TTA Standard

정보통신단체표준(국문표준)

TTAK.OT-10.0356 제정일: 2014년 12월 17일

CSS 셀렉터 레벨 3

CSS Selectors Level 3

한국정보통신기술협회
Telecommunications Technology Association

정보통신단체표준(국문표준)
TTAK.OT-10.0356 제정일: 2014년 12월 17일

CSS 셀렉터 레벨 3

CSS Selectors Level 3

한국정보통신기술협회
Telecommunications Technology Association

서 문

1. 표준의 목적

본 표준은 CSS1과 CSS2에 존재하는 셀렉터를 설명하고, CSS3와 관련한 신규 셀렉터 및 그것을 필요로 할 수 있는 기타 언어에 대해 소개한다. CSS는 스타일 속성을 문서 요소로 묶기 위해 셀렉터를 사용한다.

2. 주요 내용 요약

CSS(Cascading Style Sheets)는 HTML 및 XML 문서 렌더링을 표현하기 위한 언어이다. CSS 셀렉터는 특정 트리의 구성요소와 일치하는 패턴으로, XML 문서에서 노드를 선택하는 데 사용될 수 있는 기술이다. 셀렉터는 HTML 및 XML과 함께 사용하도록 최적화되었으며, 성능 필수 코드에서 사용 가능하도록 설계되었다.

3. 표준 적용 산업 분야 및 산업에 미치는 영향

본 표준을 통해 표준 기반의 웹 응용 활성화와 이를 통한 산업 활성화를 기대할 수 있다.

4. 참조 표준(권고)

4.1. 국외 표준(권고)

- W3C, ‘Selectors Level 3’, 2011. 09. 29., http://www.w3.org/TR/css3-selectors/

4.2. 국내 표준

- 해당 사항 없음.

5. 참조표준(권고)과의 비교

5.1. 참조 표준(권고)과의 관련성

본 표준은 W3C의 표준인 Selectors Level 3를 참조하여 번역하였다.

5.2. 참조한 표준(권고)과 본 표준의 비교표

TTAK.OT-10.0356	W3C CSS Selectors Level 3	비고
1. 개요	1. Introduction	동일
2. 표준의 구성 및 범위	-	추가
3. 참조 표준(권고)	-	추가
4. 용어 정의	-	추가
5. 셀렉터	2. Selectors	동일
6. 대소문자 구분	3. Case sensitivity	동일
7. 셀렉터 구문	4. Selector syntax	동일
8. 셀렉터 그룹	5. Groups of selectors	동일
9. 단순 셀렉터	6. Simple selectors	동일
10. 의사 요소	7. Pseudo-elements	동일
11. 콤비네이터	8. Combinators	동일
12. 셀렉터의 특이성 계산	9. Calculating a selector's specificity	동일
13. 셀렉터 문법	10. The grammar of Selectors	동일
14. 프로파일	11. Profiles	동일
15. 적합성 요구 사항	12. Conformance and requirements	동일
16. 테스트	13. Tests	동일
부록Ⅰ. 참조	14. References	동일

6. 지적 재산권 관련 사항

본 표준의 '지적 재산권 확약서' 제출 현황은 TTA 웹사이트에서 확인할 수 있다.

※본 표준을 이용하는 자는 이용함에 있어 지적 재산권이 포함되어 있을 수 있으므로, 확인 후 이용한다.

※본 표준과 관련하여 접수된 확약서 이외에도 지적 재산권이 존재할 수 있다.

7. 시험 인증 관련 사항

7.1. 시험 인증 대상 여부

- 해당 사항 없음.

7.2. 시험 표준 제정 현황

- 해당 사항 없음.

8. 표준의 이력 정보

8.1. 표준의 이력

판 수	제정·개정일	제정·개정 내역
제1판	2013.12.18.	잠정표준 제정 TTAI.OT-10.0356
제1판	2014.12.17.	단체표준 제정 TTAK.OT-10.0356

8.2. 주요 개정 사항

- 해당 사항 없음.

Preface

1. Purpose of Standard

This document describes the selectors that already exist in CSS1 and CSS2 and further introduces new selectors for CSS3 and other languages that may need them. CSS uses Selectors for binding style properties to elements in the document.

2. Summary of Contents

CSS(Cascading Style Sheets) is a language for describing the rendering of HTML and XML documents on screen, on paper, in speech, etc. Selectors are patterns that match against elements in a tree, and as such form one of several technologies that can be used to select nodes in an XML document. Selectors have been optimized for use with HTML and XML, and are designed to be usable in performance-critical code.

3. Applicable fields of industry and its effect

This standard will encourage the activation of web industry. This standard will also contribute to related technology development and activate related application service. In addition, this standard will ensure web application compatibility and consistency to gradually vitalize the web application and its services.

4. Reference Standards(Recommendations)

4.1. International Standards(Recommendations)

- W3C, “Selectors Level 3”, 29 September 2011, http://www.w3.org/TR/css3-selectors/

4.2. Domestic Standards

- None.

5. Relationship to Reference Standards(Recommendations)

5.1. Relationship of Reference Standards(Recommendations)

"W3C Selectors Level 3" is fully referred here.

5.2. Differences between Reference Standard(Recommendation) and this Standard

TTAK.OT-10.0356	W3C CSS Selectors Level 3	Remarks
1. Introduction	1. Introduction	Equivalent
2. Constitution and Scope	-	Added
3. Reference Standards (Recommendations)	-	Added
4. Terms and Definitions	-	Added
5. Selectors	2. Selectors	Equivalent
6. Case sensitivity	3. Case sensitivity	Equivalent
7. Selector syntax	4. Selector syntax	Equivalent
8. Groups of selectors	5. Groups of selectors	Equivalent
9. Simple selectors	6. Simple selectors	Equivalent
10. Pseudo-elements	7. Pseudo-elements	Equivalent
11. Combinators	8. Combinators	Equivalent
12. Calculating a selector's specificity	9. Calculating a selector's specificity	Equivalent
13. The grammar of Selectors	10. The grammar of Selectors	Equivalent
14. Profiles	11. Profiles	Equivalent
15. Conformance and requirements	12. Conformance and requirements	Equivalent
16. Tests	13. Tests	Equivalent
Appendix Ⅰ. References	14. References	Equivalent

6. Statement of Intellectual Property Rights

IPRs related to the present document may have been declared to TTA. The information pertaining to these IPRs, if any, is available on the TTA Website.

No guarantee can be given as to the existence of other IPRs not referenced on the TTA website.

And, please make sure to check before applying the standard.

7. Statement of Testing and Certification

7.1. Object of Testing and Certification

- None.

7.2. Standards of Testing and Certification

- None.

8. History of Standard

8.1. Change History

Edition	Issue Date	Outline
The 1st edition	2013.12.18.	Established TTAI.OT-10.0356
The 1st edition	2014.12.17.	Established TTAK.OT-10.0356

8.2. Revisions

- None.

목 차

Contents

CSS 셀렉터 레벨 3

(CSS Selectors Level 3)

1. 개요

본 표준은 CSS1과 CSS2에 존재하는 셀렉터를 설명하고, CSS3와 관련한 신규 셀렉터 및 그것을 필요로 할 수 있는 기타 언어에 대해 소개한다. CSS는 스타일 속성을 문서 요소로 묶기 위해 셀렉터를 사용한다.

1.1. CSS2 로부터 변경된 사항들

본 항목은 비 규범적이다.

CSS2의 셀렉터와의 주요 차이점은 다음과 같다.

- 기본 정의 목록(셀렉터, 셀렉터 그룹, 단순 셀렉터 등)가 바뀌었다. 특히 CSS2에서 단순 셀렉터로 불리던 것이 이제는 단순 셀렉터 시퀀스로 불리고, “단순 셀렉터(simple selector)”라는 용어가 이제는 이 시퀀스의 구성 요소에 대해 사용된다.
- 선택적 네임스페이스 구성 요소가 지금은 요소 타입 셀렉터(element type selector), 전체 셀렉터(universal selector) 및 특성 셀렉터(attribute selector)에서 허용된다.
- 신규 콤비네이터(new combinatory) 도입.
- 특성 셀렉터에 매칭되는 부속 문자열을 포함한 단순 셀렉터와 새로운 슈도 클래스(pseudo-class).
- 신규 슈도(pseudo) 엘리먼트 및 슈도(pseudo) 엘리먼트를 위한 "::" 규약 도입.
- 문법이 재작성됨.
- 프로파일(profile)을 규격에 추가함으로써 셀렉터를 통합시키고 실제 각 규격에 의해 지원되는 셀렉터 집합을 정의.

- 셀렉터는 이제 CSS3 모듈이자 독립적인 규격임. CSS와 무관하게 다른 규격들도 이제는 본 표준을 참조할 수 있음.
- 규격 자체 테스트 항목(test suite) 확보.

2. 표준의 구성 및 범위

본 표준은 총 17개의 장으로 구성되어 있고, 셀럭터의 표현 방법, 셀렉터 유형, 의사 요소, 셀렉터 문법, 적합성 요구사항과 테스트 방법 등에 대해 정의하고 있다

3. 참조 표준(권고)

- W3C, 'Selectors Level 3', 2011. 9. 29., http://www.w3.org/TR/css3-selectors/

4. 용어 정의

4.1. CSS(Cascading Style Sheets)

웹 문서의 전반적인 스타일을 미리 저장해 두기 위한 스타일 시트

5. 셀렉터

본 항목은 비 규범적으로, 이어지는 항목을 요약하는 데 그친다.

셀렉터는 특정 구조를 나타낸다. 이 구조는 특정 셀렉터가 문서 트리에서 매칭하는 요소를 결정하는 일종의 조건이나 그 구조와 일치하는 HTML 또는 XML 단편에 대한 평이한 설명으로 사용할 수 있다.

셀렉터의 범위는 단순 요소 명칭에서부터 맥락 표현문에 이르기까지 광범위하다.

