W3C Recommendation

Accessibility Conformance Testing (ACT) Rules Format 1.1

W3C Recommendation, 5 February 2026

▼ **More details about this document**

This version:
https://www.w3.org/TR/2026/REC-act-rules-format-1.1-20260205/

Latest published version:
https://www.w3.org/TR/act-rules-format/

Editor's Draft:
https://w3c.github.io/wcag-act/act-rules-format.html

History:
https://www.w3.org/standards/history/act-rules-format-1.1/

Implementation Report:
https://www.w3.org/2025/12/17-act-rules-format-11/implementation-report.html

Feedback:
Open Issues
GitHub

Editors:
Wilco Fiers (Deque Systems)
Kathy Eng (US Access Board)
Daniel Montalvo (W3C)

Former Editors:
Trevor Bostic (MITRE Corp.)
Maureen Kraft (IBM Corp.)
Mary Jo Mueller (IBM Corp.)
Shadi Abou-Zahra (W3C)

Errata:
Errata

See also **translations**.

Abstract

Accessibility Conformance Testing (ACT) Rules Format 1.1 defines a format for writing accessibility test rules. The test rules can be used for developing automated testing tools and manual testing methodologies. This document provides a common format that allows anyone involved in accessibility testing to document and share their testing procedures in a robust and understandable manner. This enables transparency and harmonization of testing methods, including methods implemented by accessibility test tools.

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 standards and drafts index.

This document was published by the Accessibility Guidelines Working Group as a Recommendation using the Recommendation track.

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.

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

ACT Rules Format 1.1 includes two major changes with respect to the ACT Rules Format 1.0. It now allows for:

- Two types of accessibility requirements: conformance requirements and secondary requirements.
- Subjective applicability.

ACT Rules Format 1.1 is not fully backwards compatible with 1.0. The two versions of the specification coexist, although the Working Group encourages writers to write test rules that are compatible with ACT Rules Format 1.1.

There have been no changes since ACT Rules Format 1.1 was published as a Candidate Recommendation 19 August 2025.

All ACT Rules that are published on the WAI website are now compatible with this version of the ACT Rules Format.

The group addressed feedback received during the Horizontal Review Process. For details, see:

- Changes since ACT Rules Format First Public Working Draft
- Changes since ACT Rules Format 1.0
- Complete Diff between ACT Rules Format 1.1 and 1.0

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 that 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 18 August 2025 W3C Process Document.

Table of Contents

§ 1. Introduction

There are currently many test procedures and tools available which aid their users in testing web content for conformance to accessibility standards such as the Web Content Accessibility Guidelines [WCAG22]. As the Web develops in both size and complexity, these procedures and tools are essential for managing the accessibility of resources available on the Web.

ACT Rules Format provides guidance that is intended to enable a consistent interpretation of how to test conformance to WCAG and other accessibility requirements documents, and to promote consistent results in accessibility testing. Act Rules Format is designed to describe both manual tests, and automated tests that are performed by accessibility testing tools.

Documenting how to test accessibility requirements will result in transparent accessibility tests that have reproducible test results. ACT Rules Format defines the requirements for these test descriptions, known as Accessibility Conformance Testing Rules (ACT Rules).

An ACT Rule is a plain language description of how to test a specific type of content for a specific aspect of an accessibility requirement. An ACT Rule describes what kind of content it applies to and which conditions are true about the applicable elements for them to meet all expectations of the rule. In the context of WCAG, ACT Rules test for failures in satisfying success criteria. Such failures are often described in common failures documented for WCAG. ACT Rules are written for the testing process and are usually more specific than common failures.

ACT Rules Format defines the requirements and rule structure for the types of information each rule needs to include in order to be called an ACT Rule. The structure of the ACT rule is defined in the ACT Rule Structure section. Each ACT Rule also contains a plain language description of the type of

content under test, the test to perform, and the expected result. Where the test result affects conformance, the rule documents the particular requirement being tested. The resulting outcomes from the test can be used to help determine conformance or non-conformance to the requirement. Examples are also written as part of the ACT rule to provide a way to verify that implementations of the rule can successfully determine the expected outcomes.

2. Scope

The ACT Rules Format defined in this specification is designed for rules that can be used in testing content created using web technologies, such as Hypertext Markup Language [HTML], Cascading Style Sheets [css-2018], Accessible Rich Internet Applications [WAI-ARIA], Scalable Vector Graphics [SVG2], EPUB 3 [epub-33], and more. The ACT Rules Format is designed to be technology agnostic, meaning that it can conceivably be used to describe test rules for other technologies.

The ACT Rules Format can be used to describe ACT Rules dedicated to testing the accessibility requirements defined in the Web Content Accessibility Guidelines [WCAG22], which are specifically designed for web content. Other accessibility requirements applicable to web technologies can also be testable with ACT Rules. For example, ACT Rules could be developed to test the conformance of web-based user agents to the User Agent Accessibility Guidelines [UAAG20]. The ACT Rules Format might not always be suitable to describe tests for other types of accessibility requirements.

2.1. Backward Compatibility

ACT Rules Format 1.1 is largely, but not fully, backward compatible with 1.0. ACT Rules Format 1.1 introduces requirements that are not part of 1.0. It also allows a few things that were not allowed under 1.0, such as subjective applicability. A detailed list of changes is available under Change History.

3. ACT Rule Types

In accessibility, there are often different technical solutions to make the same type of content accessible. For example, there are multiple methods for giving an `img` element in HTML an accessible name. Multiple solutions could be tested in a single rule; however, such a rule tends to be quite complex, making it difficult to understand and maintain. The ACT Rules Format solves this by providing two types of rules:

- ***Atomic rules*** describe how to test a specific type of solution. It contains a precise definition of what elements, nodes or other "parts" of a test subject are to be tested, and when those test targets

are considered to pass or fail the rule. These rules are to be kept small and *atomic*. This means that atomic rules test a single "condition" and do so without using the outcomes from other rules.

- ***Composite rules*** describe how the outcomes of multiple atomic rules can be combined into a single outcome for each test target. A composite rule can have multiple "conditions", each of these tested in a separate atomic rule. The logic in the composite rule describes how any one of these satisfying conditions, or some combination of them, is used to determine a single outcome.

Composite rules cannot contain other composite rules. Any time a nested composite rule would be needed, all of the relevant atomic rules can be combined directly into the new composite rule.

EXAMPLE 1 ¶

Using multiple input rules in a composite rule:

> Each HTML `video` element meets all expectations from at least one of the following rules:
>
> - Video elements have a transcript
> - Video elements have an audio description
> - Video elements have a description track

Not all atomic rules have to be part of a composite rule. Composite rules are used when the outcomes of multiple atomic rules need to be combined to determine if a test subject does not satisfy an accessibility requirement.

The separation between atomic rules and composite rules creates a division of responsibilities. Atomic rules test if web content is correctly implemented in a particular solution. Composite rules can test if a combination of outcomes from other atomic rules satisfy the accessibility requirement, in part or as a whole.

