↑ Jump to Table of Contents→ Pop Out Sidebar light dark auto

Non-element Selectors Module Level 1

W3C Group Note, 24 January 2024

▼ More details about this document

This version:
https://www.w3.org/TR/2024/WG-NOTE-selectors-nonelement-1-20240124/
Latest published version:
https://www.w3.org/TR/selectors-nonelement-1/
Editor's Draft:
https://drafts.csswg.org/selectors-nonelement/
Previous Versions:
https://www.w3.org/TR/2014/WD-selectors-nonelement-1-20140603/
History:
https://www.w3.org/standards/history/selectors-nonelement-1/
Feedback:
CSSWG Issues Repository
Editors:
Jirka Kosek (Invited Expert)
Tab Atkins Jr. (Google)
Suggest an Edit for this Spec:
GitHub Editor

Copyright © 2024 World Wide Web Consortium. W3C® liability, trademark and permissive document license rules apply.

Abstract

Non-element Selectors extends [SELECTORS4] and allow selecting other kinds of document nodes than elements. This is useful when selectors are used as a general document query language. Non-element Selectors are not intended to be used in CSS, but only as a separate query language in other host environments.

CSS is a language for describing the rendering of structured documents (such as HTML and XML) on screen, on paper, etc.

Status of this document

This section describes the status of this document at the time of its publication. A list of current W3C publications and the latest revision of this technical report can be found in the W3C standards and drafts

index.

Please send feedback by filing issues in GitHub (preferred), including the spec code “selectors-nonelement” in the title, like this: “[selectors-nonelement] *...summary of comment...*”. All issues and comments are archived. Alternately, feedback can be sent to the (archived) public mailing list www-style@w3.org.

This document is governed by the 18 August 2025 W3C Process Document.

This document was produced by a group operating under the W3C Patent Policy. W3C maintains a public list of any patent disclosures made in connection with the deliverables of the group; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent that the individual believes contains Essential Claim(s) must disclose the information in accordance with section 6 of the W3C Patent Policy.

Table of Contents

1. Introduction

Selectors are a very popular mechanism for selecting things in HTML and XML content. They are not used only in CSS [CSS3SYN] but also as a standalone query language in libraries like jQuery, in newer standardized browser APIs like [SELECTORS-API] and in other Web standards like ITS 2.0.

The [SELECTORS4] specification only defines selectors for selecting element nodes from the document, but some uses of Selectors would like to select other types of nodes as well. This specification extends [SELECTORS4] with additional selectors that can be used for selecting non-element nodes in a document tree.

Note: Currently the draft only defines means for selecting and matching attribute nodes, but other kinds of nodes, such as comments or processing instructions, might be supported in the future.

2. Non-element Selectors

2.1. Attribute node selector

An *attribute node selector* represents an attribute node in a document tree. Its syntax is:

```
::attr() = ::attr( <namespace-attr>? )
<namespace-attr> = [ <na-prefix>? '|' ]? <na-name>
<na-prefix> = <ident> | '*'
<na-name> = <ident> | '*'
```

No whitespace is allowed between the tokens of <namespace-attr>.

<namespace-attr> is divided into two halves: an optional prefix preceding a '|' character, and an attribute name following it.

If the <na-prefix> is provided as an <ident>, it must match a declared namespace prefix, in which case the selector only matches attributes in that namespace; if it doesn't match a declared namespace prefix, the selector matches nothing. If the <na-prefix> is provided as a '*' character, the selector matches attributes in any namespace. If the <na-prefix> is omitted, the selector only matches attributes in no namespace.

If the <na-name> is an <ident>, the selector matches attributes with that name. If the <na-name> is a '*' character, the selector matches attributes with any name.

If the <namespace-attr> is omitted entirely, the selector matches any attribute in any namespace.

The selector matches an attribute node with the given namespace and name on the originating element, if such an attribute exists.

The selector uses pseudo-element syntax.

Example 1 The following ITS rules use an attribute node selector to switch off translatability of `title` attribute on `abbr` elements. ¶

```
<rules xmlns="http://www.w3.org/2005/11/its"
       version="2.0"
       queryLanguage="css">
  <translateRule selector="abbr::attr(title)" translate="no"/>
</rules>
```

Although entirely valid in Selectors used in the scope of CSS, attribute node selectors never generate boxes.

