Responses to public comments on WCAG2ICT

This document is where the TF will develop draft responses to substantive issues brought up in the wide review of the 2 July publication of WCAG2ICT.

Issue 394 - SC Problematic for Closed Functionality 1.4.10 Reflow: Note should include “or content”

This change was suggested during the AG WG review of Reflow, as there was concern that some might assume that “viewport” equated to a “window” but that there could be other resizable controls or areas within a window where reflow is needed, such as a resizable multi-line entry field, listbox, etc.

Proposed changes to the 1.4.10 bullet in SC Problematic For Closed Functionality

Option 1: Do not change further

The current text for 1.4.10 Reflow is:

- 1.4.10 Reflow — Some software on ICT with closed functionality does not support scrolling content, or zooming, or changing the viewport (examples include, but are not limited to, software for self-service transaction machines or kiosks). Therefore, some other non-WCAG requirements would be needed for products with closed functionality to ensure that content is readable by persons with low vision without scrolling in two dimensions.

Option 2: Change to cover cases discussed in the issue

- 1.4.10 Reflow — Some software on ICT with closed functionality does not support scrolling content, or zooming, or changing the viewport (examples include, but are not limited to, software for self-service transaction machines or kiosks). Therefore, some other non-WCAG requirements would be needed for products with closed functionality to ensure that content is readable by persons with low vision without scrolling in two dimensions.

Option 3: Something else

Add your alternate proposal here.

Proposed changes to SC 1.4.10 Reflow, Note 5

Option 1: Do not change the existing text

The following is what is currently in the editor's draft for Note 5.

NOTE 5 (ADDED): The intent section refers to the ability for content to reflow (for vertical scrolling content at a width equivalent to 320 CSS pixels, or for horizontal scrolling content at a height equivalent to 256 CSS pixels) when user agent zooming is used to scale content or when the viewport changes in width. For non-web software, this means that when users scale content, adjust the size of a window or dialog, or change the screen resolution, the content will reflow without loss of information or functionality, and without requiring scrolling in two dimensions; or that the application works with platform features that meet this success criterion.

Option 2: Slight modification to include resizable controls/areas.

NOTE 5 (ADDED): The intent section refers to the ability for content to reflow (for vertical scrolling content at a width equivalent to 320 CSS pixels, or for horizontal scrolling content at a height equivalent to 256 CSS pixels) when user agent zooming is used to scale content or when the viewport changes in width. For non-web software, this means that when users scale content, adjust the size of a window, dialog, or change the screen resolution, the content will reflow without loss of information or functionality, and without requiring scrolling in two dimensions; or that the application works with platform features that meet this success criterion.

Option 3: Something else

Add your proposal here.

Proposed answer to Issue 394:

Option 1: No changes

Use this answer if we make no changes.

We have reviewed the 1.4.10 Reflow bullet in the SC Problematic for Closed Functionality section and the section titled Applying SC 1.4.10 Reflow to Non-web Documents and Software, and feel that there are no changes needed. The definition of "viewport" covers the cases of scrollable widgets and areas without further explanation needed.

Option 2: Changes are being made

Depending on whether both changes are approved or only one, the answer will have to be adjusted. This answer is written assuming both changes are approved. We can remove one or the other if one of them isn't approved.

The task force has made clarifications to the SC Problematic for Closed Functionality and Applying SC 1.4.10 Reflow to Non-web Documents and Software sections. In both cases we

have added text to make it clear this SC applies to scrollable elements as well. See PR @@@fill in PR # and link here@@ for the exact changes made. You can read the changes in-context in the editor's draft in Applying SC 1.4.10 Reflow to Non-web Documents and Software (Note 5) and SC Problematic for Closed Functionality (the bullet for 1.4.10 Reflow).

@mraccess Please review to ensure these improvements address your concern.

Option 3: Something else

Add your proposal here.

DONE: Issue 395 - Platform software definition

Issue suggests using EN 301 549 definition alone. Subsequent PR added references to both of the ISO standards this definition derived from (considering that the EN 301 549 and US Revised 508 Standards used different ISO definitions).

Proposed answer to Issue 395

In the development of the WCAG2ICT definition of "platform software", we looked to the EN 301 549, US Revised 508, and ISO standards.

- US Revised 508 Standards - Uses both ISO 9241-171:2008 and ISO/IEC 13066-1:2011
- EN 301 549 - Uses only ISO/IEC 13066-1:2011, but removed the examples

The WCAG2ICT Task Force used the definition that follows the convention of being concise and when substituted into a sentence containing the term (from ISO 9241-171:2008). We also included the examples provided by ISO. The resulting definition is a combination of what appears in the EN and 508 definitions.

We have also added document references and a note that indicates the ISO standards the definition derived from by adding this text to the definition:

> This definition is based on the definition of "platform software" found in [ISO_9241-171] and [ISO/IEC_13066-1].

See the changes in-context in the Key terms definition of "platform software".

We are closing this issue as completed. The changes are in the latest Editor's draft.

DONE: Issue 396 - virtual keyboard (as used in WCAG2ICT) - change out “like”

Issue suggests the following edit:

virtual keyboard (as used in WCAG2ICT)

any software that acts as a keyboard and generates output that is treated by other software like keystrokes from a keyboard".

Proposed answer to Issue 396

Thank you for your comment @ljoakley. We have made the modification to the editor’s draft with the change you suggested, replacing “like” with “as.” The definition now reads:

virtual keyboard (as used in WCAG2ICT)
any software that acts as a keyboard and generates output that is treated by other software as keystrokes from a keyboard".

Issue 397 - Rework the “virtual keyboard” Note for Types of Input

Issue says the user is presented with a list of input modes with the explanation of why they are listed at the end of the sentence. Issue suggests changing the Note on “virtual keyboard” definition to:

All of the following, and not limited to,have all been used by virtual keyboards as input that generates "keystroke" output: Speech, eye-gaze, sip-and-puff, sounds, switches, and codes.