§ 4. ACT Rule Structure

An ACT Rule *MUST* consist of at least the following items:

- ***Descriptive Title***
- Rule Identifier
- Rule Description
- Rule Type

- Accessibility Requirements Mapping
- Rule Input, which is one of the following:
  - Input Aspects (for atomic rules) OR
  - Input Rules (for composite rules)
- Applicability
- Expectations
- Background
  - Assumptions
  - Accessibility Support
  - Related Rules (optional)
  - Other Resources (optional)
- Examples
- Rule Versions
- ACT Rules Format Version
- Glossary

An ACT Rule MAY also contain:

- Related Rules (optional) under Background
- Other Resources (optional) under Background
- Issues List (optional)
- Implementations (optional)
- Acknowledgments (optional)

The ACT Rules format does not prescribe what format ACT Rules can be written in (e.g. HTML, DOCX, PDF, etc.). However, ACT Rules *MUST* be written in a document that conforms to the Web Content Accessibility Guidelines [WCAG22] or a comparable accessibility standard. This ensures that ACT Rules are accessible to people with disabilities. ACT Rule examples are allowed to contain inaccessible content. If any example contains accessibility issues listed in WCAG 2.2 Section 5.2.5 Non-Interference, users *MUST* be warned of this in advance. ACT Rules should use a localizable format, such that a rule can contain multiple language representations or so that translations of the rule can be created.

§ 4.1. Rule Identifier

An ACT Rule *MUST* have an identifier that is unique within its ruleset. The identifier can be any text, URL, a database identifier, etc.

EXAMPLE 2 ¶

Identifiers that are also used as filenames; They include a technology directory, followed by a handle that includes an element name or attribute:

- html+svg/video-alternative
- html+svg/meta-no-refresh
- html+svg/unique-id

In addition to the identifier, each new release of an ACT Rule *MUST* be versioned with either a date or a number. A reference to the previous version of that rule *MUST* be available. The identifier *MUST NOT* be changed when the rule is updated. For substantial changes, a new rule *SHOULD* be created with a new identifier, and the old rule *SHOULD* be deprecated.

EXAMPLE 3 ¶

Updating a rule:

> When updating a rule that is about buttons, to now also be about links, menu items and tabs, the buttons rule is deprecated. As a replacement, a new rule is created that is applicable to all those elements.

§ 4.2. Rule Description

An ACT Rule *MUST* have a description that is in plain language which provides a brief explanation of what the rule does.

Regardless of the data format in which a rule is written, the description needs to include metadata specifying its language and base paragraph direction. Most markup languages or document formats have built-in provisions for indicating these. For more details, see [string-meta].

EXAMPLE 4

Rule description:

> This rule checks that the HTML page has a title

4.3. Rule Type

An ACT Rule *MUST* have a rule type which is one of the following:

- Atomic rule
- Composite rule

Refer to the Rule Type section for detailed definitions of the rule types.

4.4. Accessibility Requirements Mapping

When an ACT Rule is designed to test conformance to one or more Accessibility requirements documents, the rule *MUST* list all accessibility requirements from those documents that are not satisfied when one or more of the outcomes of the rule is `failed`. The rule *MAY* list accessibility requirements that could be not satisfied when the rule outcome is failed. There are two types of accessibility requirements:

- Conformance Requirements
- Secondary Requirements

Each accessibility requirement in the mapping *MUST* include the following:

1. either the name, title, identifier or summary of the accessibility requirement, and
2. the name of the accessibility requirements document, and
3. a link or reference to the accessibility requirements document if one exists, and
4. the conformance level associated with the accessibility requirement, if one exists, and
5. whether the requirement is a conformance requirement or a secondary requirement.

4.4.1. Outcome Mapping

For each conformance requirement in the mapping, an ACT Rule *MUST* indicate what the outcomes of the rule mean for satisfying an accessibility requirement for that test subject.

4.4.1.1. Conformance Requirements

When a rule is designed to test a specific accessibility requirement, the accessibility requirement is mapped as a ***Conformance Requirement*** when both of the following conditions are true:

- Failed Outcomes: When one or more of the outcomes for a test subject is `failed`, the accessibility requirement is ***not satisfied*** for the test subject, and
- Passed or Inapplicable Outcomes: When all of the outcomes are `passed` or `inapplicable` for a test subject, the accessibility requirement could be ***satisfied*** or ***further testing is needed*** for the test subject.

Rules that can be used to determine if an accessibility requirement is *satisfied* are called ***satisfying tests***.

- Passed: When all of the outcomes are `passed`, the accessibility requirement is satisfied for the test subject.

> **Note:** In the Web Content Accessibility Guidelines [WCAG22], success criteria do not evaluate to `passed`, `failed` or `inapplicable`. Rather they can be *satisfied* (or not). (See the WCAG 2.2 definition: satisfies a success criterion.) If a success criterion is *not satisfied*, a web page can only conform if there is a conforming alternative version, as described in WCAG 2.2 Conformance Requirement 1.

EXAMPLE 5 ¶

Accessibility requirements mapping for a rule that was designed to test if an image button has an accessible name maps success criteria 1.1.1 Non-text content and 4.1.2 Name, Role, Value as conformance requirements:

- Success criterion 1.1.1: Non-text content
  - **Required for conformance** to WCAG 2.0, 2.1, and 2.2 level A and higher
  - Outcome mapping:
    - Any `failed` outcomes: not satisfied
    - All `passed` outcomes: further testing is needed
    - An `inapplicable` outcome: further testing is needed
- Success criterion 4.1.2: Name, Role, Value
  - **Required for conformance** to WCAG 2.0, 2.1, and 2.2 level A and higher
  - Outcome mapping:
    - Any `failed` outcomes: not satisfied
    - All `passed` outcomes: further testing is needed
    - An `inapplicable` outcome: further testing is needed

§ *4.4.1.2. Secondary Requirements*

A ***secondary accessibility requirement*** is a requirement that is correlated with the rule, but for which the rule is not designed to test. The outcome of the rule impacts the result of the accessibility requirement, but the rule is not intended to test the conformance of that requirement. This correlation often results in some of the rule's examples not satisfying the secondary accessibility requirement.

When the rule is not designed to test the accessibility requirement, or failed outcomes of the rule still require further testing for the accessibility requirement, the rule *MAY* map the accessibility requirement as Secondary. When an ACT rule maps to a Secondary requirement, it *MUST* include an explanation of why that requirement is Secondary in the rule.

When the rule is designed to test an accessibility requirement, differences in accessibility support *MUST NOT* be a reason for that requirement to be a secondary accessibility requirement.

Some scenarios when a rule will have Secondary requirements include:

Scenario 1: the rule is not as strict as a requirement

A rule was designed to test a minimum accessibility requirement and there exists a stricter requirement. The rule's failed outcomes can determine that the stricter accessibility requirement is not satisfied, and the rule's passed outcomes may not determine that the stricter requirement is satisfied. It is possible that the accessibility requirement may be not satisfied when the rule's outcomes are passed. The stricter requirement is a Secondary requirement.

EXAMPLE 6

Rule that tests if text has minimum contrast

> This rule was designed to test Success Criterion 1.4.3 Contrast Minimum (AA). A stricter requirement, Success Criterion 1.4.6 Contrast (Enhanced) (Level AAA), will be not satisfied when the rule outcome is failed, and may be not satisfied when the rule outcomes are passed. This rule's mapping:
>
> - Conformance Requirement: Success Criterion 1.4.3 Contrast (Minimum)
> - Secondary Requirement: Success Criterion 1.4.6 Contrast (Enhanced)
>   - Explanation: This success criterion is **more strict** than this rule. This is because this criterion has a higher minimum contrast. Some of the passed examples do not satisfy this success criterion.