Privacy Considerations

No new privacy considerations have been reported on this specification.

Security Considerations

No new security considerations have been reported on this specification.

Conformance

Document conventions

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

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

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

Example 2 ¶

This is an example of an informative example.

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

Note, this is an informative note.

Advisements are normative sections styled to evoke special attention and are set apart from other normative text with `<strong class="advisement">`, like this: **UAs MUST provide an accessible alternative.**

Conformance classes

Conformance to this specification is defined for three conformance classes:

style sheet
: A CSS style sheet.

renderer
: A UA that interprets the semantics of a style sheet and renders documents that use them.

authoring tool
: A UA that writes a style sheet.

A style sheet is conformant to this specification if all of its statements that use syntax defined in this module are valid according to the generic CSS grammar and the individual grammars of each feature defined in this module.

A renderer is conformant to this specification if, in addition to interpreting the style sheet as defined by the appropriate specifications, it supports all the features defined by this specification by parsing them correctly and rendering the document accordingly. However, the inability of a UA to correctly render a document due to limitations of the device does not make the UA non-conformant. (For example, a UA is not required to render color on a monochrome monitor.)

An authoring tool is conformant to this specification if it writes style sheets that are syntactically correct according to the generic CSS grammar and the individual grammars of each feature in this module, and meet all other conformance requirements of style sheets as described in this module.

Partial implementations

So that authors can exploit the forward-compatible parsing rules to assign fallback values, CSS renderers

must treat as invalid (and ignore as appropriate) any at-rules, properties, property values, keywords, and other syntactic constructs for which they have no usable level of support. In particular, user agents **must not** selectively ignore unsupported component values and honor supported values in a single multi-value property declaration: if any value is considered invalid (as unsupported values must be), CSS requires that the entire declaration be ignored.

Implementations of Unstable and Proprietary Features

To avoid clashes with future stable CSS features, the CSSWG recommends following best practices for the implementation of unstable features and proprietary extensions to CSS.

Non-experimental implementations

Once a specification reaches the Candidate Recommendation stage, non-experimental implementations are possible, and implementors should release an unprefixed implementation of any CR-level feature they can demonstrate to be correctly implemented according to spec.

To establish and maintain the interoperability of CSS across implementations, the CSS Working Group requests that non-experimental CSS renderers submit an implementation report (and, if necessary, the testcases used for that implementation report) to the W3C before releasing an unprefixed implementation of any CSS features. Testcases submitted to W3C are subject to review and correction by the CSS Working Group.

Further information on submitting testcases and implementation reports can be found from on the CSS Working Group's website at http://www.w3.org/Style/CSS/Test/. Questions should be directed to the public-css-testsuite@w3.org mailing list.

Index

Terms defined by this specification

- ::attr(), in § 2.1
- attribute node selector, in § 2.1
- <namespace-attr>, in § 2.1
- <na-name>, in § 2.1
- <na-prefix>, in § 2.1

Terms defined by reference

- [CSS-NAMESPACES-3] defines the following terms:
  - namespace prefix
- [CSS-VALUES-4] defines the following terms:
  - <ident>
  - ?
  - |
- [SELECTORS4] defines the following terms:
  - originating element
  - pseudo-elements

References

Normative References

[CSS-NAMESPACES-3]
Elika Etemad. *CSS Namespaces Module Level 3*. 20 March 2014. REC. URL: https://www.w3.org/TR/css-namespaces-3/

[CSS-VALUES-4]
Tab Atkins Jr.; Elika Etemad. *CSS Values and Units Module Level 4*. 12 March 2024. WD. URL: https://www.w3.org/TR/css-values-4/

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

[SELECTORS4]
Elika Etemad; Tab Atkins Jr.. *Selectors Level 4*. 22 January 2026. WD. URL: https://www.w3.org/TR/selectors-4/

Informative References

[CSS3SYN]
Tab Atkins Jr.; Simon Sapin. *CSS Syntax Module Level 3*. 24 December 2021. CRD. URL: https://www.w3.org/TR/css-syntax-3/

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