아래의 표에 셀렉터 구문이 요약되어 있다:

패턴	의미	항목 설명	CSS 레벨 최초 정의
*	모든 요소	유니버셜 셀렉터	2
E	E 타입 요소	타입 셀렉터	1
E[foo]	"foo" 속성을 갖춘 E 요소	어트리뷰트 셀렉터	2
E[foo="bar"]	"foo" 속성 값이 “bar”와 정확히 일치하는 E 요소	어트리뷰트 셀렉터	2
E[foo~="bar"]	"foo" 속성 값이 공백으로 구분된 값 목록으로, 그 중 하나가 “bar”와 정확히 일치하는 E 요소	어트리뷰트 셀렉터	2
E[foo^="bar"]	"foo" 속성 값이 정확히 문자열 "bar"로 시작하는 E 요소	어트리뷰트 셀렉터	3
E[foo$="bar"]	"foo" 속성 값이 정확히 문자열 "bar"로 끝나는 E 요소	어트리뷰트 셀렉터	3
E[foo*="bar"]	"foo" 속성 값에 부속 문자열 "bar"가 들어있는 E 요소	어트리뷰트 셀렉터	3
E[foo|="en"]	"foo" 속성 값에 (왼쪽부터) “en”으로 시작하는 하이픈으로 구분된 값이 들어있는 E 요소	어트리뷰트 셀렉터	2
E:root	특정 E 요소, 문서의 루트	구조적 슈도 클래스	3
E:nth-child(n)	특정 E 요소, 부모의 n번째 자식	구조적 슈도 클래스	3
E:nth-last-child (n)	특정 E 요소, 부모의 n번째 자식, 마지막부터 카운트	구조적 슈도 클래스	3
E:nth-of-type(n)	특정 E 요소, 이런 타입의 n번째 형제	구조적 슈도 클래스	3
E:nth-last-	특정 E 요소, 이런 타입의 n번째 형	구조적 슈도 클래스	3

패턴	의미	항목 설명	CSS 레벨 최초 정의
of-type(n)	제, 마지막부터 카운트		
E:first- child	특정 E 요소, 부모의 첫 번째 자식	구조적 슈도 클래스	2
E:last- child	특정 E 요소, 부모의 마지막 자식	구조적 슈도 클래스	3
E:first-of-type	특정 E 요소, 이런 타입의 첫 번째 형제	구조적 슈도 클래스	3
E:last-of-type	특정 E 요소, 이런 타입의 마지막 형제	구조적 슈도 클래스	3
E:only- child	특정 E 요소, 부모의 유일한 자식	구조적 슈도 클래스	3
E:only-of-type	특정 E 요소, 이런 타입의 유일한 형제	구조적 슈도 클래스	3
E:empty	자식이 없는 E 요소 (텍스트 노드 포함)	구조적 슈도 클래스	3
E:link E:visited	타깃이 아직 방문되지 않았거나 (:link) 이미 방문된 (:visited) 하이퍼링크의 소스 앵커가 되는 E 요소	The link 슈도 클래스	1
E:active E:hover E:focus	특정 사용자 조치 중인 E 요소	사용자 조치 슈도 클래스	1 & 2
E:target	Referring URI의 타깃이 되는 E 요소	target 슈도 클래스	3
E:lang(fr)	"fr" 언어로 된 타입 E 요소 (문서 언어는 언어 결정 메서드를 지정)	:lang() 슈도 클래스	2
E:enabled E:disabled	활성화 또는 해제되는 사용자 인터페이스 요소 E	UI element states 슈도 클래스	3
E:checked	체크되는 사용자 인터페이스 요소 E (예; 라디오 버튼이나 체크박스)	The UI element states 슈도 클래스	3

패턴	의미	항목 설명	CSS 레벨 최초 정의
E::first-line	최초의 E 요소 포맷 문자열	::first-line 의사 요소	1
E::first-letter	최초의 E 요소 포맷 글자	::first-letter 의사 요소	1
E::before	E 요소 전 생성된 내용	::before 의사 요소	2
E::after	E 요소 후 생성된 내용	::after 의사 요소	2
E.warning	클래스가 “경고”인 E 요소 (문서 언어는 클래스 결정 메서드를 지정).	클래스 셀렉터	1
E#myid	"myid"와 동일한 ID의 E 요소.	ID 셀렉터	1
E:not(s)	단순 셀렉터를 매칭하지 않는 E 요소	부인 슈도 클래스	3
E F	E 요소에 종속되는 F 요소	하위 콤비네이터	1
E > F	E 요소의 F 요소 자식	자식 콤비네이터	2
E + F	E 요소 직계의 F 요소	근친 형제 콤비네이터	2
E ~ F	E 요소 후손의 F 요소	일반 형제 콤비네이터	3

각 셀렉터의 의미는 “의미” 칸의 각 셀 내용 앞에 “매치”를 배치함으로써 위의 표에서 도출된다.

6. 대소문자 구분

셀렉터의 통제력을 벗어나는 부분을 제외하고, 모든 셀렉터 구문은 ASCII 범위(즉, a-z 및 A-Z가 해당됨) 내에서 케이스에 둔감하다. 셀렉터에서 문서 요소 명칭, 속성 명칭 및 속성 값의 케이스 감도는 문서 언어에 좌우된다. 예를 들어 HTML에서는 요소가 케이스에 둔감하지만, XML에서는 케이스에 민감하다. 네임스페이스 접두사의 케이스 감도는 [CSS3NAMESPACE]에서 정의된다.

7. 셀렉터 구문

셀렉터는 콤비네이터로 구분되는 하나 이상의 연쇄 단순 셀렉터 시퀀스이다. 특정 셀렉터의 단순 셀렉터 중 마지막 시퀀스에 하나의 의사 요소가 부가될 수 있다.

단순 셀렉터 시퀀스는 콤비네이터로 구분되지 않는 단순 셀렉터 체인이다. 항상 타입 셀렉터 또는 유니버셜 셀렉터로 시작한다. 다른 타입 셀렉터나 유니버셜 셀렉터는 이 시퀀스에서 허용되지 않는다.

단순 셀렉터는 타입 셀렉터, 유니버셜 셀렉터, 어트리뷰트 셀렉터, 클래스 셀렉터, ID 셀렉터, 또는 슈도 클래스이다.

콤비네이터: whitespace, "greater-than sign" (U+003E, >), "plus sign" (U+002B, +) 및 "tilde" (U+007E, ~). 공백은 콤비네이터와 그 주위의 단순 셀렉터 사이에서 나타난다. 공백에서는 글자 "space" (U+0020), "tab" (U+0009), "line feed" (U+000A), "carriage return" (U+000D), 그리고 "form feed" (U+000C)만이 이루어질 수 있다. "em-space" (U+2003) 및 "ideographic space" (U+3000)와 같은 기타 공간과 유사한 글자는 공백의 일부가 되지 못한다.

셀렉터로 표현되는 문서 트리 요소는 셀렉터의 대상이다. 하나의 단순 셀렉터 시퀀스로 이루어진 셀렉터는 요건을 충족시키는 모든 요소들을 나타낸다. 특정 시퀀스 앞에 다른 단순 셀렉터 시퀀스와 콤비네이터를 배치하면 추가적인 매칭 제약이 부여되므로, 셀렉터의 대상은 항상 단순 셀렉터의 마지막 시퀀스로 표현되는 요소의 부분 집합이다.

단순 셀렉터 시퀀스와 의사 요소가 들어있지 않은 빈 셀렉터는 무효한 셀렉터이다.

셀렉터의 글자는 CSS와 동일한 이스케이핑 규칙에 따라 백슬래시로 벗어날 수 있다. [CSS21].

특정 셀렉터는 네임스페이스 접두사를 지원한다. 네임스페이스 접두사가 선언되는 메커니즘은 셀렉터를 사용하는 언어에 의해 지정되어야 한다. 언어가 네임스페이스 접두사 선언 메커니즘을 지정하지 않을 경우, 접두사는 선언되지 않는다. CSS에서 네임스페이스 접두사는 @namespace 규칙에 따라 선언된다. [CSS3NAMESPACE]

8. 셀렉터 그룹

콤마로 구분되는 셀렉터 목록은 목록에 수록된 개별 셀렉터 각각에 의해 선택된 모든 요소의 결합체를 나타낸다. (콤마는 U+002C.) 예를 들어, CSS에서 여러 개의 셀렉터가 동일한 선언을 공유할 경우, 콤마로 구분되는 하나의 목록으로 묶을 수 있다. 콤마 앞뒤에 공백이 나타날 수 있다.

CSS 예시:

이 예시에서, 우리는 세 가지 선언이 일치하는 규칙을 하나로 집약시킨다. 따라서,

```
h1 { font-family: sans-serif }
h2 { font-family: sans-serif }
h3 { font-family: sans-serif }
```

는 아래에 해당한다:

```
h1, h2, h3 { font-family: sans-serif }
```

경고: 이 예시에서는 모든 셀렉터들이 유효한 셀렉터이므로 이러한 동일성은 참이다. 이 셀렉터들 중 하나만 무효화해도 셀렉터 그룹 전체가 무효화된다. 이렇게 되면 세 가지 헤딩 요소 규칙이 모두 무효화되는 반면, 전자의 케이스에서는 세 가지 개별 헤딩 중 하나만 무효화된다.

무효 CSS 예시:

```
h1 { font-family: sans-serif }
h2..foo { font-family: sans-serif }
h3 { font-family: sans-serif }
```

는 아래와 같지 않다:

```
h1, h2..foo, h3 { font-family: sans-serif }
```

위의 셀렉터(h1, h2..foo, h3)는 완전히 무효화되고 스타일 규칙 전체가 무시되기 때문이다(셀렉터가 집약되지 않은 경우, h2..foo 규칙만이 무시된다).

9. 단순 셀렉터

9.1. 타입 셀렉터

타입 셀렉터는 CSS 공인 명칭의 구문을 사용하여 작성되는 문서 언어 요소 타입의 명칭이다[CSS3NAMESPACE]. 타입 셀렉터는 문서 트리에서 요소 타입 인스턴스를 나타낸다.

예시: 아래의 셀렉터는 문서 트리에서 h1 요소를 나타낸다:

```
h1
```

9.1.1. 타입 셀렉터와 네임스페이스

타입 셀렉터는 네임스페이스 옵션 구성요소를 허용한다: 이전에 선언된 네임스페이스 접두사가 네임스페이스 분리 기호 "세로 막대"(U+007C, |)로 분리되는 요소 앞에 배치할 수 있다(예를 들면 XML 네임스페이스 사용은 [XML-NAMES] 참조).

네임스페이스 구성요소를 비워둠으로써(네임스페이스 구분 기호 앞에 접두사 없음) 셀렉터가 네임스페이스가 없는 요소만을 표현한다는 사실을 나타낼 수 있다.