Scenario 2: the rule is stricter than a requirement

A rule was designed to test a specific solution for an accessibility requirement, but the requirement can be satisfied by other solutions that are not included in the rule. In this scenario, the rule's failed outcomes cannot determine that an accessibility requirement is not satisfied because further testing is needed. The rule's passed outcomes can determine that an accessibility requirement is satisfied. The accessibility requirement is a secondary requirement.

EXAMPLE 7 ¶

Rule that tests if a focusable element has no keyboard trap via standard navigation

> This rule was designed to test for a specific solution that can be used to meet Success Criterion 2.1.2: No Keyboard Trap. The rule does not test for the use of other possible solutions, such as non-standard navigation, that can satisfy the success criterion. Its failed outcomes cannot determine that the accessibility requirement is not satisfied. This rule's mapping:
>
> - Secondary Requirement: Success Criterion 2.1.2: No Keyboard Trap
>   - Explanation: This success criterion is **less strict** than this rule. This is because this criterion allows for the use of non-standard keyboard navigation to exit a keyboard trap. Some of the failed examples may satisfy this success criterion.

A rule was designed to test an accessibility requirement and there exists a less strict accessibility requirement. In this scenario, the rule's passed outcomes can determine that the less strict requirement is satisfied, and the rule's failed outcomes may not determine that the less strict requirement is not satisfied. It is possible that the accessibility requirement may be satisfied when the rule's outcome is failed. The less strict accessibility requirement is a secondary requirement.

EXAMPLE 8 ¶

Rule that tests Enhanced Contrast

> This rule was designed to test Success Criterion 1.4.6 Contrast (Enhanced) (Level AAA). A less strict requirement, Success Criterion 1.4.3 Contrast Minimum (AA), will be satisfied when the rule outcomes are passed, and may be satisfied when the rule outcomes are failed. This rule's mapping:
>
> - Conformance Requirement: Success Criterion 1.4.6 Contrast (Enhanced)
> - Secondary Requirement: Success Criterion 1.4.3 Contrast (Minimum)
>   - Explanation: This success criterion is **less strict** than this rule. This is because this criterion has a lower minimum contrast. Some of the failed examples may satisfy this success criterion.

Scenario 3: the rule may impact the result of the accessibility requirement, but the rule is not intended to test the conformance of that requirement

A rule was designed to test an accessibility requirement and under certain conditions, other accessibility requirements apply. In this scenario, the other accessibility requirements are sometimes, but not always, satisfied or not satisfied because they are not always applicable in the rule. These other accessibility requirements are secondary requirements.

EXAMPLE 9 ¶

Rule that tests if a link has a non-empty accessible name

This rule was designed to test links for accessibility requirements 4.1.2 Name, Role, Value (Level A), 2.4.4 Link Purpose (In Context) (Level A), 2.4.9 Link Purpose (Link Only) (Level AAA). In an image map, <area> elements that define regions within the map are both links and images. Testing for non-empty accessible name for <area> elements would map to another accessibility requirement 1.1.1 Non-text Content. Because the rule tests for all links and Success Criterion 1.1.1 only applies certain links, this rule's mapping:

- Conformance Requirement: Success Criteria 4.1.2 Name, Role, Value (Level A), 2.4.4 Link Purpose (In Context) (Level A), 2.4.9 Link Purpose (Link Only) (Level AAA)
- Secondary Requirement: Success Criterion 1.1.1 Non-text Content
  - Explanation: This success criterion is **related** to this rule. This is because HTML <area> elements are both links and non-text content. Most failed examples satisfy this success criterion.

EXAMPLE 10 ¶

Rule that tests that elements that have an explicit role also specify all required states and properties

This rule was designed to test a requirement of WAI-ARIA. In some cases, a failed outcome for this rule may result in WCAG 2 Success Criterion 4.1.2 Name, Role, Value being not satisfied, but not always. This rule's mapping:

- Conformance Requirement: WAI-ARIA 1.2
- Secondary Requirement: Success Criterion 4.1.2 Name, Role, Value
  - Background Section: Success Criterion 4.1.2 Name, Role, Value is mapped as a Secondary requirement because it only applies to user controls.

4.4.2. Mapping Outside WCAG

ACT Rules can be used to test accessibility requirements that are not part of a W3C accessibility standard, such as accessibility requirements in Hypertext Markup Language [HTML], or tests in a methodology like RGAA 3 2016. An ACT Rule *MUST* indicate whether or not the accessibility requirement it maps to is required for conformance in its accessibility requirements document. Examples of accessibility requirements that are not required for conformance are WCAG sufficient techniques, or a company style guide that includes both requirements and optional "best practices". The distinction between what is required and what is optional has to be clear.

EXAMPLE 11

Accessibility requirements mapping for a rule that tests that each `img` element has an `alt` attribute:

- Technique H37: Using alt attributes on img elements
  - **Not required** for conformance to WCAG 2.0, 2.1, and 2.2 at any level
  - Outcome mapping:
    - Any `failed` outcomes: not satisfied
    - All `passed` outcomes: satisfied
    - An `inapplicable` outcome: satisfied
- RGAA 3, Test 1.1.1: Does each image have a text alternative
  - **Required for conformance** to RGAA 3 level A and higher
  - Outcome mapping:
    - Any `failed` outcomes: not satisfied
    - All `passed` outcomes: satisfied
    - An `inapplicable` outcome: satisfied

4.4.3. Rules Without Accessibility Requirements

If the rule does not map to any accessibility requirement, the accessibility requirement mapping will only contain the explainer that it is not required for conformance to the accessibility requirements document. This is common with atomic rules used in composite rules.

EXAMPLE 12

Rule that tests if `role=alert` is used to satisfy WCAG 2.2 success criterion 4.1.3 Status Messages:

> This rule is **not required** for conformance to WCAG 2.2 at any level.

If the `failed` outcome cannot be mapped to an accessibility requirement, there *MUST NOT* be an accessibility requirement in the accessibility requirements mapping. The optional Background section could be used to list accessibility requirements documents when they are thematically related, but for which the rule is not a failure test.

4.4.4. External Accessibility Requirements Mapping

This section is *non-normative*.

While rules are often designed for one, or possibly a small collection of accessibility requirements documents, it is likely that other accessibility requirements documents also map to those ACT Rules. For example, rules can be written for the Web Content Accessibility Guidelines [WCAG22], but many of those could also map to a company's internal accessibility policy. In such a scenario, an external accessibility requirements mapping could be created. An external accessibility requirements mapping amends the accessibility requirements mapping of an ACT rule by adding mapping to a different accessibility requirements document. An external accessibility requirements mapping avoids duplication of a rule for the sole purpose of changing the mapping.

4.5. Rule Input

To evaluate content using an ACT Rule, the rule requires some information from the test subject. This is the input for the rule. What input is required is made explicit, to help testers understand what capabilities are required to use a rule. Atomic rules and composite rules have different input.

- Atomic rules have Input Aspects
- Composite rules have Input Rules

4.5.1. Input Aspects (Atomic rules only)

An input aspect is a distinct part of the test subject. Rendering a particular piece of content to an end user commonly involves different technologies, some or all of which are required as input for an atomic rule. For example, some rules need to operate directly on the Hypertext Transfer Protocol [http11] messages exchanged between a server and a client, while others need to operate on the Document Object Model [DOM] tree exposed by a web browser.

Atomic rules *MUST* list the aspects used as input for the applicability and expectations of the atomic rule. Rules can operate on several aspects simultaneously, such as both the HTTP messages and the DOM tree.

