Standard Standard Ecma-XXX

1st Edition / 31 May 2025

Minimum Common Web Platform API

Rue du Rhône 114 CH-1204 Geneva T: +41 22 849 6000 F: +41 22 849 6001

Ecma International
Rue du Rhone 114 CH-1204 Geneva
Tel: +41 22 849 6000
Fax: +41 22 849 6001
Web: https://www.ecma-international.org

is the registered trademark of Ecma International

COPYRIGHT PROTECTED DOCUMENT

© Ecma International

Table of Contents

page

1 Terminology 4
2 Common API Index 4
3 The Global Scope 6
4 Requirements for navigator.userAgent 6
5 Extensions 7
References 7
Normative References 7
Index 8
Terms defined by this specification 8
Terms defined by reference 8

© Ecma International 2025

Introduction

The Minimum Common Web Platform API is a curated subset of standardized Web Platform APIs intended to define a minimum set of capabilities common to Browser and Non-Browser JavaScript-based runtime environments.

© Ecma International 2025

Contributing to this Specification

This version:
https://min-common-api.proposal.wintercg.org/

Issue Tracking:
GitHub

Editor:
James M Snell (Cloudflare)

1. Terminology

The Web Platform is the combination of technology standards defined by organizations such as the W3C, the WHATWG, and others as implemented by Web Browsers.

A *Web-interoperable Runtime* is any ECMAScript-based application runtime environment that implements the subset of Web Platform APIs outlined in this specification. While this term is intentionally broad to also encompass Web Browsers, the primary focus here is on outlining expectations for non-browser runtimes.

2. Common API Index

All Web-interoperable Runtimes conforming to this specification SHALL implement each of the following Web Platform APIs in accordance with their normative requirements except where modified here. Where any conforming runtime environment chooses (either by necessity or otherwise) to diverge from a normative requirement of the specification, clear explanations of such divergence MUST be made clearly and readily available in the documentation.

Interfaces:

- `AbortController`
- `AbortSignal`
- `Blob`
- `ByteLengthQueuingStrategy`
- `CompressionStream`
- `CountQueuingStrategy`
- `Crypto`
- `CryptoKey`
- `DecompressionStream`
- `DOMException`
- `Event`
- `EventTarget`
- `File`

© Ecma International 2025

- `FormData`

  The `FormData` constructor optionally takes `HTMLFormElement` and `HTMLElement` as parameters. TODO: Figure out what implementations without DOM support should do here. Node.js and Deno throw if the first parameter is not `undefined` but ignore the second parameter. Cloudflare Workers ignores all parameters.

- `Headers`
- `ReadableByteStreamController`
- `ReadableStream`
- `ReadableStreamBYOBReader`
- `ReadableStreamBYOBRequest`
- `ReadableStreamDefaultController`
- `ReadableStreamDefaultReader`
- `Request`
- `Response`
- `SubtleCrypto`
- `TextDecoder`
- `TextDecoderStream`
- `TextEncoder`
- `TextEncoderStream`
- `TransformStream`
- `TransformStreamDefaultController`
- `URL`
- `URLPattern`
- `URLSearchParams`
- `WebAssembly.Global`
- `WebAssembly.Instance`
- `WebAssembly.Memory`
- `WebAssembly.Module`
- `WebAssembly.Table`
- `WritableStream`
- `WritableStreamDefaultController`

Global methods / properties:

© Ecma International 2025

- `globalThis`
- globalThis.`atob()`
- globalThis.`btoa()`
- globalThis.`console`
- globalThis.`crypto`
- globalThis.`fetch()`
- globalThis.`navigator.userAgent`
- globalThis.`performance.now()`
- globalThis.`performance.timeOrigin`
- globalThis.`queueMicrotask()`
- globalThis.`setTimeout()` / globalThis.`clearTimeout()`
- globalThis.`setInterval()` / globalThis.`clearInterval()`
- globalThis.`structuredClone()`
- globalThis.`WebAssembly.compile()`
- globalThis.`WebAssembly.compileStreaming()`
- globalThis.`WebAssembly.instantiate()`
- globalThis.`WebAssembly.instantiateStreaming()`
- globalThis.`WebAssembly.validate()`

3. The Global Scope

The exact type of the global scope (**`globalThis`**) can vary across runtimes. Most Web Platform APIs are defined in terms that assume Web Browser environments that specifically expose types like `Window`, `Worker`, `WorkerGlobalScope`, and so forth. To simplify conformance, all Interfaces, methods, and properties defined by this specification MUST be exposed on the runtime's relevant global scope (e,g., **`globalThis.crypto`**, **`globalThis.ReadableStream`**, etc).

With many runtimes, adding a new global-scoped property can introduce breaking changes when the new global conflicts with existing application code. Many Web Platform APIs define global properties using the **`readonly`** attribute. To avoid introducing breaking changes, runtimes conforming to this specification MAY choose to ignore the **`readonly`** attribute for properties being added to the global scope.