Current text in the definition of “virtual keyboard”

The term **virtual keyboard**, as used in WCAG2ICT, has the meaning below:

virtual keyboard (as used in WCAG2ICT)

> any software that acts as a keyboard and generates output that is treated by other software like keystrokes from a keyboard

NOTE: Speech, eye-gaze, sip-and-puff, sounds, switches, and codes have all been used by virtual keyboards as input that generates "keystroke" output.

Proposed changes to the Note to address Issue 397

Option 1: Do not change the text. It is clear enough as-is.

NOTE: Speech, eye-gaze, sip-and-puff, sounds, switches, and codes have all been used by virtual keyboards as input that generates "keystroke" output.

Option 2: Proposed changes in PR 445

NOTE: Speech, eye-gaze, sip-and-puff, sounds, switches, and codes have all been used by virtual keyboards as input that generates "keystroke" output.

Clean read of 2: Example: Speech, eye-gaze, sip-and-puff (and other kinds of switches), sounds, and morse code have all been used as ways to generate keystroke output.

Option 3: Change the Note to put the “why” first (as suggested in the issue)

This proposal also takes some of the edits from PR 445.
NOTE: Speech, eye-gaze, sip-and-puff, sounds, switches, and codes have all been used by virtual keyboards as input that generates "keystroke" output.

Clean 3: Example: Some ways to generate "keystroke" output include virtual keyboards that take input from various sources such as speech, eye-gaze, sip-and-puff (and other kinds of switches), sounds, morse code, and so on.

Option 4: Proposal from Gregg in the issue

NOTE: Speech, eye-gaze, sip-and-puff, sounds, switches, and codes have all been used by virtual keyboards as input that generates "keystroke" output.

Clean read of 4: NOTE: Some of the many ways to generate keystroke input include speech, eye-gaze, sip-and-puff (and other kinds of switches), sounds, morse code, and, of course, keyboards (small, large, physical, on-screen, floating in the air, etc.)

Option 5: Add alternate proposal here

Provide info on which proposal this is modified from, and add the text with modifications indicated.

Option 5a: Proposal from Olivia - reword Option 3, put list in alphabetical order

Example: Some common ways to generate "keystroke" output for virtual keyboards include eye-gaze, morse code, sounds, speech, and switches (*e.g.*, sip-and-puff).

Option 5b: Should be input from keyboards -- since that is what the SC is about. So 5a becomes 5b

Example: Some common ways to generate "keystroke" input from virtual keyboards include eye-gaze, morse code, sounds, speech, and switches (*e.g.*, sip-and-puff).

Proposed answer to go into Issue 397

Option 1: Initial proposed answer to Issue 397

Thank you for your review and comment. We have updated the definition of “virtual keyboard” to make the sentence you identified easier to read. It has been changed to:

@@put in the TF agreed text here

It can be read in-context in the editor’s draft definition of “virtual keyboard”.

Option 2: Add alternate proposal here

Thank you for your review and comment. We have decided not to update the definition of “virtual keyboard”.

Issue 437 - Success Criterion Applying SC 2.4.2 Page Titled to Non-Web Documents and Software

Current text in the draft for 2.4.2

6.2.5.3.1 Applying SC 2.4.2 Page Titled to Non-Web Documents and Software

This applies directly as written, and as described in Intent from Understanding Success Criterion 2.4.2 replacing “Web pages” with “non-web documents or software”.

With this substitution, it would read:

2.4.2 Page Titled: [Non-web documents or software] have titles that describe topic or purpose.

NOTE 1: As described in the WCAG intent, the name of a non-web software application or non-web document (e.g. document, media file, etc.) is a sufficient title if it describes the topic or purpose.

NOTE 2: See also the Comments on Closed Functionality.

Proposed changes (if any)

On 11 July, the TF discussed this issue and thought no changes were needed. However, since that time this has been an active issue with multiple postings.

Option 1: New note indicating “best practice”

One possible addition could be a note that is similar to what is added to “sets of” criteria:

Note: Although not required by this success criterion, ensuring that individual windows or screens have a title that describes the topic or purpose addresses the user needs identified in the Understanding Success Criterion 2.4.2 Intent section, and is generally considered a best practice.

Option 2: Add a new note from Mitch’s comment in Issue 437

Note: When software content is presented as separate screens that resemble pages, and the technology supports a separate title for each screen, it is a best practice to provide screen titles and ensure that they describe the topic or purpose of each screen. Where screen titles are provided, a title for the overall software program is still necessary.

Option 3: Something else - add your proposal title

Add text of alternate proposal here.

Proposed answer to the issue 437:

Option 1: The answer initially added to the issue

Thank you @stevefaulkner for the feedback. The TF discussed and we have consensus that 2013 approach is appropriate and sufficient. Please see 2.4.2 Page Titled in the editor's draft.

Option 2: Answer to use IF the TF approves any changes

Appreciate your comment @stevefaulkner. The TF has agreed to add a note to the editor’s draft to indicate that when an application has different views or windows it is a best practice for them to have a title. The exact verbiage we have added to the section Applying 2.4.2 Page Titled to Non-web Documents and Software is:

@@Add the quoted note we agree upon here.@@

Option 3: Something else

Add text of alternate proposal here.

Issue 412 - Some WCAG2ICT content after word substitution does not match WCAG

Issues:

This content in WCAG2ICT was created by copy-and-paste from WCAG:

- success criteria with word substitutions
- glossary terms with word substitutions

In some cases, copy-and-paste based content was created before changes were made to WCAG. Examples of changes:

- WCAG 2.1 errata corrections
- Any other changes not listed in the errata corrections (e.g., the change to 'changes of context')

In these cases, the errata corrections are appearing in WCAG2ICT in content pulled automatically from WCAG, but the errata corrections are not appearing in WCAG2ICT in the older copy-and-paste based content.

