

# Deprecating SC 4.1.1 #770

New issue

**!** Open

WilcoFiers opened this issue on Jun 6, 2019 · 58 comments



WilcoFiers commented on Jun 6, 2019 · edited

## The top card of this issue will be updated as this work progresses

The purpose of this issue is to determine whether or not [WCAG 2.1, success criterion 4.1.1 Parsing](#) is still relevant to the accessibility of web content for people with disabilities. The web accessibility landscape has changes significantly since this criterion was proposed in 2008. Many accessibility experts are of the opinion that this success criterion is largely irrelevant, and that the part that is still relevant is addressed by other success criteria. If this is true, success criterion 4.1.1 could potentially be deprecated as part of WCAG 2.2.

## Plan of Action

For this proposal, we'll go through the following steps:

- Identify categories of conformance issues that need to be tested **[ready for review]**
- Identify which markup languages require testing **[in progress]**
- Determine which assistive technologies need to be tested
- Create test cases, and perform assistive technology tests
- Analyse the results and create a proposal for WCAG 2.2

### Assignees

DavidMacDonald

### Labels

Normative  
WCAG 2.2

### Projects

**WCAG 2.2**  
Assigned

### Milestone

No milestone

### Linked pull requests

Successfully merging a pull request may close this issue.

None yet

12 participants

## Background

Web technologies have developed significantly since the creation of WCAG 2.0, where this success criterion was introduced. In HTML 4.01 and older versions, there was no description of what to do with malformed HTML. This caused HTML parsers to behave differently in different environments. HTML 5.0 included a detailed definition of how to parse malformed HTML. This has ensured that all modern browsers parse HTML exactly the same way.

The second reason why 4.1.1 was relevant in 2008, was because assistive technologies frequently had their own build-in HTML parser. Rather than rely on the browser to build up the accessibility tree, assistive technologies would build their own accessibility tree based on the HTML of a page. Improvements in browsers and standardisation efforts have made the accessibility trees more reliable. Assistive technologies in 2019 use the accessibility tree directly, rather than parse HTML and build their own.

## Step 1: Breakdown of 4.1.1 Parsing

### Scope

Success criterion 4.1.1 is applicable to all content implemented in markup languages.

### Requirement 1

Elements have complete start and end tags.

This requirement can be broken down further:

1. A closing tag exists for each element that require a closing tag (such as `main` , `h1` , `strong` , etc.)
2. None of the closing tags are for elements that do not allow closing tags (such as `img` , `input` , `meta` , etc.)



3. Each closing tag has a corresponding opening tag positioned before the closing tag of its parent node (e.g. `<div><h1></h1></div>` rather than `<div><h1></div></h1>` )

## Requirement 2

Elements are nested according to their specifications.

1. The root node of a document is an element that is allowed as the root node (e.g. `html` is the root of HTML documents, and `svg` for SVG documents)
2. Each child node of an element is of a type allowed in its content model. (e.g. `p` is allowed `strong` , but not `h1` )

## Requirement 3

Elements do not contain duplicate attributes.

This requirement is often "expanded" to mean attributes must be marked up correctly.

1. Elements do not contain duplicate attributes (e.g. `<p class="foo" class="bar">...</p>` should be `<p class="foo bar">...</p>` , and `<input readonly="readonly" readonly>` should be `<input readonly>` )
2. Unquoted attributes (under certain circumstances) \* (e.g. `<p id=alexander-'the-great'>` should be `<p id="alexander-'the-great'">` or `<p id=alexander-the-great>` )
3. Mismatched quotes on attributes \* (e.g. `<p id="main'>` should be `<p id="main">` or `<p id='main'>` )
4. Attributes not space separated \* (e.g. `<p id="main"class="main">` should be `<p id="main" class="main">` )

5. Attributes are not malformed \* (e.g. `<p id="main">` should be `<p id="main">`  )

\*: It is debatable whether or not points 2 through 5 are required by the success criterion. Either way it is worth exploring whether or not these cause accessibility issues.

The same attribute and attribute + value combination can exist on different elements, just not on the same element.

## Requirement 4

Any IDs are unique.