네임스페이스 접두사에 별표를 사용하여 셀렉터가(네임스페이스가 없는 요소를 포함한) 모든 네임스페이스 요소를 표현함을 나타낼 수도 있다.

네임스페이스 구성요소가 없는(네임스페이스 구분 기호 없음) 요소 타입 셀렉터는 네임스페이스 셀렉터에 대하여 기본 네임스페이스가 선언되지 않는 한, ("*|"에 해당하는) 요소의 네임스페이스의 구애를 받지 않고 요소를 나타낸다. 기본 네임스페이스가 선언되면, 해당 셀렉터는 기본 네임스페이스 내의 요소만을 나타낸다.

네임스페이스 셀렉터에 대하여 이전에 선언되지 않은 네임스페이스 접두사가 들어있는 타입 셀렉터는 무효한 셀렉터이다.

네임스페이스 인식 클라이언트에서, 요소 타입 셀렉터의 이름 파트(네임스페이스 구분 기호 다음의 파트, 존재할 경우)는 요소 공인 이름 중 로컬 파트에 대해서만 매칭이 이루

어질 것이다.

요약:

ns|E

네임스페이스 ns에 이름 E가 있는 요소

*|E

모든 네임스페이스에 이름 E가 있는 요소 (네임스페이스가 없는 요소 포함)

|E

네임스페이스 없이 이름 E가 있는 요소

E

셀렉터에 대하여 기본 네임스페이스가 선언되지 않은 경우, *|E에 해당한다. 그렇지 않으면 ns가 기본 네임스페이스인 ns|E에 해당한다.

CSS 예시:

```
@namespace foo url(http://www.example.com);
 foo|h1 { color: blue } /* first rule */
 foo|* { color: yellow } /* second rule */
 |h1 { color: red } /* ...*/
 *|h1 { color: green }
 h1 { color: green }
```

첫 번째 규칙(not counting the @namespace at-rule)은 http://www.example.com 네임스페이스에서 h1 요소와만 매칭한다.

두 번째 규칙은 http://www.example.com 네임스페이스에서 모든 요소들을 매칭한다.

세 번째 규칙은 네임스페이스 없이 h1 요소하고만 매칭한다.

네 번째 규칙은 모든 네임스페이스에서 h1 요소들을 매칭한다(네임스페이스가 없는 요소 포함).

마지막 규칙은 기본 네임스페이스가 정의되지 않았으므로 네 번째 규칙에 해당한다.

9.2. 유니버셜 셀렉터

로컬 이름으로 별표(* U+002A)가 사용된 CSS 공인 명칭[CSS3NAMESPACE]으로 작성된 유니버셜 셀렉터는 요소 타입의 공인 명칭을 나타낸다. 셀렉터에 대하여 기본 네임스페이스가 지정되지 않은 경우, 임의의 네임스페이스에서 문서 트리의 모든 단일 요소를 나타낸다(네임스페이스가 없는 요소 포함). 기본 네임스페이스가 지정된 경우는 유니버셜 셀렉터 및 네임스페이스 참조.

별표로 표시된 (즉 네임스페이스 접두사가 없는) 유니버셜 셀렉터가 단순 셀렉터 시퀀스 셀렉터의 유일한 구성요소가 아니거나 바로 뒤에 의사 요소가 붙을 경우, 별표가 생략되고 유니버셜 셀렉터의 존재가 암시된다.

예시:

*[hreflang|=en]와 [hreflang|=en]는 대등,

*.warning와 .warning 대등,

*#myid와 #myid 대등.

주: 예를 들면 div:first-child와 div:first-child 간에 있을 수 있는 혼선을 줄여주므로, 별표를 생략하지 않는 것이 좋다. 이 경우, div *:first-child이 더 가독성이 좋다.

9.2.1. 유니버셜 셀렉터와 네임스페이스

유니버셜 셀렉터는 옵션 네임스페이스 요소를 허용한다. 다음과 같이 사용된다:

ns|*
네임스페이스 ns 의 모든 요소

|
모든 요소

|*
네임스페이스 없는 모든 요소

*

기본 네임스페이스가 지정되지 않은 경우, 이것은 *|*에 해당한다. 그렇지 않으면 ns|*에 해당하고, ns는 기본 네임스페이스다.

이전에 선언되지 않은 네임스페이스 접두사가 들어있는 유니버셜 셀렉터는 무효한 셀렉터이다.

9.3. 어트리뷰트 셀렉터

셀렉터는 요소 속성의 표현문을 허용한다. 셀렉터를 일종의 표현문으로 활용하여 요소에 대한 매칭을 구현할 경우, 어트리뷰트 셀렉터는 그 요소가 어트리뷰트 셀렉터로 표현되는 속성과 일치하는 속성을 가진 것처럼 요소와 일치하는 것으로 간주되어야 한다.

9.3.1. 속성 존재 및 값 셀렉터

CSS2, 4개의 어트리뷰트 셀렉터 도입:

[att]

속성 값에 구애 받지 않고 att 속성을 갖춘 요소를 나타낸다.

[att=val]

값이 정확히 "val"인 att 속성을 갖춘 요소를 나타낸다.

[att~=val]

값 중 하나가 정확히 "val"인 공백으로 구분되는 단어 목록인 att 속성을 갖춘 요소를 나타낸다. "val"에 공백이 들어있을 경우, (단어가 공간에 의해 구분되므로) 어떤 것도 나타내지 않는다. 또한 "val"이 빈 문자열일 경우에도 아무 것도 나타내지 않는다.

[att|=val]

att 속성이 있거나, 값이 정확히 "val"이거나, "val"로 시작하고 바로 뒤에 "-" (U+002D)가 붙는 요소를 나타낸다. 주 목적은 BCP 47 ([BCP47]) 또는 후속에 설명된 바와 같이 언어 서브 코드 매치를 허용하는 데 있다(예; HTML로 된 a 요소의 hreflang 속성). lang (또는 xml:lang) 언어 서브코드 매칭의 경우, the :lang pseudo-class 참조.

속성 값은 반드시 CSS 식별자 또는 문자열이어야 한다. [CSS21] 속성 명칭과 셀렉터 값의 케이스 감도는 문서 언어에 좌우된다.

예시:

아래의 어트리뷰트 셀렉터는 값과 무관하게 title 속성을 지닌 h1 요소를 나타낸다:

```
h1[title]
```

아래 예시에서, 셀렉터는 class서속성에 정확히 "example" 값이 있는 span 요소를 나타낸다:

```
span[class="example"]
```

복수의 어트리뷰트 셀렉터를 활용하여 특정 요소의 여러 가지 속성이나 동일한 속성의 여러 가지 상태를 나타낼 수 있다. 여기서, 셀렉터는 hello 속성에 정확히 "Cleveland" 값이 있고 goodbye 속성에 정확히 "Columbus" 값이 있는 요소를 나타낸다:

```
span[hello="Cleveland"][goodbye="Columbus"]
```

아래의 CSS 규칙은 "="과 "~=" 간의 차이를 설명한다. 첫 번째 셀렉터는 예를 들어 rel 속성에서 "copyright copyleft copyeditor" 값의 a 요소를 매칭할 것이다. 두 번째 셀렉터는 정확한 값이 "http://www.w3.org/인 속성의 a 요소와만 매칭할 것이다.

```
a[rel~="copyright"] { ... }
a[href="http://www.w3.org/"] { ... }
```

아래의 셀렉터는 hreflang 속성이 정확히 "fr"인 a 요소를 나타낸다.

```
a[hreflang=fr]
```

아래 셀렉터는 hreflang 속성 값이 "en", "en-US" 및 "en-scouse"를 포함해서 "en"으로 시작하는 a 요소를 나타낸다:

```
a[hreflang|="en"]
```

아래 셀렉터는 특정 속성 character에 대하여 두 개의 각기 다른 값이 있는 DIALOGUE 요소를 나타낸다:

DIALOGUE[character=romeo]

DIALOGUE[character=juliet]

9.3.2. 부속 문자열 매칭 어트리뷰트 셀렉터

특정 속성 값의 부속 문자열 매칭을 위해 추가로 세 개의 어트리뷰트 셀렉터가 제공된다:

[att^=val]

값이 접두사 "val"로 시작되는 att 속성의 요소를 나타낸다. "val"이 빈 문자열이면 셀렉터는 아무 것도 나타내지 않는다.

[att$=val]

값이 접미사 "val"로 끝나는 att 속성의 요소를 나타낸다. "val"이 빈 문자열이면 셀렉터는 아무 것도 나타내지 않는다.

[att*=val]

값에 최소 하나 이상의 부속 문자열 "val" 인스턴스가 들어있는 att 속성의 요소를 나타낸다. "val"이 빈 문자열이면 셀렉터는 아무 것도 나타내지 않는다.

속성 값은 반드시 CSS 식별자 또는 문자열이어야 한다. [CSS21] 속성 명칭과 셀렉터 값의 케이스 감도는 문서 언어에 좌우된다.

예시:

아래의 셀렉터는 특정 이미지를 참조하는 HTML object를 나타낸다:

object[type^="image/"]

아래의 셀렉터는 값이 ".html"로 끝나는 href 속성의 HTML 앵커를 나타낸다.

a[href$=".html"]

아래의 셀렉터는 값에 부속 문자열 "hello"가 들어있는 title 속성의 HTML 단락을 나타낸다.

```
p[title*="hello"]
```

9.3.3. 어트리뷰트 셀렉터와 네임스페이스

어트리뷰트 셀렉터의 속성 이름은 CSS 공인 명칭으로 제시된다: 이전에 선언된 네임스페이스 접두사가 네임스페이스 분리 기호 "세로 막대(vertical bar)"(|)로 분리되는 요소 앞에 배치할 수 있다. XML 권고의 네임스페이스와 부합하도록 기본 네임스페이스는 속성에 적용되지 않으므로, 네임스페이스 요소가 없는 어트리뷰트 셀렉터는 네임스페이스에 없는 속성에만 적용한다("|attr"에 해당; 이러한 속성을 요소 타입 당 네임스페이스 파티션(per-element-typenamespace partition)이라고 한다). 네임스페이스 접두사에 별표를 사용하여, 셀렉터가 속성의 네임스페이스와 무관하게 모든 속성 이름에 매칭됨을 나타낼 수도 있다.