Some input aspects are well defined in a formal specification, such as HTTP messages, the DOM tree, and CSS styling [css-2018]. For these, a reference to the corresponding section in the Common Input Aspects note is sufficient as a description of the aspect. For input aspects that are not well defined, an ACT Rule *MUST* include either a detailed description of the aspect in question, or a reference to a well defined description.

EXAMPLE 13

Input aspects for a rule that checks if a transcript is available for videos:

- DOM Tree
- CSS Styling
- Audio output
- Visual output

EXAMPLE 14

Input aspects for a rule that checks for use of (language specific) generic link texts like "click here" and "more":

- DOM Tree
- CSS Styling
- Language

The method through which an input aspect is served is not relevant. For example a DOM tree can be served through HTTP as HTML, it can be bundled as several pages in an EPUB publication, or it can be inferred from a JSX source file. All rules that have only DOM tree as an input aspect can be applied to those technologies.

4.5.2. Input Rules (Composite rules only)

A composite rule uses outcomes from atomic rules and applies logic to them so that a single outcome can be determined for each test target. The identifier and descriptive title of all the atomic rules used in the expectations *MUST* be listed in the composite rule. The input rules describe the input for composite rules, similar to how input aspects describe the input for atomic rules.

4.6. Applicability

The applicability describes what parts of the test subject are tested.

4.6.1. Applicability for Atomic Rules

The applicability section is a required part of an atomic rule. It *MUST* contain a precise description of the parts of the test subject to which the rule applies. For example, specific nodes in the DOM [DOM] tree, or tags that are incorrectly closed in an HTML [HTML] document. These are known as the test targets. The applicability *MUST* only use information made available through the listed input aspects in the rule. No other information can be used in the applicability.

Applicability *MUST* be unambiguous so that the applicability can only be understood in one way. For example, concepts like headings and images are ambiguous since they could refer to the tag name, the semantic role, or the element's purpose on the web page. Conversely, a rule that specifically only applies to elements with a heading tag would satisfy the unambiguous requirement.

Additionally, the applicability *SHOULD* be described objectively and in plain language. An objective description is one that can be resolved without uncertainty, in a given technology. Examples of objective properties in HTML are tag names, their computed role, and the distance between two elements.

Subjective properties are concepts like decorative, navigation mechanism, and pre-recorded that may be defined in a variety of ways, often relying on human judgement. When possible, avoid subjectivity since it can easily be misunderstood. In cases where it is impossible to objectively define the applicability, the use of a subjective description in the applicability is acceptable. When crafting a

subjective applicability, if possible, split the objective and subjective parts of the applicability into separate rules. This will ensure the designed rule meets the atomic rule requirement of testing a single condition. An example would be testing headings with correct semantic markup in a different rule from those without semantic markup.

Definitions can be put in the rule glossary, or they can be defined in the section where they are used.

EXAMPLE 15 ¶

Objective applicability of an atomic rule testing WCAG 2.2 success criterion 1.4.2 Audio Control:

> Applicability: Each `video` or `audio` element with the `autoplay` attribute, as well as each `object` element that is used for automatically playing video or audio when the web page loads.
>
> Note: A web page is considered "loaded" when the `document.readyState` is set to `complete`.

EXAMPLE 16 ¶

Objective applicability of a rule with the page as a test target

> Applicability: The rule applies to any page where the document element is an `html` element, and the `html` element is rendered in a top-level context (i.e. the `html` element is not embedded in another page, such as through `iframe` or `object` elements).

EXAMPLE 17 ¶

Objective applicability of a rule with a DOM attribute as a test target

> Applicability: The rule applies to any `role` attribute that is specified on an HTML or SVG element.

EXAMPLE 18 ¶

Subjective applicability for a rule testing that elements styled as a heading use correct heading markup. This rule applicability cannot be written objectively since "styled as a heading" is subjective and depends on the context of the page.

> Applicability: The rule applies to any HTML element that is styled as a heading. Elements that are styled as a heading may include features such as a larger font-size than nearby text, bolding or changing of the font, or the use of whitespace or shapes that visually distinguish the element text.

EXAMPLE 19 ¶

Non-conforming: This example demonstrates an applicability that attempts to be subjective, but does not meet the "must be unambiguous" requirement. For example, non-text content could refer to images, interactive elements, or even a text smiling face emoji using a colon and parenthesis like ":)". Due to the ambiguity in the applicability, it is impossible to tell.

> Applicability: This rule applies to any non-text content.

§ *4.6.1.1. Applicability Type Designation (optional)*

Rules can optionally include an applicability type identifier signifying whether the rule contains an objective or a subjective applicability. This identifier is intended to benefit rule readers and implementers by clearly stating the rule author’s intention of the applicability and reducing confusion due to different reader and implementer interpretations.

§ 4.6.2. Applicability for Composite Rules

The applicability of a composite rule is defined as the union of all applicability definitions from the rules listed in the input rules. Rule authors *MAY* omit a description of the applicability for composite rules. This can be useful if it is difficult to express the combined applicability in plain language. If the composite rule includes applicability, it *MUST* be the union of all the applicability in the input rules.

Note that input rules in a composite rule *MAY* have different applicability. Because of this, not every test target applicable within the composite rule is tested by every input rule.

EXAMPLE 20

Union of applicability of input rules in a composite rule:

Input applicability:

- **Input Rule 1:** All `img` element *with* an `alt` attribute
- **Input Rule 2:** All `img` element *without* an `alt` attribute

Combined applicability:

All `img` elements.

4.6.2.1. Applicability Type Designation (optional)

Composite rules can optionally include an applicability type identifier signifying whether the rule contains an objective or a subjective applicability. The applicability type of a composite rule is calculated as subjective if any of the input rules contain a subjective applicability or objective otherwise.

4.7. Expectations

An ACT Rule *MUST* contain one or more expectations. The expectations describe what the requirements are for the test targets derived from the applicability. An expectation is an assertion about a test target. When a test target meets all expectations, the test target `passed` the rule. If the test target does not meet all expectations, the test target `failed` the rule. If there are no test targets, the outcome for the rule is `inapplicable`.

Each expectation *MUST* be distinct, unambiguous, and be written in plain language.

4.7.1. Expectations for Atomic Rules

All expectations of an atomic rule *MUST* describe the logic that is used to determine a single `passed` or `failed` outcome for a test target. The expectation *MUST* only use information available in the input aspects, from the applicability, and other expectations of the same rule. No other information can be used in the expectation. So for instance, an expectation could be "Expectation 1 is true and ...", but it can’t be "Rule XYZ passed and ...". This ensures that atomic rules are encapsulated.

EXAMPLE 21 ¶

Expectations of a rule to test for accessible names of HTML `input` elements:

1. Each HTML `input` element has an accessible name (as described in Accessible Name and Description: Computation and API Mappings 1.1). [accname-aam-1.1]
2. The accessible name describes the purpose of each HTML `input` element.

EXAMPLE 22 ¶

Expectation of a rule to test if a `role` attribute is valid:

1. Each `role` attribute has a value that corresponds to a non-abstract WAI-ARIA 1.2 role.

Note: Sometimes there is the need for rules with more complex aggregation, for example that X% of all images on a web page are expected to have text alternatives. In this case, the page itself needs to become the test target. The expectation would then be "The test target (the page) has a text alternative for X% of all img elements". The logic for calculating the expectations in such rules is left to the implementations, to avoid over-complexity of this rules format.