1. IDs used in IDREFs are unique (e.g. `<label for="name">name</label><input id="name">`  )
2. IDs used referenced by internal links are unique (e.g. `<h1 id="hd1"><a href="#hd1">Heading 1</a></h1>`  )
3. IDs used on user interface components are unique (e.g. `<button id="submit">Submit</button>`  )
4. IDs used on any element unique (e.g. `<p id="my-style-be4d-c66d4b51819f">...</p>`  )

**Note 1:** IDs in different DOM trees are **not** required to be unique. The same ID can exist in different shadow DOM trees, since ID referencing across different trees is not valid.

**Note 2:** Elements hidden through CSS are not allowed duplicate IDs. IDREFs work regardless of if an element is visible. For example, `aria-labelledby` can use hidden elements for the computation of an accessible name.

## Step 2: Scope "Markup languages"

In order for any markup language to be relevant for SC 4.1.1 Parsing, it must meet the following requirements:

- It can be served as a [web page](#)
- It is supported by [assistive technologies](#)

- It is implemented in a [user agent](#)

Markup Language	Web Page	User Agent	Assistive Technology
HTML: The Markup Language	Yes	Web Browsers	
SVG: Scalable Vector Graphics	Yes	Web Browsers	
MathML: Mathematical Markup Language			
SMIL: Synchronized Multimedia Integration Language			
TTML: Timed Text Markup Language			
EmotionML: Emotion Markup Language			
EMMA: Extensible MultiModal Annotation markup language			
InkML: Ink Markup Language			
VoiceXML: Voice Extensible Markup Language			
SSML: Speech Synthesis Markup Language			

To do: Fill out the table

### Step 3: Assistive Technologies

---

TBD as part of step 3.

### Step 4: Testing

---



TBD as part of step 4.

### Step 5: Conclusion And Recommendations

---

TBD as part of step 5.

  **WilcoFiers** added the **WCAG 2.2** label on Jun 6, 2019

  **WilcoFiers** self-assigned this on Jun 6, 2019

  **WilcoFiers** mentioned this issue on Jun 6, 2019

**4.1.1 Parsing - Duplicate attributes with no user impact cause a conformance failure #542**

 Closed



**bruce-usab** commented on Jun 6, 2019

Let's be sure to drag [@stevefaulkner](#) into this discussion! Here is a link to his [parsing bookmarklet \(2/25/19\)](#). Per his comments, Steve seems to think 4.1.1 might still have utility.



mraccess77 commented on Jun 6, 2019

Contributor

We often see links inside of buttons and other nesting issues that aren't clearly covered by SC but don't work well with assistive technology. This is the type of issue that 4.1.1 may still be needed for.



DavidMacDonald commented on Jun 6, 2019

Contributor

• edited by WilcoFiers ▾

Excellent start of this discussion @WilcoFiers . There are two minor corrections about the intent of the Success Criteria parts, from my memory of when we wrote it.

**Requirement 1:** Elements have complete start and end tags.

This was a very simple requirement that anything that opened needed to be closed (if closing it is in the spec). Your example of `<div><h1></div></h1>` would fall under your examples of the second requirement about nesting. An example of this that I would give would be

```
<h1>La la<p> more lala</p>
```

with no close to the h1.

**Requirement 2:**

I would put your example from [#1](#) here.

```
<div><h1></div></h1>
```

In addition to your discussion of nodes.



**mraccess77** commented on Jun 6, 2019

Contributor

The other part of this discussion was that issues with duplicate ids that impacted the user such as references to labels and aria-labelledby, etc. would be caught under other SC like 1.3.1 and 4.1.2. So I think people agree that duplicate ids can be an issue -- but they generally result in violations to other SC -- @WilcoFiers this is an important concept to capture in the doc.



**bruce-usab** commented on Jun 6, 2019 · edited ▾

@mraccess77 wrote:

We often see links inside of buttons and other nesting issues that aren't clearly covered by SC but don't work well with assistive technology. This is the type of issue that 4.1.1 may still be needed for.

Jon, could you post a cleanish example of this? (On a live page somewhere.) It seems to me like the simplest way to resolve this issue would be a single example demonstrating that we still need 4.1.1 (at least for now).



**jake-abma** commented on Jun 6, 2019 · edited ▾

Contributor

I know of lots of wrongly nested roles when using ARIA which this SC should cover, but doesn't, as it only speaks of "elements". Same Jon talks about... 4.1.2. Passes but still confuses UA/AT like role button nested in role textbox. Enough to add/extend in Understanding doc OR new SC?





**bruce-usab** commented on Jun 6, 2019 · edited ▾

Enough to add/extend in Understanding doc OR new SC?