이전에 선언되지 않은 네임스페이스 접두사가 들어있는 어트리뷰트 셀렉터는 무효한 셀렉터이다.

CSS 예시:

```
@namespace foo "http://www.example.com";
[foo|att=val] { color: blue }
[*|att] { color: yellow }
[|att] { color: green }
[att] { color: green }
```

첫 번째 규칙은 http://www.example.com 네임스페이스에서 att 속성을 갖춘 요소만을 "val" 값과 매칭시킨다.

두 번째 규칙은 속성의 네임스페이스와 무관하게 att 속성을 갖춘 요소하고만 매칭시킨다(네임스페이스 없음 포함).

마지막 두 가지 규칙은 대등하며, 네임스페이스에 속성이 없는 att 속성을 갖춘 요소하

고만 매칭시킨다.

9.3.4. DTD의 기본 속성 값

어트리뷰트 셀렉터는 문서 트리의 속성 값을 나타낸다. 문서 구축 메서드는 셀렉터의 범위를 벗어난다. 일부 문서 포맷에서는 기본 속성 값을 DTD나 그 밖의 장소에서 정의할 수 있지만, 문서 트리에 나타날 경우에는 어트리뷰트 셀렉터에 의해서만 선택될 수 있다. 셀렉터는 기본 값을 문서 트리에 포함시킬지 여부를 작업하도록 설계되어야 한다.

예를 들어, XML UA는 DTD의 “외부 서브세트”을 읽을 수 있지만, (단, 반드시 읽어야 하는 것은 아님), 문서 “내부 서브세트”의 기본 속성 값은 반드시 찾아야 한다." (이러한 서브세트 정의는 예를 들면 [XML11] 참조) UA에 따라, DTD의 외부 서브세트에 정의된 기본 속성 값은 문서 트리에 나타날 수도 있고 그렇지 않을 수도 있다.

UA는 XML 네임스페이스를 인식하되 그 네임스페이스에 대한 지식을 활용하여 기본 속성 값을 처리해야 할 필요는 없다. (예를 들어, XHTML UA는 XHTML DTD에 대한 내장된 지식을 사용할 필요가 없다. XML 1.0의 네임스페이스에 대한 자세한 내용은 예를 들면 [XML-NAMES] 참조)

주: 통상적으로, 구현은 외부 서브세트를 무시하는 쪽을 선택한다. 이는 XML 규격으로 정의되는 미 검증 프로세서의 거동과 일치한다.

예시:

기본 값이 "decimal"인 radix 속성의 EXAMPLE 요소를 고려해보자. DTD 조각은 <!ATTLIST EXAMPLE radix (decimal,octal) "decimal">이 될 수 있을 것이다.

스타일 시트에 다음과 같은 규칙이 들어있을 경우

```
EXAMPLE[radix=decimal] { /*... default property settings ...*/ }
EXAMPLE[radix=octal] { /*... other settings...*/ }
```

첫 번째 규칙은 radix 속성이 기본으로 설정되는 (즉 명시적으로 설정되지 않은) 요소를 매칭하지 않을 것이다. 모든 케이스를 포착하기 위해, 기본 값에 대하여 어트리뷰트

셀렉터를 반드시 드롭시켜야 한다:

```
EXAMPLE { /*... default property settings ...*/ }
EXAMPLE[radix=octal] { /*... other settings...*/ }
```

셀렉터 EXAMPLE[radix=octal]는 타입 셀렉터보다 더욱 특정하므로, 두 번째 규칙의 스타일 선언은 "octal"의 radix 속성 값을 가진 요소에 대하여 첫 번째 규칙의 스타일 선언을 대체한다. 기본 케이스에만 적용되는 모든 속성 선언이 비 기본 케이스의 스타일 규칙에서 대체된다는 점을 유념해야 한다.

9.4. 클래스 셀렉터

HTML과 공조하여, 저작자들은 class 속성을 표현문할 때 ("full stop", U+002E, .로 알려진) "period" notation을 ~= notation 대신 활용할 수 있다. 따라서 HTML의 경우, div.value와 div[class~=value]는 같은 의미를 가진다. 속성 값 바로 뒤에는 반드시 full stop (.)이 붙어야 한다.

UA에 각 네임스페이스의 "class" 속성을 파악할 수 있는 네임스페이스 전용 지식이 있을 경우, UA는 XML 문서에서 period (.) notation을 활용하여 셀렉터를 적용할 수 있다. 이러한 네임스페이스 전용 지식의 예로 특정 네임스페이스 규격의 산문을 들 수 있다(예를 들어, SVG 1.0 [SVG11]은 SVG class 속성 및 UA의 해석 메서드를 설명하고, 마찬가지로 MathML 1.01[MATHML]은 MathML class 속성을 설명한다.)

CSS 예시:

class~="pastoral"의 모든 요소에 다음과 같이 스타일 정보를 지정할 수 있다:

```
*.pastoral { color: green } /* all elements with class~=pastoral */
```

또는 단지

```
.pastoral { color: green } /* all elements with class~=pastoral */
```

다음은 class~="pastoral"의 H1 요소에만 스타일을 지정한다:

```
H1.pastoral { color: green } /* H1 elements with class~=pastoral */
```

이러한 규칙을 감안하면, 아래의 첫 번째 H1 인스턴스는 녹색 글자(green text)를 가지지 않는 반면, 두 번째 인스턴스는:

```
<H1>Not green</H1>
<H1 class="pastoral">Very green</H1>
```

아래 규칙은 class 속성에 pastoral과 marine을 모두 포함하는 공백 구분된 값이 지정된 모든 P 요소를 매칭한다:

```
p.pastoral.marine { color: green }
```

이 규칙은 class="pastoral blue aqua marine"일 때 매칭하지만 class="pastoral blue"의 경우에는 매칭하지 않는다.

주: CSS는 "class" 속성에 상당한 권한을 부여하므로, 저작자는 (HTML DIV 및 SPAN과 같은) 관련 표현문이 거의 없는 요소를 기반으로 자체 “문서 언어”를 설계하고 "class" 속성을 통해 스타일 정보를 지정할 수 있을 것이다. 문서 언어의 구조적 요소는 인지 및 수용되는 의미를 가지는 경우가 많지만 저작자가 정의한 클래스는 그렇지 않으므로, 이런 관행은 피해야 한다.

주: 특정 요소에 여러 개의 클래스 속성이 있을 경우, 클래스를 검색하기 전에 값 사이 공간으로 값이 연결되어야 한다. 현재 워킹 그룹은 이 상황에 도달할 수 있는 메서드를 알지 못하므로, 본 규격에서 이런 거동은 분명히 비 규범적이다.

9.5. ID 셀렉터

문서 언어에는 type ID로 선언되는 속성이 들어있을 수 있다. type ID special 속성을 특별하게 만드는 것은 요소의 종류와 무관하게 문서에서 두 개의 해당 속성이 같은 값을 가질 수 없다는 데 있다; 문서 언어가 무엇이건 ID typed 속성을 활용하여 요소를 고유하게 식별할 수 있다. HTML에서는 모든 ID 속성이 "id"로 명명된다. XML 애플리케이션은 ID 속성을 다르게 명명할 수 있지만, 동일한 제약이 적용된다.

문서 언어의 ID typed 속성을 통해 저작자는 문서 트리의 특정 요소 인스턴스에 식별자를 지정할 수 있다. ID 셀렉터에는 숫자 기호”(U+0023, #)가 들어있고 바로 뒤에 반드시 CSS 식별자가 되어야 하는 ID 값이 붙는다. ID 셀렉터는 ID 셀렉터의 식별자와 일치하는 식별자가 있는 요소 인스턴스를 나타낸다.

셀렉터는 UA가 특정 요소의 ID typed 속성을 파악하는 메서드를 지정하지 않는다. UA는 예를 들어 문서의 DTD를 읽고, 정보를 하드코딩하거나, 사용자에게 묻는다.

예시:

아래의 ID 셀렉터는 ID typed 속성에 "chapter1" 값이 들어있는 h1 요소를 나타낸다:

```
h1#chapter1
```

아래의 ID 셀렉터는 ID typed 속성에 "chapter1" 값이 들어있는 모든 요소를 나타낸다:

```
#chapter1
```

아래의 셀렉터는 ID typed 속성에 "z98y" 값이 들어있는 모든 요소를 나타낸다.

```
*#z98y
```

주: XML 1.0 [XML10]에서, 특정 요소의 ID가 들어있는 속성에 관한 정보가 DTD나 스키마에 들어있다. XML를 분석할 때 UA는 항상 DTD를 읽지는 않으므로, 특정 요소의 ID가 무엇인지 알지 못한다(단, UA는 해당 네임스페이스의 ID 속성에 해당하는 속성을 파악할 수 있는 네임스페이스 지식을 가지고 있을 수 있다). UA가 특정 요소의 ID를 알지 못한다는 사실을 스타일 시트 저작자가 알고 있거나 의심할 경우, #p371대신 정상적인 어트리뷰트 셀렉터 [name=p371]를 사용해야 한다.

특정 요소에 여러 개의 ID 속성이 있을 경우, ID 셀렉터를 목적으로 그것들 전체를 해당 요소의 ID로 취급해야 한다. xml:id, DOM3 Core, XML DTD 및 네임스페이스 고유 지식의 믹스를 활용하여 그러한 상황에 도달할 수 있다.

9.6. 슈도 클래스

문서 트리를 벗어나거나 다른 단순 셀렉터를 활용하여 표현할 수 없는 정보를 토대로 한 선택이 가능하도록 슈도 클래스 개념이 도입되었다.

슈도 클래스는 항상 “콜론” (:) 뒤에 붙는 슈도 클래스의 이름과 (옵션으로) 괄호 사이 값으로 이루어진다.

슈도 클래스는 특정 셀렉터에 들어있는 모든 단순 셀렉터 시퀀스에서 허용된다. 슈도 클래스는 리딩 타입 셀렉터나 유니버셜 셀렉터(생략 가능) 다음 단순 셀렉터 시퀀스의 아무 곳에서나 허용된다. 상호 배타적인 슈도 클래스도 있고, 같은 요소에 동시적으로 적용되는 슈도 클래스도 있다. 사용자가 문서와 소통하는 동안 특정 요소가 슈도 클래스를 획득하거나 잃을 수 있다는 점에서 슈도 클래스는 동적이다.