I pulled out all WCAG2ICT content with word substitutions, and diffed them with WCAG. The following differences should be fixed in WCAG2ICT.

1.4.2 Audio Control: In WCAG the first instance of 'mechanism' is a link. In WCAG2ICT the second instance of 'mechanism' is a link.

1.4.10 Reflow: Note 2 differs

2.2.2 Pause, Stop, Hide: "Content" in Notes 2 and 3 has glossary links added in WCAG2ICT, which are not there in WCAG.

2.4.3 Focus Order: 'navigated sequentially' should be a glossary link

2.5.7 Dragging Movements: WCAG has four links to definitions that are not in WCAG2ICT

2.5.8 Target Size (Minimum):

- WCAG has "target" where WCAG2ICT has "target and target offset"
- The last list item should end with a period, not a semicolon
- In Note 1, WCAG has "sliders" where WCAG2ICT has "sliders with granular values"

3.3.4 Error Prevention (Legal, Financial, Data): WCAG2ICT has added glossary links, which are not there in WCAG.

Definition of 'accessibility supported': In bullet (1), 'content' has a glossary link added in WCAG2ICT, which is not there in WCAG

Definition of 'changes of context':

- WCAG changed "major changes in the content of the Web page" to "major changes"
- In bullet (4), 'content' has a glossary link in WCAG, which is not there in WCAG2ICT

Definition of 'relative luminance':

    - In WCAG2ICT, in Note 1, the text before the first unordered list has extra slash and asterisk characters.
    - WCAG changed the red threshold from 0.03928 to 0.04045
    - At the end of Note 1, ")." should be ".)"
    - WCAG has changed Note 6.

  Definition of 'programmatically determined': "Software" has a glossary link added in WCAG2ICT, which is not there in WCAG.

  Definition of 'same functionality': Two instances of the word 'Web' should be capitalized

  Definition of 'user interface component': "content" has a glossary link added in WCAG2ICT, which is not there in WCAG.

  Definition of 'viewport':

    - "content" has a glossary link added in WCAG2ICT, which is not there in WCAG.
    - Note 2 in WCAG2ICT is missing a footnote link to WCAG.

A couple of more, not word substitutions, but direct quotes from WCAG: In 1.2.2 Captions (Prerecorded) and 1.2.4 Captions (Live), an added Note quotes the WCAG definition of 'captions'. The quotation has these issues:

- WCAG has a link to the definition of 'text alternative' but WCAG2ICT does not
- WCAG2ICT has a link to the definition of 'content' but WCAG does not
- Whitespace difference: "and/or" in WCAG, "and / or" in WCAG2ICT

Proposal

Accept the above tasks and incorporate in appropriate markdown files within WCAG2ICT code.

Issue 414 - Issues with the 'platform software' notes for 2.5.1, 2.5.2, 2.5.7

This section describes the problem space.

Note as it appears in WCAG:
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).

The note appears in three WCAG SCs:

- 2.5.1 Pointer Gestures
- 2.5.2 Pointer Cancellation
- 2.5.7 Dragging Movements

WCAG2ICT should and does apply these three SCs to the platform software itself, e.g. a browser, but it is confusing about how the three SCs apply to platform software.

However, there are issues:

- The "user agents" mentioned in Note 5 are one kind of platform software. To this extent Notes 5 and 6 are redundant.
- The "platform software" in Note 6 can also have further underlying platform software, but Note 6 does not mention this.
- (minor issue) Note 6 has these further edits not mentioned in the list of word substitutions: added "also"; removed "user agent or".

Example: Current text in 2.5.2

This example is excerpted from current WCAG2ICT content for 2.5.2, quoting just the parts relevant to platform software appears below.

This applies directly as written, and as described in Intent from Understanding Success Criterion 2.5.2, making changes to the notes for non-web documents by replacing “web content” with "content", for non-web software applications by replacing "web content that interprets" with "user agents and other software applications that interpret" and "user agent" with "underlying platform software", and for non-web platform software by replacing "web content" with "platform software".

With these substitutions, the notes would read:

(for non-web software)

Note 5: This requirement applies to [user agents and other software applications that interpret] pointer actions (i.e. this does not apply to actions that are required to operate the [underlying platform software] or assistive technology).

Note 6: This requirement also applies to [platform software] that interprets pointer actions (i.e. this does not apply to actions that are required to operate the assistive technology).

Proposed changes for SC 2.5.2 Pointer Cancellation

Note that, if approved, the same text changes would apply to 2.5.1, 2.5.2, and 2.5.7.

Option 1 (Note 5): No change to note 5

Note 5: This requirement applies to [user agents and other software applications that interpret] pointer actions (i.e. this does not apply to actions that are required to operate the [underlying platform software] or assistive technology).

Option 2 (Note 5): Use Note 5 only for applications that are not platform software

Note 5: This requirement applies to [user agents and other software applications that interpret] pointer actions (i.e. this does not apply to actions that are required to operate the [underlying platform software] or assistive technology).

Option 1 (Note 6): No change to Note 6, word substitution language

This means that the word substitutions would be unchanged.

Note 6: This requirement also applies to [platform software] that interprets pointer actions (i.e. this does not apply to actions that are required to operate the assistive technology).

Option 2 (Note 6): Use Note 6 for all platform software

This would be an additional Note (and would no longer be mentioned in the word substitutions). The verbiage for the word substitutions would be:

This applies directly as written, and as described in Intent from Understanding Success Criterion 2.5.2, making changes to the notes for non-web documents by replacing “web content” with "content", for non-web software applications by replacing "web content that interprets" with "user agents and other software applications that interpret" and "user agent" with "underlying platform software", and for non-web platform software by replacing "web content" with "platform software".

