as a mouse, single finger on a touch screen, or stylus.

NOTE

Single pointer interactions include clicks, double clicks, taps, dragging motions, and single-finger swipe gestures. In contrast, multipoint interactions involve the use of two or more pointers at the same time, such as two-finger interactions on a touchscreen, or the simultaneous use of a mouse and stylus.

specific sensory experience

a sensory experience that is not purely decorative and does not primarily convey important information or perform a function

EXAMPLE

Examples include a performance of a flute solo, works of visual art etc.

state

dynamic property expressing characteristics of a [user interface component](#) that may change in response to user action or automated processes

States do not affect the nature of the component, but represent data associated with the component or user interaction possibilities. Examples include focus, hover, select, press, check, visited/unvisited, and expand/collapse.

status message

change in content that is not a [change of context](#), and that provides information to the user on the success or results of an action, on the waiting state of an application, on the progress of a [process](#), or on the existence of errors

structure

- The way the parts of a [web page](#) are organized in relation to each other; and
- The way a collection of [web pages](#) is organized

style property

property whose value determines the presentation (e.g. font, color, size, location, padding, volume, synthesized speech prosody) of content elements as they are rendered (e.g. onscreen, via loudspeaker, via braille display) by user agents

Style properties can have several origins:

- User agent default styles: The default style property values applied in the absence of any author or user styles. Some web content technologies specify a default rendering, others do not;
- Author styles: Style property values that are set by the author as part of the content (e.g. inline styles, author style sheets);
- User styles: Style property values that are set by the user (e.g. via user agent interface settings, user style sheets)

supplemental content

additional [content](#) that illustrates or clarifies the primary content

EXAMPLE 1

An audio version of a [web page](#).

EXAMPLE 2

An illustration of a complex [process](#).

EXAMPLE 3

A paragraph summarizing the major outcomes and recommendations made in a research study.

synchronized media

[audio](#) or [video](#) synchronized with another format for presenting information and/or with time-based interactive components, unless the media is a [media alternative for text](#) that is clearly labeled as such

target

region of the display that will accept a pointer action, such as the interactive area of a [user interface component](#)

NOTE

If two or more targets are overlapping, the overlapping area should not be included in the measurement of the target size, except when the overlapping targets perform the same action or open the same page.

technology (web content)

[mechanism](#) for encoding instructions to be rendered, played or executed by [user agents](#)

NOTE 1

As used in these guidelines "web technology" and the word "technology" (when used alone) both refer to web content technologies.

NOTE 2

Web content technologies may include markup languages, data formats, or programming languages that authors may use alone or in combination to create end-user experiences that range from static web pages to synchronized media presentations to dynamic Web applications.

EXAMPLE

Some common examples of web content technologies include [HTML](#), [CSS](#), [SVG](#), [PNG](#), [PDF](#), [Flash](#), and [JavaScript](#).

text

sequence of characters that can be [programmatically determined](#), where the sequence is expressing something in [human language](#)

text alternative

[Text](#) that is programmatically associated with [non-text content](#) or referred to from text that is programmatically associated with non-text content. Programmatically associated text is text whose location can be [programmatically determined](#) from the non-text content.

EXAMPLE

An image of a chart is described in text in the paragraph after the chart. The short text alternative for the chart indicates that a description follows.

NOTE

Refer to [Understanding Text Alternatives](#) for more information.

up-event

platform event that occurs when the trigger stimulus of a pointer is released

The up-event may have different names on different platforms, such as "touchend" or "mouseup".

used in an unusual or restricted way

words used in such a way that requires users to know exactly which definition to apply in order to understand the content correctly

EXAMPLE

The term "gig" means something different if it occurs in a discussion of music concerts than it does in article about computer hard drive space, but the appropriate definition can be determined from context. By contrast, the word "text" is used in a very specific way in WCAG 2, so a definition is supplied in the glossary.

user agent

any software that retrieves and presents web content for users

EXAMPLE

Web browsers, media players, plug-ins, and other programs — including [assistive technologies](#) — that help in retrieving, rendering, and interacting with web content.

user-controllable

data that is intended to be accessed by users

NOTE

This does not refer to such things as Internet logs and search engine monitoring data.

EXAMPLE

Name and address fields for a user's account.

user interface component

a part of the content that is perceived by users as a single control for a distinct function

NOTE 1

Multiple user interface components may be implemented as a single programmatic element. "Components" here is not tied to programming techniques, but rather to what the user perceives as separate controls.

NOTE 2

User interface components include form elements and links as well as components generated by scripts.

NOTE 3

What is meant by "component" or "user interface component" here is also sometimes called "user interface element".

EXAMPLE

An applet has a "control" that can be used to move through content by line or page or random access. Since each of these would need to have a name and be settable independently, they would each be a "user interface component."

user inactivity

any continuous period of time where no user actions occur

The method of tracking will be determined by the website or application.

video

the technology of moving or sequenced pictures or images

NOTE

Video can be made up of animated or photographic images, or both.

video-only

a time-based presentation that contains only [video](#) (no [audio](#) and no interaction)

viewport

object in which the [user agent](#) presents content

NOTE 1

The user agent presents content through one or more viewports. Viewports include windows, frames, loudspeakers, and virtual magnifying glasses. A viewport may contain another viewport (e.g., nested frames). Interface components created by the user agent such as prompts, menus, and alerts are not viewports.

NOTE 2

This definition is based on [User Agent Accessibility Guidelines 1.0 Glossary \[UAAG10\]](#).

visually customized

the font, size, color, and background can be set

web page

a non-embedded resource obtained from a single URI using HTTP plus any other resources that are used in the rendering or intended to be rendered together with it by a [user agent](#)

NOTE 1

Although any "other resources" would be rendered together with the primary resource, they would not necessarily be rendered simultaneously with each other.

NOTE 2

For the purposes of conformance with these guidelines, a resource must be "non-embedded" within the scope of conformance to be considered a web page.

EXAMPLE 1

A web resource including all embedded images and media.

EXAMPLE 2

A web mail program built using Asynchronous JavaScript and XML (AJAX). The program lives entirely at <http://example.com/mail>, but includes an inbox, a contacts area and a calendar. Links or buttons are provided that cause the inbox, contacts, or calendar to display, but do not change the URI of the page as a whole.

EXAMPLE 3

A customizable portal site, where users can choose content to display from a set of different content modules.

EXAMPLE 4

When you enter "<http://shopping.example.com/>" in your browser, you enter a movie-like interactive shopping environment where you visually move around in a store dragging products off of the shelves around you and into a visual shopping cart in front of you. Clicking on a product causes it to be demonstrated with a specification sheet floating alongside. This might be a single-page website or just one page within a website.

7. Input Purposes for User Interface Components

This section contains a listing of common user interface component input purposes. The terms below are not keywords that must be used, but instead represent purposes that must be captured in the taxonomy adopted by a webpage. Where applicable, authors mark up controls with the chosen taxonomy to indicate the semantic purpose. This provides the potential for user agents and assistive technologies to apply personalized presentations that can enable more people to understand and use the content.

NOTE

The list of input type purposes is based on the control purposes defined in the [HTML specification's Autocomplete section](#), but it is important to understand that a different technology may have some or all of the same concepts defined in its specification and only the concepts that are mapped to the meanings below are required.

The following input control purposes are intended to relate to the user of the content and pertain only to information related to that individual.

- **name** - Full name
- **honorific-prefix** - Prefix or title (e.g., "Mr.", "Ms.", "Dr.", "Mlle")
- **given-name** - Given name (in some Western cultures, also known as the *first name*)
- **additional-name** - Additional names (in some Western cultures, also known as *middle names*, forenames other than the first name)
- **family-name** - Family name (in some Western cultures, also known as the *last name* or *surname*)
- **honorific-suffix** - Suffix (e.g., "Jr.", "B.Sc.", "MBASW", "II")
- **nickname** - Nickname, screen name, handle: a typically short name used instead of the full name
- **organization-title** - Job title (e.g., "Software Engineer", "Senior Vice President", "Deputy Managing Director")
- **username** - A username
- **new-password** - A new password (e.g., when creating an account or changing a password)
- **current-password** - The current password for the account identified by the **username** field (e.g., when logging in)
- **organization** - Company name corresponding to the person, address, or contact information in the other fields associated with this field
- **street-address** - Street address (multiple lines, newlines preserved)
- **address-line1** - Street address (one line per field, line 1)
- **address-line2** - Street address (one line per field, line 2)
- **address-line3** - Street address (one line per field, line 3)
- **address-level4** - The most fine-grained administrative level, in addresses with four

administrative levels

- **address-level3** - The third administrative level, in addresses with three or more administrative levels
- **address-level2** - The second administrative level, in addresses with two or more administrative levels; in the countries with two administrative levels, this would typically be the city, town, village, or other locality within which the relevant street address is found
- **address-level1** - The broadest administrative level in the address, i.e., the province within which the locality is found; for example, in the US, this would be the state; in Switzerland it would be the canton; in the UK, the post town
- **country** - Country code
- **country-name** - Country name
- **postal-code** - Postal code, post code, ZIP code, CEDEX code (if CEDEX, append "CEDEX", and the *arrondissement*, if relevant, to the **address-level2** field)
- **cc-name** - Full name as given on the payment instrument
- **cc-given-name** - Given name as given on the payment instrument (in some Western cultures, also known as the *first name*)
- **cc-additional-name** - Additional names given on the payment instrument (in some Western cultures, also known as *middle names*, forenames other than the first name)
- **cc-family-name** - Family name given on the payment instrument (in some Western cultures, also known as the *last name* or *surname*)
- **cc-number** - Code identifying the payment instrument (e.g., the credit card number)
- **cc-exp** - Expiration date of the payment instrument
- **cc-exp-month** - Month component of the expiration date of the payment instrument
- **cc-exp-year** - Year component of the expiration date of the payment instrument
- **cc-csc** - Security code for the payment instrument (also known as the card security code (CSC), card validation code (CVC), card verification value (CVV), signature panel code (SPC), credit card ID (CCID), etc)
- **cc-type** - Type of payment instrument
- **transaction-currency** - The currency that the user would prefer the transaction to use
- **transaction-amount** - The amount that the user would like for the transaction (e.g., when entering a bid or sale price)

- **language** - Preferred language
- **bday** - Birthday
- **bday-day** - Day component of birthday
- **bday-month** - Month component of birthday
- **bday-year** - Year component of birthday
- **sex** - Gender identity (e.g., Female, Fa'afafine)
- **url** - Home page or other web page corresponding to the company, person, address, or contact information in the other fields associated with this field
- **photo** - Photograph, icon, or other image corresponding to the company, person, address, or contact information in the other fields associated with this field
- **tel** - Full telephone number, including country code
- **tel-country-code** - Country code component of the telephone number
- **tel-national** - Telephone number without the country code component, with a country-internal prefix applied if applicable
- **tel-area-code** - Area code component of the telephone number, with a country-internal prefix applied if applicable
- **tel-local** - Telephone number without the country code and area code components
- **tel-local-prefix** - First part of the component of the telephone number that follows the area code, when that component is split into two components
- **tel-local-suffix** - Second part of the component of the telephone number that follows the area code, when that component is split into two components
- **tel-extension** - Telephone number internal extension code
- **email** - E-mail address
- **impp** - URL representing an instant messaging protocol endpoint (for example, "aim:goim?screenname=example" or "xmpp:fred@example.net")

§ A. Change Log

This section shows substantive changes incorporated into WCAG 2.2 since WCAG 2.1, as well as

changes made to 2.2 since its original publication on 05 October 2023. [Errata fixes to WCAG 2.1](#) have also been incorporated into WCAG 2.2.

The full [commit history to WCAG 2.2](#) is available.