9.6.1. 동적 슈도 클래스

동적 슈도 클래스는 이름, 속성, 내용 이외의 다른 기능 (원칙적으로 문서 트리에서 추론될 수 없는 기능) 관련 요소를 분류한다.

동적 슈도 클래스는 문서 소스나 문서 트리에 나타나지 않는다.

9.6.1.1. pseudo class: :link & :visited

사용자 에이전트는 공통적으로 이전의 방문지와 다른 미 방문 링크를 표시한다. 셀렉터는 pseudo class:link와 :visited를 제공하여 그것을 구별한다:

:link 슈도 클래스는 아직 방문하지 않은 링크에 적용된다.

:visited 슈도 클래스는 사용자가 링크를 방문한 후 적용된다.

일정 시간이 지나면, 사용자 에이전트는 방문된 링크를 (미 방문) ‘:link이 상태로 되돌릴 수 있다.

두 상태는 상호 배타적이다.

예시:

아래 셀렉터는 external클래스가 들어있고 이미 방문한 링크를 나타낸다:

```
a.external:visited
```

주: 스타일 시트 저작자는 :link 및 :visited 슈도 클래스를 남용하여 사용자의 동의 없이 사용자가 방문한 사이트를 파악할 수 있다.

따라서 UA는 모든 링크를 미 방문 링크로 취급하거나, 사용자 개인 정보를 보존하는 한편 방문 링크와 미 방문 링크를 달리 하기 위한 다른 조치를 구현할 수 있다.

9.6.1.2. 사용자 조치 pseudo class: hover, :active, :focus

쌍방향 사용자 에이전트는 때로 사용자의 조치에 대응하여 렌더링을 변경한다. 셀렉터는 사용자가 활동하는 특정 요소의 선택을 위해 세 가지 슈도 클래스를 제공한다.

:hover 슈도 클래스는 사용자가 포인팅 장치로 특정 요소를 지정하는 동안 적용되지만, 반드시 활성화되는 것은 아니다. 예를 들어, 시각적 사용자 에이전트는 커서(마우스 포인터)를 그 요소에 의해 생성된 박스 위에 가져갈 때 이 슈도 클래스를 적용할 수 있다. 쌍방향 매체를 지원하지 않는 사용자 에이전트는 이 슈도 클래스를 지원하지 않는다. 쌍방향 매체를 지원하더라도 일부 사용자 에이전트는 이 슈도 클래스를 지원하지 못할 수 있다(예: 하버링을 탐지하지 못하는 펜 장치).

:active 슈도 클래스는 사용자에 의해 특정 요소가 활성화될 때 적용된다. 사용자가 마우스 버튼을 눌렀다가 놓는 사이의 시간을 예로 들 수 있다. 마우스 버튼이 하나 이상인 시스템에서는 :active가 1차 또는 1차 활성화 버튼(통상 “왼쪽” 마우스 버튼) 및 관련 위 신호에만 적용된다.

:focus 슈도 클래스는 특정 요소에 포커스가 있는 동안 적용된다(키보드나 마우스 이벤트, 또는 기타 입력 형태 수용).

요소가 :active 되거나 :focus를 획득할 수 있는 문서 언어 또는 구현 고유 한계가 있을 수 있다.

이러한 슈도 클래스는 상호 배타적이지 않다. 하나의 요소가 한 번에 여러 개의 슈도 클래스를 매칭할 수 있다.

셀렉터는 특정 요소의 부모가 ‘:active 인지 여부나 ‘:hover’ 또한 그 상태인지 여부를 정의하지 않는다.

주: 포인팅 장치에 의해 자식이 지정되므로 ‘:hover’ 상태가 특정 요소에 적용될 경우, ‘:hover’는 포인팅 장치 아래 있지 않은 요소에도 적용할 수 있다.

예시:

```
a:link /* unvisited links */
a:visited /* visited links */
a:hover /* user hovers */
a:active /* active links */
```

동적 슈도 클래스 접목 예시:

```
a:focus
a:focus:hover
```

마지막 셀렉터는 슈도 클래스 :focus와 슈도 클래스 :hover에 있는 a 요소를 매칭한다.

주: 요소는 ‘:visited’과 ‘:active’ (또는 ‘:link’과 ‘:active’) 모두가 될 수 있다.

9.6.2. 타깃 슈도 클래스 :target

일부 URI는 리소스 내의 특정 위치를 참조한다. 이런 종류의 URI는 뒤에 (단편 식별자라고 하는) 앵커 식별자가 붙는 “숫자 기호” (#)로 끝난다.

단편 식별자가 있는 URI는 타깃 요소라고 하는 문서 내의 특정 요소로 연결된다. 특정 HTML 문서에서 section_2라는 이름의 앵커를 가리키는 URI를 예로 들어보자:

```
http://example.com/html/top.html#section_2
```

타깃 요소는 :target 슈도 클래스로 표현할 수 있다. 문서 URI에 단편 식별자가 없으면, 그 문서에는 타깃 요소가 없는 것이다.

예시:

```
p.note:target
```

이 셀렉터는 조회하는 URI의 타깃 요소인 note 클래스의 p 요소를 나타낸다.

CSS 예시:

여기서는 :target 슈도 클래스를 활용하여 타깃 요소를 적색으로 만들고 이미지가 있을 경우 그 앞에 이미지를 놓는다:

```
*:target { color : red }
*:target::before { content : url(target.png) }
```

9.6.3. 언어 슈도 클래스 :lang

문서 언어가 특정 요소의 인간 언어 결정 메서드를 지정할 경우, 그 언어를 토대로 특정 요소를 나타내는 셀렉터를 작성할 수 있다. 예를 들어 HTML [HTML401]에서, 언어는 lang 속성과 meta요소 또는 (HTTP 헤더와 같은) 프로토콜 정보의 조합에 의해 결정된다. XML은 xml:lang이라고 하는 속성을 사용하며, 언어를 결정하기 위한 다른 문서 언어 고유의 메서드가 있을 수 있다.

슈도 클래스 :lang(C)은 C 언어로 되어있는 요소를 나타낸다. a :lang() 셀렉터로 표현되는 특정 요소가 식별자 C와 동일한 그 요소의 언어 값(필요 시 BCP 47 구문으로 평준화)만을 기반으로 하거나, 또는 바로 뒤에 "-" (U+002D)가 붙는 식별자 C로 시작된다. C와 요소의 언어 값 매칭은 케이스에 둔감하게 구현된다. 식별자 C는 반드시 유효한 언어 이름일 필요가 없다.

C는 반드시 유효한 CSS 식별자 [CSS21]여야 하고 비어있으면 안 된다. (그렇지 않으면 셀렉터는 무효화된다.)

주: BCP 47 [BCP47]이나 후속 코드를 활용하여 문서 및 프로토콜에서 언어를 표시하고, XML 기반 문서[XML10]의 경우에는 "xml:lang" 속성을 활용할 것을 권고한다. "FAQ: 2자 또는 3자 언어 코드 " 참조.

예시:

아래의 두 셀렉터는 벨기에 프랑스어 또는 독일어로 된 HTML 문서를 보여주고 있다. 다음의 두 셀렉터는 임시 요소의 q 인용문을 벨기에 프랑스어 또는 독일어로 표현한다.

```
html:lang(fr-be)
html:lang(de)
:lang(fr-be) > q
:lang(de) > q
```

:lang(C)과 '|=' 연산자의 차이는 '|=' 연산자가 그 요소와 관련하여 주어진 속성에 대한 비교만을 구현하는 반면, :lang(C) 슈도 클래스는 문서의 시멘틱에 대한 UA의 지식을 활용하여 비교를 구현한다는 것이다.

HTML 예시에서는(LANG 속성이 있기 때문에) BODY만이 [lang|=fr]를 매칭하지만, (둘 모두 프랑스어이므로) BODY와 P 모두 [lang|=fr]를 매칭한다. P에는 LANG 속성이 없기 때문에 [lang|=fr]를 매칭하지 않는다.

```
<body lang=fr>
 <p>Je suis français.</p>
</body>
```

9.6.4. UI 요소 설명 슈도 클래스

9.6.4.1. :enabled 및 :disabled 슈도 클래스

:enabled 슈도 클래스는 활성화된 상태의 사용자 인터페이스 요소를 나타낸다; 요소에는 그에 해당하는 활성화 해제 상태가 있다.

반대로 :disabled 슈도 클래스는 해제된 상태의 사용자 인터페이스 요소를 나타낸다; 요소에는 그에 해당하는 활성화 상태가 있다.

황성화 상태, 해제 상태 및 사용자 인터페이스 요소를 구성하는 것은 언어에 좌우된다. 통상적인 문서에서는 대부분의 요소들이 :enabled고 아니고 :disabled도 아니다.

주: 사용자의 인터페이스 요소 소통 능력에 영향을 미칠 수 있는 CSS 속성은 :enabled 또는:disabled; 매칭 여부에 영향을 주지 않는다. 예를 들어, display 및 visibility 속성은 특정 요소의 활성화/해제 상태에 영향을 미치지 않는다.

9.6.4.2. :checked 슈도 클래스

사용자는 라디오 및 체크박스 요소를 토글할 수 있다. 사용자가 선택하면 일부 메뉴 항목이 “체크 표시”된다. 해당 요소를 토글 "on"하면 :checked 슈도 클래스가 적용된다. :checked 슈도 클래스는 문서의 시멘틱 속성 존재를 기반으로 활 수도 있으므로 성격상 동적이고 사용자에 의해 변경될 수 있는 반면, 모든 매체에 적용된다. 예를 들어, :checked 슈도 클래스는 처음 HTML4의 항목 17.2.1에 설명된 HTML4 selected 및 checked 속성이 있는 요소에 적용되고, 사용자는 해당 요소를 토글 "off" 할 수 있으며, 이 경우 :checked 슈도 클래스는 더 이상 적용되지 않는다.

9.6.4.3. :indeterminate 슈도 클래스

주: 라디오 및 체크박스 요소는 사용자가 토글할 수 있지만, 가끔은 체크 상태도 아니고 미 체크 상태도 아닌 애매한 상태가 되기도 한다. 이는 특정 요소 속성이나 DOM 조작으로 인한 것일 수 있다.
향후 규격 버전에서는 이러한 요소에 적용되는 :indeterminate 슈도 클래스를 도입할 수도 있다.

