Immutable ArrayBuffers for stage 2.7

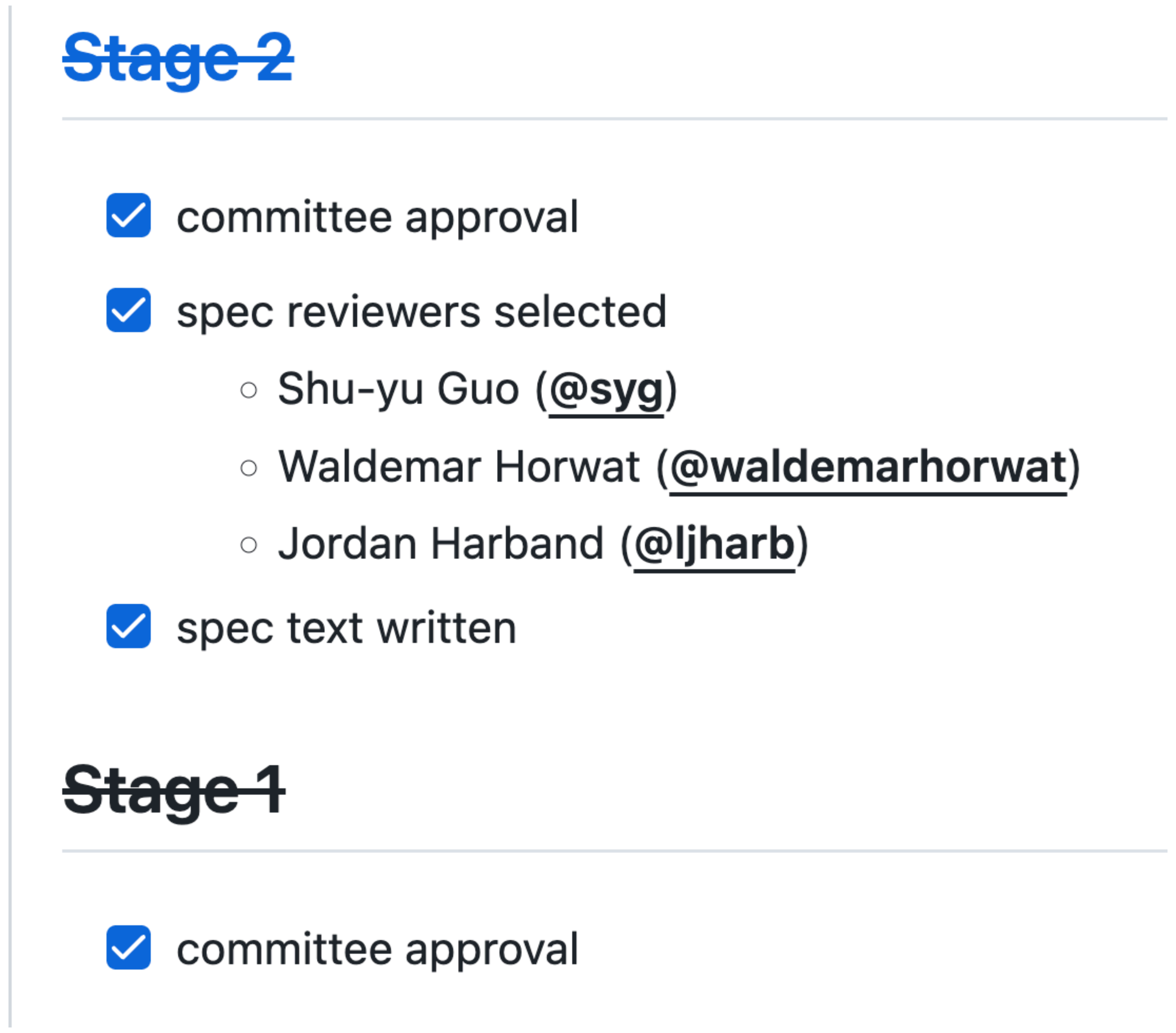
Mark S. Miller Agoric
Peter Hoddie moddable
Richard Gibson Agoric
Jack-Works