- 2020-03-30: Added [Accessible Authentication \(Minimum\)](#).
- 2020-05-27: Added “Dragging” (later renamed [Dragging Movements](#)).
- 2020-07-19: Added “Findable Help” (later renamed to [Consistent Help](#)), “Pointer Target Spacing” (later renamed [Target Size \(Minimum\)](#)), and [Redundant Entry](#).
- 2020-08-04: Added “Focus Appearance (Minimum)” (later renamed to [Focus Appearance](#)).
- 2021-09-21: Added “Accessible Authentication (No Exception)” (later renamed [Accessible Authentication \(Enhanced\)](#)).
- 2022-03-22: Added [Focus Not Obscured \(Minimum\)](#).
- 2022-05-30: Added [Focus Not Obscured \(Enhanced\)](#).
- 2023-06-05: Added privacy and security sections within conformance.
- 2024-12-12: Republished WCAG 2.2, incorporating the following errata:
 - modified the definitions of [single pointer](#), [used in an unusual or restricted way](#), [motion animation](#), and [programmatically determined](#)
 - modified the formatting of definitions for [changes of context](#), [general flash and red flash thresholds](#), [cognitive function test](#), and [structure](#)
 - removed the defunct “encloses” definition
 - corrected typo in [input purposes](#) list
 - modified the formatting of Target Size (Minimum) and Accessible Authentication (Minimum)
 - modified the visual presentation for content identified as New
 - modified the language covering devices in the [Abstract](#)
 - made editorial changes to improve consistent use of definitions in the success criteria
 - made editorial changes to improve consistent use of the terms “success criteria/criterion”, “web”, “website”, and “web page”

§ B. Acknowledgments

This section is non-normative.

Additional information about participation in the Accessibility Guidelines Working Group (AG WG) can be found on the [Working Group home page](#).

§ B.1 Participants of the AG WG active in the development of this document:

- Jake Abma (Invited Expert)
- Shadi Abou-Zahra (Amazon)
- Chuck Adams (Oracle Corporation)
- Amani Ali (Nomensa)
- Jim Allan (Invited Expert)
- Jon Avila (Level Access)
- Bruce Bailey (U.S. Access Board)
- Renaldo Bernard (University of Southampton)
- Dan Bjorge (Deque Systems, Inc.)
- Peter Bossley (Thomson Reuters)
- Rachael Bradley Montgomery (Library of Congress)
- Judy Brewer (W3C)
- Shari Butler (Pearson plc)
- Thaddeus Cambron (Invited Expert)
- Alastair Campbell (Nomensa)
- Laura Carlson (Invited Expert)
- Sukriti Chadha (Invited Expert)
- Rafal Charlampowicz (AccessibilityOZ)
- Michael Cooper (W3C)

- Jennifer Delisi (Invited Expert)
- Wayne Dick (Knowbility, Inc)
- Kim Dirks (Thomson Reuters)
- E.A. Draffan (University of Southampton)
- Eric Eggert (W3C)
- Michael Elledge (Invited Expert)
- Steve Faulkner (TPGi)
- David Fazio (Invited Expert)
- Wilco Fiers (Deque Systems, Inc.)
- Detlev Fischer (Invited Expert)
- John Foliot (Invited Expert)
- Matt Garrish (DAISY Consortium)
- Alistair Garrison (Level Access)
- Jaunita George (Navy Federal Credit Union)
- Michael Gower (IBM Corporation)
- Markku Hakkinen (Educational Testing Service)
- Charles Hall (Invited Expert)
- Katie Haritos-Shea (Knowbility, Inc)
- Dan Harper-Wain (HM Government)
- Shawn Henry (W3C)
- Sarah Horton (Invited Expert)
- Abi James (University of Southampton)
- Marc Johlic (IBM Corporation)
- Oliver Keim (SAP SE)
- Andrew Kirkpatrick (Adobe)
- John Kirkwood (Invited Expert)
- JaEun Jemma Ku (University of Illinois Chicago)

- Patrick H. Lauke (TetraLogical)
- Shawn Lauriat (Google, Inc.)
- Steve Lee (Invited Expert)
- Chris Loiselle (Invited Expert)
- David MacDonald (Invited Expert)
- Jan McSorley (Pearson plc)
- Rain Breaw Michaels (Google LLC)
- Neil Milliken (Unify Software and Solutions)
- Mary Jo Mueller (IBM Corporation)
- Jay Mullen (College Board)
- Brooks Newton (Thomson Reuters)
- Gundula Niemann (SAP SE)
- James Nurthen (Oracle Corporation)
- Lori Oakley (Oracle Corporation)
- Joshue O Connor (Invited Expert)
- Scott O'Hara (Microsoft)
- Sailesh Panchang (Deque Systems, Inc.)
- Kim Patch (Invited Expert)
- Melanie Philipp (Deque Systems, Inc.)
- Mike Pluke (Invited Expert)
- Ian Pouncey (TetraLogical)
- Ruoxi Ran (W3C)
- Stephen Repsher (Invited Expert)
- John Rochford (Invited Expert)
- Stefan Schnabel (SAP SE)
- Ayelet Seeman (Invited Expert)
- Lisa Seeman-Kestenbaum (Invited Expert)

- Glenda Sims (Deque Systems, Inc.)
- Avneesh Singh (DAISY Consortium)
- David Sloan (TPGi)
- Andrew Somers (Invited Expert)
- Jeanne Spellman (TetraLogical)
- Francis Storr (Intel)
- Poornima Badhan Subramanian (Invited Expert)
- Ben Tillyer (Invited Expert)
- Makoto Ueki (Invited Expert)
- Gregg Vanderheiden (Raising the Floor)
- Kathleen Wahlbin (Invited Expert)
- Léonie Watson (TetraLogical)
- Jason White (Educational Testing Service)
- White, Kevin (W3C Staff)
- Mark Wilcock (Unify Software and Solutions)

B.2 Other previously active WCAG WG participants and other contributors to WCAG 2.0, WCAG 2.1, or supporting resources

Paul Adam, Jenae Andershonis, Wilhelm Joys Andersen, Andrew Arch, Avi Arditti, Aries Arditi, Tom Babinszki, Mark Barratt, Mike Barta, Sandy Bartell, Kynn Bartlett, Chris Beer, Charles Belov, Marco Bertoni, Harvey Bingham, Chris Blouch, Paul Bohman, Frederick Boland, Denis Boudreau, Patrice Bourlon, Andy Brown, Dick Brown, Doyle Burnett, Raven Calais, Ben Caldwell, Tomas Caspers, Roberto Castaldo, Sofia Celic-Li, Sambhavi Chandrashekhar, Mike Cherim, Jonathan Chetwynd, Wendy Chisholm, Alan Chuter, David M Clark, Joe Clark, Darcy Clarke, James Coltham, Earl Cousins, James Craig, Tom Croucher, Pierce Crowell, Nir Dagan, Daniel Dardailler, Geoff Deering, Sébastien Delorme, Pete DeVasto, Iyad Abu Doush, Sylvie Duchateau, Cherie Eckholm, Roberto Ellero, Don Evans, Gavin Evans, Neal Ewers, Steve Faulkner, Bengt Farre, Lainey Feingold, Wilco Fiers, Michel Fitos, Alan J. Flavell, Nikolaos Floratos, Kentarou Fukuda, Miguel Garcia, P.J. Gardner, Alistair Garrison, Greg Gay, Becky Gibson, Al Gilman, Kerstin Goldsmith, Michael Grade, Karl Groves, Loretta Guarino Reid, Jon Gunderson, Emmanuelle Gutiérrez y Restrepo, Brian Hardy, Eric

Hansen, Benjamin Hawkes-Lewis, Sean Hayes, Shawn Henry, Hans Hillen, Donovan Hipke, Bjoern Hoehrmann, Allen Hoffman, Chris Hofstader, Yvette Hoitink, Martijn Houtepen, Carlos Iglesias, Richard Ishida, Jonas Jacek, Ian Jacobs, Phill Jenkins, Barry Johnson, Duff Johnson, Jyotsna Kaki, Shilpi Kapoor, Leonard R. Kasday, Kazuhito Kidachi, Ken Kipness, Johannes Koch, Marja-Riitta Koivunen, Maureen Kraft, Preety Kumar, Kristjan Kure, Andrew LaHart, Gez Lemon, Chuck Letourneau, Aurélien Levy, Harry Loots, Scott Luebking, Tim Lacy, Jim Ley, Alex Li, William Loughborough, N Maffeo, Mark Magennis, Erich Manser, Kapsi Maria, Luca Mascaro, Matt May, Sheena McCullagh, Liam McGee, Jens Oliver Meiert, Niqui Merret, Jonathan Metz, Alessandro Miele, Steven Miller, Mathew J Mirabella, Matt May, Marti McCuller, Sorcha Moore, Charles F. Munat, Robert Neff, Charles Nevile, Liddy Nevile, Dylan Nicholson, Bruno von Niman, Tim Noonan, Sebastiano Nutarelli, Graham Oliver, Sean B. Palmer, Charu Pandhi, evarshi Pant, Nigel Peck, Anne Pemberton, David Poehlman, Ian Pouncey, Charles Pritchard, Kerstin Probiesch, W Reagan, Adam Victor Reed, Chris Reeve, Chris Ridpath, Lee Roberts, Mark Rogers, Raph de Rooij, Gregory J. Rosmaita, Matthew Ross, Sharron Rush, Joel Sanda, Janina Sajka, Roberto Scano, Gordon Schantz, Tim van Schie, Wolf Schmidt, Stefan Schnabel, Cynthia Shelly, Glenda Sims, John Slatin, Becky Smith, Jared Smith, Andi Snow-Weaver, Neil Soiffer, Mike Squillace, Michael Stenitzer, Diane Stottlemeyer, Christophe Strobbe, Sarah J Swierenga, Jim Thatcher, Terry Thompson, Justin Thorp, David Todd, Mary Utt, Jean Vanderdonckt, Carlos A Velasco, Eric Velleman, Gijs Veyfeyken, Dena Wainwright, Paul Walsch, Daman Wandke, Richard Warren, Elle Waters, Takayuki Watanabe, Gian Wild, David Wooley, Wu Wei, Kenny Zhang, Leona Zumbo.

§ B.3 Enabling funders

This publication has been funded in part with U.S. Federal funds from the Health and Human Services, National Institute on Disability, Independent Living, and Rehabilitation Research (NIDILRR), initially under contract number ED-OSE-10-C-0067, then under contract number HHSP23301500054C, and now under HHS75P00120P00168. The content of this publication does not necessarily reflect the views or policies of the U.S. Department of Health and Human Services or the U.S. Department of Education, nor does mention of trade names, commercial products, or organizations imply endorsement by the U.S. Government.

§ C. References

§ C.1 Informative references

[css3-values]

[*CSS Values and Units Module Level 3*](#). Tab Atkins Jr.; Elika Etemad. W3C. 22 March 2024. W3C Candidate Recommendation. URL: <https://www.w3.org/TR/css-values-3/>

[HTML]

[*HTML Standard*](#). Anne van Kesteren; Domenic Denicola; Dominic Farolino; Ian Hickson; Philip Jägenstedt; Simon Pieters. WHATWG. Living Standard. URL: <https://html.spec.whatwg.org/multipage/>

[ISO_9241-391]

[*Ergonomics of human-system interaction—Part 391: Requirements, analysis and compliance test methods for the reduction of photosensitive seizures*](#). International Standards Organization. URL: <https://www.iso.org/standard/56350.html>

[pointerevents]

[*Pointer Events*](#). Jacob Rossi; Matt Brubeck. W3C. 4 April 2019. W3C Recommendation. URL: <https://www.w3.org/TR/pointerevents/>

[RFC2119]

[*Key words for use in RFCs to Indicate Requirement Levels*](#). S. Bradner. IETF. March 1997. Best Current Practice. URL: <https://www.rfc-editor.org/rfc/rfc2119>

[SRGB]

[*Multimedia systems and equipment - Colour measurement and management - Part 2-1: Colour management - Default RGB colour space - sRGB*](#). IEC. URL: <https://webstore.iec.ch/publication/6169>

[UAAG10]

[*User Agent Accessibility Guidelines 1.0*](#). Ian Jacobs; Jon Gunderson; Eric Hansen. W3C. 17 December 2002. W3C Recommendation. URL: <https://www.w3.org/TR/UAAG10/>

[UNESCO]

[*International Standard Classification of Education*](#). 2011. URL: <https://unesdoc.unesco.org/ark:/48223/pf0000219109>

[WAI-WEBCONTENT]

[*Web Content Accessibility Guidelines 1.0*](#). Wendy Chisholm; Gregg Vanderheiden; Ian Jacobs. W3C. 5 May 1999. W3C Recommendation. URL: <https://www.w3.org/TR/WAI-WEBCONTENT/>

[WCAG20]

[*Web Content Accessibility Guidelines \(WCAG\) 2.0*](#). Ben Caldwell; Michael Cooper; Loretta Guarino Reid; Gregg Vanderheiden et al. W3C. 11 December 2008. W3C Recommendation. URL: <https://www.w3.org/TR/WCAG20/>

[WCAG21]

Web Content Accessibility Guidelines (WCAG) 2.1. Michael Cooper; Andrew Kirkpatrick; Joshue O'Connor; Alastair Campbell. W3C. 21 September 2023. W3C Recommendation. URL: <https://www.w3.org/TR/WCAG21/>

↑



✍️ Writing for Web Accessibility

in [Tips for Getting Started](https://www.w3.org/WAI/tips/) (<https://www.w3.org/WAI/tips/>)

Summary

This page introduces some basic considerations to help you get started writing web content that is more accessible to people with disabilities. These tips are good practice to help you meet Web Content Accessibility Guidelines (WCAG) requirements. Follow the links to the related WCAG requirements, detailed background in the “Understanding” document, guidance from Tutorials, user stories, and more.