§ 4.7.2. Expectations for Composite Rules

All expectations of a composite rule *MUST* describe the logic that is used to determine a single `passed` or `failed` outcome for a test target, based on the outcomes of rules in its input rules. A composite rule expectation *MUST NOT* use information from input aspects. The outcome for a composite rule is `inapplicable` when all input rules are inapplicable.

EXAMPLE 23 ¶

Expectations for the composite rule 'video elements have an audio description or media alternative' (WCAG 2.2 success criterion 1.2.3 Audio Description or Media Alternative):

> Each HTML `video` element meets all expectations from at least one of the following rules:
>
> - video elements have a transcript
> - video elements have an audio description
> - video elements have a description track

EXAMPLE 24 ¶

Expectations for a composite rule that checks if a mechanism is available to escape a keyboard trap; Either through a standard mechanism, or one for which instructions are available:

> For each focusable element, the outcome of one of the following rules is `passed`:
>
> - Keyboard trap with standard escape mechanism
> - Keyboard trap with escape instructions

§ 4.8. Background

An ACT Rule's Background *MUST* contain the Assumptions and Accessibility Support sections. Additional information *MAY* be included about the development of the rule, or references to relevant reading in Other Resources, or Related Rules. Whenever a reference is included in the rule, the relationship to the relevant reading can be specified. Examples of relevant background references for a rule for a Web Content Accessibility Guidelines [WCAG22] success criterion could be WCAG 2.2 Understanding documents, WCAG 2.2 Techniques, or WAI-ARIA 1.2, CSS [css-2018], or HTML [HTML] specifications.

§ 4.8.1. Assumptions

An ACT Rule *MUST* list any known assumptions, limitations or any exceptions for the evaluation, the test environment, technologies being used or the subject being tested. For example, a rule that would

partially test WCAG 2.2 success criterion 1.4.3 Contrast (Minimum) based on the inspection of CSS properties could state that it is only applicable to HTML text content styleable with CSS, and that the rule does not support images of text.

Sometimes there are multiple plausible ways that an accessibility requirement can be interpreted. For instance, it is not immediately obvious if emoji characters are "text" or "non-text content" in the Web Content Accessibility Guidelines [WCAG22]. Whatever the interpretation is, this *MUST* be documented in the rule.

While the assumptions *MUST* be included in the ACT Rule, it *MAY* be empty when there are no known assumptions, limitations or exceptions.

§ 4.8.2. Accessibility Support

Content can be designed to rely on the support for particular accessibility features by different assistive technologies and user agents. For example, content using a particular WAI-ARIA 1.2 feature relies on that feature to be supported by assistive technologies and user agents. This content would not work for assistive technologies and user agents that do not support WAI-ARIA. See the WCAG definition for accessibility supported use of a web technology.

An ACT Rule *MUST* include known limitations on accessibility support.

EXAMPLE 25 ¶

Rule that checks if `aria-errormessage` is used to satisfy WCAG 2.2 success criterion 4.1.3 Status messages:

> The `aria-errormessage` property is known to have limited support with several major screen readers. This method cannot be relied on for support. Alternatives, like using live regions, could serve as fallback. (January 2019)

While an accessibility support section *MUST* be included in the ACT Rule, it *MAY* be empty when there are no known accessibility support issues.

Note: The Website Accessibility Conformance Evaluation Methodology (WCAG-EM) provides guidance on defining an accessibility support baseline for test scenarios, which can help users of ACT Rules to select the appropriate rules for their own circumstance.

4.8.3. Related Rules (optional)

Related rules are other rules that test the same accessibility requirement. For example, two related rules of a composite rule can be the two atomic rules that contribute to its outcome. Similarly, each atomic rule in this example can list the other atomic rule and the composite rule as its related rules.

4.8.4. Other Resources (optional)

Whenever a resource is included in the rule, the relationship to the relevant reading can be specified. Examples of relevant background references for a rule for a Web Content Accessibility Guidelines [WCAG22] success criterion could be WCAG 2.2 Understanding documents, WCAG 2.2 Techniques, or WAI-ARIA 1.2, CSS [css-2018], or HTML [HTML] specifications.

4.9. Examples

Examples are (snippets of) content that can be used to validate the implementation of ACT Rules. Each rule *MUST* have one or more examples for `passed`, `failed`, and `inapplicable` outcomes. An example consists of two pieces of data, a snippet of each input aspect for a rule, and the expected outcome for that rule. Examples serve two functions, firstly as example scenarios for readers to understand when the outcome of a rule is `passed`, `failed`, or `inapplicable`. It also serves developers and users of accessibility testing tools and methodologies to validate that a rule is correctly implemented.

Each `passed` and `inapplicable` example of an ACT Rule *MUST* satisfy all the rule's conformance requirements. For each `failed` example, all conformance requirements *MUST* be *not satisfied*. For composite rules, all examples of input rules *MUST* satisfy those same requirements for the conformance requirements of the composite rule.

EXAMPLE 26

HTML examples for a rule that checks if `img` elements have a text alternative:

Example of a `passed` outcome:

```
<img alt="W3C Logo" src="image/w3c.png">
```

Example of a `failed` outcome:

```
<img src="image/w3c.png">
```

Example of an `inapplicable` outcome:

```
<input type="image" alt="W3C Logo" src="image/w3c.png">
```

4.10. Rule Versions

It is important to keep track of versions of ACT Rules so that users of the rules can understand if changes in test results are due to changes in the rules used when performing the tests, or from changes in the content itself. All previous versions of an ACT Rule *MUST* be recorded either in the rule document or referenced from it.

This section was previously named "Change Log". Either section name is acceptable.

4.11. ACT Rules Format Version

An ACT Rule is written for a specific version of the ACT Rules Format. Each ACT Rule *MUST* indicate the version of the ACT Rules Format for which it is written. This is important because the ACT Rules Format is not fully backward compatible.

4.12. Glossary

ACT Rules *MUST* have a glossary section. The glossary *MUST* contain the outcome definition, as well as any definitions used in applicability and expectations sections in the rule. Since changes to the definition change the rule, those definitions cannot be maintained independently of the rule. If a shared glossary is used for rules, any definition changes *MUST* result in a new rule version of all rules that use that definition.

> **Note:** Rules can use multiple glossaries, some in the rule, and some outside. These can also be glossaries in other documents such as WCAG 2.2 or HTML 5. Where an external glossary term can change without a change to the rule, some indication of the glossary term's version is necessary. For example the HTML 5 standard is regularly updated, so a date can be added to the link to indicate the version that was used in authoring the rule.

4.13. Issues List (optional)

An ACT Rule *MAY* include a list or a reference to a list of any known issues. The issues list would be used to record cases where an outcome of an ACT Rule was `failed` where it ought to have been `passed` or `inapplicable`, or vice versa. There are several reasons why this might occur. See rule accuracy for more information.

The issues list serves two purposes. For users of ACT Rules, the issues list gives insight into why an inaccurate result occurred, as well as provide confidence in the result of that rule. For the designer of the rule, the issues list is also useful to plan future updates to the rule. In a new version of the rule, resolved issues would be moved to the change log.

4.14. Implementations (optional)

An ACT Rule *MAY* contain a list of implementations, where each implementation has one or more checks that are consistent or partially consistent with that rule. An implementation is an accessibility test methodology or test tool that uses checks to compute outcomes indicating whether an accessibility requirement could be satisfied. These checks can be fully automated, completely manual, or some combination of the two.