106th Plenary
February 2025
F5, Seattle, WA
TC39

Recap: Proposed ArrayBuffer API

```
transfer(len?: number) :ArrayBuffer
transferToFixedLength(len?: number) :ArrayBuffer
resize(len: number) :void
slice(start?: number, end?: number) :ArrayBuffer
transferToImmutable() :ArrayBuffer
get immutable: boolean
get detached: boolean
get resizable: boolean
get byteLength: number
get maxByteLength: number
```

Status Recap



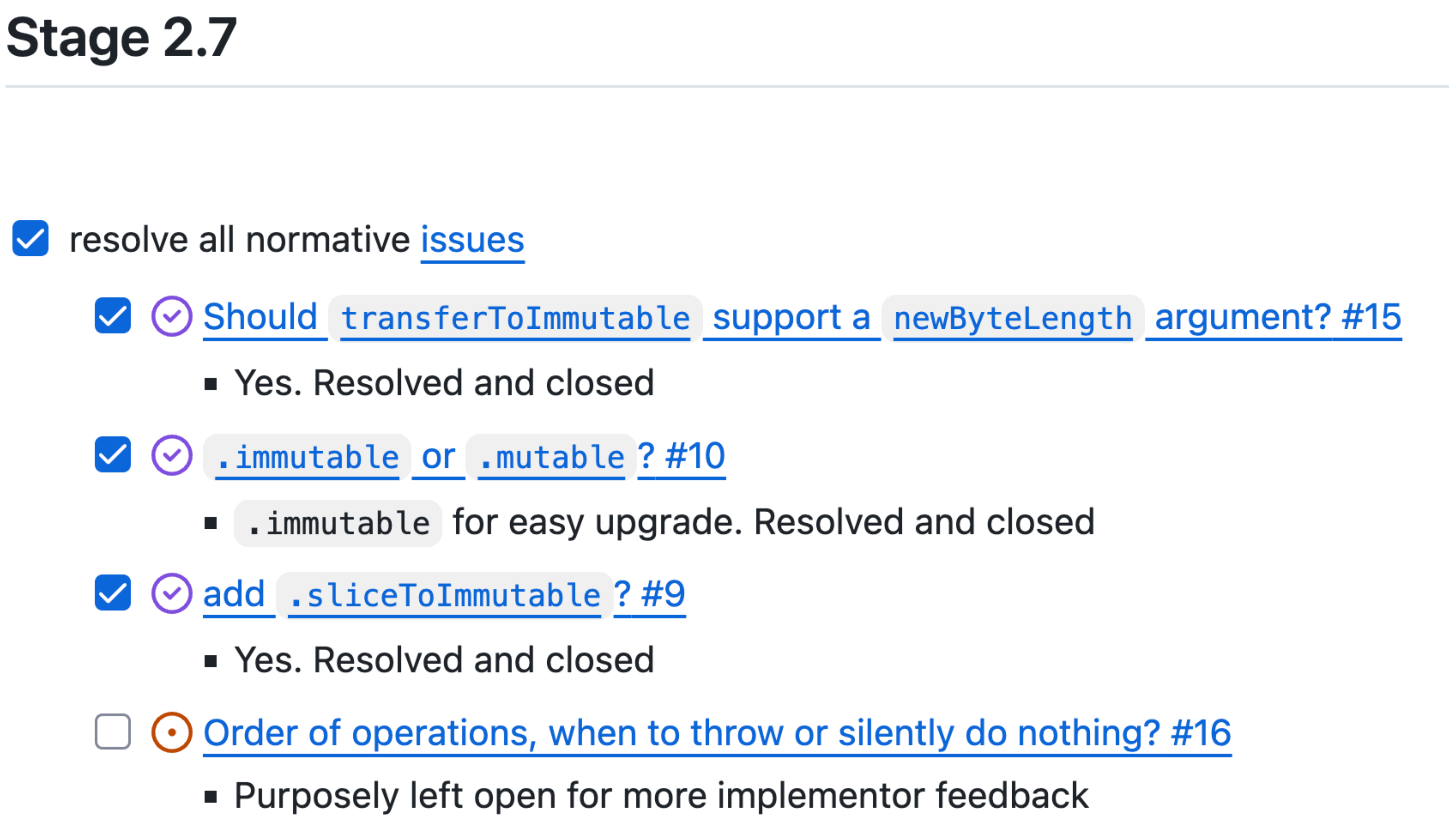
~~Stage 2~~

- [x] committee approval
- [x] spec reviewers selected
    - Shu-yu Guo (**@syg**)
    - Waldemar Horwat (**@waldemarhorwat**)
    - Jordan Harband (**@ljharb**)
- [x] spec text written

~~Stage 1~~

- [x] committee approval

Normative Issues



Stage 2.7

- [x] resolve all normative issues
  - [x] Should `transferToImmutable` support a `newByteLength` argument? #15
    - Yes. Resolved and closed
  - [x] `.immutable` or `.mutable`? #10
    - `.immutable` for easy upgrade. Resolved and closed
  - [x] add `.sliceToImmutable`? #9
    - Yes. Resolved and closed
  - [ ] Order of operations, when to throw or silently do nothing? #16
    - Purposely left open for more implementor feedback

Recap: Proposed ArrayBuffer API

```
transfer(len?: number) :ArrayBuffer
transferToFixedLength(len?: number) :ArrayBuffer
resize(len: number) :void
slice(start?: number, end?: number) :ArrayBuffer
transferToImmutable() :ArrayBuffer
get immutable: boolean
get detached: boolean
get resizable: boolean
get byteLength: number
get maxByteLength: number
```

Proposed ArrayBuffer API

```
transfer(len?: number) :ArrayBuffer
transferToFixedLength(len?: number) :ArrayBuffer
resize(len: number) :void
slice(start?: number, end?: number) :ArrayBuffer
transferToImmutable(len?: number) :ArrayBuffer
sliceToImmutable(start?: number, end?: number) :ArrayBuffer
get immutable: boolean
get detached: boolean
get resizable: boolean
get byteLength: number
get maxByteLength: number
```