Page Contents

- [Provide informative, unique page titles](#)(<#provide-informative-unique-page-titles>)
- [Use headings to convey meaning and structure](#)(<#use-headings-to-convey-meaning-and-structure>)
- [Make link text meaningful](#)(<#make-link-text-meaningful>)
- [Write meaningful text alternatives for images](#)(<#write-meaningful-text-alternatives-for-images>)
- [Create transcripts and captions for multimedia](#)(<#create-transcripts-and-captions-for-multimedia>)
- [Provide clear instructions](#)(<#provide-clear-instructions>)
- [Keep content clear and concise](#)(<#keep-content-clear-and-concise>)

Provide informative, unique page titles

For each web page, provide a short title that describes the page content and distinguishes it from other pages. The page title is often the same as the main heading of the page. Put the unique and most relevant information first; for example, put the name of the page before the name of the organization. For pages that are part of a multi-step process,

include the current step in the page title.

Example: Page Titles

Home page title

Space Teddy Inc. — □ X

Page name followed by organization name

Latest News • Space Teddy Inc. — □ X

Page name including step in a process

Buy Your Bear (Step 1 of 3) • Space Teddy Inc. — □ X

More Information

- WCAG

- [Page Titled 2.4.2 \(https://www.w3.org/WAI/WCAG21/quickref/#page-titled\)](https://www.w3.org/WAI/WCAG21/quickref/#page-titled)
[Understanding 2.4.2 \(https://www.w3.org/WAI/WCAG21/Understanding/page-titled\)](https://www.w3.org/WAI/WCAG21/Understanding/page-titled)

Use headings to convey meaning and structure

Use short headings to group related paragraphs and clearly describe the sections. Good headings provide an outline of the content.

Example: Using headings to organize content

✖ Lack of headings

Headings and Subheadings

HTML elements provide information on structural hierarchy of a document. Using elements correctly will help convey additional meaning to assistive technology. In many cases, doing so will also make your document easier to edit.

For documents longer than three or four paragraphs, headings and subheadings are important for usability and accessibility. They help readers to determine the overall outline of a document and to navigate to specific information of interest.

Headings are classified into levels from one to six. The highest level is "Level 1" and often corresponds to the title of the page or major document sections. Sub-headers proceed through increasing header levels. The lower the number, the smaller and more detailed a section.

Visual readers identify headers by scanning pages for text of a larger size or a different style. Assistive technology users are not able to see these visual changes, so changing the style is not a sufficient cue.

Instead, the headings must be semantically "tagged" so that assistive technology can identify headings. This can be presented to the user as a navigation aid.

This makes adding headings one of the most important tools for a screen reader user so that he or she can learn

 View inline example

✔ Using headings and subheadings

Headings and Subheadings

HTML elements provide information on structural hierarchy of a document. Using elements correctly will help convey additional meaning to assistive technology. In many cases, doing so will also make your document easier to edit.

Purpose of Headings

For documents longer than three or four paragraphs, headings and subheadings are important for usability and accessibility. They help readers to determine the overall outline of a document and to navigate to specific information of interest.

Heading Levels

Headings are classified into levels from one to six. The highest level is "Level 1" and often corresponds to the title of the page or major document section. Sub-headers proceed through increasing header levels. The lower the number, the smaller and more detailed a section.

Meaning vs. Formatting

 View inline example

More Information

• WCAG

- [Headings and Labels 2.4.6 \(https://www.w3.org/WAI/WCAG21/quickref/#headings-and-labels\)](https://www.w3.org/WAI/WCAG21/quickref/#headings-and-labels) ([Understanding 2.4.6 \(https://www.w3.org/WAI/WCAG21/Understanding/headings-and-labels\)](https://www.w3.org/WAI/WCAG21/Understanding/headings-and-labels))
- [Section Headings 2.4.10 \(https://www.w3.org/WAI/WCAG21/quickref/#section-headings\)](https://www.w3.org/WAI/WCAG21/quickref/#section-headings) ([Understanding 2.4.10 \(https://www.w3.org/WAI/WCAG21/Understanding/section-headings\)](https://www.w3.org/WAI/WCAG21/Understanding/section-headings))
- [Info and Relationships 1.3.1 \(https://www.w3.org/WAI/WCAG21/quickref/#info-and-relationships\)](https://www.w3.org/WAI/WCAG21/quickref/#info-and-relationships) ([Understanding 1.3.1 \(https://www.w3.org/WAI/WCAG21/Understanding/info-and-relationships\)](https://www.w3.org/WAI/WCAG21/Understanding/info-and-relationships))

• User Stories

- [Ade, reporter with limited use of his arms \(https://www.w3.org/WAI/people-use-web/user-stories/story-one/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-one/)
- [Ian, data entry clerk with autism \(https://www.w3.org/WAI/people-use-web/user-stories/story-two/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-two/)
- [Lakshmi, senior accountant who is blind \(https://www.w3.org/WAI/people-use-web/user-stories/story-three/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-three/)
- [Stefan, student with attention deficit hyperactivity disorder and dyslexia \(https://www.w3.org/WAI/people-use-web/user-stories/story-eight/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-eight/)

Make link text meaningful

Write link text so that it describes the content of the link target. Avoid using ambiguous link text, such as ‘click here’ or ‘read more’. Indicate relevant information about the link target, such as document type and size, for example, ‘Proposal Documents (RTF, 20MB)’.

Example: Using link text to describe target page

No information

For more information on device independence, [click here](#).

Meaningful information

Read more [about device independence](#).

More Information

- WCAG

- [Link Purpose \(In Context\) 2.4.4 \(https://www.w3.org/WAI/WCAG21/quickref/#link-purpose-in-context\) \(Understanding 2.4.4 \(https://www.w3.org/WAI/WCAG21/Understanding/link-purpose-in-context\)\)](#)
- [Link Purpose \(Link Only\) 2.4.9 \(https://www.w3.org/WAI/WCAG21/quickref/#link-purpose-link-only\) \(Understanding 2.4.9 \(https://www.w3.org/WAI/WCAG21/Understanding/link-purpose-link-only\)\)](#)

- User Stories

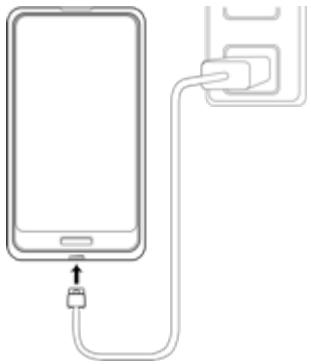
- [Ade, reporter with limited use of his arms \(https://www.w3.org/WAI/people-use-web/user-stories/story-one/\)](#)
- [Ian, data entry clerk with autism \(https://www.w3.org/WAI/people-use-web/user-stories/story-two/\)](#)
- [Lakshmi, senior accountant who is blind \(https://www.w3.org/WAI/people-use-web/user-stories/story-three/\)](#)
- [Stefan, student with attention deficit hyperactivity disorder and dyslexia \(https://www.w3.org/WAI/people-use-web/user-stories/story-eight/\)](#)
- [Elias, retiree with low vision, hand tremor, and mild short-term memory loss \(https://www.w3.org/WAI/people-use-web/user-stories/story-nine/\)](#)

Write meaningful text alternatives for images

For every image, write alternative text that provides the information or function of the image. For purely decorative images, there is no need to write alternative text.

Example: Using alternative text to communicate important information

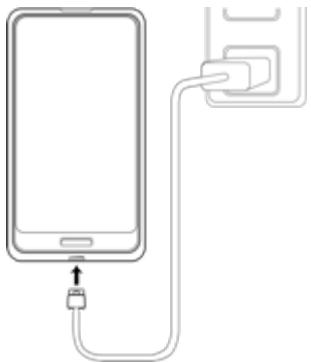
Uninformative



Charging the phone: Connect the phone to a power outlet using the cable and power adaptor provided.

Alternative text for image: "Charging phone"

In informative



Charging the phone: Connect the phone to a power outlet using the cable and power adaptor provided.

Alternative text for image: "Plug cable into the bottom edge of the phone."

Alternative text is usually not visible; it is included in this example just so you can see what it is.

More Information

- **WCAG**

- [Non-text Content 1.1.1 \(https://www.w3.org/WAI/WCAG21/quickref/#non-text-content\)](https://www.w3.org/WAI/WCAG21/quickref/#non-text-content) ([Understanding 1.1.1 \(https://www.w3.org/WAI/WCAG21/Understanding/non-text-content\)](https://www.w3.org/WAI/WCAG21/Understanding/non-text-content))

- **Tutorial**

- [Images \(https://www.w3.org/WAI/tutorials/images/\)](https://www.w3.org/WAI/tutorials/images/)

- **User Story**

- [Lakshmi, senior accountant who is blind \(https://www.w3.org/WAI/people-use-web/user-stories/story-three/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-three/)

Create transcripts and captions for multimedia

For audio-only content, such as a podcast, provide a transcript. For audio and visual content, such as training videos, also provide captions. Include in the transcripts and captions the spoken information and sounds that are important for understanding the content, for example, 'door creaks'. For video transcripts, also include a description of the important visual content, for example 'Athan leaves the room'.

More Information

- [Making Audio and Video Media Accessible \(https://www.w3.org/WAI/media/av/\)](https://www.w3.org/WAI/media/av/)

- **WCAG**

- [Captions \(Prerecorded\) 1.2.2 \(https://www.w3.org/WAI/WCAG21/quickref/#captions-prerecorded\)](https://www.w3.org/WAI/WCAG21/quickref/#captions-prerecorded) ([Understanding 1.2.2 \(https://www.w3.org/WAI/WCAG21/Understanding/captions-prerecorded\)](https://www.w3.org/WAI/WCAG21/Understanding/captions-prerecorded))
- [Audio Description or Media Alternative \(Prerecorded\) 1.2.3 \(https://www.w3.org/WAI/WCAG21/quickref/#audio-description-or-media-alternative-prerecorded\)](https://www.w3.org/WAI/WCAG21/quickref/#audio-description-or-media-alternative-prerecorded) ([Understanding 1.2.3 \(https://www.w3.org/WAI/WCAG21/Understanding/audio-description-or-media-alternative-prerecorded\)](https://www.w3.org/WAI/WCAG21/Understanding/audio-description-or-media-alternative-prerecorded))

- **User Stories**

- [Dhruv, older adult student who is deaf \(https://www.w3.org/WAI/people-use-web/user-stories/story-six/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-six/)
- [Marta, marketing assistant who is deaf and blind \(https://www.w3.org/WAI/\)](https://www.w3.org/WAI/)

[people-use-web/user-stories/story-seven/](#)

Provide clear instructions

Ensure that instructions, guidance, and error messages are clear, easy to understand, and avoid unnecessarily technical language. Describe input requirements, such as date formats.

Example: Instructions communicate what information the user should provide

Password should be at least six characters with at least one number (0-9).