Checks are not required to have a one-to-one mapping to an ACT Rule. A single check can be consistent with multiple ACT Rules or a set of checks can together be consistent with a single ACT Rule.

For each implementation, information such as the following could be included:

- Name, title or identifier of any consistent or partially consistent checks
- Consistency
- test modes of the checks (e.g. automatic, manual, semiAuto)
- Name of implementation
- Version used in testing consistency and coverage
- Name of the vendor

4.14.1. Consistency

Consistency describes how well a check is aligned with the intent of an ACT Rule. If consistency is included in an ACT Rule it *MUST* be determined as defined in this section. A consistent check of an ACT Rule is one that can be used to identify some or all non-conformance issues as described by the rule. A check is ***consistent*** when all the following are true:

1. ***True positives***: There are no inconsistencies with the examples, meaning:
    1. The `passed` and `inapplicable` examples are **not** reported as `failed`; and
    2. The `failed` examples are **not** reported as `passed` or `inapplicable`; and
2. ***Completeness***: There are no gaps in coverage, meaning:
    1. None of the check's outcomes are `untested`; and
    2. At least one of the `failed` examples is reported as `failed`; and
3. ***Rule Mapping***: The accessibility requirements are reported consistently, meaning:
    1. The check is reported as testing for all the rule's conformance requirements, except requirements of a level or standard not supported by the implementation; and
    2. All accessibility requirements the rule reports to be a part of are either conformance requirements or secondary requirements, except for requirements of accessibility requirements documents not mentioned in the rule.

When a check reports more than one outcome for an example, the first outcome that appears on the following ordered list is considered for determining consistency:

1. `failed`
2. `untested`

3. `cantTell`
4. `passed`
5. `inapplicable`

§ 4.14.2. Partial Consistency

A check that is not consistent is considered ***partially consistent*** if the true positive condition is true and not all outcomes it reports for the rule's examples are `cantTell` or `untested`.

§ 4.14.3. Consistency with Multiple Checks

An implementation can include checks that test only parts of a single ACT Rule. The combination of those checks can be consistent if all parts of the ACT Rule are covered. The consistency of a ***set of checks*** can be determined by treating all partially consistent checks as though they were a single check.

A set of checks is consistent with an ACT Rule if it meets all requirements for a consistent check. The outcomes for this set of checks is a list of all outcomes of checks that are partially consistent with the ACT Rule. The accessibility requirements of this set of checks is a list of all accessibility requirements of checks that are partially consistent with the ACT rule.

EXAMPLE 27

Consistent set of checks:

> An ACT Rule checks that all native button elements, and elements with a role="button" have an accessible name. The ACT Rule has examples including both native buttons and role="button" elements.
>
> The ACME Conformance Tool has a check for native button elements. This check passes all native buttons with an accessible name, it fails all native buttons without an accessible name, and gives inapplicable for pages with only elements with role="button". Because it reports inapplicable for some of the failed examples of the ACT Rule, this check is partially consistent.
>
> ACME Conformance Tool has a second check for elements with role="button". This works similarly to the previous check, except it passes and fails role="button" elements, and is inapplicable for pages with only native buttons. This check is also partially consistent. The combination of these two checks is consistent though, because all failed examples of the ACT Rule are failed by one of the checks.

4.15. Acknowledgments (optional)

An ACT Rule *MAY* contain acknowledgments. This can include, but is not limited to:

- List of rule authors
- List of rule reviewers/contributors
- Funding or other support

5. Rule Accuracy

This section is *non-normative*.

While examples can be used to determine if an ACT Rule was correctly implemented, they do not guarantee that implementations will never produce incorrect results. When writing ACT Rules, it is almost inevitable that edge cases will be overlooked. Technologies are always evolving, and content authors are constantly coming up with new and unexpected ways to use them. Some examples of causes for inaccuracy are:

- Assumptions were made about the test subject that turned out to be untrue
- Technologies were used in an unusual and difficult to predict manner
- Technologies have changes, or aspects of the technologies were overlooked
- The accessibility requirement was not correctly interpreted

There are two types of inaccuracies that can produce incorrect results. Inaccuracies in the **implementation** can be addressed with examples, but inaccuracies in the **ACT Rule itself** cannot. After all, rule inaccuracies come from the rule author being unaware of a particular edge case.

When a test result incorrectly indicates non-conformance to an accessibility requirement, this is known as a false positive. Opposite, when a rule incorrectly indicates conformance, this is a false negative. A percentage of false positives and false negatives can be calculated by comparing it to results from an accessibility audit:

- **False positives:** This is the percentage of test targets, that were `failed` by the rule, but satisfy the accessibility requirements.
- **False negatives:** This is the percentage of test targets, that were `passed` by the rule, but do not satisfy the accessibility requirements.

The ever present possibility of false positives and false negatives with ACT Rules means they will likely require ongoing maintenance. Designing a process for maintaining ACT Rules is outside the scope of the ACT Rules Format, which is limited to the rules themselves. Nevertheless, it is suggested that rule authors work out a process for maintaining their rules.

§ 6. Harmonization

This section is *non-normative*.

While the ACT Rules Format is designed to stimulate harmonization, there are no direct requirement in the ACT Rules Format that prevent a rule author from writing rules inconsistent with already established ACT Rules. Neither are there requirements for ACT Rules to have a certain number of implementations, or to have a certain level of accuracy. This allows quality requirements to be different for different rulesets, and allows them to develop over time.

Harmonization occurs when a group of rule implementors collectively accept the validity of an ACT Rule. For example, a community group of accessibility testing tool vendors could declare they have harmonized on a particular set of ACT Rules. Such a group can set acceptance criteria for rules, and have quality requirements that go beyond the ACT Rules Format.

EXAMPLE 28

Acceptance criteria for a group working on EPUB rules:

> - An ACT EPUB Rule is harmonized when it is approved by members of 3 organizations, AND
> - An ACT EPUB Rule is harmonized when it has 2 independent implementations

An example of such a process is the WCAG ACT Review Process.

7. Privacy Considerations

This specification provides Accessibility Conformance Testing (ACT) Rules to test content conformity with the Web Content Accessibility Guidelines. This specification is not intended to access user information or expose user information directly to user agents or assistive technologies.

8. Security Considerations

This specification does not provide means for protocols, domains, and ports to communicate directly with underlying platforms (including browsers and operating systems), neither does it allow access to device sensors. The specification does not enable new script execution/loading mechanisms. This Group is not aware of other security considerations.

9. Definitions

Accessibility requirement

An accessibility requirement is a requirement aimed at improving access for people with disabilities to an ICT product. In the context of ACT rules mapping, a requirement can be compulsory or advisory. When compulsory, it has to be satisfied in order to conform to a standard, or to comply with a contract, policy or regulation. When advisory, it is recommended, but not satisfying it does not lead to non-conformance or non-compliance.

A common example of accessibility requirements are the WCAG success criteria. There are other standards, including W3C standards, that have recommendations for accessibility, such as WAI-

ARIA and HTML. Accessibility requirements are also often found in company policies, regional standards or in legislation.