And verbiage for Note 6 would be:

Note 6: This requirement also applies to [platform software] that interprets pointer actions (i.e. this does not apply to actions that are required to operate the assistive technology).

Option 3: Something else - Add your proposal

Provide your detailed changes and if it is based on an existing proposal, indicate which one.

Issue 431 - 2.5.2: An example has been inserted into a WCAG Note and not listed as a substitution

Current Note 4 and 5 text in 2.5.2 Pointer Cancellation

In 2.5.2 Pointer Cancellation, under "(for non-web software)", Note 4 and Note 5 are notes from WCAG with word substitutions. They currently appear like this:

Current markdown:

Functions that emulate a keyboard or numeric keypad key press are considered essential.

Examples of essential functionality for non-web software are features for meeting environmental energy usage requirements (like waking a device from sleep, power saver mode, and low power state).

Issue: The example shouldn't be nested. While it is an example of the essential exception, it is not an example that would illustrate the note. The note is about keyboard while the example is about power saving modes.

Issue: The example has been inserted into a WCAG Note, but this insertion is not mentioned in the description of word substitutions.

This applies directly as written, and as described in Intent from Understanding Success Criterion 2.5.2, making changes to the notes for non-web documents by replacing “web content” with "content", for non-web software applications by replacing "web content that interprets" with "user agents and other software applications that interpret" and "user agent" with "underlying platform software", and for non-web platform software by replacing "web content" with "platform software".

Proposed changes to 2.5.2 Pointer Cancellation

NOTE: I created PR 454 so you could visualize and read in-context what the proposals would look like in the document, if approved. You'll need to scroll down to the (for non-web software) part to see the notes this talks about. Once consensus is reached, the options not chosen will be cleaned out of the PR before merging it.

Option 1: Leave example text embedded as-is, and add mention of the inserted example

Update the word substitution to read:
This applies directly as written, and as described in Intent from Understanding Success Criterion 2.5.2, making changes to the notes for non-web documents by replacing “web content” with "content", for non-web software applications by replacing "web content that interprets" with "user agents and other software applications that interpret", and "user agent" with "underlying platform software", and for non-web platform software by replacing "web content" with "platform software".

Option 2: Move the added example below Note 4, and explain in word substitution

Move the added example out of the WCAG note. Keep issue #414 in mind, which also affects this SC's notes.

Update the word substitution to read:
This applies directly as written, and as described in Intent from Understanding Success Criterion 2.5.2, making changes to the notes for non-web documents by replacing “web content” with "content", for non-web software applications by replacing "web content that interprets" with "user agents and other software applications that interpret" and "user agent" with "underlying platform software", and for non-web platform software by replacing "web content" with "platform software".

Option 3: Move the added example below all the notes

This has the disadvantage of separating the information about essential exceptions, but it has the advantage of the example not getting inserted in the middle of WCAG content.

Option 4: Something else - Add your proposal

Add your proposed text changes here.

Issue 428 - 3.2.6 Consistent Help: does it need to be added in ‘problematic for closed’

Proposed changes to SC Problematic for Closed Functionality

Option 1: Add in 2.5.2 using text similar to 2.4.1 Bypass Blocks

- 3.2.6 Consistent Help — The WCAG2ICT interpretation of this success criterion replaces "sets of Web pages" with "sets of software programs" which are extremely rare - especially for closed functionality software. However, providing consistent access to help is generally considered best practice.

Option 2: Something else - Add your proposal

Add the text of your alternate proposal.

Issue 419 - Definition of 'style property' needs different word substitution

WCAG definition of ‘style property’

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. in-line 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)

Proposed changes to WCAG2ICT '*style property*'

Option 1: Minor changes - keep existing WCAG2ICT content (with a couple of fixes)

This applies directly as written and as described in the WCAG 2 glossary, replacing "user agent(s)" with "user agent(s) or platform software", "web content" with "content", replacing "in-line styles, author style sheets" with "programmatically-set styles", and replacing "user agent interface settings, user style sheets" with "user agent, platform software or other software settings".

With these substitutions, it would read:

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 or platform software]**

Style properties can have several origins:

- **[User agent or platform software] default styles:** The default style property values applied in the absence of any author or user styles. Some **[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. **[programmatically-set styles]**);
- **User styles:** Style property values that are set by the user (e.g. via **[user agent, platform software or other software]** interface settings)

Option 2: The following substitutions (proposed by Mitch)

Description of issues and changes in GitHub with an addition of the word substitution language which would also need to change to match the proposal.

This applies directly as written and as described in the WCAG 2 glossary, replacing "user agent(s)" with "user agent(s) or platform software", "web content" with "content", replacing "in-line styles, author style sheets" with "programmatically-set styles", and replacing "user agent interface settings, user style sheets" with "user agent, platform software or other software settings".

With these substitutions, it would read:

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 or software]

Style properties can have several origins:

- [**User agent or platform software**] default styles: The default style property values applied in the absence of any author or user styles. Some [**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. [**programmatically-set styles**]);
- User styles: Style property values that are set by the user (e.g. via [**user agent, platform software or other software**] interface settings)

Option 3: Additional “or” added where Gregg suggested instead of moving the”or”

Word substitution text would be:

This applies directly as written and as described in the WCAG 2 glossary, replacing “user agent(s)” with “user agent(s) or platform software”, "web content" with "content", replacing "in-line styles, author style sheets" with "programmatically-set styles", and replacing "user agent interface settings, user style sheets" with "user agent, platform software, or other software settings".