9.6.5. 구조적 슈도 클래스

셀렉터는 문서 트리에 존재하지만 다른 단순 셀렉터나 콤비네이터로 표현할 수 없는 추가 정보를 토대로 한 선택을 허용하기 위해 구조적 슈도 클래스 개념을 도입한다.

부모의 자식ren 목록에서 특정 요소의 포지션을 계산할 때 독립 텍스트를 비롯한 기타 비 요소 노드는 감안하지 않는다. 부모의 자식ren 목록에서 특정 요소의 포지션을 계산할 때 색인 번호는 1부터 시작한다.

9.6.5.1. :root 슈도 클래스

:root 슈도 클래스는 문서의 뿌리에 해당하는 요소를 표시한다. HTML 4에서 이것은 항상 HTML 요소이다.

9.6.5.2. :nth-child() 슈도 클래스

:nth- child (an+b) 슈도 클래스 표기는 문서 트리에서 양수 정수 또는 n의 제로 값에 대하여 앞에 an+b-1 형제가 있고 부모 요소를 갖춘 요소를 나타낸다. 0보다 큰 a와 b 값의 경우, 요소의 자식ren을 a 요소의 그룹으로 효과적으로 나누는 한편, 각 그룹의 b 번째 요소를 선택한다. 예를 들어, 이를 통해 셀렉터는 특정 테이블의 다른 모든 줄을 어드레싱할 수 있고, 4주기로 문장 텍스트 색상을 교차시킬 수 있다. A와 b 값은 반드시 정수가 되어야 한다(양수, 음수, 또는 제로). 첫 번째 요소 자식의 인덱스는 1이다.

이와 더불어, :nth- child ()는 인자로 'odd'와 'even'을 취할 수도 있다. 'odd'는 2n+1과 동일한 의미를 가지며, 'even'은 2n과 같은 의미를 가진다.

:nth- child ()의 인자는 아래의 문법을 매칭시켜야 한다. 여기서 INTEGER는 토큰 [0-9]+를 매칭시키고 나머지 토큰화는 항목 10.2의 Lexical scanner에서 제시된다:

```
nth
  : S* [ ['-'|'+']? INTEGER? {N} [ S* ['-'|'+'] S* INTEGER ]? |
  ['-'|'+']? INTEGER | {O}{D}{D} | {E}{V}{E}{N} ] S*
  ;
```

예시:

```
tr:nth- child (2n+ 1) /* represents every odd row of an HTML table */
tr:nth- child (odd) /* same */
tr:nth- child (2n+ 0) /* represents every even row of an HTML table */
tr:nth- child (even) /* same */

/* Alternate paragraph colours in CSS */
```

```
p:nth- child (4n+ 1) { color: navy; }
p:nth- child (4n+ 2) { color: green; }
p:nth- child (4n+ 3) { color: maroon; }
p:nth- child (4n+ 4) { color: purple; }
```

값 b 앞에 음수 표식이 붙을 경우, 그 표현문에서 "+" 글자를 반드시 없애야 한다(음수 b 값을 나타내는 "-" 글자로 효과적으로 대체된다).

예시:

```
:nth- child (10n-1) /* represents the 9th, 19th, 29th, etc, element */
:nth- child (10n+ 9) /* Same */
:nth- child (10n+ -1) /* Syntactically invalid, and would be ignored */
```

a=0의 경우, (b 부분이 이미 생략되지 않은 한) an 부분을 포함시킬 필요가 없다. An이 포함되지 않고 b가 음수가 아닐 경우, b 앞의 + 기호를 생략해도 무방하다. 이 경우 구문은:nth- child (b)로 간략해진다.

예시:

```
foo:nth- child (0n+ 5) /* represents an element foo that is the 5th child
 of its parents element */
foo:nth- child (5) /* same */
```

a=1 또는 a=-1의 경우, 규칙에서 숫자를 생략할 수 있다.

예시:

따라서 아래의 셀렉터들은 등가이다:

```
bar:nth- child (1n+ 0) /* represents all bar elements, specificity (0,1,1) */
bar:nth- child (n+ 0) /* same */
bar:nth- child (n) /* same */
bar /* same but lower specificity (0,0,1) */
```

b=0일 경우, 모든 a번째 요소가 채택된다. 이 경우 a 파트가 이미 생략되지 않은 한 +b (또는 -b) 파트는 생략될 수 있다.

예시:

tr:nth- child (2n+ 0) /* represents every even row of an HTML table */

tr:nth- child (2n) /* same */

앞뒤 및 "+" 또는 "-" 양쪽에 an과 b 파트를 구분해주는 공백이 허용된다.

공백이 있는 유효한 예시:

:nth- child (3n + 1)

:nth- child (+3n - 2)

:nth- child (-n+ 6)

:nth- child (+6)

공백이 있는 무효한 예시:

:nth- child (3 n)

:nth- child (+ 2n)

:nth- child (+ 2)

a와 b가 똑같이 0일 경우, 슈도 클래스 문서 트리의 어떠한 요소도 나타내지 않는다.

a 값은 음수가 될 수 있지만, n≥0일 경우 양수 값 an+b 만이 문서 트리의 특정 요소를 나타낼 수 있다.

예시:

html|tr:nth- child (-n+ 6) /* represents the 6 first rows of XHTML tables */

9.6.5.3. :nth-last- child () 슈도 클래스

:nth-last- child (an+b) 슈도 클래스 표기는 문서 트리에서 양수 정수 또는 n의 제로 값에 대하여 뒤에 an+b-1 형제가 있고 부모 요소를 갖춘 요소를 나타낸다. 구문은 :nth- child () 슈도 클래스 참조. 인자로 'even' 및 'odd' 값도 수용한다.

예시:

tr:nth-last- child (-n+ 2) /* represents the two last rows of an HTML table */

```
foo:nth-last- child (odd) /* represents all odd foo elements in their parents element,
    counting from the last one */
```

9.6.5.4. :nth-of-type() 슈도 클래스

:nth-of-type(an+b) 슈도 클래스 표기는 문서 트리에서 양수 정수 또는 n의 제로 값에 대하여 앞에 동일한 확장 요소 명의 an+b-1 형제가 있고 부모 요소를 갖춘 요소를 나타낸다. 구문은 :nth- child () 슈도 클래스 참조. 인자로 'even' 및 'odd' 값도 수용한다.

CSS 예시:

이를 통해 저작자는 부유하는 이미지의 포지션을 바꿀 수 있다:

```
img:nth-of-type(2n+ 1) { float: right; }
  img:nth-of-type(2n) { float: left; }
```

9.6.5.5. :nth-last-of-type() 슈도 클래스

:nth-of-type(an+b) 슈도 클래스 표기는 문서 트리에서 양수 정수 또는 n의 제로 값에 대하여 뒤에 동일한 확장 요소 명의 an+b-1 형제가 있고 부모 요소를 갖춘 요소를 나타낸다. 구문은 :nth- child () 슈도 클래스 참조. 인자로 'even' 및 'odd' 값도 수용한다.

예시:

첫째와 마지막을 제외한 XHTML body의 h2 자식을 모두 나타내기 위해, 다음과 같은 셀렉터를 사용할 수 있다:

```
body > h2:nth-of-type(n+ 2):nth-last-of-type(n+ 2)
```

이 경우, 셀렉터가 길게 끝날 수도 있지만 :not()를 사용하면 된다:

```
body > h2:not(:first-of-type):not(:last-of-type)
```

9.6.5.6. :first- child 슈도 클래스

:nth- child (1)와 같다. :first-자식 슈도 클래스는 다른 일부 요소의 첫 번째 자식인 특

정 요소를 나타낸다.

예시:

아래 셀렉터는 div 요소의 첫 번째 자식인 p 요소를 나타낸다:

```
div > p:first- child
```

이 셀렉터는 아래 단편의 div 내부 p를 나타낼 수 있다:

```
<p> The last P before the note.</p>
<div class="note">
 <p> The first P inside the note.</p>
</div>
```

하지만 아래 단편의 두 번째 p는 표현문할 수 없다:

```
<p> The last P before the note.</p>
<div class="note">
 <h2> Note </h2>
 <p> The first P inside the note.</p>
</div>
```

아래 두 개의 셀렉터는 일반적으로 대등하다:

```
* > a:first- child /* a is first child of any element */
  a:first- child /* Same (assuming a is not the root element) */
```

9.6.5.7. :last- child 슈도 클래스

:nth-last- child (1)와 같다. :last- child 슈도 클래스는 다른 일부 요소의 마지막 자식인 요소를 나타낸다.

예시:

아래 셀렉터는 순서 목록 ol의 마지막 자식인 목록 항목 li을 나타낸다.

```
ol > li:last- child
```

9.6.5.8. :first-of-type 슈도 클래스

:nth-of-type(1)와 같다. :first-of-type 슈도 클래스는 부모 요소의 자식 목록 중 해당 유형의 첫 번째 형제인 요소를 나타낸다.

예시:

아래 셀렉터는 정의 목록 dl 내의 정의 제목 dt를 나타내며, 이 dt는 부모 요소의 자식 목록에서 첫 번째가 된다.

```
dl dt:first-of-type
```

아래 예시에서 첫 두 개의 dt 요소에 대한 설명은 유효하지만, 세 번째는 유효하지 않다:

```
<dl>
 <dt>gigogne</dt>
 <dd>
 <dl>
 <dt>fusée</dt>
 <dd>multistage rocket</dd>
 <dt>table</dt>
 <dd>nest of tables</dd>
 </dl>
 </dd>
  </dl>
```

9.6.5.9. :last-of-type 슈도 클래스

:nth-last-of-type(1)와 같다. :last-of-type 슈도 클래스는 부모 요소의 자식ren 목록 중 해당 유형의 마지막 형제인 요소를 나타낸다.

예시:

아래 셀렉터는 table row.tr 중 마지막 데이터 셀 td를 나타낸다.

tr > td:last-of-type

9.6.5.10. :only-child 슈도 클래스

부모 요소가 있고 부모 요소에는 다른 요소 자식이 없는 요소를 나타낸다. :first-자식:last-자식 또는 :nth-자식(1):nth-last-child(1)와 같지만, 특이성은 낮다.

9.6.5.11. :only-of-type 슈도 클래스

부모 요소가 있고 부모 요소에는 동일한 확장 요소 명의 다른 요소 children이 없는 요소를 나타낸다. :first-of-type:last-of-type 또는 :nth-of-type(1):nth-last-of-type(1)와 같지만, 특이성은 낮다.