Password

Example: Error indicates what the problem is and, possibly, how to fix it

1.  [The username 'superbear' is already in use.](#)
2.  [The password needs to include at least one number.](#)

More Information

- **WCAG**
 - [Labels or Instructions 3.3.2 \(<https://www.w3.org/WAI/WCAG21/quickref/#labels-or-instructions>\) \(Understanding 3.3.2 \(<https://www.w3.org/WAI/WCAG21/Understanding/labels-or-instructions>\)\)](#)
- **User Stories**
 - [Ian, data entry clerk with autism \(<https://www.w3.org/WAI/people-use-web/user-stories/story-two/>\)](#)
 - [Stefan, student with attention deficit hyperactivity disorder and dyslexia \(<https://www.w3.org/WAI/people-use-web/user-stories/story-eight/>\)](#)
 - [Elias, retiree with low vision, hand tremor, and mild short-term memory loss \(<https://www.w3.org/WAI/people-use-web/user-stories/story-nine/>\)](#)

Keep content clear and concise

Use simple language and formatting, as appropriate for the context.

- Write in short, clear sentences and paragraphs.
- Avoid using unnecessarily complex words and phrases.
- Expand acronyms on first use. For example, Web Content Accessibility Guidelines (WCAG).
- Consider providing a glossary for terms readers may not know.
- Use list formatting as appropriate.
- Consider using images, illustrations, video, audio, and symbols to help clarify meaning.

Example: Making content readable and understandable

Unnecessarily complex

CPP: In the event of a vehicular collision, a company assigned representative will seek to ascertain the extent and cause of damages to property belonging to all parties involved. Once our representative obtains information that allows us to understand the causality, we may or may not assign appropriate monetary compensation. The resulting decision may occasion one of the following options: the claim is not approved and is assigned a rejected status, the status of the claim is ambiguous and will require additional information before further processing can occur, the claim is partially approved and reduced payment is assigned and issued, or claim is fully approved and total claim payment is assigned and issued.

Easier to understand

Claims Processing Procedure (CPP): If you have a car accident, our agent will investigate. Findings will determine any claim payment. This could result in:

- Approved claim - full payment
- Partially approved claim - reduced payment
- Undetermined claim - more information needed
- Rejected claim - no payment



More Information

- **WCAG**

- [Reading Level 3.1.5 \(https://www.w3.org/WAI/WCAG21/quickref/#reading-level\)](https://www.w3.org/WAI/WCAG21/quickref/#reading-level) ([Understanding 3.1.5 \(https://www.w3.org/WAI/WCAG21/Understanding/reading-level\)](https://www.w3.org/WAI/WCAG21/Understanding/reading-level))
- [Unusual Words 3.1.3 \(https://www.w3.org/WAI/WCAG21/quickref/#unusual-words\)](https://www.w3.org/WAI/WCAG21/quickref/#unusual-words) ([Understanding 3.1.3 \(https://www.w3.org/WAI/WCAG21/Understanding/unusual-words\)](https://www.w3.org/WAI/WCAG21/Understanding/unusual-words))
- [Abbreviations 3.1.4 \(https://www.w3.org/WAI/WCAG21/quickref/#abbreviations\)](https://www.w3.org/WAI/WCAG21/quickref/#abbreviations) ([Understanding 3.1.4 \(https://www.w3.org/WAI/WCAG21/Understanding/abbreviations\)](https://www.w3.org/WAI/WCAG21/Understanding/abbreviations))

- **User Stories**

- [Ian, data entry clerk with autism \(https://www.w3.org/WAI/people-use-web/user-stories/story-two/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-two/)
- [Sophie, basketball fan with Down syndrome \(https://www.w3.org/WAI/people-use-web/user-stories/story-five/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-five/)
- [Stefan, student with attention deficit hyperactivity disorder and dyslexia \(https://www.w3.org/WAI/people-use-web/user-stories/story-eight/\)](https://www.w3.org/WAI/people-use-web/user-stories/story-eight/)

bookmark icon Learn More About Accessibility

These tips are a few of the things you need to consider for web accessibility. Additional guidance on writing that meets the accessibility needs of people with cognitive and learning disabilities is in [Use Clear and Understandable Content \(https://www.w3.org/WAI/WCAG2/supplemental/objectives/o3-clear-content/\)](https://www.w3.org/WAI/WCAG2/supplemental/objectives/o3-clear-content/).

The following resources help you learn why accessibility is important, and about

guidelines for making the web more accessible to people with disabilities.

- [Accessibility Introduction](https://www.w3.org/WAI/fundamentals/accessibility-intro/) (<https://www.w3.org/WAI/fundamentals/accessibility-intro/>) — Introduces accessibility and provides links to many helpful resources
- [Accessibility Principles](https://www.w3.org/WAI/fundamentals/accessibility-principles/) (<https://www.w3.org/WAI/fundamentals/accessibility-principles/>) — An introduction to the WCAG requirements
- [How people with disabilities use the web](https://www.w3.org/WAI/people-use-web/) (<https://www.w3.org/WAI/people-use-web/>) — Real-life examples showing the importance of accessibility for people with disabilities
- [How to Meet WCAG \(Quick Reference\)](https://www.w3.org/WAI/WCAG21/quickref/) (<https://www.w3.org/WAI/WCAG21/quickref/>) — customizable reference of all WCAG requirements and techniques

← Previous:
[Overview](#)

(<https://www.w3.org/WAI/tips/>)

(<https://www.w3.org/WAI/tips/designing/>)

Next:
[Tips for Designing](#) →

Updated: 16 July 2024. [Latest changes](https://www.w3.org/WAI/tips/changelog/) (<https://www.w3.org/WAI/tips/changelog/>).

Date: Minor update 16 July 2024. Updated 5 August 2022. First published September 2015.

Editors: [Kevin White](https://www.w3.org/People/kevin) (<https://www.w3.org/People/kevin>), [Shadi Abou-Zahra](https://www.w3.org/People/shadi) (<https://www.w3.org/People/shadi>), and [Shawn Lawton Henry](https://www.w3.org/People/Shawn) (<https://www.w3.org/People/Shawn>). [Acknowledgements](https://www.w3.org/WAI/tips/acknowledgements/) (<https://www.w3.org/WAI/tips/acknowledgements/>).

Developed by the [Education and Outreach Working Group \(EOWG\)](https://www.w3.org/WAI/EO/) (<https://www.w3.org/WAI/EO/>). Developed with support from the [WAI-DEV project](https://www.w3.org/WAI/DEV/) (<https://www.w3.org/WAI/DEV/>), co-funded by the European Commission IST Programme.

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities. Copyright © 2025 World Wide Web Consortium ([W3C®](#)). See [Permission to Use WAI Material](#).



DRAFT Roles Involved in Accessibility

in Accessibility Roles and Responsibilities Mapping (ARRM) (<https://www.w3.org/WAI/planning/arrm/>)

This is an in-progress draft. We welcome your comments via GitHub or email from the links below under [Help improve this page](#)(`#helpimprove`). You are also welcome to join the [ARRM Community Group](#) (<https://www.w3.org/community/arrm/>) to contribute.

Summary

This page describes typical roles that have responsibilities for ensuring accessibility.

Page Contents

- [Introduction](#)(`#introduction`)
- [1. Business Role Group](#)(`#business-role-group`)
 - [Business Analysis](#)(`#business-analysis`)
- [2. Author Role Group](#)(`#author-role-group`)
 - [Content Authoring](#)(`#content-authoring`)
 - [Content Publishing](#)(`#content-publishing`)
- [3. Design Role Group](#)(`#design-role-group`)
 - [User Experience Researcher](#)(`#user-experience-researcher`)
 - [User Experience Design](#)(`#user-experience-design`)
 - [Visual Design](#)(`#visual-design`)
- [4. Development Role Group](#)(`#development-role-group`)
 - [Front-End Development](#)(`#front-end-development`)
 - [Back-End Development](#)(`#back-end-development`)
- [5. Testing Role Group](#)(`#testing-role-group`)
 - [QA Testing Roles \(overall\)](#)(`#qa-testing-roles-overall`)
 - [Automated QA Testing](#)(`#automated-qa-testing`)

- [Manual QA Testing\(↴ #manual-qa-testing\)](#)
- [6. Administration Role Group\(↴ #administration-role-group\)](#)
 - [Product Ownership\(↴ #product-ownership\)](#)
 - [Project Management\(↴ #project-management\)](#)
 - [Governance\(↴ #governance\)](#)
- [Draft Review\(↴ #draft-review\)](#)

Introduction

This page describes typical roles that have responsibilities for ensuring accessibility. These role descriptions are generalized on purpose. An individual may fulfill more than one role.

Each role description includes areas of focus in decision making, decision ownership, notes on the primary tasks of that role, a shortlist of common job titles, and a general description of the accessibility responsibilities of that role.

You are welcome to use this information as is, or change it for your situation.

1. Business Role Group

Writes business requirements and/or initial user stories, are concerned with ensuring that the project delivers the agreed-upon business benefits.

Business Analysis

Business analysts are involved in the design or modification of business systems or IT systems. They interact with business stakeholders and subject matter experts in order to understand their problems and needs. They gather, document and analyze business needs and requirements to help steer the team towards an end result that meets the organization's needs and expectations.

Example job titles for this role:

Business Analyst, Client Sponsor

2. Author Role Group

This role includes content strategy and content authoring, including audio/video production.

Responsible for creating all text presented by the deliverable in all forms (HTML, audio, video). Defines or selects the standards the content should meet and processes for its review and preparation.

Content Authoring

Content creation is often used in marketing, but can also be a task assigned to a role within a product team. Content creation involves defining a content strategy, the writing or creation of the content or media for a product. The person who authors the content is responsible for making sure that content is accessible to people with disabilities.

Content creation extends to roles specific audio and video media production. These begin with the writing of scripts but can extend to the production of the media files or support of live streaming content.

Key deliverable examples:

Body copy, managed content, scripts, taxonomies, writing guidelines, media files including PDF, audio and video

Tasks include:

Content authoring, media and documentation creation, content strategies definition

Example job titles for this role:

Content Strategist, Content Creator, Content Designer, Content Author, Digital Copywriter, UX Writer, Content Producer, Technical Writer, Script Writer, Video Producer, Podcast Host

Content Publishing

Description (not a definition, needs a more polished “Definition” statement): Team members who are not front-end web developers that are tasked with preparing content for publishing to websites, products, applications, etc. These team members do not author content, instead they work with the content within specific business enterprise systems or software tools and apply edits to material generated by content authors.

Content Publishers may perform some tasks that are a part of the front end web developer skillset, however they are not web developers and are often applying this markup using proprietary tools and widget.

Key deliverable examples:

Content editing and formatting, metadata and SEO optimization, content publishing schedule, cross-platform compatibility, and quality assurance and testing

Tasks include:

Advanced tagging in a PDF document, adjusting timing of captions, creating and editing transcripts, conversion to ePUB documentation, applying tags through a proprietary software to generate heading structure, SSML

Example job titles for this role:

Content Strategist, Content Producer, Content Publisher, Publishing Technology Specialist, Content Designer, Video Editor, Accessible Materials Specialist, Information Developer, Content Developer, Publishing Implementation Manager

3. Design Role Group

This group includes user research, user experience (UX), and presentation decisions.

Some roles within this group define the user experience of a product, designing its behaviors and interactions with the end user and/or their assistive technologies. Some roles provide the general “look and feel” of the products and features, covering presentation, fonts and colors. Others outline the functionality of features, and their operation through assistive technologies (ATs). They translate input from business roles into user stories, requirements, specifications, documentation, and guidelines used by other roles (mainly Development) to build the finished product.

This includes applying user research that has been conducted and an understanding of the intended audiences for a quality experience.

User Experience Researcher

UX Researchers conduct both exploratory and generative research with end users (e.g. user interviews, ethnographic research, etc.) to gather their feedback. The responsibility of this role is to continuously feed user feedback to those creating products. They test existing concepts and summarize user insights to inform the assets that will be built by the roles in

UX Design, Visual Design, Front-end Development and Content Authoring.

Key deliverable examples:

Findings from moderated and unmoderated versions of user research studies, usability testing and user interviews (i.e. a moderator or facilitator was present during the test, or the end user was alone)

Tasks include:

Conducting user research with a partial prototype, concept, or completed product

Example job titles for this role:

User Researcher, UXR (User Experience Researcher), Usability Analyst

User Experience Design

UX Designers can potentially cover numerous related areas, from conceptualizing the user journey to partial front-end development. For the purposes of this resource, UX Design is defined by its core responsibilities, such as information architecture, creating wireframes (low fidelity screen mockups), and creating prototypes that define interactions.