All else is the same as proposal 2, substituting the User Styles bullet (3rd bullet) with:

- User styles: Style property values that are set by the user (e.g. via [**user agent, platform software, or other software** interface **settings],** user style sheets)

Issue 436 - The definition of ‘large scale’ needs substitutions or notes

WCAG original text

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.

Current WCAG2ICT guidance section text

‘Large scale’ is listed in the section Glossary Items that Apply to All Technologies. No word substitutions, no interpretation.

Proposed changes to ‘large scale’ definition interpretation where ‘user agent’ is used

Option 1: Leave as-is (like it was in the 2013 WCAG2ICT note)

Option 2: Provide word replacements for ‘user agent’ and ‘ in Note 3 & 4

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 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.

Option 3: Mary Jo edits - Updates to Option 2 as I tried to implement the changes

***Added on 4 Sept.** Mary Jo: Trying to address multiple issues I found:*

- *Missing a word replacement for "user agent" in Note 3.*
- *The proposal did not include the language describing the word replacements.*

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 or non-web software]** for display. The point size should be obtained from the **[user agent or non-web software]**, or calculated based on font metrics as the **[user agent or non-web software]** 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, **[user agents, or platform software]** 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, **[user agents, or platform software]**, then it would be reasonable to assume it is large text. Relative scaling can be calculated from the default sizes in a similar fashion.

Option 4: Something else

Proposed changes to 'large scale' to address 'pt' and 'point' interpretation

Option 1: Leave as-is

Option 2: Mitch proposal

NOTE: When evaluating non-web documents and software, 1 point means 1.333 CSS pixels.

Option 3: Something else

Add your alternate proposal or edits here.

Issue 432 - Need clearer distinction between WCAG notes and WCAG2ICT added notes

There's five scenarios for Notes in WCAG2ICT *Applying SC x.x.x to Non-web Documents and Software* and term definition sections.

Scenarios for SC notes (and some term definitions)

1. WCAG note(s) not copied into WCAG2ICT "*Applying SC*…" section but is still in effect, because the SC or term applies as written. WCAG2ICT doesn't make another copy of the WCAG SC text - it would be redundant.
    - Examples of when WCAG2ICT adds notes: 1.4.12 Text Spacing, 1.3.4 Orientation, definition of 'keyboard interface', 1.4.12, 2.1.1, 2.4.4, 2.5.3, "contrast ratio", "keyboard interface", "keyboard shortcut", web page
    - Examples where WCAG2ICT doesn't add any notes: 1.3.3 Sensory *Characteristics*, *2.4.11 Focus not Obscured*

    NOTE: This is potentially the most confusing case, as WCAG2ICT Notes can have the same note number(s) as WCAG notes but the WCAG2ICT notes do not replace the WCAG notes.

    Question: Would adding "WCAG2ICT" to our added notes sufficiently indicate this Note is in addition to WCAG's notes? Should it also be described in text, e.g. "Though numbered the same, the notes identified with "WCAG2ICT NOTE" are in addition to the WCAG notes for this success criterion."

2. WCAG note is copied, unchanged, into WCAG2ICT "*Applying SC*…" section with no word substitutions.
    - Example: 2.2.1 Timing Adjustable
      Unchanged WCAG notes are not marked as a WCAG2ICT note.
    - *Examples where WCAG2ICT doesn't change notes but does add notes: 2.5.8 Target size (Minimum), 3.3.8 Accessible Authentication (Minimum), "changes of context"*

3. WCAG note is copied and WCAG2ICT edits them with word substitutions.
    - Examples where WCAG2ICT edits WCAG note and adds notes: 1.4.2 Audio control, 1.4.10 Reflow, 2.1.2, 2.2.2, 2.5.1, 2.5.2, 2.5.7, 3.2.6, "name", programmatically determined, relative luminance, role, same functionality (added an example)

    - Examples where WCAG2ICT edits and does not add notes: 1.4.13 Content on Hover or Focus (Note 1 edited, others are not), 2.3.1 Three Flashes or Below Threshold

Edits to the WCAG note are described in the word substitution description. The note is NOT marked with WCAG2ICT.

4. The WCAG note is replaced by a different WCAG2ICT note (Examples: WCAG 2.2 guidance for 4.1.1 Parsing - no word replacement noted - should it be? Another example: 4.1.2 Name, Role, Value where the note replacement is clearly indicated…and there's additional notes added by WCAG2ICT, "accessibility supported" - if we mark this with WCAG2ICT, do we need to use the INS markdown to visually highlight it?), "set of Web pages"
   Word substitution description indicates the note is replaced. Should the Note also be marked with WCAG2ICT?

5. WCAG SC has no notes at all.
    - Examples where WCAG2ICT adds notes: *1.1.1 Non-text Content*, *1.2.1 Audio-only and Video-only (Prerecorded)*, 1.2.2 Captions (Prerecorded), 1.2.3, 1.2.4, 1.2.5, 1.3.1, 1.3.2, 1.3.5 Identify Input purpose, 1.4.3, 1.4.4, 1.4.5, 1.4.11, 2.1.4, 2.4.1, 2.4.2, 2.4.5, 2.4.6, 2.4.7, 3.1.1, 3.1.2, 3.2.1, 3.3.1, 4.1.3, "CSS pixel", "down-event", "general flash and red flash thresholds", programmatically determined, structure, up-event
    - Examples where WCAG2ICT doesn't add notes: 1.4.1 Use of color, 2.4.3 Focus order, 2.5.4, 3.2.2, 3.2.3, 3.2.4, 3.3.2, 3.3.3, 3.3.4, 3.3.7

   Causes no problems - no potential clashes between note numbers. When WCAG2ICT adds new notes, they should get marked for consistency and to make it clear these are additions.

Proposal 1: New notes are marked "WCAG2ICT"

Any time WCAG2ICT adds notes (that are not replacing a WCAG note), they are marked with WCAG2ICT and not mentioned in the word substitution description.