I feel pretty strongly that this should be new SC. I think it is okay that an SC written in 2008 misses catching common mistakes developers make with ARIA. I am not really okay with changing Understanding to have new normative requirements. Likewise, I feel like the note attached to 4.1.2 should really guide our saying if some particularly poor code fails 4.1.2 or not.



✉ **johnfoliot** commented on Jun 6, 2019

+1 to Bruce - a new SC rather than modifying the intent & scope of the original 4.1.1

JF

...



✉ **Ryladog** commented on Jun 6, 2019

+1 to new SC

...



**jake-abma** commented on Jun 6, 2019

Contributor

Clear, added to MATF doc to discuss, thanks!



**awkawk** commented on Jun 6, 2019

Member

Much of the discussion on [#542](#) was around deprecating 4.1.1 if it can be demonstrated that issues that are flagged as 4.1.1 issues all either are issues against other SC. It sounds like the discussion is at least partly around adding a new SC to capture aria-related issues. Is it reasonable to address these as separate questions and issues?



**brand** commented on Jun 6, 2019

+1 for a new SC.

I've seen 4.1.1 used to argue that a PDF document should have a Document element at the root of it's tag tree. While I think it is a bit of a stretch to see a PDF tag structure as a markup language, it clearly has properties in common with markup languages.

So, if we go the route of a new SC to express that only valid nested roles are allowed, it would be good to think of the above case as well. Basically, I think we would like to see that people don't put things in something that is eventually rendered to an accessibility tree that mess up the resulting role structure. It being HTML, ARIA, a PDF tag tree or may be in the future an accessibility tree generated by some javascript.



 **Ryladog** commented on Jun 6, 2019

I would like it to also be relevant to ePub.

...



**dd8** commented on Jun 7, 2019 · edited ▼

The other part of this discussion was that issues with duplicate ids that impacted the user such as references to labels and aria-labelledby, etc. would be caught under other SC like 1.3.1 and 4.1.2. So I think people agree that duplicate ids can be an issue -- but they generally result in violations to other SC

Here's a list of HTML / SVG / MathML features that use ID references taken from the relevant recommendations - that should help in figuring out whether duplicate IDs can have an impact not covered by other SC:

Element	Attribute	Spec
a	href # id	HTML 4.01
label	for	HTML 4.01
td	headers	HTML 4.01
th	headers	HTML 4.01
input	list	HTML 5
output	for	HTML 5
(any form element)	form	HTML 5
(any element)	itemref	HTML 5
(any MathML element)	xref	MathML 3
(any SVG element)	(various)	SVG 2

- HTML 4.01 - ID references use IDREF type: <https://www.w3.org/TR/html4/sgml/loosedtd.html>
- XHTML 1.0 - ID references use IDREF type: <https://www.w3.org/TR/xhtml1/dtds.html>
- HTML living standard - ID references scattered around normative text <https://html.spec.whatwg.org/multipage/indices.html#att>

### ributes-3

The SVG recommendation says IDs should be unique within the node tree - that has implications for SVG embedded in HTML:

<https://www.w3.org/TR/2018/CR-SVG2-20181004/struct.html#Core.attrib>

The MathML 3 recommendation is inconsistent about whether IDs should be unique. This normative text says `id` is a unique reference:

<https://www.w3.org/TR/MathML3/chapter2.html#fund.globatt> but the MathML DTD *intentionally allows* IDs to be duplicated - see MMLIDTYPE in the DTD:

<https://www.w3.org/Math/DTD/mathml3/mathml3.dtd>

Of note: there's also a section in the MathML 3 recommendation explaining how things like complete end tags and attribute quotes are not required for MathML in HTML:

<https://www.w3.org/TR/MathML3/chapter6.html#interf.html>



1



**dd8** commented on Jun 7, 2019

The use of duplicate IDs referenced in links via `href=otherpage.html#targetid` may be hard to map to another SC because the conformance requirements apply to full pages or complete processes, and the link may come from outside those.

Ending up on the wrong part of the target page is difficult for people using magnifiers or screen readers.



WilcoFiers commented on Jun 7, 2019 · edited ▾ Author

@dd8 Thanks for the list. That would have taken me a while to put together.

Enough to add/extend in Understanding doc OR new SC?

I agree with what most people are saying here, a new SC would be preferable.