Accessibility requirements document

A document, such as a standard, contract, policy or regulation, that includes accessibility requirements. For example, WCAG 2.2, WAI-ARIA 1.2, HTML 5.2, EPUB Accessibility 1.0, BBC Guide to Accessible HTML Documents

Check

A procedure resulting in one or more outcomes when used to evaluate the accessibility of a web page or other test subject. The procedure can be a step by step description of how to manually perform a test, a fully automated test, or some combination of manual and automated testing.

Implementation

An accessibility test methodology or test tool, containing a set of checks which can be used to determine (non-)conformance of accessibility requirements.

Outcome

A conclusion that comes from evaluating an ACT Rule on a test subject or one of its constituent test target. An outcome can be one of the five following types:

- **inapplicable:** No part of the test subject matches the applicability
- **passed:** A test target meets all expectations
- **failed:** A test target does not meet all expectations
- **cantTell:** Whether the rule is applicable, or whether all expectations were met could not be fully determined by the tester.
- **untested:** The test subject was not evaluated for the rule.

> **Note:** A rule has one `passed` or `failed` outcome for every test target. When a tester evaluates a test target it can also be reported as `cantTell` if the rule cannot be tested in its entirety. For example, when applicability was automated, but the expectations have to be evaluated manually.
>
> When there are no test targets the rule has one `inapplicable` outcome. If the tester is unable to determine whether there are test targets there will be one `cantTell` outcome. And when no evaluation has occurred the test target has one `untested` outcome. This means that each test subject always has one or more outcomes.
>
> Outcomes used in ACT Rules can be expressed using the outcome property of the [EARL10-Schema].

Test Subject

A resource or collection of resources that can be evaluated by an ACT Rule.

EXAMPLE 29 ¶

Test subjects:

- An HTML page, including all embedded scripts, style and images
- An EPUB publication
- A web component file

Note: Implementers using the [EARL10-Schema] can express the test subject with the subject property

Test Target

A distinct part of the test subject, as defined by the applicability.

EXAMPLE 30 ¶

Test targets:

- Nodes within an HTML page
- A period of time within a video
- A range of characters within a text document

Note: Implementers using the [EARL10-Schema] can express the test target with the pointer property

§ Appendix 1: Expressing ACT Rule Results with JSON-LD and EARL

This section is *non-normative*.

This section provides examples of expressing results from carrying out ACT Rules using EARL and JSON-LD. For details, see Evaluation and Report Language and A JSON-based Serialization for

Linked Data (JSON-LD). More examples and background is provided on the JSON-LD serialization of EARL GitHub repository.

Examples in this section include:

- Minimal outcome from one assertion
- Results from more than one assertion
- Aggregating based on requirement
- Aggregating based on 'Test Subject'
- Assumed context for this section

EXAMPLE 31 ¶

Minimal outcome for one assertion

```
{
  "@context": "context.json",
  "@type": "Assertion",
  "assertedBy": "https://example.org/MyTool",
  "subject": "https://example.org/page1.html",
  "test": "ACT-CG:rules/23a2a8",
  "result": {
    "outcome": "earl:failed",
    "pointer": "html > body > h1:first-child"
  }
}
```

EXAMPLE 32

Results for more than one assertion

```
{
  "@context": "context.json",
  "@graph": [{
    "@type": "Assertion",
    "assertedBy": "https://example.org/MyTool",
    "subject": "https://example.org/page1.html",
    "test": "ACT-CG:rules/23a2a8",
    "result": {
      "outcome": "earl:failed",
      "pointer": "html > body > h1:first-child"
    }
  }, {
    "@type": "Assertion",
    "assertedBy": "https://example.org/AnotherTool",
    "subject": "https://example.org/page1.html",
    "test": "ACT-CG:rules/23a2a8",
    "result": {
      "outcome": "earl:passed",
      "pointer": "html > body > h1:nth-child(2)"
    }
  }]
}
```

EXAMPLE 33 ¶

Aggregating based on requirement (eg. WCAG Success Criteria)

```
{
  "@context": "context.json",
  "@type": "Assertion",
  "assertedBy": "https://example.org/MyTool",
  "subject": "https://example.org/page1.html",
  "test": {
    "@type": "earl:TestRequirement",
    "@id": "WCAG22:non-text-content"
  },
  "result": {
    "outcome": "earl:failed",
    "source": [{
      "test": "ACT-CG:rules/23a2a8",
      "result": {
        "outcome": "earl:failed",
        "pointer": "html > body > h1:first-child"
      }
    }, {
      "test": "ACT-RULES-CG:rules/23a2a8",
      "result" : {
        "outcome": "earl:passed",
        "pointer": "html > body > h1:nth-child(2)"
      }
    }]
  }
}
```

EXAMPLE 34

Aggregating based on 'Test Subject' (eg. for a website)

```
{
  "@context": "context.json",
  "@type": "Assertion",
  "assertedBy": {
    "@type": "Organization",
    "@id": "_:myOrg",
    "title": "My Organization",
    "description" : "Accessibility testing service",
    "homepage" : "http://example.org/myOrg/"
  },
  "subject": {
    "@type": ["WebSite", "TestSubject"],
    "@id": "https://example.org/"
  },
  "test": {
    "@type": "earl:TestRequirement",
    "@id": "http://www.w3.org/WAI/WCAG2A-Conformance"
  },
  "result": {
    "outcome": "earl:failed",
    "source": [{
      "test": {
        "@type": "earl:TestRequirement",
        "@id": "WCAG22:non-text-content"
      },
      "result": {
        "outcome": "earl:failed",
        "source": [ … ]
      }
    }, {
      "test": {
        "@type": "earl:TestRequirement",
        "@id": "WCAG22:audio-only-and-video-only-prerecorded"
      },
      "result": {
        "outcome": "earl:passed",
        "source": [ … ]
      }
    }, {
```

```
      "test": {
        "@type": "earl:TestRequirement",
        "@id": "WCAG22:captions-prerecorded"
      },
      "result" : {
        "outcome": "earl:passed",
        "source": [ … ]
      }
    }, … ]
  }
}
```

EXAMPLE 35

Assumed context for this section

```
{
  "@vocab": "http://www.w3.org/ns/earl#",

  "cnt": "http://www.w3.org/2011/content#",
  "dct": "http://purl.org/dc/terms/",
  "earl": "http://www.w3.org/ns/earl#",
  "foaf": "http://xmlns.com/foaf/0.1/",
  "http": "http://www.w3.org/2011/http#",
  "ptr": "https://www.w3.org/2009/pointers#",
  "schema": "http://schema.org/",
  "xsd": "https://www.w3.org/2001/XMLSchema#",

  "WCAG20": "https://www.w3.org/TR/WCAG20#",
  "WCAG22": "https://www.w3.org/TR/WCAG22#",
  "ACT-RULES-CG": "https://act-rules.github.io/",

  "WebSite": "schema:WebSite",
  "WebPage": "schema:WebPage",

  "title": "dct:title",
  "description": "dct:description",
  "date": "dct:date",
  "hasVersion": "dct:hasVesion",
  "isPartOf": "dct:isPartOf",
  "hasPart": "dct:hasPart",
  "source": "dct:source",

  "Agent": "foaf:Agent",
  "Person": "foaf:Person",
  "Organization": "foaf:Organization",
  "Group": "foaf:Group",
  "Document": "foaf:Document",
  "name": "foaf:name",
  "firstName": "foaf:firstName",
  "surname": "foaf:surname",
  "mbox": "foaf:mbox",
  "mbox_sha1sum": "foaf:mbox_sha1sum",
  "member": "foaf:member",
  "homepage": "foaf:homepage",
```

```
  "assertedBy": { "@type": "@id" },
  "subject": { "@type": "@id" },
  "test": { "@type": "@id" },
  "mode": { "@type": "@id" },
  "outcome": { "@type": "@id" },
  "pointer": {
    "@id": "earl:pointer",
    "@type": "ptr:CSSSelectorPointer"
  }
}
```

Appendix 2: Acknowledgments

This section is *non-normative*.