9.6.5.12. :empty 슈도 클래스

:empty 슈도 클래스는 children이 전혀 없는 요소를 나타낸다. 문서 트리 측면에서, (DOM [DOM-LEVEL-3-CORE] 텍스트 노드, CDATA 노드 및 개체 참조와 같이) 데이터 길이가 0이 아닌 요소 노드와 내용 노드만이 영향을 미치는 공란으로 고려되어야 한다; 코멘트, 처리 지시 및 기타 노드는 특정 요소가 공란으로 간주되는지 여부에 영향을 미치지 않아야 한다.

예시:

p:empty는 아래와 같은 단편의 유효한 표현문이다:

```
<p></p>
```

foo:empty 는 아래와 같은 단편의 유효한 표현문이 아니다:

```
<foo>bar</foo>
<foo><bar>bla</bar></foo>
  <foo>this is not <bar>:empty</bar></foo>
```

9.6.6. 공백

이 항목은 의도적으로 공백으로 남겨둔다. (이전에는 :contains() 슈도 클래스를 정의했다.)

9.6.7. 부인 슈도 클래스

부인 슈도 클래스 :not(X)는 단순 셀렉터(부인 슈도 클래스 자체 제외)를 인자로 채택하는 기능적 표기이다. 인자로 표현되지 않는 특정 요소를 나타낸다.

부인은 군락을 이루지 않는다; :not(:not(...))는 무효하다. 또한 의사 요소는 단순 셀렉터가 아니므로, :not()에 대하여 유효한 인자가 아니라는 점에 유의해야 한다.

예시:

아래 셀렉터는 해제되지 않은 HTML 문서의 모든 button 요소를 매칭한다.

```
button:not([DISABLED])
```

아래 셀렉터는 FOO 요소만을 제외한 모두를 나타낸다.

```
*:not(FOO)
```

아래 셀렉터 그룹은 링크를 제외한 모든 HTML 요소를 나타낸다.

```
html|*:not(:link):not(:visited)
```

인자가 유니버셜 셀렉터 또는 타입 셀렉터가 아닌 이상, 기본 네임스페이스 선언은 부인 슈도 클래스의 인자에 영향을 미치지 않는다.

예시:

기본 네임스페이스가 http://example.com/와 연계되는 것으로 가정하면, 아래의 셀렉터는 그 네임스페이스에 속하지 않은 모든 요소를 나타낸다:

```
*|*:not(*)
```

아래 셀렉터는 네임스페이스와 무관하게 배회하지 않는 모든 요소를 매칭한다. 특히 배회하지 않는 기본 네임스페이스의 매칭 요소만으로 국한되지 않으며, 기본 네임스페이스에 속하지 않는 요소는 배회할 경우에도 규칙을 매칭하지 않는다.

```
*|*:not(:hover)
```

주: :not() 의사를 이용하여 쓸모 없는 셀렉터를 작성할 수 있다. 어떤 요소도 나타내지 않는 :not(*|*), 또는 foo에 해당하지만 특이성이 더 높은 foo:not(bar)를 예로 들 수 있다.

10. 의사 요소(pseudo-element)

의사 요소는 문서 언어에 의해 지정되지 않은 문서 트리에 관한 추상적 개념을 생성한다. 예를 들어, 문서 언어는 특정 요소 내용의 첫 번째 글자 또는 첫 번째 문자열 접근을 위한 메커니즘을 제공하지 않는다. 의사 요소를 통해 저작자는 달리 접근이 불가능한 이 정보를 참조할 수 있다. 또한 의사 요소는 소스 문서에 존재하지 않는 내용을 참조할 수단을 제공하기도 한다(예를 들어, ::before 및 ::after 의사 요소는 생성된 내용 접근 권한을 부여한다).

의사 요소는 뒤에 의사 요소 이름이 붙는 두 개의 콜론(::)으로 이루어진다.

:: 표기는 슈도 클래스와 의사 요소를 확실히 구분하기 위해 현재 문서에 의해 도입되었다. 기존 스타일 시트와의 호환성을 위해, 사용자는 반드시 CSS levels 1과 2에서 도입된 이전의 1개 콜론 의사 요소 표기를 수용해야 한다(즉, :first-line, :first-letter, :before 및 :after). 본 규격서에서 도입된 새로운 의사 요소에는 이 호환성이 허용되지 않는다.

셀렉터 당 한 개의 의사 요소만이 나타날 수 있으며, 반드시 셀렉터의 대상을 나타내는 단순 셀렉터 시퀀스 다음에 나타나야 한다.

주: 본 규격서의 향후 버전에서는 셀렉터 당 복수의 의사 요소를 허용할 것이다.

10.1. ::first-line 의사 요소

::first-line 의사 요소는 처음으로 포맷된 특정 요소 문자열의 내용을 설명한다.

CSS 예시:

```
p::first-line { text-transform: uppercase }
```

위의 규칙은 “모든 p 요소의 첫 번째 문자열 글자를 대문자로 변경함”을 의미한다.

셀렉터 p::first-line는 실제 문서 요소를 매칭하지 않는다. 모든 p 요소의 도입부에 삽입되는 사용자 에이전트와 부합하는 의사 요소를 매칭한다.

첫 번째 문자열의 길이는 페이지 폭, 폰트 사이즈 등을 포함한 여러 가지 요소에 좌우된다는 점을 유념해야 한다. 따라서 보통의 HTML 단락은 다음과 같다:

```
<P>This is a somewhat long HTML
paragraph that will be broken into several
lines. The first line will be identified
by a fictional tag sequence. The other lines
will be treated as ordinary lines in the
paragraph.</P>
```

문자열은 다음과 같이 분해된다:

```
THIS IS A SOMEWHAT LONG HTML PARAGRAPH THAT
will be broken into several lines. The first
line will be identified by a fictional tag
sequence. The other lines will be treated as
ordinary lines in the paragraph.
```

이 단락은 ::first-line에 대한 가공의 태그 시퀀스를 포함하도록 사용자 에이전트에 의해 재 작성될 수 있다. 이 가공의 태그 시퀀스는 속성이 계승되는 과정을 보여주는 데 도움이 된다.

```
<P><P::first-line> This is a somewhat long HTML
paragraph that </P::first-line> will be broken into several
lines. The first line will be identified
by a fictional tag sequence. The other lines
will be treated as ordinary lines in the
paragraph.</P>
```

특정 의사 요소가 실제 요소를 분해시킬 경우, 해당 요소를 닫았다가 다시 여는 가공의 태그 시퀀스로 원하는 효과를 설명할 수 있는 경우가 많다. 따라서 span 요소로 앞의 단락을 마크업하면 아래와 같다:

```
<P><SPAN class="test"> This is a somewhat long HTML
paragraph that will be broken into several
lines.</SPAN> The first line will be identified
by a fictional tag sequence. The other lines
will be treated as ordinary lines in the
paragraph.</P>
```

사용자 에이전트는 ::first-line에 대하여 가공의 태그 시퀀스를 삽입할 때 span에 대한 시작 및 종료 태그를 가장할 수 있다.

```
<P><P::first-line><SPAN class="test"> This is a
somewhat long HTML
paragraph that will </SPAN></P::first-line><SPAN class="test"> be
broken into several
lines.</SPAN> The first line will be identified
by a fictional tag sequence. The other lines
will be treated as ordinary lines in the
paragraph.</P>
```

10.1.1. CSS의 첫 번째 문자열 정의

CSS에서 ::first-line 의사 요소는 블록박스, 인라인 블록, 테이블 캡션, 또는 테이블 셀 등 블록과 유사한 컨테이너에 붙을 때에만 효과를 가질 수 있다.

특정 요소의 첫 번째 문자열은 같은 계열의 블록 차원 후손 (즉 floating이나 positioning으로 인해 흐름에서 벗어나지 않는 블록 차원 후손) 내에서 발생할 수 있다. 예를 들어, <DIV><P>This line...</P></DIV> 에서 DIV의 첫 번째 문자열은 P의 첫 번째 문자열이다(P와 DIV모두 블록 차원으로 가정).

테이블 셀이나 인라인 블록의 첫 번째 문자열은 조상 요소의 첫 번째 문자열이 될 수 없다. 따라서 <DIV><P STYLE="display: inline-block">Hello
Goodbye</P> etcetera </DIV> 에서 DIV의 첫 번째 문자열은 "Hello" 문자열이 아니다.

주: 이 단락에서 p의 첫 번째 문자열을 주목할 것: <p>
First...에는 어떠한 글자도 들어있지 않다(HTML 4의 br에 대한 기본 스타일 가정). 단어 "First"는 첫 번째 문자열에 없다.

UA는 ::first-line 의사 요소의 가공의 시작 태그가 가장 안쪽을 둘러싼 블록 차원 요소 바로 안쪽에 있는 듯이 거동한다. (이 경우 CSS1와 CSS2는 조용하므로, 저작자는 이 거동에 의존해서는 안 된다.) 예를 들어,

```
<DIV>
 <P>First paragraph</P>
 <P>Second paragraph</P>
</DIV>
```

에 대한 가공의 시작 태그는 다음과 같다:

```
<DIV>
 <P><DIV::first-line><P::first-line>First paragraph</P::first-line></DIV::first-line></P>
 <P><P::first-line>Second paragraph</P::first-line></P>
</DIV>
```

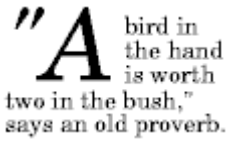
::first-line 의사 요소는 인라인 레벨 요소와 유사하지만, 일정한 제약이 따른다. 다음의 CSS 속성은 ::first-line의사 요소: 폰트 속성, 컬러 속성, 배경 속성, ‘word-spacing’, ‘letter-spacing’, ‘text-decoration’, ‘vertical-align’, ‘text-transform’, ‘line-height’. UA는 다른 속성도 적용할 수 있다.

CSS 계승 과정에서, 첫 번째 문자열에서 발생하는 자식 요소 부분은 ::first-line 의사 요소에 해당하는 속성만을 ::first-line 의사 요소로부터 계승한다. 그 외 다른 요소들은 모두 첫 번째 문자열 의사 요소 중 비 의사 요소 부모로부터 계승된다. (첫 번째 문자열에서 발생하지 않는 자식 요소 부문은 항상 그 자식의 부모로부터 계승된다.)