Key deliverable examples:

User journeys, wireframes, prototypes, interaction guidelines, information architecture

Tasks include:

User workflow / process maps, designing user experiences, user task and workflow mapping, creating and maintaining user personas

Example job titles for this role:

User Experience (UX) Designer, Product Designer, Web Designer, Service Designer

Visual Design

Visual Designers focus largely on the look and feel of a product, as an end user would experience it, visually or otherwise. This includes specifying original design of interface elements and layout, choosing fonts and colors, and more. While UX design is focused on how something works, visual design is focused on how it looks and feels.

Key deliverable examples:

Style guides, page comps, design mockups, image files

Tasks include:

Visual styling, logos and branding, animation, and iconography design

Example job titles for this role:

Visual Designer, User Interface (UI) Designer, Interaction Designer, Graphic Designer

4. Development Role Group

Oversees the creation, coding and delivery of the product based upon the requirements provided. Responsible for all user-facing and supporting systems, along with all related infrastructure selection, setup and deployment.

Front-End Development

Front-end development typically builds the parts of a product that will be interacted with by the user - specifically, the user interface. For the purpose of this resource, front-end development refers to the implementation or codification of the design in functional templates for a product using technologies such as HTML, CSS and JavaScript.

Key deliverable examples:

HTML and CSS files, client-side scripting, JavaScript libraries and frameworks

Tasks include:

Pattern libraries and prototypes, template functionalities, semantically-rich HTML document structures and widgets, use and adapt frameworks and content management systems

Example job titles for this role:

Front-End Developer, Web Developer, Full-Stack Developer, UI/UX Developer, JavaScript Developer, UI/UX Engineer

Back-End Development

Back-end developers have a smaller, more indirect involvement with accessibility work, but still play a critical role in delivering accessible products, as the underlying product architecture can inform accessibility solutions. It's important for back-end developers to be involved in accessibility discussions so that any potential issues stemming from how the database is organized can be caught and fixed, or better yet, avoided altogether.

Example job titles for this role:

Back-End Developer, Middleware Developer, Database Architect, Data Engineer

5. Testing Role Group

QA Testers run automated test frameworks or manually test products to confirm correct operation based upon provided requirement.

QA Testing Roles (overall)

Quality Assurance (QA) Testers typically don't contribute directly to the design and development phases of a product. They may have the opportunity to review and sign off on designs before they are implemented. The main accessibility role of a QA Tester in the accessibility lifecycle is to understand the accessibility requirements that exist and to run tests to ensure the product or feature conforms to those requirements. The testing roles have been split between automated and manual, but a number of QA professionals will do both.

Automated QA Testing

Quality Assurance (QA) automation frameworks typically run against products in order to test features and functionality that would otherwise fall to a manual QA tester. For the purpose of this resource, a QA Tester performing automated tests is responsible for running tests within an automation framework that covers accessibility features and requirements. This can be accomplished either through automation of functional tests, and/or the inclusion of an accessibility testing library within the framework. QA Testers responsible for automated testing may also typically run automated accessibility testing tools, such as browser extensions or add-ons. It is expected that automated tools will uncover about 30 to 35% of potential accessibility issues on a screen. The rest will be identified through the team's manual accessibility testing methodology.

Example job titles for this role:

QA Automation Engineer, Quality Engineer, Automation Engineer

Manual QA Testing

Quality Assurance Testers responsible for manual testing will typically handle the testing that cannot be covered through the use of automated tools. They will typically be resources that are more knowledgeable about accessibility, with a deeper understanding of the requirements, and some experience operating assistive technologies for testing, such as

screen readers. They will typically run a series of test cases to validate the degree of inclusion of the components of a screen and will be charged with making sure that the overall user experience is positive for people with disabilities. It is expected that manual testing will build on top of an automated testing process, in order to cover the remaining 65 to 70% of potential accessibility issues on any given screen.

Example job titles for this role:

QA Analyst, QA Specialist, User Acceptance Tester, Functional Tester

6. Administration Role Group

The roles in this section cover managing the product and project, as well as other bureaucratic functions of the broader organization that often have a larger mandate than any individual project. Most administrative roles, as defined in this resource, have very little, if anything, to do with the design, the implementation, or the testing of accessibility principles to create more inclusive applications and websites. These roles, however, are still instrumental in ensuring that the team members who are actively taking part in making content accessible and conformant with WCAG can be successful at doing so. This begins with project managers and product owners working hand in hand with the design, development and testing teams, but also other governance roles contributing to steering the organization's culture in a direction that is aligned with the goals pursued by accessibility guidelines.

Product Ownership

Product owners own individual products and define their features and are key to defining the importance of accessibility on a project. It is their job to make sure that products are built and delivered in a way that meets business needs and user needs. They should have some basic understanding of the accessibility implications of the UI requirements they request, such as cost and required infrastructure. But, as with features in general, they delegate design and implementation decisions to other roles. As a result, they are typically not directly involved and do not have ownership in the ARRM model.

Example job titles for this role:

Product Manager, Product Owner, Release Manager, Business Owner

Project Management

Project managers are primarily responsible for keeping everything about the process of building a product organized and on track. They have little decision-making power directly impacting accessibility. The person managing the project should make sure that accessibility is built into estimates, user stories (if Agile), and requirements documents. In smaller teams, the roles of product owners and project managers often overlap.

Example job titles for this role:

Project Manager, Scrum Master (Agile), Team Lead

Governance

Unlike most other administrative roles, governance is not part of an individual project team. Their mandate focuses on the big picture, often enterprise-wide initiatives. Most importantly to individual projects, governance roles are deliberative (often committee-based) and are not bound to the deadlines of a singular project without outside, executive influence to encourage higher-velocity decision making.

Example job titles for this role:

Legal, procurement, conformance, administrative, governance, branding

Draft Review

We welcome your input

We are particularly interested in feedback to better cover roles in a wide range of organizations throughout the world. For example:

- Do these role groups and role descriptions work for most project teams?
- Do you have suggestions for other examples of key deliverables, tasks, or job titles that might be important to include?

We welcome your input by email or GitHub from the links below [Help improve this page](#)([#helpimprove](#))

Updated: 6 March 2025.

Editors and contributors: See [Acknowledgements](#) (<https://www.w3.org/WAI/planning/arm/acknowledgements/>).

Developed through the [Accessibility Roles and Responsibilities Mapping \(ARRM\) Community Group](#) (<https://www.w3.org/community/arm/>) at W3C. Initially developed with the [Accessibility Education and Outreach Working Group \(EOWG\)](#) (<https://www.w3.org/WAI/about/>)

[groups/eowg/](#)).

© This work is licensed under a [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/) (<https://creativecommons.org/licenses/by/4.0/>).

W3C Web Accessibility Initiative (WAI)

Strategies, standards, and supporting resources to make the Web accessible to people with disabilities.

Copyright © 2025 World Wide Web Consortium ([W3C](#)[®]).
See [Permission to Use WAI Material](#).



DRAFT Tasks Involved in Accessibility

In Accessibility Roles and Responsibilities Mapping (ARRM) (<https://www.w3.org/WAI/planning/arrm/>)

This is an in-progress draft. We welcome your comments via GitHub or email from the links below under [Help improve this page](#) ([#helpimprove](#)). You are also welcome to join the [ARRM Community Group](#) (<https://www.w3.org/community/arrm/>) to contribute.

Summary

This page provides an approach to addressing accessibility requirements in WCAG as tasks for specific roles.

Page Contents

- [Introduction](#) ([#introduction](#))
- [Images and Graphs](#) ([#images-and-graphs](#))
- [Semantic Structure](#) ([#semantic-structure](#))
- [Input Modalities](#) ([#input-modalities](#))
- [Form Interactions](#) ([#form-interactions](#))
- [CSS and Presentation](#) ([#css-and-presentation](#))
- [Navigation](#) ([#navigation](#))
- [Data Tables](#) ([#data-tables](#))
- [Animation and Movement](#) ([#animation-and-movement](#))
- [Static Content](#) ([#static-content](#))
- [Dynamic Interactions](#) ([#dynamic-interactions](#))

Introduction

⚠ Important

This is not a definitive or complete list of accessibility tasks.

These tasks offer a starting point for a role-based approach to addressing Web Content Accessibility Guidelines ([WCAG \(<https://www.w3.org/WAI/standards-guidelines/wcag/>\)](#)) 2.1 success criteria (SC). A later iteration will include the success criteria added in WCAG 2.2.

This information is also available to download as a [single CSV file](#) (<https://www.w3.org/WAI/content-assets/wai-arrm/arrm-all-tasks.csv>).

Images and Graphs

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
IMG-001	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Informative alternate text is provided for images (i.e. not "spacer" or image file name).	Content Authoring	User Experience (UX) Design	none
IMG-002	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Informative images are described with a clear and meaningful text equivalent (alt attribute or other equivalent means).	Content Authoring	none	none
IMG-003	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Purely decorative images are provided with null alt attribute values (or other equivalent means).	Front-End Development	Content Authoring	none
IMG-004	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Null alt attribute values are used for images that are already described in text in adjacent page content.	Front-End Development	Content Authoring	none
IMG-005	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Adjacent linked images and text links pointing the same URL are combined into single links.	Front-End Development	User Experience (UX) Design	none
IMG-006	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Alt text used for images of text include all relevant text found in the image.	Content Authoring	none	none
IMG-007	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Informative images are marked up as foreground images, and not embedded as part of the CSS.	Front-End Development	Content Authoring	none
IMG-008	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	The purpose or function of complex images is accurately described in text.	Content Authoring	User Experience (UX) Design	none

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
IMG-009	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	The purpose or function of complex images is conveyed using a text description, via an alt attribute.	Front-End Development	Content Authoring	User Experience (UX) Design
IMG-010	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	The full explanation of complex images is accurately described in text.	Content Authoring	none	none
IMG-011	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	A mechanism that conveys the way through which the full explanation of complex images is defined.	User Experience (UX) Design	none	none
IMG-012	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	The full explanation of complex images is provided through the longdesc attribute (or other equivalent means).	Front-End Development	none	none
IMG-013	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Images primarily conveying function use alternative text to describe their purpose, rather than what they look like.	Content Authoring	User Experience (UX) Design	none
IMG-014	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Text alternatives of static and linked images do not replicate any information that is already being conveyed by screen reader technology.	Content Authoring	Front End Development	none
IMG-015	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Text alternatives of dynamically updated images are simultaneously updated as the images change.	Front-End Development	User Experience (UX) Design	Content Authoring
IMG-016	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Alternate means of accessing CAPTCHA information are provided, such as audio CAPTCHA, logical question, or other equivalent means.	Business Analyst	User Experience (UX) Design	none
IMG-017	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Images which do not convey information are defined as decorative.	Content Authoring	none	none
IMG-018	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Charts, graphs, infographics and other visual representations of information don't rely on color alone to convey information.	Visual Design	User Experience (UX) Design	none
IMG-019	1.4.5 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/images-of-text)	Text content that conveys information is not part of images.	User Experience (UX) Design	Visual Design	Content Authoring
IMG-020	1.4.5 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/images-of-text)	Text that is placed on top of an image is handled semantically through HTML and CSS instead.	Front-End Development	User Experience (UX) Design	none
IMG-021	1.4.5 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/images-of-text)	Unless a particular presentation of text is essential to the information being conveyed, all images that contain text are only used for purely decorative purposes.	Visual Design	User Experience (UX) Design	none
IMG-022	1.4.9 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/images-of-text-no-exception)	With the exception of logos, all images that contain text are only used for purely decorative purposes.	Visual Design	Content Authoring	none

Semantic Structure

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
SEM-001	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Informative content is provided through HTML markup.	Front-End Development	User Experience (UX) Design Content Authoring	none
SEM-002	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	HTML elements are used according to the HTML specification.	Front-End Development	User Experience (UX) Design	none