## Update

---

I've updated **Requirement 3** in the top post, to include some of the attribute tests that Steve Faulkner's bookmarklet treats as failures for SC 4.1.1. I'm going to create a Doodle poll next for some of the people who volunteered to help work on this to come up with next steps.



dd8 commented on Jun 7, 2019 · edited ▼

@dd8 Thanks for the list. That would have taken me a while to put together.

There are some missing since IDs can be used in any URL after the # character, so that means any href or src attribute (and there are many other attributes that contain URLs such as longdesc ).

Fragments in URLs can only cause dupe id problems if the content type referenced can contain IDs. For example, the fragment in `<img src='map.png#iceland'>` has no meaning for PNG graphics and works identically to `<img src='map.png'>` , but `<img src='map.svg#iceland'>` should display Iceland if map.svg contains `id='iceland'` and there's obviously a problem for the user if more than one country on the map is identified as Iceland.

An example that currently fails SC 4.1.1 but needs considered if the SC is removed/replaced are skip links linked to duplicated IDs:

```
<a href='#main'>Skip to content</a>
<div id='main'>...</div>
<div id='main'>...</div>
```

There's also an ambiguity on `img usemap` due to spec churn:

- in HTML 5 `usemap` refers to map name again but via a 'hash-name reference' instead of a URL
- in XHTML 1.1 `usemap` refers to map id via a URL and name was removed
- in XHTML 1.0 `usemap` refers to map id via a URL and name was deprecated
- in HTML 4.01 `usemap` refers to map name via a URL



**awkawk** commented on Jun 7, 2019

Member

**@dd8** This doesn't necessarily fail any other SC - it may fail 2.4.1 Bypass blocks if the user agents make the skip link go to the wrong place, but if the user agents make the skip link go to the right place then functionally it is fine for users.



**awkawk** commented on Jun 7, 2019

Member

I would like it to also be relevant to ePub.

**@Ryladog** Epub uses markup so it does apply already.



**patrickhlauke** commented on Jun 8, 2019

Member

The use of duplicate IDs referenced in links via `href=otherpage.html#targetid` may be hard to map to another SC because the conformance requirements apply to full pages or complete processes, and the link may come from outside those.

Ending up on the wrong part of the target page is difficult for people using magnifiers or screen readers.

Question is probably if it's a problem for all users, or something that disproportionately affects users with disabilities?



**patrickhlauke** commented on Jun 8, 2019

Member

We often see links inside of buttons and other nesting issues that aren't clearly covered by SC but don't work well with assistive technology. This is the type of issue that 4.1.1 may still be needed for.

I'd flag that under focus order, generally



✉ **Ryladog** commented on Jun 8, 2019 · edited by alastc ▼

Andrew,

Oops, you are right, it does have elements! Thanks!

\*\* katie \*\*





**mraccess77** commented on Jun 11, 2019 ·  
edited ▼

Contributor

Bruce, we come across situations like the following

```
<a href="http://www.google.com">  
<input type="checkbox" aria-label="I agree to the terms"> I  
agree to the terms  
</a>
```

```
<div role="button" tabindex="0"aria-label="Visit Google">  
<a href="http://www.google.com"> <span  
class="backgroundimage"></span> </a>  
</div>
```

```
<a href="http://www.google.com">  
<button><span class="backgroundimage"></span>  
</button>  
</a>
```



**bruce-usab** commented on Jun 12, 2019

@mraccess77 is that three examples or just one? I am reading it as three, but also I am not seeing how any of those fail against what 4.1.1 catches. Maybe not "nested according to their specification"? I feel like I need a more complete code sample to evaluate that.



mraccess77 commented on Jun 12, 2019

Contributor

@bruce-usab HTML specification does not allow button inside of anchor, anchors inside of buttons, or inputs inside of anchors. So yes, these are 3 examples of nesting issues that can have impact on some AT in different ways.

3



awkawk commented on Jun 12, 2019

Member

@mraccess77 It would be great to see functioning examples for those cases.

1



jake-abma commented on Jun 12, 2019 · edited ▼

Contributor

@mraccess77 It would be great to see functioning examples for those cases.

I've been involved in features where this had been implemented dozens of times the last 2/3 years. Specially for sites where they use / create custom components / web components (with lots of ARIA). Problem is focus / programmatic associated descriptions / label / names gets mixed, not communicated properly or skipping of elements when Tabbing / Shift tabbing UICs. Also inconsistency in Tabbing OR Shift tabbing. (skipping one direction, not other direction).