Additional term definitions problem mentioned as 2(a) in the issue

See the noted term definitions in the sidebar comments where indentation issues are noted.

Issue 424 - 4.1.1 Parsing: The WCAG 2.1 version of the SC is not quoted, yet a word substitution is provided for it

See PR #451 for a mock-up of how embedding the original text for 4.1.1 Parsing in WCAG 2.0 and 2.1 vs. the text for 4.1.1 Parsing in WCAG 2.2 might be done.

You can read the changes in-context in the built document for WCAG 2.0 & 2.1 version of Parsing and the WCAG 2.2 version of parsing.

Questions when reviewing the mock-up

- Are the headings clear?
- In the document, what order should the versions appear?
    a. WCAG 2.0/2.1 version first, followed by the WCAG 2.2 version OR
    b. WCAG 2.2 version first, followed by WCAG 2.0 & 2.1 version

Issue 383 - Adjust links in Guidance section to link to all task force and AG publications

See Issue 383.

Existing text excerpted from Guidance in this Document section

Full Guidance in this Document section, in case you wish to see it in context.

Although this document covers a wide range of issues, it is not able to address all the needs of all people with disabilities. Since WCAG 2 was developed for the Web, addressing accessibility for non-web documents and software may involve requirements and considerations beyond those included in this document. Authors and developers are encouraged to seek relevant advice about current best practices to ensure that non-web documents and software are accessible, as much as possible, to people with disabilities. The following WCAG 2 supporting documents, though they have not been changed to fully apply in non-web contexts, contain helpful information to learn about the user needs, intent, and generalized implementation techniques:

- WCAG 2 Overview
- Techniques for WCAG 2.2 [WCAG22-TECHS]
- How to Meet WCAG (Quick Reference)
- Mobile Accessibility: How WCAG 2.0 and other W3C/WAI Guidelines Apply to Mobile - This draft resource, as of the date of the WCAG2ICT Note publication, is undergoing an update by the Mobile Accessibility Task Force to cover WCAG 2.2.

Proposed changes to **text before the list** in Guidance in this Document

Option 1: Keep current text in Guidance in this Document

The following WCAG 2 supporting documents, though they have not been changed to fully apply in non-web contexts, contain helpful information to learn about the user needs, intent, and generalized implementation techniques:

Option 2: Add exact verbiage suggested in Issue 383 to introductory text

Although this document covers a wide range of issues, it is not able to address all the needs of all people with disabilities. Since WCAG 2 was developed for the Web, addressing accessibility for non-web documents and software may involve requirements and considerations beyond those included in this document. Authors and developers are encouraged to seek relevant advice about current best practices to ensure that non-web documents and software are accessible, as much as possible, to people with disabilities. The following WCAG 2 supporting documents, though they have not been changed to fully apply in non-web contexts, contain helpful information to learn about the user needs, intent, and generalized implementation techniques:

Option 3: (Gregg's)

Add your alternate proposal here.

Although the Task Force has not suggested changes in this document to apply in non-web contexts, the WCAG AAA success criteria and the following supporting documents contain helpful information to learn about the user needs, intent, and generalized implementation techniques to support a wider range of people with disabilities.

Option 4: Option 2 with AAA removed

Although this document covers a wide range of issues, it is not able to address all the needs of all people with disabilities. Since WCAG 2 was developed for the Web, addressing accessibility for non-web documents and software may involve requirements and considerations beyond those included in this document. Authors and developers are encouraged to seek relevant advice about current best practices to ensure that non-web documents and software are accessible, as much as possible, to people with disabilities. Although they have not been changed to fully apply in non-web contexts, the following supporting documents contain helpful information to learn about the user needs, intent, and generalized implementation techniques to support a wider range of people with disabilities:

Proposals for changing the **bulleted list** in the Guidance in this Document

Option 1: Keep current list in Guidance in this Document

This proposes no changes to the list - leave as-is.

Option 2: Remove mobile doc link and add to top of list a link to all AG publications

-
- WCAG 2 Overview
- Techniques for WCAG 2.2 [WCAG22-TECHS]
- How to Meet WCAG (Quick Reference)
- Mobile Accessibility: How WCAG 2.0 and other W3C/WAI Guidelines Apply to Mobile - This draft resource, as of the date of the WCAG2ICT Note publication, is undergoing an update by the Mobile Accessibility Task Force to cover WCAG 2.2.

Option 3: Remove mobile doc link and add to bottom of list the link to all AG publications

- Additional Accessibility Guidelines Working Group - Publications
- WCAG 2 Overview
- Techniques for WCAG 2.2 [WCAG22-TECHS]
- How to Meet WCAG (Quick Reference)

Option 4: Create a different evergreen page where only relevant documents are listed

Have all draft and published documents from each task force listed on a new page that would be useful reference documents. Would prefer the documents are organized by category (by task force or simply topic) rather than by document status (published, draft, FPWD). The document status could be added with a brief description of each document. Then add a link to that references page to the list, and also deleting the Mobile-specific link from the list.

- WCAG 2 Overview
- Techniques for WCAG 2.2 [WCAG22-TECHS]
- How to Meet WCAG (Quick Reference)
- Relevant publications by W3C Web Accessibility Initiative (WAI) Groups

Option 5: Link to the AG WG Task forces page

- WCAG 2 Overview
- Techniques for WCAG 2.2 [WCAG22-TECHS]
- How to Meet WCAG (Quick Reference)
-

Option 6: Link to the WAI Group Task Forces page

This is because APA task forces also cover more broad topics than simply WCAG that can help groups meet the WCAG criteria.