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
SEM-003	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Navigation groupings are marked up using HTML list or nav elements (or other equivalent means).	Front-End Development	User Experience (UX) Design	none
SEM-004	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Header sections are marked up using HTML header elements (or other equivalent means).	Front-End Development	User Experience (UX) Design	none
SEM-005	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	The main section of a page is marked up using a HTML main element (or other equivalent means).	Front-End Development	User Experience (UX) Design	none
SEM-006	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	The footer of the page is marked up using a HTML footer element (or other equivalent means).	Front-End Development	User Experience (UX) Design	none
SEM-007	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Content that is complementary to the main section is marked up using HTML aside elements (or other equivalent means).	Front-End Development	User Experience (UX) Design	none
SEM-008	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	HTML elements are used based on the semantics they provide, not based on what they look like.	Front-End Development	User Experience (UX) Design	none
SEM-009	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Decorative elements are embedded through the CSS presentation layer (or other equivalent means).	Front-End Development	Content Authoring	none
SEM-010	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	All scripting behaviors are handled through JavaScript.	Front-End Development	none	none
SEM-011	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Elements that act as headings are marked up as such.	Front-End Development	User Experience (UX) Design	none
SEM-012	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Headings follow a hierarchical sequence without skipping any levels.	Content Authoring	User Experience (UX) Design	none
SEM-013	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Headings are marked up using h1 to h6 elements or other equivalent means.	Front-End Development	User Experience (UX) Design Content Authoring	none
SEM-014	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Page contains a level 1 heading that describes the page content.	Front-End Development	User Experience (UX) Design Content Authoring	none
SEM-015	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Whitespace or pre-formatted text is not used to render content to appear as multiple columns or tabular information.	Front-End Development	none	none
SEM-016	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Use of native, semantic HTML elements are prioritized over other methods.	Front-End Development	User Experience (UX) Design	none
SEM-017	1.3.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/meaningful-sequence)	The source code (or DOM) order matches the suggested visual order of the design.	Front-End Development	User Experience (UX) Design	none
SEM-018	2.4.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/bypass-blocks)	iFrames displaying content are provided with a clear, informative title attribute value.	User Experience (UX) Design	Content Authoring	none
SEM-019	2.4.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/page-titled)	Page title text matches the level 1 heading text.	Content Authoring	none	none
SEM-020	2.4.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/page-titled)	Pages are described using unique and descriptive page title values.	Content Authoring	none	none
SEM-021	2.4.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/focus-order)	The tab order logically and predictably follows the expected interaction order of the visual design.	Front-End Development	User Experience (UX) Design	none
SEM-022	2.4.6 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/headings-and-labels)	Heading text meaningfully describes the content's topic or purpose.	Content Authoring	none	none
SEM-023	2.4.6 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/headings-and-labels)	The main heading of the page describes the content of the page.	Content Authoring	none	none
SEM-024	4.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/parsing)	Source code is properly nested, according to its specification.	Front-End Development	none	none
SEM-025	4.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/parsing)	Source code elements are provided with complete start and end tags (or are self-closed) according to	Front-End Development	none	none

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
		specification.			
SEM-026	4.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/parsing)	ID attribute values assigned to elements are unique.	Front-End Development	none	none
SEM-027	4.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/parsing)	Elements do not contain duplicate attributes.	Front-End Development	none	none
SEM-028	4.1.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/name-role-value)	iFrames are given a title that describes their content or purpose.	User Experience (UX) Design	Content Authoring	none
SEM-029	4.1.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/name-role-value)	iFrames are implemented using title attribute values that describe their content or purpose.	Front-End Development	Content Authoring	none

Input Modalities

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
INP-001	1.4.13 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/content-on-hover-or-focus)	Additional content triggered by focus or hover that covers other information can be dismissed by the user.	User Experience (UX) Design	Front-End Development	none
INP-002	1.4.13 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/content-on-hover-or-focus)	Additional content triggered by pointer hover does not disappear when trying to move the pointer over it.	User Experience (UX) Design	Front-End Development	none
INP-003	1.4.13 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/content-on-hover-or-focus)	Additional content triggered by hover or focus stays visible until purposefully un-triggered, dismissed, or is no longer valid.	User Experience (UX) Design	Front-End Development	none
INP-004	2.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/keyboard)	All actionable elements can be reached, using only the keyboard.	Front-End Development	User Experience (UX) Design	none
INP-005	2.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/keyboard)	All active elements can be triggered, using only the keyboard.	Front-End Development	User Experience (UX) Design	none
INP-006	2.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/keyboard)	Device-specific programmatic event handlers are not used as the only way to trigger interactions.	Front-End Development	none	none
INP-007	2.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/keyboard)	Behaviors for hover and focus states are planned and included with the design assets.	User Experience (UX) Design	Visual Design	none
INP-008	2.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/keyboard)	Keyboard focus states are planned for every active element.	User Experience (UX) Design	Visual Design	none
INP-009	2.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/keyboard)	Keyboard focus is not applied to non-active or static elements.	Front-End Development	User Experience (UX) Design	none
INP-010	2.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/keyboard)	Custom active elements replicate all inherent keyboard behaviors of native active HTML elements.	Front-End Development	User Experience (UX) Design	none
INP-011	2.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/keyboard)	Non-interactive elements are not assigned JavaScript event handlers.	Front-End Development	none	none
INP-012	2.1.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/no-keyboard-trap)	Users can navigate away from all active elements, using only the keyboard.	Front-End Development	none	none
INP-013	2.1.3 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/keyboard-no-exception)	All actionable elements can be reached using only the keyboard without requiring specific timings for individual keystrokes.	User Experience (UX) Design	Front-End Development	none
INP-014	2.1.4 (A) (https://www.w3.org/WAI/WCAG22/Understanding/character-key-shortcuts)	Single-key keyboard shortcuts can be disabled or remapped, unless they are only active on keyboard focus.	User Experience (UX) Design	none	none
INP-015	2.4.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/focus-order)	Users can tab through active elements in an order that reflects the intended interaction order of the design.	Front-End Development	User Experience (UX) Design Visual Design	none

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
INP-016	2.4.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/focus-order)	Tabindex attributes are not assigned positive integer values.	Front-End Development	none	none
INP-017	2.4.7 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/focus-visible)	Every element that receives keyboard focus is designed to display a visible focus indicator.	Visual Design	User Experience (UX) Design	none
INP-018	2.4.7 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/focus-visible)	Every element that receives keyboard focus displays a visible focus indicator.	Front-End Development	Visual Design	none
INP-019	2.5.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/pointer-gestures)	Non-essential multipoint, path- or gesture-based functions have single-pointer alternatives.	User Experience (UX) Design	none	none
INP-020	2.5.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/pointer-cancellation)	Non-essential single pointer functionality is not triggered on down events, unless the functionality can be canceled, undone, or reversed.	User Experience (UX) Design	Front-End Development	none
INP-021	2.5.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/label-in-name)	Text or images of text that is usually included in a user interface control is part of its accessible name.	Content Authoring	User Experience (UX) Design	none
INP-022	2.5.4 (A) (https://www.w3.org/WAI/WCAG22/Understanding/motion-actuation)	All non-essential motion-related functionalities have alternative user interface controls that allow for equivalent actions.	User Experience (UX) Design	none	none
INP-023	2.5.4 (A) (https://www.w3.org/WAI/WCAG22/Understanding/motion-actuation)	All non-essential motion-related functionality can be turned off to prevent accidental actuation.	User Experience (UX) Design	none	none
INP-024	3.2.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/on-input)	Keyboard focus does not move automatically from one form control to the next.	User Experience (UX) Design	Front End Development	none

Form Interactions

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
FRM-001	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Text labels are marked up using the label element or other equivalent means.	Front-End Development	none	none
FRM-002	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Text labels and form controls are programmatically associated (using the FOR and ID attributes, or equivalent means).	Front-End Development	User Experience (UX) Design	none
FRM-003	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Submit buttons in forms rely on a submit input type, a button element, or other equivalent means.	Front-End Development	none	none
FRM-004	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Related form controls are programmatically associated using fieldset and legend elements or other equivalent means.	Front-End Development	User Experience (UX) Design	none
FRM-005	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Long option lists in select elements are grouped semantically (using the optgroup element, or other equivalent means).	Front-End Development	User Experience (UX) Design	none
FRM-006	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Common group label text is informative, meaningful and provides context for the grouping.	Content Authoring	User Experience (UX) Design	none
FRM-007	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Instructions and messages are programmatically conveyed to assistive technologies.	Front-End Development	User Experience (UX) Design	none

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
FRM-008	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Use of native HTML controls are prioritized over other methods.	Front-End Development	none	none
FRM-009	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Required fields are programmatically conveyed as such to assistive technologies.	Front-End Development	User Experience (UX) Design	none
FRM-010	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Instructions on how to use forms are programmatically conveyed to assistive technologies.	Front-End Development	User Experience (UX) Design	none
FRM-011	1.3.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/meaningful-sequence)	Information that is relevant to a form does not appear after the submit button.	User Experience (UX) Design	none	none
FRM-012	1.3.5 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/identify-input-purpose)	Data entry fields are designed to autofill previously entered user information when that information is available.	User Experience (UX) Design	Front-End Development	none
FRM-013	1.3.5 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/identify-input-purpose)	Data entry fields using the autocomplete attribute for previously entered information are set to the appropriate value.	Front-End Development	User Experience (UX) Design	none
FRM-014	1.3.6 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/identify-purpose)	The purpose of user interface components, icons, and regions is implemented using markup languages, in a way that can be programmatically determined.	Front-End Development	User Experience (UX) Design	none
FRM-015	2.4.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/focus-order)	Keyboard focus is dynamically moved to the error message when errors are returned.	Front-End Development	User Experience (UX) Design	none
FRM-016	2.4.6 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/headings-and-labels)	The purpose of the form control is clearly described in text.	Content Authoring	User Experience (UX) Design	none
FRM-017	2.5.6 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/concurrent-input-mechanisms)	The content is designed in a way that does not restrict input modalities available.	User Experience (UX) Design	Front-End Development	none
FRM-018	3.2.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/on-focus)	Changes of context are not initiated automatically as user interface components receive focus.	User Experience (UX) Design	Front-End Development	none
FRM-019	3.2.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/on-input)	Event handlers are not used to automatically trigger a change of context upon input that would otherwise require explicit user action unless previously communicated.	Front-End Development	User Experience (UX) Design	none
FRM-020	3.2.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/on-input)	Form interactions are not designed to include automatic changes of context upon input that would otherwise require explicit user action unless previously communicated.	User Experience (UX) Design	Front-End Development	none
FRM-021	3.2.4 (A) (https://www.w3.org/WAI/WCAG22/Understanding/consistent-identification)	Error messages and alerts are visually displayed across the site in a consistent manner.	User Experience (UX) Design	Visual Design	none
FRM-022	3.2.4 (A) (https://www.w3.org/WAI/WCAG22/Understanding/consistent-identification)	Visual indicators are presented to support error messages when errors are returned.	Visual Design	User Experience (UX) Design	none

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
FRM-023	3.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/error-identification)	Inline error messages are displayed next to their related form controls.	User Experience (UX) Design	none	none
FRM-024	3.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/error-identification)	Error messages are grouped as a list at the top of the form.	User Experience (UX) Design	Visual Design	none
FRM-025	3.3.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/labels-or-instructions)	Radio buttons and checkbox labels are positioned to the right of their respective form controls (for left-to-right languages).	User Experience (UX) Design	none	none
FRM-026	3.3.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/labels-or-instructions)	Related form controls and their respective labels are grouped together visually.	User Experience (UX) Design	Visual Design	none
FRM-027	3.3.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/labels-or-instructions)	Instructions are in close visual proximity to their related controls.	User Experience (UX) Design	none	none
FRM-028	3.3.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/labels-or-instructions)	Form controls are coded to have persistent visual labels.	Front-End Development	User Experience (UX) Design	Visual Design
FRM-029	3.3.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/labels-or-instructions)	Form controls are designed to have persistent visual labels.	User Experience (UX) Design	Visual Design	none
FRM-030	3.3.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/labels-or-instructions)	Clear text-based instructions are provided on how to use the form controls.	Content Authoring	User Experience (UX) Design	Visual Design
FRM-031	3.3.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/labels-or-instructions)	Instructions provided in forms are displayed in a clear and unambiguous way.	Visual Design	User Experience (UX) Design	none
FRM-032	3.3.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/labels-or-instructions)	Placeholder text is not used in lieu of regular text label elements.	User Experience (UX) Design	none	none
FRM-033	3.3.3 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/error-suggestion)	Error messages returned provide clear instructions on how to fix them.	Content Authoring	User Experience (UX) Design	none
FRM-034	3.3.3 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/error-suggestion)	Instructions intending to prevent errors are provided in text and available.	Front-End Development	User Experience (UX) Design	Content Authoring
FRM-035	3.3.3 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/error-suggestion)	Text-based instructions are provided to help users correct errors.	User Experience (UX) Design	Content Authoring Visual Design	none
FRM-036	3.3.4 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/error-prevention-legal-financial-data)	Users are provided with means to prevent and correct form errors when legal, financial, or data information is involved.	User Experience (UX) Design	Business	none
FRM-037	3.3.4 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/error-prevention-legal-financial-data)	When legal, financial, or data information is involved, confirmation screens are provided prior to any final form submission.	User Experience (UX) Design	Business	Front-End Development
FRM-038	3.3.5 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/help)	Context-sensitive help text is available.	User Experience (UX) Design	Content Authoring	none
FRM-039	3.3.6 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/error-prevention-all)	Users are provided with means to prevent and correct form errors.	User Experience (UX) Design	Business	none