4. Requirements for navigator.userAgent

The globalThis.`navigator.userAgent` property is provided such that application code can reliably identify the runtime within which it is running. The value of the property is a string conforming to the `User-Agent` construction in RFC 7231:

© Ecma International 2025

```
User-Agent      = product *( RWS ( product / comment ) )
product         = token ["/" product-version]
product-version = token
```

While runtimes that implement globalThis.**`navigator.userAgent`** MUST provide a value that is conformant with the structure defined by RFC 7231, the value SHOULD be treated as a single, complete, opaque, unstructured value. It is RECOMMENDED that the value be limited to a single **`product`** token excluding the optional **`product-version`**. For instance, **`navigator.userAgent = 'MyRuntime'`**. The value SHOULD NOT include any **`comment`** components.

5. Extensions

Runtime-specific extensions to any Web Platform API MAY be implemented by conforming runtimes. Such extensions MUST be defined so that their use neither contradicts nor causes the non-conformance of normative functionality of any Web Platform API.

Application use of such extensions must be carefully considered, as doing so reduces interoperability and portability of code across runtimes.

References

Normative References

[CONSOLE]
Dominic Farolino; Robert Kowalski; Terin Stock. *Console Standard*. Living Standard. URL: https://console.spec.whatwg.org/

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

[ENCODING]
Anne van Kesteren. *Encoding Standard*. Living Standard. URL: https://encoding.spec.whatwg.org/

[FETCH]
Anne van Kesteren. *Fetch Standard*. Living Standard. URL: https://fetch.spec.whatwg.org/

[FileAPI]
Marijn Kruisselbrink. *File API*. URL: https://w3c.github.io/FileAPI/

[HR-TIME-3]
Yoav Weiss. *High Resolution Time*. URL: https://w3c.github.io/hr-time/

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

[STREAMS]
Adam Rice; et al. *Streams Standard*. Living Standard. URL: https://streams.spec.whatwg.org/

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

[URLPATTERN]
Ben Kelly; Jeremy Roman; 宍戸俊哉 (Shunya Shishido). *URL Pattern Standard*. Living Standard. URL: https://urlpattern.spec.whatwg.org/

[WASM-JS-API-2]
. Ms2ger. *WebAssembly JavaScript Interface*. URL: https://webassembly.github.io/spec/js-api/

© Ecma International 2025

[WASM-WEB-API-2]
. Ms2ger. *WebAssembly Web API*. URL: https://webassembly.github.io/spec/web-api/

[WEBCRYPTO-2]
Daniel Huigens. *Web Cryptography Level 2*. URL: https://w3c.github.io/webcrypto/

[WEBIDL]
Edgar Chen; Timothy Gu. *Web IDL Standard*. Living Standard. URL: https://webidl.spec.whatwg.org/

[XHR]
Anne van Kesteren. *XMLHttpRequest Standard*. Living Standard. URL: https://xhr.spec.whatwg.org/

Index

Terms defined by this specification

- Web-interoperable Runtime, in § 1

Terms defined by reference

- [CONSOLE] defines the following terms:
  - console
- [DOM] defines the following terms:
  - AbortController
  - AbortSignal
  - Event
  - EventTarget
- [ENCODING] defines the following terms:
  - TextDecoder
  - TextDecoderStream
  - TextEncoder
  - TextEncoderStream
- [FETCH] defines the following terms:
  - Headers
  - Request
  - Response
  - fetch(input)
- [FileAPI] defines the following terms:
  - Blob
  - File
- [HR-TIME-3] defines the following terms:
  - now()
  - performance
  - timeOrigin
- [HTML] defines the following terms:
  - HTMLElement
  - HTMLFormElement
  - Window
  - Worker
  - WorkerGlobalScope
  - atob(data)
  - btoa(data)
  - clearInterval(id)
  - clearTimeout(id)
  - navigator
  - queueMicrotask(callback)
  - setInterval(handler, timeout, ...arguments)
  - setTimeout(handler, timeout, ...arguments)

© Ecma International 2025

    - structuredClone(value, options)
    - userAgent
- [STREAMS] defines the following terms:
    - ByteLengthQueuingStrategy
    - CountQueuingStrategy
    - ReadableByteStreamController
    - ReadableStream
    - ReadableStreamBYOBReader
    - ReadableStreamBYOBRequest
    - ReadableStreamDefaultController
    - ReadableStreamDefaultReader
    - TransformStream
    - TransformStreamDefaultController
    - WritableStream
    - WritableStreamDefaultController
- [URL] defines the following terms:
    - URL
    - URLSearchParams
- [URLPATTERN] defines the following terms:
    - URLPattern
- [WASM-JS-API-2] defines the following terms:
    - Global
    - Instance
    - Memory
    - Module
    - Table
    - WebAssembly
    - compile(bytes)
    - instantiate(bytes)
    - validate(bytes)
- [WASM-WEB-API-2] defines the following terms:
    - compileStreaming(source)
    - instantiateStreaming(source)
- [WEBCRYPTO-2] defines the following terms:
    - Crypto
    - CryptoKey
    - SubtleCrypto
    - crypto
- [WEBIDL] defines the following terms:
    - DOMException
- [XHR] defines the following terms:
    - FormData

© Ecma International 2025