- WCAG 2 Overview
- Techniques for WCAG 2.2 [WCAG22-TECHS]
- How to Meet WCAG (Quick Reference)

Option 7: Link to the W3C document inventory page using “accessibility” filter

This provides a link to a page where the user can filter the full list of documents by topic of interest using keywords (e.g. mobile, cognitive, low vision, etc.) and get just a few specific documents of interest.

- WCAG 2 Overview
- Techniques for WCAG 2.2 [WCAG22-TECHS]
- How to Meet WCAG (Quick Reference)
-

Option 8: Something else

Add your proposal here.

Proposed TF response to be added in a comment in Issue 383

Option 1: If we accept all of Rachael’s changes as-is.

@rachaelbradley thank you for your review of WCAG2ICT and your comment. The WCAG2ICT task force has included your suggested changes in the guidance section as-is. You can read the changes in-context in the Guidance in this Document section of the editor’s draft.

Option 2: Assuming there are some changes to the link

@rachaelbradley thank you for your review of WCAG2ICT and your comment. The WCAG2ICT Task Force has included your editorial changes for the Guidance in this Document section as-is with the exception of the document link. We agree to remove the link to the MATF draft document but think that the *@@@insert which link we use@@@* is a preferred link because *@@@add reason, depending on which link was chosen@@@.*

Please let us know if this sufficiently addresses the issue.

Option 3: WCAG2ICT document is not changing

@rachaelbradley Thank you for your review of WCAG2ICT and your comment. While we understand the desire to provide all possible resources to readers of WCAG2ICT, we previously added the specific bullet with a link to the draft Mobile Accessibility Task Force document due to previous comments received from their the MATF Task Force and multiple questions asking about whether our two groups and documents are in synch. Therefore, the WCAG2ICT Task Force has decided to refrain from making the suggested changes to our document. We have noted in the bullet that the MATF document is a draft and is undergoing an update by the Mobile

Accessibility Task Force. You can read the current text in-context in the editor's draft section called Guidance in this Document.

Please let us know if this answer is acceptable.

Option 4: Something else

@rachaelbradley thank you for your review of WCAG2ICT and your comment. The WCAG2ICT task force agreed to remove the link to the MATF draft document. We have included your alternative link and some of the suggested changes to the sentence prior to that list (with a few edits). You can read the changes in-context in the Guidance in this Document section of the editor's draft.

Issue 394: Reflow as SC problematic for closed functionality

SC problematic for closed functionality 1.4.10 Reflow: Note should include "or content" · Issue #394 · w3c/wcag2ict (github.com)

Discuss on call 8/1. Folks in attendance concur that we are content with current treatment no change is needed.

Proposals to address issue 465

***Note, the proposals have been moved into the Horizontal Review Google doc, as we have to keep separate track of those issues.*

Issue 473: Definitions and explanations for "Set of Documents" and "Set of Software Programs" produces strange corner cases that should be addressed or explained

Proposals for answering Issue 473

Proposal 1: No change to the content, with the following proposed answer

Thank you for your question regarding WCAG2ICT. The Task Force considered making the advisory content regarding "sets of software" and "sets of documents" more formal, but decided against making any changes. While overlapping sets could occur, we do not foresee a problem in evaluating each overlapping set as a set.

Proposal 2: Something else - give it a title

Add your proposal here.

Issue 427: 4.1.1 Parsing: does it need to be added in ‘problematic for closed’?

See Issue 427.

Proposed content changes to address Issue 427

Option 0: No change - don’t add a bullet to SC Problematic for Closed Functionality

Current draft has no bullet for SC 4.1.1 Parsing in the SC Problematic for Closed Functionality section.

Option 1: Add in what was there in the 2013 WCAG2ICT, clarifying it’s only for WCAG 2.0, 2.1

- 4.1.1 Parsing—(WCAG 2.0 and 2.1 only) The Intent of 4.1.1 is to provide consistency so that different user agents or assistive technologies will yield the same result.

Option 2: Also include a WCAG 2.2 part to indicate this SC is not relevant

- 4.1.1 Parsing—
    - (WCAG 2.0 and 2.1 only) The Intent of 4.1.1 is to provide consistency so that different user agents or assistive technologies will yield the same result.
    - (WCAG 2.2) 4.1.1 Parsing was made obsolete and WCAG 2.2 removed it as a requirement.

Option 3: Something else

- 4.1.1 Parsing—
    - When WCAG 2.0 and 2.1 were written, The Intent of 4.1.1 is to provide consistency so that different user agents or assistive technologies wlll yield the same result.
    - By the time WCAG 2.2 the problem 4.1.1 was intended to fix no longer existed. Therefore in 2.2 4.1.1 Parsing was made obsolete and WCAG 2.2 removed it as a requirement. The working group also recommended that it no longer be enforced in 2.0 or 2.1 since the enforcement created a lot of work with no accessibility benefit.

Option 4: Loic's

- *4.1.1 Parsing—*
    - *When WCAG 2.0 and 2.1 were written, the Intent of 4.1.1 was to provide consistency so that different user agents or assistive technologies would yield the same result.*
    - *In WCAG 2.2, 4.1.1 Parsing was made obsolete and WCAG 2.2 removed it as a requirement, so this success criterion is not applicable.*

Add your proposal here.

Issue 421: In definition of 'keyboard interface' and in 2.1.1, "would satisfy the success criterion" is incomplete

Note: Question 1 in the survey also changes this same note. This proposal assumes the changes from question 1 will be accepted.

Proposed content changes to ‘keyboard interface’ due to issue 421

Option 0: Original text

Note 1 (Added): Keyboard interface does not refer to a physical device but to the interface between the software and any keyboard or keyboard substitute (i.e., an interface where the software accepts text or other keystroke input). Platform software may provide device independent input services to applications that enable operation via such a keyboard interface. Software that supports operation via such platform device independent services would be operable via a keyboard interface and would satisfy the success criterion.