CSS and Presentation

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
CSS-001	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Icon fonts used to convey information are provided with a text equivalent.	User Experience (UX) Design	Content Authoring	none
CSS-002	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	The meaning of icon fonts is determined programmatically using the aria-label attribute (or other equivalent means).	Front-End Development	Content Authoring	none
CSS-003	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Icon fonts whose default accessible name do not represent what the icon is, are overwritten using an aria-hidden attribute (or other equivalent means).	Front-End Development	none	none
CSS-004	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Background images identified as decorative are implemented as such.	Front-End Development	Visual Design	none
CSS-005	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	CSS pseudo-selectors such as :before and :after are not used to integrate informative content.	Front-End Development	none	none
CSS-006	1.3.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/sensory-characteristics)	Shape and location are never used as the only way to convey information and relationships between page components.	Visual Design	User Experience (UX) Design	Content Authoring
CSS-007	1.3.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/sensory-characteristics)	Users relying on High Contrast themes don't lose information as a result of doing so.	Front-End Development	Visual Design	none
CSS-008	1.3.4 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/orientation)	Content is viewable in both portrait and landscape orientation, unless a particular orientation is essential.	User Experience (UX) Design	Visual Design	none
CSS-009	1.4.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/use-of-color)	Color is never used as the only way to convey information, context, indicate selection or the presence of errors.	Visual Design	User Experience (UX) Design	Content Authoring
CSS-010	1.4.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/use-of-color)	For a link that is not underlined and in a paragraph text, its text color is sufficiently contrasted by providing a luminosity ratio of at least 3:1 against its surrounding text.	Visual Design	none	none
CSS-011	1.4.3 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/contrast-minimum)	Regular-sized text is sufficiently contrasted against its background, with a luminosity ratio of at least 4.5:1.	Visual Design	none	none
CSS-012	1.4.3 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/contrast-minimum)	Large-sized text is sufficiently contrasted against its background, with a luminosity ratio of at least 3:1.	Visual Design	none	none
CSS-013	1.4.4 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/resize-text)	Users can resize the text on the page up to 200% without any loss of content or functionality.	Visual Design	Front End Development	User Experience (UX) Design
CSS-014	1.4.4 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/resize-text)	CSS techniques are used to ensure that content doesn't overflow, overlap or get truncated as a result of increasing the text size.	Front-End Development	Visual Design	User Experience (UX) Design
CSS-015	1.4.5 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/images-of-text)	CSS background sprites don't include images of text, unless equivalent text alternatives are also provided as part of the HTML.	Front-End Development	Content Authoring Visual Design	none
CSS-016	1.4.6 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/contrast-enhanced)	Regular-sized text is sufficiently contrasted against its background, with a luminosity ratio of at least 7:1.	Visual Design	none	none
CSS-017	1.4.6 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/contrast-enhanced)	Large-sized text is sufficiently contrasted against its background, with a luminosity ratio of at least 4.5:1.	Visual Design	none	none
CSS-018	1.4.10 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/reflow)	The design makes it possible for end users to enlarge the text so that it reflows into a single column without any loss of information or functionality.	Visual Design	User Experience (UX) Design	none
CSS-019	1.4.10 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/reflow)	Displaying content on narrower screens or magnifying it does not cause multidirectional scrolling.	Visual Design	Front-End Development	User Experience (UX) Design
CSS-020	1.4.11 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-contrast)	All non-text user interface components and graphical objects are sufficiently contrasted against their background,	Visual Design	none	none

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
		with a luminosity ratio of at least 3:1.			
CSS-021	1.4.12 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/text-spacing)	Adjusting spacing between letters, words, or paragraphs, or adjusting line height does not cause a loss of content or functionality.	Visual Design	Front-End Development	none
CSS-022	2.4.7 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/focus-visible)	The CSS outline property of objects that receive keyboard focus are not set to zero or none.	Front-End Development	Visual Design	none
CSS-023	2.5.5 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/target-size-enhanced)	Unless the interactive element is part of a sentence or a block of text, its size must be at least 44 x 44 pixels.	Visual Design	none	none

Navigation

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
NAV-001	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Active objects and other calls to action are visually identifiable as such.	Visual Design	none	none
NAV-002	1.3.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/sensory-characteristics)	Instructions are conveyed through more than shape, size, position, or sound alone.	Content Authoring	User Experience (UX) Design	none
NAV-003	1.4.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/use-of-color)	Additional visual and/or textual cues are provided when color is used to convey information.	User Experience (UX) Design	Visual Design Content Authoring	none
NAV-004	2.2.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/timing-adjustable)	Users are notified when time limits are about to expire.	User Experience (UX) Design	Business	none
NAV-005	2.2.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/timing-adjustable)	Options to extend, or even turn off time limits are provided.	User Experience (UX) Design	Business	none
NAV-006	2.2.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/pause-stop-hide)	Users are given means to pause, stop or hide content that automatically updates.	User Experience (UX) Design	none	none
NAV-007	2.2.4 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/interruptions)	Users are provided with means to turn off all updates, except in case of emergencies.	User Experience (UX) Design	none	none
NAV-008	2.2.5 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/re-authenticating)	Users are provided with means to re-authenticate sessions without loss of data.	User Experience (UX) Design	none	none
NAV-009	2.4.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/bypass-blocks)	Users can bypass blocks of content using skip links or similar mechanisms.	User Experience (UX) Design	none	none
NAV-010	2.4.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/bypass-blocks)	Skip links (and similar mechanisms) for main content and navigation are provided at the most effective location in the interface (such as the very first tab stop).	User Experience (UX) Design	none	none
NAV-011	2.4.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/bypass-blocks)	The functionality and expected destination of skip links and similar mechanisms is clearly defined.	User Experience (UX) Design	Content Authoring	none
NAV-012	2.4.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/bypass-blocks)	Skip links and similar mechanisms point to the expected destination.	Front-End Development	User Experience (UX) Design	none
NAV-013	2.4.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/focus-order)	All active elements receive focus in a logical and predictable order that is prescribed by the visual presentation.	Front-End Development	User Experience (UX) Design	User Experience (UX) Design, Visual Design
NAV-014	2.4.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/focus-order)	A logical and predictable focus order is defined for complex interactions.	User Experience (UX) Design	none	none
NAV-015	2.4.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/focus-order)	Objects that are not actionable are not part of the tabbing order.	Front-End Development	User Experience (UX) Design	none
NAV-016	2.4.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/focus-order)	Focus is sent back to the initiating	Front-End	User Experience	none

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
		point when modal dialogs and controls are dismissed.	Development	(UX) Design	
NAV-017	2.4.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/focus-order)	Event handlers do not unexpectedly send the focus somewhere else on the page.	Front-End Development	none	none
NAV-018	2.4.4 (A) (https://www.w3.org/WAI/WCAG22/Understanding/link-purpose-in-context)	Link text and alternate text for images, when used as links, describe the destination or purpose of the link.	Content Authoring	none	none
NAV-019	2.4.4 (A) (https://www.w3.org/WAI/WCAG22/Understanding/link-purpose-in-context)	Links are marked up using the anchor element and have a valid href attribute value (or use other equivalent means).	Front-End Development	none	none
NAV-020	2.4.5 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/multiple-ways)	Multiple mechanisms are provided for wayfinding, such as navigation menus, breadcrumbs, search features, site map, progress bar, steps, etc.	User Experience (UX) Design	none	none
NAV-021	2.4.8 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/location)	Indications are provided to help users identify their current location within the site.	User Experience (UX) Design	Visual Design	none
NAV-022	2.4.9 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/link-purpose-link-only)	The purpose of each link can be identified in its immediate context, or from the link text alone.	Content Authoring	User Experience (UX) Design	none
NAV-023	2.4.10 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/section-headings)	Content is logically organized using section headings.	Content Authoring	User Experience (UX) Design	Front-End Development
NAV-024	3.2.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/on-focus)	Setting the focus to a new element doesn't automatically trigger a context change, such as content updates or the opening of new windows.	User Experience (UX) Design	Front-End Development	none
NAV-025	3.2.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/on-input)	Interacting with input controls or other equivalent elements doesn't automatically trigger a change of context, unless the user has been notified ahead of time.	User Experience (UX) Design	Front-End Development	none
NAV-026	3.2.3 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/consistent-navigation)	Navigation mechanisms are repeated consistently throughout the site or application in the same relative order.	User Experience (UX) Design	Visual Design	none
NAV-027	3.2.4 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/consistent-identification)	Navigational graphics and icons used throughout the site or application are designed to always serve the same function and or meaning.	Visual Design	User Experience (UX) Design	none
NAV-028	3.2.4 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/consistent-identification)	The accessible name of user interface components used across the site or application are defined consistently.	Content Authoring	User Experience (UX) Design	none
NAV-029	3.2.4 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/consistent-identification)	Users can consistently distinguish between links internal to a page and links going to different locations.	Visual Design	none	none
NAV-030	3.2.5 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/change-on-request)	Links that open new windows visually indicate they will do so.	User Experience (UX) Design	Visual Design	none
NAV-031	3.2.5 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/change-on-request)	Links that open new windows indicate they will do so, either as part of the link text, or using aria-label or equivalent means.	Front-End Development	Content Authoring	none

Data Tables

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
TAB-001	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Tables are only to be used to lay out	Front-End	User Experience	none

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
		tabular information or data.	Development	(UX) Design	
TAB-002	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Table row and/or column headers provide context for data within the table.	User Experience (UX) Design	none	none
TAB-003	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Data table structure is appropriate for the data being included.	User Experience (UX) Design	none	none
TAB-004	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Tabular data and corresponding header cells for that data are part of the same table.	Front-End Development	User Experience (UX) Design	none
TAB-005	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Header cells for rows are marked up using THEAD elements.	Front-End Development	none	none
TAB-006	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Header cells for columns are marked up using TH elements.	Front-End Development	none	none
TAB-007	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	The relationship between table header rows and table header columns with data cells in simple data tables is provided through the SCOPE attributes.	Front-End Development	User Experience (UX) Design	none
TAB-008	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	The relationship between table header rows and table header columns with data cells in complex data tables is provided through the HEADERS and ID attributes.	Front-End Development	User Experience (UX) Design	none
TAB-009	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Caption elements are used to associate caption information with data tables.	Front-End Development	none	none
TAB-010	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Large, complex data tables are broken into smaller, simpler data tables (when possible).	User Experience (UX) Design	Visual Design Content Authoring	none
TAB-011	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Unrelated data is not included in the same data table.	User Experience (UX) Design	Content Authoring	none
TAB-012	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Tables are not used for layout purposes.	Front-End Development	none	none
TAB-013	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Tables are not used to layout lists.	Front-End Development	none	none
TAB-014	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Caption elements, aria-labelledby attributes or other equivalent means are used to explain the structure of data tables.	Front-End Development	none	none
TAB-015	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	A meaningful description of the structure of data tables is provided.	Content Authoring	User Experience (UX) Design	none
TAB-016	1.2.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/meaningful-sequence)	The programmatic order of the table content matches the intended reading order, so not to affect its meaning.	Front-End Development	User Experience (UX) Design	none
TAB-017	2.4.6 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/headings-and-labels)	All data table header cells are identified.	User Experience (UX) Design	Front-End Development	none