10 hidden items

[Load more...](#)



patrickhlauke commented on Aug 14, 2019

Member

@dd8 would that not fail 4.1.2?



dd8 commented on Aug 14, 2019 · edited

@dd8 would that not fail 4.1.2?

Not sure - this affects the accessible description and 4.1.2 covers Name, Role, Value, but doesn't mention description, unless the definition of name is intended to cover both accessible name and accessible description:  
<https://www.w3.org/WAI/WCAG21/Understanding/name-role-value.html#dfn-name>

You could fail this under 3.3.2 Labels or Instructions if you thought the differences between the instructions caused problems.

If both descriptions were identical (or both aria-describedby attributes referred to the same ID) this would fail 4.1.1 currently, but it wouldn't have any impact on accessibility and I don't think it would fail any other SC (with the possible exception of 4.1.2)



patrickhlauke commented on Aug 14, 2019

Member

i'd caution that 3.3.2 labels and instructions is mostly a "visual" test, as currently defined, as it doesn't really concern itself with programmatic association etc and arguably AT users would also reach the actual visible label text in browse mode. 4.1.2 seems the stronger candidate for failing it



**dd8** commented on Aug 14, 2019 · edited ▾

Not sure - one of the sufficient techniques for 3.3.2 explicitly mentions aria-describedby in conjunction with a visible label: <https://www.w3.org/WAI/WCAG21/Understanding/labels-or-instructions.html>

A very strict reading of 4.1.2 in conjunction with other specs like AccName and ARIA might exclude accessible description from 4.1.2. This is probably not the intent, so there might be a specification gap here...



**patrickhlauke** commented on Aug 14, 2019

Member

@**dd8** but also "This Success Criterion does not require that labels or instructions be correctly marked up, identified, or associated with their respective controls", meaning that, conversely, things that aren't associated with a particular control can still count as a "label or instruction"



**dd8** commented on Aug 14, 2019

@**patrickhlauke** that makes sense - the text says "Further, this Success Criterion does not take into consideration whether or not alternative methods of providing an accessible name or *description* for form controls and inputs has been used - this aspect is covered separately by 4.1.2: Name, Role and Value. "

I'll file a bug on the omission of description from 4.1.2 - the WCAG definition made sense before the accessibility API was formalised, but is less clear now.





**DavidMacDonald** commented on Sep 13, 2019 Contributor

Personally, I like the idea of retiring 4.1.1 with an note that the issues it identified that were relevant to Accessibility, are covered in SC 1.3.1, and SC 4.1.2.

Anyone else on board?



**mraccess77** commented on Sep 13, 2019 Contributor

**@DavidMacDonald** As mentioned above -- there are issues related to nesting of controls and non-controls elements like lists that could fall outside of 1.3.1 and 4.1.2. Perhaps as part of this we create failure techniques for some of these nesting items and map them to 1.3.1 or 4.1.2 as appropriate?



**bruce-usab** commented on Sep 15, 2019 · edited ▼

Anyone else on board?

FWIW, I can live with dropping it. Back in the day, I was as strong an advocate for 4.1.1 as anyone.

Perhaps as part of this we create failure techniques for some of these nesting items and map them to 1.3.1 or 4.1.2 as appropriate?


Even a single documented failure (mapping to 4.1.2 rather than 1.3.1 would be my strong preference) I feel would provide sufficient rational for deprecating 4.1.1.

  **JAWS-test** mentioned this issue on Nov 30, 2019

**What does nested according to the specification mean in SC 4.1.1 #978**

 Open

  **alastc** added this to **To do** in **WCAG 2.2** on Jan 6, 2020

  **alastc** moved this from **To do** to **In progress** in **WCAG 2.2** on Jan 6, 2020



**dd8** commented on Feb 4, 2020 · edited ▾

There's another issue with nested controls which is currently picked up by 4.1.1:

```
<a href='/'> View our privacy policy or <button>  
sign up </button> for our newsletter</a>
```

It's much easier to hit the wrong target because if you hit near the edge of the inner control you may activate the outer control..

That's a problem for folks with Parkinson's. A button embedded in normal text that just passes 2.5.5 Target Size is much easier to use than an identically sized button nested inside a link.