Option 1: Mitch’s proposed changes from the issue

In context of the glossary term ‘keyboard interface’:
Note 1 (Added): Keyboard interface does not refer to a physical device but to the interface between the software and any keyboard or keyboard substitute (i.e., an interface where the software accepts text or other keystroke input). Platform software may provide device independent input services to applications that enable operation via such a keyboard interface. Software that supports operation via such platform device independent services would be operable via a keyboard interface and would satisfy the success criterion.

Option 2: An alternate proposal (taking into account some of Gregg’s comments and Mary Jo’s thoughts)

In context of the glossary term ‘keyboard interface’ delete the last sentence.
Note 1 (Added): Keyboard interface does not refer to a physical device but to the interface between the software and any keyboard or keyboard substitute (i.e., an interface where the software accepts text or other keystroke input). Platform software may provide device

independent input services to applications that enable operation via such a keyboard interface. Software that supports operation via such platform device independent services would be operable via a keyboard interface and would satisfy the success criterion.

Option 3: Gregg's edit to Option 2 (change to parenthetical)

Note 1 (Added): Keyboard interface does not refer to a physical device but to the interface between the software and any keyboard or keyboard substitute (i.e., the interface where the software accepts text or other keystroke input from the platform which may come from a keyboard or from a keyboard alternative). Platform software may provide device independent input services to applications that enable operation via such a keyboard interface.

Add your alternate proposal here.

Proposed content changes to SC 2.1.1 Keyboard due to issue 421

Option 0: Original text

NOTE 1 (Added): Keyboard interface does not refer to a physical device but to the interface between the software and any keyboard or keyboard substitute (i.e., an interface where the software accepts text or other keystroke input). Platform software may provide device independent input services to applications that enable operation via such a keyboard interface. Software that supports operation via such platform device independent services would be operable via a keyboard interface and would satisfy the success criterion.

Option 1: Mitch's proposal from the issue

Note 1 (Added): Keyboard interface does not refer to a physical device but to the interface between the software and any keyboard or keyboard substitute (i.e., an interface where the software accepts text or other keystroke input). Platform software may provide device independent input services to applications that enable operation via such a keyboard interface. Software that supports operation via such platform device independent services would be operable via a keyboard interface and would satisfy the success criterion.

Clean version of Mitch's proposal:

Note 1 (Added): Keyboard interface does not refer to a physical device but to the interface between the software and any keyboard or keyboard substitute (i.e., an interface where the software accepts text or other keystroke input). Platform software may provide device independent input services to applications that enable operation via such a keyboard interface. When software supports such a device-independent service of the platform, and the software's functionality is made operable through the service, then this success criterion would be satisfied.

Option 2: Gregg's alternate proposal (variation of Mitch's)

Note 1 (Added): Keyboard interface does not refer to a physical device but to the interface between the software and any keyboard or keyboard substitute (i.e., an interface where the software accepts text or other keystroke input). Platform software may provide device independent input services to applications that enable operation via such a keyboard interface. When software supports such a device-independent service of the platform, and the software's functionality is made operable through the service, then this success criterion would be satisfied.

Option 3: An alternate proposal (taking into account Gregg's issue comment and Mary Jo's thoughts)

Note 1 (Added): Keyboard interface does not refer to a physical device but to the interface between the software and any keyboard or keyboard substitute (i.e., an interface where the software accepts text or other keystroke input). Platform software may provide device independent input services to applications that enable operation via such a keyboard interface. When software supports such a device-independent service of the platform, and the software's functionality is made operable through the service, then this success criterion would be satisfied.

Option 4:Same as 3 but with new i.e. parenthetical

Note 1 (Added): Keyboard interface does not refer to a physical device but to the interface between the software and any keyboard or keyboard substitute (i.e., the interface where the software accepts text or other keystroke input from the platform which may come from a keyboard or from a keyboard alternative). Platform software may provide device independent input services to applications that enable operation via such a keyboard interface. When software supports such a device-independent service of the platform, and the software or non-web document functionality is made fully operable through the service, then this success criterion would be satisfied.

Issue 383: Adjust links in Guidance Section to link to all task force and AG publications

Option 0: Leave Guidance section unchanged

Current content from last sentence of paragraph 3 onwards:

The following WCAG 2 supporting documents, though they have not been changed to fully apply in non-web contexts, contain helpful information to learn about the user needs, intent, and generalized implementation techniques:

- WCAG 2 Overview
- Techniques for WCAG 2.2 [WCAG22-TECHS]
- How to Meet WCAG (Quick Reference)
- Mobile Accessibility: How WCAG 2.0 and other W3C/WAI Guidelines Apply to Mobile - This draft resource, as of the date of the WCAG2ICT Note publication, is undergoing an update by the Mobile Accessibility Task Force to cover WCAG 2.2.

Option 1: Add reference to individual TF publications

Proposed change: remove mobile draft guidance, point to individual TF publications

The following WCAG 2 supporting documents, though they have not been changed to fully apply in non-web contexts, contain helpful information to learn about the user needs, intent, and generalized implementation techniques:

- WCAG 2 Overview
- Techniques for WCAG 2.2 [WCAG22-TECHS]
- How to Meet WCAG (Quick Reference)

Publications from other task forces may also contain relevant information for specific application areas (such as designing for mobile).

Option 2: Add reference to evergreen AG page of links (proposal from Rachael)

Proposed change: remove mobile draft guidance, point to additional AG publications page

The following WCAG 2 supporting documents, though they have not been changed to fully apply in non-web contexts, contain helpful information to learn about the user needs, intent, and generalized implementation techniques:

- WCAG 2 Overview
- Techniques for WCAG 2.2 [WCAG22-TECHS]
- How to Meet WCAG (Quick Reference)
- Additional Accessibility Guidelines Working Group - Publications