Animation and Movement

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
ANM-001	1.2.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/audio-only-and-video-only-prerecorded)	Text transcripts are provided for prerecorded audio-only files.	Content Authoring	User Experience (UX) Design	none
ANM-002	1.2.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/audio-only-and-video-only-prerecorded)	Text transcripts are provided for prerecorded video-only files.	Content Authoring	User Experience (UX) Design	none
ANM-003	1.2.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/audio-only-and-video-only-prerecorded)	Links to transcript files are provided in close proximity to the audio-only files.	User Experience (UX) Design	none	none
ANM-004	1.2.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/audio-only-and-video-only-prerecorded)	The relationship	User	Front-End	none

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
		between multimedia files and their associated transcript is clearly communicated through content sequence or focus order.	Experience (UX) Design	Development	
ANM-005	1.2.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/audio-only-and-video-only-prerecorded)	Copy on the page identifies when video content has no sound.	User Experience (UX) Design	Content Authoring	none
ANM-006	1.2.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/audio-only-and-video-only-prerecorded)	When appropriate, further information about the multimedia file is provided in proximity to the multimedia file.	User Experience (UX) Design	none	none
ANM-007	1.2.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/captions-prerecorded)	Synchronized captions are provided for all prerecorded video content.	Content Authoring	User Experience (UX) Design	none
ANM-008	1.2.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/captions-prerecorded)	Captions do not skip dialogues or important sounds.	Content Authoring	none	none
ANM-009	1.2.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/captions-prerecorded)	Multimedia player controls are provided to turn captions on or off.	User Experience (UX) Design	none	none
ANM-010	1.2.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/audio-description-or-media-alternative-prerecorded)	Text transcripts report all significant information from the audio track.	Content Authoring	none	none
ANM-011	1.2.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/audio-description-or-media-alternative-prerecorded)	Text transcripts or audio descriptions report all significant information from the visual track.	Content Authoring	none	none
ANM-012	1.2.4 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/captions-live)	Synchronized captions are provided for all live video content.	Content Authoring	none	none
ANM-013	1.2.4 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/captions-live)	Captions for live audio content are generated using real-time text transcription services.	Content Authoring	User Experience (UX) Design	none
ANM-014	1.2.5 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/audio-description-prerecorded)	Prerecorded videos have audio descriptions that captures all significant information from the visual track.	Content Authoring	none	none
ANM-015	1.2.5 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/audio-description-prerecorded)	Controls to toggle audio descriptions features are provided as part of the media	User Experience (UX) Design	none	none

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
		player controls.			
ANM-016	1.2.5 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/audio-description-prerecorded)	Multimedia player controls are provided to access a version of the video with audio description.	User Experience (UX) Design	none	none
ANM-017	1.2.6 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/sign-language-prerecorded)	Sign language interpretation is provided for all prerecorded audio content in synchronized media.	Content Authoring	User Experience (UX) Design	none
ANM-018	1.2.7 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/extended-audio-description-prerecorded)	Extended audio description is provided for all prerecorded synchronized video content when pauses in foreground audio are insufficient to allow audio descriptions to convey the sense of the video.	Content Authoring	User Experience (UX) Design	none
ANM-019	1.2.8 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/media-alternative-prerecorded)	Text alternatives are provided for all prerecorded audio and video files.	Content Authoring	User Experience (UX) Design	none
ANM-020	1.2.8 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/media-alternative-prerecorded)	Brief descriptions summarizing multimedia content are provided in close proximity to the audio and video files.	User Experience (UX) Design	Content Authoring	none
ANM-021	1.2.9 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/audio-only-live)	Live transcripts are provided for all audio content happening in real-time.	Content Authoring	User Experience (UX) Design	none
ANM-022	1.4.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/audio-control)	Multimedia player controls are provided to turn sound on and off.	User Experience (UX) Design	none	none
ANM-023	1.4.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/audio-control)	Volume controls for page-level multimedia files are independent from general computer audio controls.	Front-End Development	User Experience (UX) Design	none
ANM-024	1.4.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/audio-control)	Volume controls for page-level multimedia files are visually located at the top of the page.	User Experience (UX) Design	none	none
ANM-025	1.4.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/audio-control)	Audio content that automatically starts on page load lasts no longer than 3 seconds.	Front-End Development	User Experience (UX) Design	none
ANM-026	1.4.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/audio-control)	Prerecorded or live video content is not set to auto-	User Experience (UX) Design	none	none

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
		play.			
ANM-027	1.4.7 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/low-or-no-background-audio)	Prerecorded audio-only background sounds can be controlled by the user.	User Experience (UX) Design	Front-End Development	none
ANM-028	1.4.7 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/low-or-no-background-audio)	Prerecorded audio-only speeches contain no background sound, or if they do, must be at least 20 decibels lower than the foreground speech content.	Content Authoring	none	none
ANM-029	2.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/keyboard)	Multimedia player controls can be fully operated using only the keyboard.	Front-End Development	User Experience (UX) Design	none
ANM-030	2.2.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/pause-stop-hide)	Multimedia player controls are provided to pause or play the multimedia file.	User Experience (UX) Design	none	none
ANM-031	2.2.3 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/no-timing)	Timing is not an essential part of the event or activity presented by the content.	User Experience (UX) Design	Content Authoring	Business
ANM-032	2.2.6 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/timeouts)	Users are warned that more than 20 hours of inactivity could lead to data loss.	User Experience (UX) Design	Content Authoring	none
ANM-033	2.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/three-flashes-or-below-threshold)	Content on the screen does not flash or blink more than three times in any one-second period, or the flash is below the general flash and red flash thresholds.	Visual Design	none	none
ANM-034	2.3.2 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/three-flashes)	Content on the screen does not flash or blink at a rate that is higher than three times in any one-second period.	Visual Design	none	none
ANM-035	2.3.3 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/animation-from-interactions)	Animation features that create the illusion of movement are defined so that users can disable them.	User Experience (UX) Design	Front-End Development	none
ANM-036	2.3.3 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/animation-from-interactions)	Animation features that create the illusion of movement support prefers-reduced-motion (or other equivalent means).	Front-End Development	User Experience (UX) Design	none
ANM-037	4.1.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/name-role-value)	Information conveyed by multimedia	Front-End Development	User Experience (UX) Design	none

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
		player controls are programmatically announced through assistive technologies.			

Static Content

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
SCT-001	1.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/non-text-content)	Emoticons, emojis, ASCII art, and other non-markup language constructs are supported with equivalent text alternatives and conveyed to assistive technologies.	User Experience (UX) Design	Content Authoring	Front-End Development
SCT-002	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Emoticons, emojis, ASCII art, and other non-markup language constructs are not used as the only way to structure content or convey information.	User Experience (UX) Design	Front-End Development	none
SCT-003	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Proper markup is used to render emphasized, bolded text and other stylistic or presentational effects.	Front-End Development	none	none
SCT-004	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Proper markup is used to structure quotes, blockquotes and citations.	Front-End Development	Content Authoring	none
SCT-005	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Heading markup is only used for text which acts as a section heading.	Front-End Development	User Experience (UX) Design	none
SCT-006	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	Heading markup is not used for formatting effects.	Front-End Development	none	none
SCT-007	1.3.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/info-and-relationships)	The headings used in a page provide a logical outline for the document.	Content Authoring	User Experience (UX) Design	none
SCT-008	1.3.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/meaningful-sequence)	The intended reading order of the content remains logical when CSS and images are turned off.	Front-End Development	User Experience (UX) Design	none
SCT-009	1.3.2 (A) (https://www.w3.org/WAI/WCAG22/Understanding/meaningful-sequence)	The source code order reflects the intended reading order of the document.	Front-End Development	User Experience (UX) Design	none
SCT-010	1.3.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/sensory-characteristics)	Site supports internationalization with multiple languages, including right-to-left languages.	Business Analysis	User Experience (UX) Design	Content Authoring Visual Design
SCT-011	1.3.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/sensory-characteristics)	Text direction is properly marked up as such, especially for right-to-left languages.	Front-End Development	none	none
SCT-012	1.3.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/sensory-characteristics)	Objects that rely on shape to be properly perceived are supported with additional text information.	User Experience (UX) Design	Visual Design	Content Authoring
SCT-013	1.3.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/sensory-characteristics)	Objects that rely on size to be properly perceived are supported with additional text information.	User Experience (UX) Design	Visual Design	Content Authoring
SCT-014	1.3.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/sensory-characteristics)	Objects that rely on visual location to be properly perceived are supported with additional text information.	User Experience (UX) Design	Visual Design	Content Authoring
SCT-015	1.3.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/sensory-characteristics)	Objects that rely on orientation to be properly perceived are supported with additional text information.	User Experience (UX) Design	Visual Design	Content Authoring
SCT-016	1.3.3 (A) (https://www.w3.org/WAI/WCAG22/Understanding/sensory-characteristics)	Objects that rely on sound to be properly perceived are supported with additional text information.	User Experience (UX) Design	Content Authoring	none
SCT-017	1.4.5 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/images-of-text)	Mathematical formulas are marked up using MathML.	Front-End Development	Content Authoring	none
SCT-018	1.4.8 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/visual-presentation)	Text content in design assets such as content blocks is left-aligned in left-to-right languages.	Visual Design	User Experience (UX) Design	none

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
SCT-019	1.4.8 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/visual-presentation)	Fully justified text can easily be changed to ragged right text with a simple action.	User Experience (UX) Design	Front-End Development	none
SCT-020	3.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/language-of-page)	The primary language used in the document is properly identified using the lang attribute.	Front-End Development	none	none
SCT-021	3.1.1 (A) (https://www.w3.org/WAI/WCAG22/Understanding/language-of-page)	The language definition of the document uses the correct value for language and locale.	Content Authoring	none	none
SCT-022	3.1.2 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/language-of-parts)	Content passages that differ from the default are identified with the correct value for language and locale.	Content Authoring	Front-End Development	none
SCT-023	3.1.3 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/unusual-words)	Unusual words, phrases, and abbreviations are organized into a glossary.	User Experience (UX) Design	Content Authoring	none
SCT-024	3.1.3 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/unusual-words)	Unusual words found in the content are linked to their definitions in a glossary.	User Experience (UX) Design	Content Authoring	none
SCT-025	3.1.4 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/abbreviations)	Abbreviations are programmatically associated with their definition.	Front-End Development	none	none
SCT-026	3.1.5 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/reading-level)	Content is written in plain language using everyday words, to help users with different literacy levels and access needs.	Content Authoring	none	none
SCT-027	3.1.5 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/reading-level)	Ideas conveyed through text are supported with illustrations or other visuals.	Visual Design	User Experience (UX) Design Content Authoring	none
SCT-028	3.1.5 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/reading-level)	Content is displayed and structured in a way that makes it easier to read.	User Experience (UX) Design	Visual Design	none
SCT-029	3.1.6 (AAA) (https://www.w3.org/WAI/WCAG22/Understanding/pronunciation)	Ambiguous words are supported by a mechanism that helps users identify their specific pronunciation.	User Experience (UX) Design	Front-End Development	Content Authoring

Dynamic Interactions

ID	WCAG SC	Task	Primary Ownership	Secondary Ownership	Contributor
DYN-001	4.1.3 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/status-messages)	Information conveyed by multimedia player controls are programmatically announced through assistive technologies.	Front-End Development	User Experience (UX) Design	none
DYN-002	4.1.3 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/status-messages)	Status messages are announced by assistive technologies without affecting the focus.	User Experience (UX) Design	Front-End Development	none
DYN-003	4.1.3 (AA) (https://www.w3.org/WAI/WCAG22/Understanding/status-messages)	Status, toast, or similar messages are programmatically determined through WAI-ARIA roles or properties, so they can be presented to assistive technology users without receiving focus.	Front-End Development	User Experience (UX) Design	none

Updated: 24 July 2025.

Editors and contributors: See Acknowledgements (<https://www.w3.org/WAI/planning/arm/acknowledgements/>).

Developed through the Accessibility Roles and Responsibilities Mapping (ARRM) Community Group (<https://www.w3.org/community/arm/>) at W3C. Initially developed with the Accessibility Education and Outreach Working Group (EOWG (<https://www.w3.org/WAI/about/groups/eowg/>)).

© This work is licensed under a Creative Commons Attribution 4.0 International License (<https://creativecommons.org/licenses/by/4.0/>).