Immutable ArrayBuffer Flavor

~~transfer(len?: number) :ArrayBuffer~~

~~transferToFixedLength(len?: number) :ArrayBuffer~~

~~resize(len: number) :void~~

slice(start?: number, end?: number) :ArrayBuffer

~~**transferToImmutable(*len?: number*) :ArrayBuffer**~~

sliceToImmutable(start?: number, end?: number) :ArrayBuffer

get immutable: true

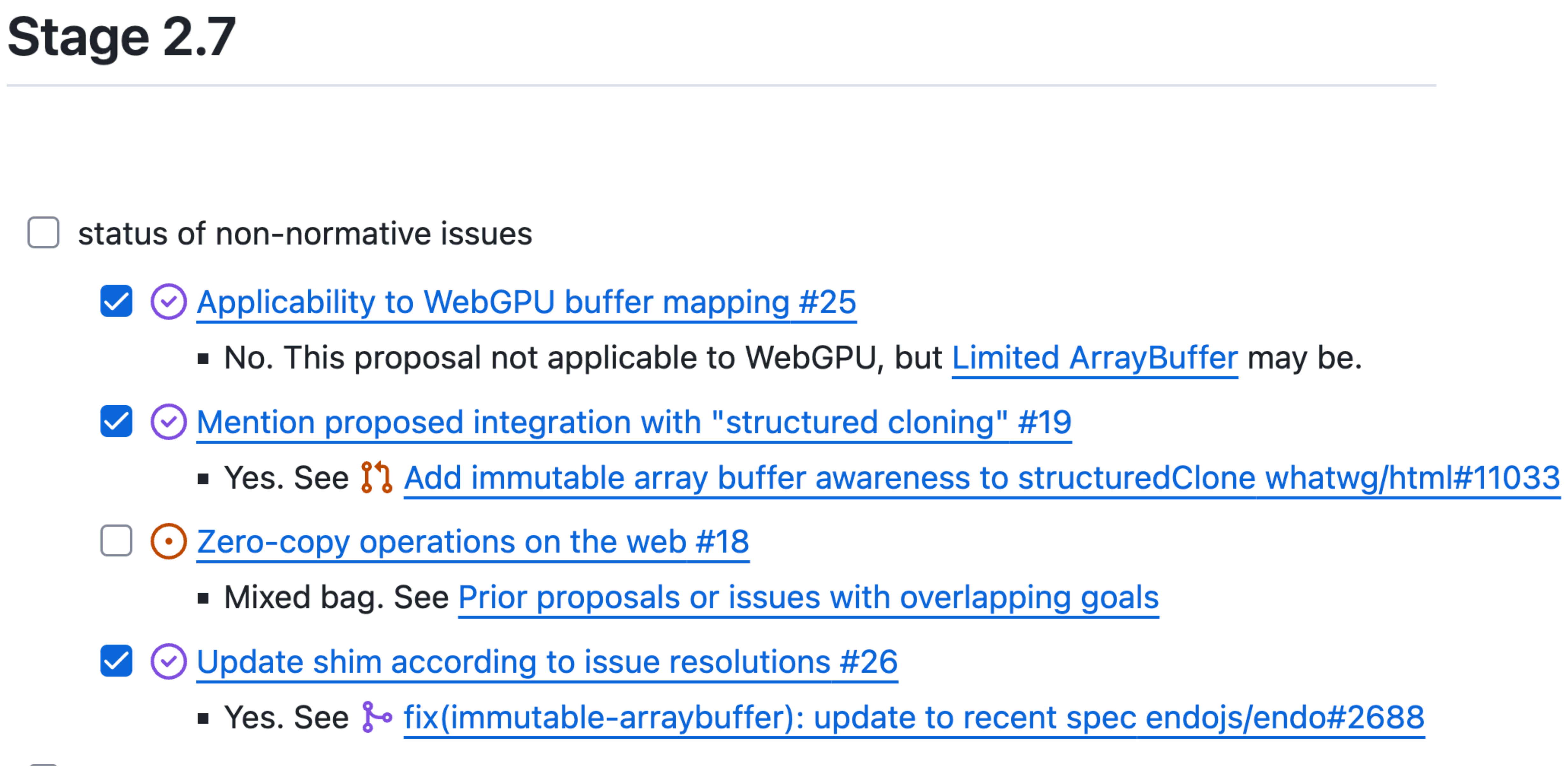
get detached: false

get resizable: false

get byteLength: number

get maxByteLength: same number

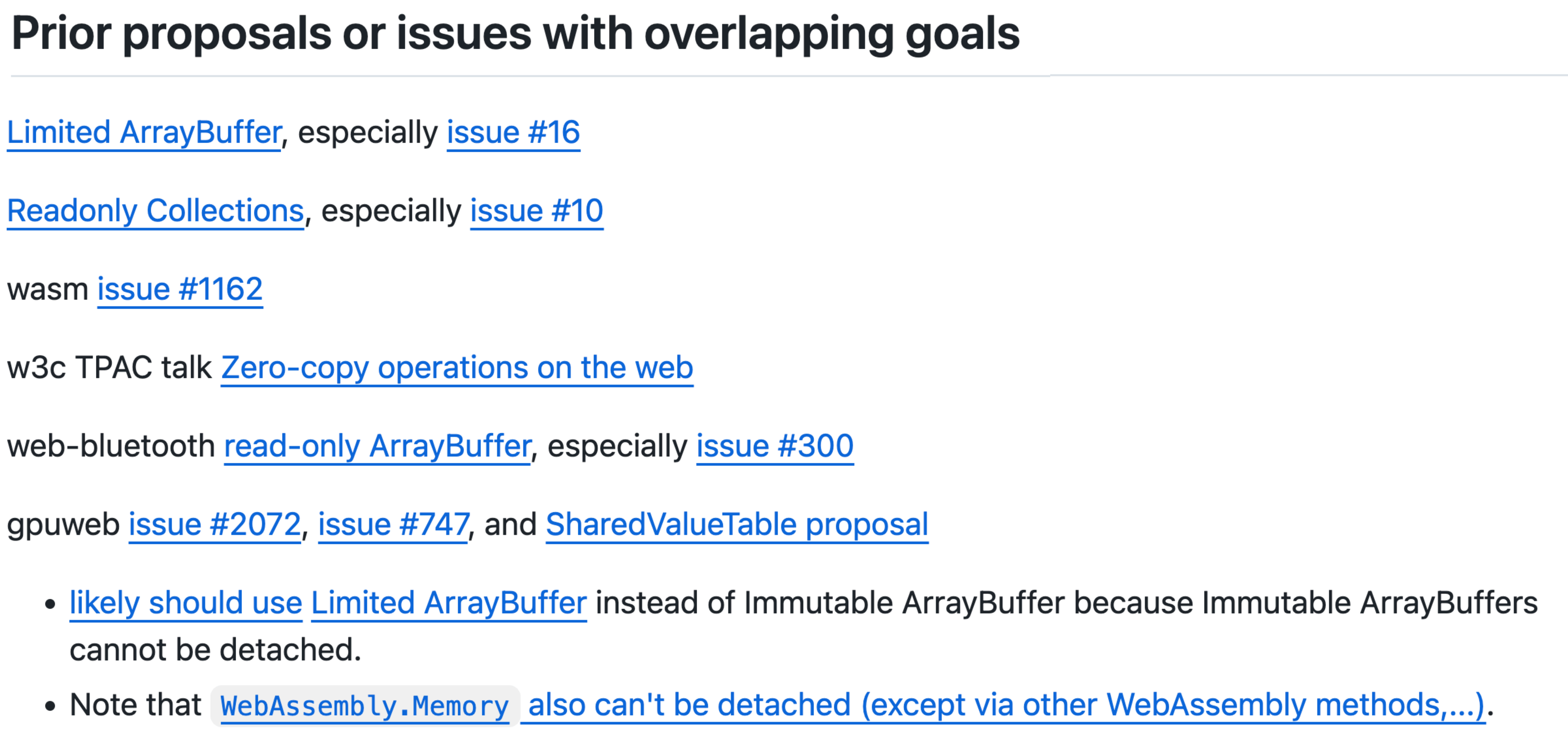
Non-Normative Issues



Stage 2.7

- [ ] status of non-normative issues
    - [x] Applicability to WebGPU buffer mapping #25
        - No. This proposal not applicable to WebGPU, but Limited ArrayBuffer may be.
    - [x] Mention proposed integration with "structured cloning" #19
        - Yes. See Add immutable array buffer awareness to structuredClone whatwg/html#11033
    - [ ] Zero-copy operations on the web #18
        - Mixed bag. See Prior proposals or issues with overlapping goals
    - [x] Update shim according to issue resolutions #26
        - Yes. See fix(immutable-arraybuffer): update to recent spec endojs/endo#2688

Zero-copy operations on the web 1/2



Prior proposals or issues with overlapping goals

Limited ArrayBuffer, especially issue #16

Readonly Collections, especially issue #10

wasm issue #1162

w3c TPAC talk Zero-copy operations on the web

web-bluetooth read-only ArrayBuffer, especially issue #300

gpuweb issue #2072, issue #747, and SharedValueTable proposal

- likely should use Limited ArrayBuffer instead of Immutable ArrayBuffer because Immutable ArrayBuffers cannot be detached.
- Note that `WebAssembly.Memory` also can't be detached (except via other WebAssembly methods,...).