Participants of the AG WG who Have Contributed to the Development of This Document

Shadi Abou-Zahra, Trevor Bostic, Helen Burge, Tom Brunet, Romain Deltour, Catherine Droege, Kathy Eng, Wilco Fiers, Alistair Garrison, Kasper Isager, Shunguo Yan, Todd Libby, Chris Loiselle, Sage Keriazes, Maureen Kraft, Daniel Montalvo, Mary Jo Mueller, Jey Nandakumar, Charu Pandhi, Stein Erik Skotkjerra, Suji Sreerama, Anne Thyme Nørregaard, and Kathleen Wahlbin.

Enabling Funders

This publication has been developed with support from the WAI-Tools Project, co-funded by the European Commission (EC) under the Horizon 2020 Program (Grant Agreement 780057). The content of this publication does not necessarily reflect the views or policies of the European Commission (EC) or any of the European Union (EU) Member States.

Appendix 3: Change History

This section is *non-normative*.

§ Changes Since ACT Rules Format 1.1 First Public Working Draft

- **2 Scope** Clarify that ACT Rules Format 1.1 is not fully backward compatible with 1.0
- **4.9 Examples** Clarify that input rule test cases must be consistent with conformance requirements
- **All** Test cases renamed to examples
- **All** Examples now only follow Bikeshed's automatic numbering
- **Appendix 1: Expressing ACT Rule results with JSON-LD and EARL** Add markup for the examples in this section to render correctly
- **4.2. Rule Description** Add assumption that text has metadata specifying its language and writing direction
- **4. ACT Rules Structure** Reinforce that ACT rules are localizable
- **4.1. Rule Identifier** Clarifications for determining if a rule identifier is unique
- **4.4.1.2. Secondary Requirements** Secondary requirements explanation no longer required in background section.
- **9. Definitions** Update cantTell and Untested definitions.
- **4.12. Glossary** Allow for internal and external
- **7. Privacy Considerations** Added
- **8. Security Considerations** Added

§ Changes Since ACT Rules Format 1.0

- **4. Rule Structure** Accessibility Support and Assumptions sections are now subsections of Background.
- **4.4. Accessibility Requirements Mapping** Accessibility requirements are now categorized as either Conformance requirements or Secondary requirements.
- **4.6 Applicability** Subjective applicability statements are now allowed. Objective and plain language requirements have been reduced to *SHOULD* instead of *MUST*.
- **4.8. Background** New optional Related Rules and Other Resources subsections have been added
- **4.10. Change Log** The Change Log has been renamed to Rule Versions
- **4.11 ACT Rules Format Version** New requirement to identify ACT Rules Format version compatibility

- **4.14. Implementations** New section has been added, including a method for determining consistency with ACT Rules
- **7. Definitions** The Outcome definition is updated to include cantTell and untested
- **Overall** Links to W3C specifications are updated to their latest recommendation

All changes from the previous published version can be viewed using the Editor's draft to Rules Format 1.0 Recommendation diff link

§ Conformance

§ Document conventions

Conformance requirements are expressed with a combination of descriptive assertions and RFC 2119 terminology. The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in the normative parts of this document are to be interpreted as described in RFC 2119. However, for readability, these words do not appear in all uppercase letters in this specification.

All of the text of this specification is normative except sections explicitly marked as non-normative, examples, and notes. [RFC2119]

Examples in this specification are introduced with the words "for example" or are set apart from the normative text with `class="example"`, like this:

EXAMPLE 36 ¶

This is an example of an informative example.

Informative notes begin with the word "Note" and are set apart from the normative text with `class="note"`, like this:

Note, this is an informative note.

§ Conformant Algorithms

Requirements phrased in the imperative as part of algorithms (such as "strip any leading space characters" or "return false and abort these steps") are to be interpreted with the meaning of the key

word ("must", "should", "may", etc) used in introducing the algorithm.

Conformance requirements phrased as algorithms or specific steps can be implemented in any manner, so long as the end result is equivalent. In particular, the algorithms defined in this specification are intended to be easy to understand and are not intended to be performant. Implementers are encouraged to optimize.

Index

Terms defined by this specification

Terms defined by reference

[I18N-GLOSSARY] defines the following terms:

localisation

§ References

§ Normative References

[I18N-GLOSSARY]
Richard Ishida; Addison Phillips. *Internationalization Glossary*. 17 October 2024. NOTE. URL: https://www.w3.org/TR/i18n-glossary/

[RFC2119]
S. Bradner. *Key words for use in RFCs to Indicate Requirement Levels*. March 1997. Best Current Practice. URL: https://datatracker.ietf.org/doc/html/rfc2119

§ Informative References

[ACCNAME-AAM-1.1]
Joanmarie Diggs; Bryan Garaventa; Michael Cooper. *Accessible Name and Description Computation 1.1*. 18 December 2018. REC. URL: https://www.w3.org/TR/accname-1.1/

[CSS-2018]
Tab Atkins Jr.; Elika Etemad; Florian Rivoal. *CSS Snapshot 2018*. 22 January 2019. NOTE. URL: https://www.w3.org/TR/css-2018/

[DOM]
Anne van Kesteren. *DOM Standard*. Living Standard. URL: https://dom.spec.whatwg.org/

[EARL10-Schema]
Shadi Abou-Zahra. *Evaluation and Report Language (EARL) 1.0 Schema*. 2 February 2017. NOTE. URL: https://www.w3.org/TR/EARL10-Schema/

[EPUB-33]
Ivan Herman; Matt Garrish; Dave Cramer. *EPUB 3.3*. 13 January 2026. REC. URL: https://www.w3.org/TR/epub-33/

[HTML]
Anne van Kesteren; et al. *HTML Standard*. Living Standard. URL: https://html.spec.whatwg.org/multipage/

[HTTP11]
R. Fielding, Ed.; M. Nottingham, Ed.; J. Reschke, Ed.. *HTTP/1.1*. June 2022. Internet Standard. URL: https://httpwg.org/specs/rfc9112.html

[STRING-META]
Richard Ishida; Addison Phillips. *Strings on the Web: Language and Direction Metadata*. 17 October 2024. NOTE. URL: https://www.w3.org/TR/string-meta/

[SVG2]

Amelia Bellamy-Royds; et al. *Scalable Vector Graphics (SVG) 2*. 4 October 2018. CR. URL: https://www.w3.org/TR/SVG2/

[UAAG20]

James Allan; et al. *User Agent Accessibility Guidelines (UAAG) 2.0*. 15 December 2015. NOTE. URL: https://www.w3.org/TR/UAAG20/

[WAI-ARIA]

James Craig; Michael Cooper; et al. *Accessible Rich Internet Applications (WAI-ARIA) 1.0*. 20 March 2014. REC. URL: https://www.w3.org/TR/wai-aria/

[WCAG22]

Michael Cooper; et al. *Web Content Accessibility Guidelines (WCAG) 2.2*. 12 December 2024. REC. URL: https://www.w3.org/TR/WCAG22/