**patrickhlauke** commented on Feb 5, 2020

Member

This scenario would likely be picked up/failed under the new proposed target spacing SC (though i have my reservations about that one for its weird attempts at defining hard spacing requirements ... but something relating to targets contained inside other targets would likely be an easier fail scenario to define)

👍 1



**jake-abma** commented on Feb 5, 2020 ·

Contributor

edited ▼

would likely be picked up/failed under the new proposed target spacing SC

It depends on interpretation, but I don't think it will be applicable.

The exception is:

except when:  
 Inline The target is in a sentence or block of text;

So

```
> <a href='/'> View our privacy policy or <button>
sign up </button> for our newsletter</a>
```

Seems like a sentence to me.



**patrickhlauke** commented on Feb 5, 2020

Member

then that target spacing SC needs even more work to explicitly cover nested targets then, as that exception would make little sense in this context...



**alastc** moved this from **In progress** to **To do** in **WCAG 2.2** on Feb 22, 2020



**dd8** commented on Mar 3, 2020

Another example which is currently caught by 4.1.1 parsing. This fails 4.1.1 but looks like it should work with the `div listitem` role matching the implicit role of `li` elements:

```
<ul>
  <div role="listitem">List item 1</div>
</ul>
```

It causes problems in practice:

- not read as a list in Safari 13 with VoiceOver on macOS 10.15 (reads list item as plain text)
- list item not announced by JAWS 2019 with IE11 (reads 'list start', 'list end' without reading list item)





**alastc** commented on Mar 6, 2020

Contributor

Interesting example, presumably that would work if the `<ul>` was a `div` with a role of list?

I would have thought that could be caught with 1.3.1 + accessibility supported. I.e. that method of conveying a list isn't well supported.

On the other hand, that could make testing harder as you'd need a support matrix for various methods, or need to do a lot of device/AT testing.



**awkawk** commented on Mar 6, 2020 · edited

Member

I would view this example as a 1.3.1 issue as the lack of the `aria-list` role in the parent makes the list that we can assume is shown visually not represented in the DOM.

I am not advocating for this code, but what if the author did this:

```
<ul role="list">
  <div role="listitem">List item 1</div>
</ul>
```

The list would function correctly AND it would flag a 4.1.1 issue.



mraccess77 commented on Mar 6, 2020

Contributor

Seems like if it's accessibility supported then it would pass SC 1.3.1. In fact this is a good way to fix things that might not be fixable at the source without adding extra roles that are not needed -- e.g. not adding roles that duplicate the default implied semantics of the ul element. My bigger issues with 4.1.1 are around buttons and form fields inside of links and links inside buttons, and odd bases like that.



dd8 commented on Mar 6, 2020

On the other hand, that could make testing harder as you'd need a support matrix for various methods, or need to do a lot of device/AT testing.

I think the problem with an accessibility support matrix is it gets out of date really quickly.

Stuff gets broken at nearly the same rate as it gets fixed in AT (e.g. `aria-label` on links worked reliably up to JAWS 18 / FF 52, didn't work in JAWS 2018 / FF 60, then worked again in JAWS 2019 with FF68). Similar problems have happened with NVDA and VoiceOver.

That points to a difficult problem with relying on accessibility support - what happens when a vendor breaks something? Does a page that was previously compliant become non-compliant?



alastc assigned **DavidMacDonald** and unassigned **WilcoFiers** on Jul 14, 2020



alastc moved this from **To do** to **In progress** in **WCAG 2.2** on Jul 15, 2020



**DavidMacDonald** commented on Sep 10, 2020

Contributor

• edited ▾

I'm actively communicating with WCAG team members on this issue to see if there are any outstanding issues that cannot be addressed as failure techniques or updates to the Understanding docs of other SC. Will report back.



**mraccess77** commented on Sep 10, 2020

Contributor

How would deprecating SC 4.1.1 impact affect backwards compatibility of WCAG 2.2?



**awkawk** commented on Sep 10, 2020

Member

@mraccess77 The idea as I understand it, is that removing 4.1.1 is based on the idea that all of the failures that are currently assigned to 4.1.1 which impact end users are captured by other SC. I don't think that we've arrived at consensus that all user-impacting 4.1.1 issues are caught by other SC yet.



**awkawk** commented on Sep 10, 2020

Member

@mraccess77 to finish that thought, if the issues for 4.1.1 are caught by other SC, then there is negligible or no backward compatibility impact.



**patrickhlauke** mentioned this issue on Oct 29, 2020

**F17 removal lost important tests #1492**

Open