Zero-copy operations on the web 2/2

Prior proposals or issues with overlapping goals

webidl Frozen Array

webcodecs issue #80, issue #104, and issue #212

web transport issue #131

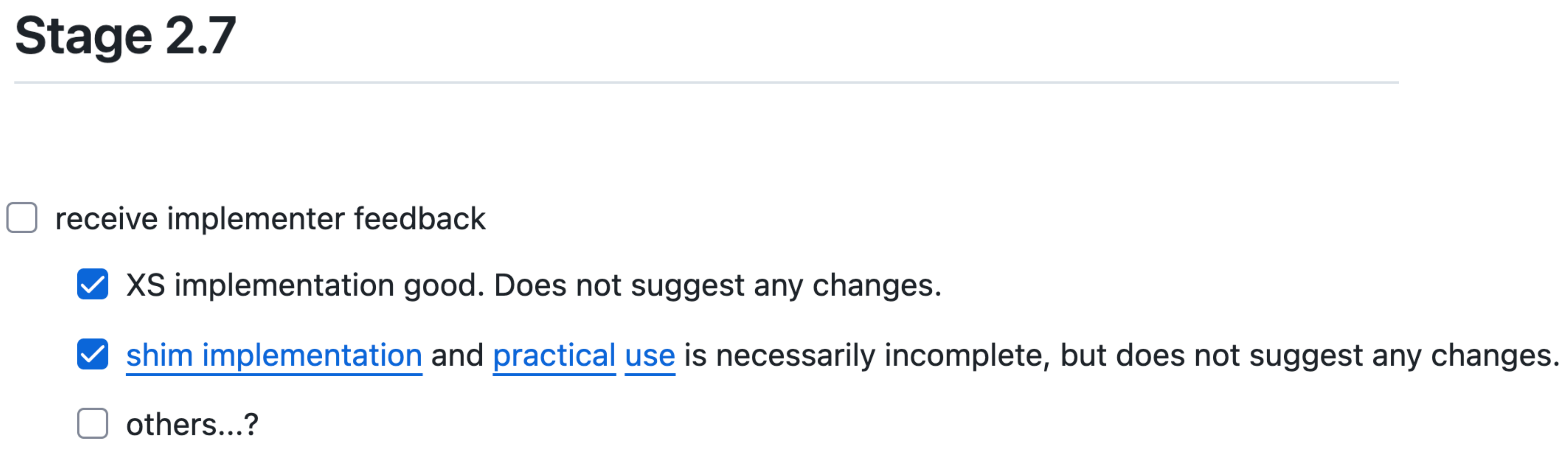
- unlikely because Chrome (and likely others) copy when crossing address spaces.
- But possible: see Even when talking between different processes, each with their own address space, for a huge enough buffer ...

whatwg streams issue #495

- unlikely because, well, they are streams, not buffers.

w3c machine learning workshop issue #93

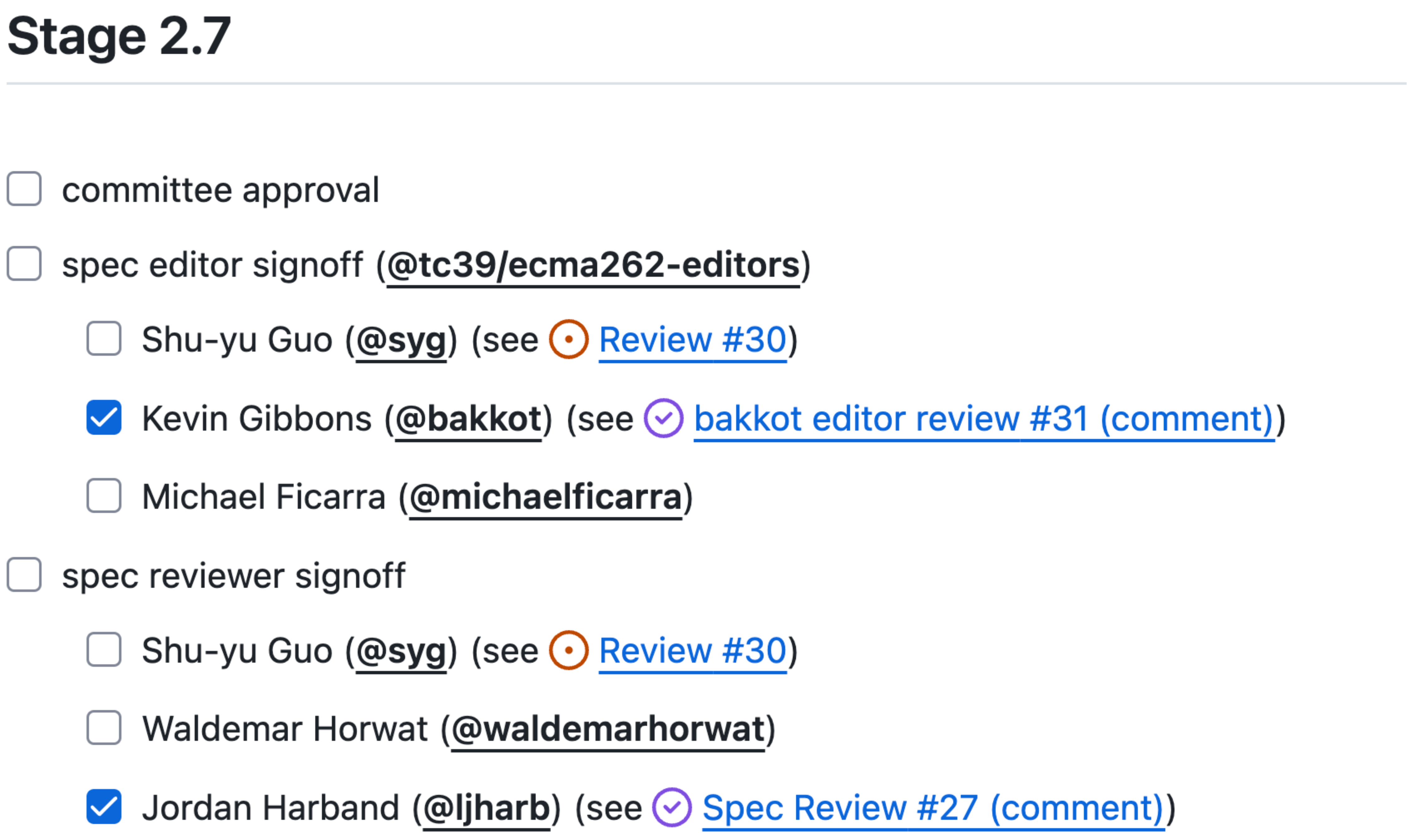
Implementor Feedback



Stage 2.7

- [ ] receive implementer feedback
  - [x] XS implementation good. Does not suggest any changes.
  - [x] shim implementation and practical use is necessarily incomplete, but does not suggest any changes.
  - [ ] others...?

Approval Status



Stage 2.7

- [ ] committee approval
- [ ] spec editor signoff (**@tc39/ecma262-editors**)
  - [ ] Shu-yu Guo (**@syg**) (see ⊙ Review #30)
  - [x] Kevin Gibbons (**@bakkot**) (see ✓ bakkot editor review #31 (comment))
  - [ ] Michael Ficarra (**@michaelficarra**)
- [ ] spec reviewer signoff
  - [ ] Shu-yu Guo (**@syg**) (see ⊙ Review #30)
  - [ ] Waldemar Horwat (**@waldemarhorwat**)
  - [x] Jordan Harband (**@ljharb**) (see ✓ Spec Review #27 (comment))

Road to Future Stages

Stage 3

- [ ] committee approval
- [ ] merge test262 tests
- [ ] write test262 tests
- [ ] receive implementer feedback

Stage 4

- [ ] committee approval
- [ ] two implementations
  - [ ] JavaScriptCore
  - [ ] SpiderMonkey
  - [x] XS
  - [ ] V8
- [ ] significant in-the-field experience
- [ ] ecma262 PR approved
- [ ] prepare ecma262 PR

Questions? Stage 2.7?