10.2. ::first-letter 의사 요소

::first-letter 의사 요소는 문자열에서 (이미지나 인라인 테이블과 같은) 다른 내용이 앞에 오지 않으면 특정 요소의 첫 번째 글자를 나타낸다. ::first-letter 의사 요소는 일반적인 인쇄 효과에 해당하는 "initial caps"와 "drop caps"에도 활용할 수 있다.

첫 번째 글자의 앞이나 뒤에 붙는 구두법(즉 "open" (Ps), "close" (Pe), "initial" (Pi), "final" (Pf) 및 "other" (Po) 구두법 클래스에서 Unicode로 정의되는 글자)이 포함되어야 한다. [UNICODE]



첫 번째 글자가 사실상 숫자일 경우에도 ::first-letter가 적용된다. 예를 들어, 6,700만 달러에서 6은 아주 많은 돈이다.

주: 일부 경우 ::first-letter 의사 요소가 한 문자열에서 첫 번째 비 구두 글자 이상을 포함해야 할 수도 있다. 예를 들어, 기본 문자와 함께 결합 문자도 유지되어야 한다. 더불어, 일부 언어에는 특정 글자 조합 취급 메서드에 관한 구체적인 규칙이 있을 수 있다. UA 정의 ::first-letter는 최소한 UAX에 의해 정의되는 기본 서기소(grapheme) 클러스터

를 포함해야 하고, 적절할 경우 그 이상을 포함할 수도 있다. 예를 들어 네덜란드에서 특정 요소의 도입부에 글자 조합 "ij"가 나타나면, 두 자 모두 ::first-letter 의사 요소에 속하는 것으로 간주해야 한다. [UAX29]

<p>'T...의 T와 같이 ::first-letter를 형성하는 글자가 같은 요소에 있지 않을 경우, UA는 그 중 한 요소나 두 요소 모두로부터 ::first-letter 의사 요소를 생성하거나 아예 의사 요소를 생성하지 않을 수 있다.

마찬가지로, (예를 들면 양방향 재정렬로 인해) 블록의 첫 글자가 문자열의 시작 지점에 있지 않을 경우, UA는 의사 요소를 생성할 필요가 없다.

예시:

아래의 CSS 및 HTML 예시는 중복 의사 요소가 소통할 수 있는 메서드를 보여준다. 각 P 요소의 첫 글자는 폰트 사이즈 '24pt'에 녹색이 된다. 첫 번째 문자열의 나머지는 'blue'가 되고, 나머지 단락은 'red'가 된다.

```
p { color: red; font-size: 12pt }
p::first-letter { color: green; font-size: 200% }
p::first-line { color: blue }

<P>Some text that ends up on two lines</P>
```

단어 "ends" 앞에서 줄 바꿈이 이루어진다고 가정하면, 이 단락에 대한 가상의 태그 시퀀스는 아래와 같아질 것이다:

```
<P>
<P::first-line>
<P::first-letter>
S
</P::first-letter>ome text that
</P::first-line>
ends up on two lines
</P>
```

::first-letter 요소가 ::first-line 안에 있다는 점을 유념해야 한다. ::first-line에서 설정된 속성은 ::first-letter에 의해 계승되지만, ::first-letter 에서 같은 속성이 설정되면 대체한다.

첫 글자는 반드시 첫 번째 문자열에서 발생해야 한다. 예를 들어, 이 HTML 단락: <p>
First...에서, 첫 번째 문자열에는 글자가 들어있지 않고 ::first-letter는 아무것도 매칭하지 않는다(HTML4로 br의 기본 스타일 가정). 특히 "First."의 “F”를 매칭하지 않는다.

10.2.1. CSS 적용

CSS에서 ::first-letter 의사 요소는 블록, 목록 항목, 테이블 셀, 테이블 캡션 및 인라인 블록 요소와 같이 블록과 유사한 컨테이너에 적용된다. **주:** 본 규격서 향후 버전에서는 이 의사 요소를 더 많은 디스플레이 타입에 적용하도록 허용할 것이다.

::first-letter 의사 요소는 텍스트가 들어있거나, 텍스트가 들어있는 동일한 계열의 후손이 있는 요소와 함께 사용될 수 있다. UA는 첫 번째 텍스트가 후손에 속하더라도 요소의 첫 텍스트 바로 앞에 ::first-letter 의사 요소의 가공 시작 태그가 있는 것처럼 거동해야 한다.

예시:

아래 HTML 단락의 가공의 태그 시퀀스

```
<div>
<p>The first text.
```

는 아래와 같다:

```
<div>
<p><div::first-letter><p::first-letter>T</...></...>he first text.
```

CSS에서 테이블 셀이나 인라인 블록의 첫 번째 글자는 조상 요소의 첫 번째 글자가 될 수 없다. 따라서 <DIV><P STYLE="display: inline-block">Hello
Goodbye</P>

etcetera</DIV>에서 DIV의 첫 글자는 글자 "H"가 아니다. 사실 DIV에는 첫 글자가 없다.

특정 요소가 목록 항목('display: list-item')일 경우, ::first-letter는 마커 뒤 principal box의 첫 글자에 적용된다. UA는 'list-style-position: inside' 목록 항목에서 ::first-letter를 무시한다. 특정 요소에 ::before 또는 ::after 내용이 들어있을 경우, ::first-letter는 그 내용을 포함한 요소의 첫 글자에 적용된다.

예시:

p::before {content: "Note: "} 규칙에 따라 셀렉터 p::first-letter는 "Note"의 "N"을 매칭한다.

CSS에서 'float' 속성이 'none'일 경우 ::first-line 의사 요소는 인라인 요소와 유사해진다. 그렇지 않을 경우, 부동 요소와 유사해진다. ::first-letter 의사 요소: 폰트 속성, 'text-decoration', 'text-transform', 'letter-spacing', 'word-spacing' (해당될 경우), 'line-height', 'float', 'vertical-align' ('float'가 'none'일 경우에 한함), 여백 속성, 패딩 속성, 보더 속성, 컬러 속성, 배경 속성 등에 적용된다. UA는 다른 속성도 적용할 수 있다. UA가 인쇄상으로 올바른 drop cap or initial cap을 나타낼 수 있도록, UA는 일반 요소와 달리 글자의 모양을 토대로 문자열 높이, 가로 세로를 선택할 수 있다.

예시:

이 CSS 및 HTML 요소는 initial cap의 가능한 렌더링을 보여준다. ::first-letter 의사 요소에 의해 계승되는 'line-height'는 1.1이지만, 이 예시에서 UA는 첫 두 줄 사이에서 불필요한 공백이 유발되지 않도록 첫 글자의 높이를 다르게 산출했다. 또한 첫 글자의 가공의 시작 태그가 span 내에 있으므로, 첫 글자의 무게가 span만큼 굵지 않고 정상적이다:

```
p { line-height: 1.1 }
p::first-letter { font-size: 3em; font-weight: normal }
span { font-weight: bold }
```

```
...
<p><span>Het hemelsche</span> gerecht heeft zich ten lange lesten<br>
Erbarremt over my en mijn benaeuwde vesten<br>
En arme burgery, en op mijn volcx gebed<br>
  En dagelix geschrey de bange stad ontzet.
```

Het hemelsche gerecht heeft zich ten lange lesten
Erbarremt over my en mijn benaeuwde vesten
En arme burgery, en op mijn volcx gebed
En dagelix geschrey de bange stad ontzet.

아래의 CSS는 두 문자열에 관한 drop cap 이니셜 글자를 만든다:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<HTML>
 <HEAD>
 <TITLE>Drop cap initial letter</TITLE>
 <STYLE type="text/css">
 P { font-size: 12pt; line-height: 1.2 }
 P::first-letter { font-size: 200%; font-weight: bold; float: left }
 SPAN { text-transform: uppercase }
 </STYLE>
 </HEAD>
 <BODY>
 <P><SPAN>The first</SPAN> few words of an article
 in The Economist.</P>
 </BODY>
</HTML>
```

이 예시는 아래와 같이 포맷될 수 있다:

THE FIRST few words of an article in the Economist

가공의 태그 시퀀스는 아래와 같다:

```
<P>
<SPAN>
<P::first-letter>
T
</P::first-letter>he first
</SPAN>
few words of an article in the Economist.
</P>
```

::first-letter 의사 요소는 내용에 관한 태그인 반면(즉 이니셜 글자), ::first-line 의사 요소 시작 태그는 블록 요소 시작 태그 바로 뒤에 삽입된다.

전형적인 drop cap 포맷을 달성하기 위해, 사용자 에이전트는 폰트 사이즈를 어림잡아 예를 들면 베이스라인을 정렬시킬 수 있다. 또한 포맷 시에 글리프(glyph) 아웃라인을 감안할 수도 있다.

10.3. Blank

이 항목은 의도적으로 공백으로 남겨둔다. (이전에는 ::selection 슈도 클래스를 정의했다.)

10.4. ::before 및 ::after 의사 요소

::before 및 ::after 의사 요소를 활용하여 특정 요소 내용의 앞이나 뒤에 생성되는 내용을 설명할 수 있다. 이것들은 CSS 2.1 [CSS21]에 설명되어 있다.

::before 또는 ::after를 사용하여 생성된 내용이 들어있는 요소에 ::first-letter 및 ::first-line 의사 요소가 적용될 경우, 생성된 내용을 포함한 요소의 첫 글자나 첫 문자열에 적용한다.

11. 콤비네이터

11.1. 하위 콤비네이터

저작자들은 셀렉터를 통해 문서 트리에서 다른 요소의 후손인 특정 요소를 설명하려 할 것이다(예를 들면 H1 요소 내에 들어있는 EM 요소). 후손 콤비네이터는 그러한 관계를 표현한다. 후손 콤비네이터는 두 개의 단순 셀렉터 시퀀스를 구분해주는 공백이다. A selector of the form "A B" 형식의 셀렉터는 일부 조상 요소 A의 임의 후손인 B 요소를 나타낸다.

예시:

예를 들어, 아래의 셀렉터를 고려해보자:

```
h1 em
```

이것은 h1요소의 후손이 되는 em 요소를 나타낸다. 올바르고 유효하지만, 다음과 같은 문단의 부분 설명에 불과하다:

```
<h1>This <span class="myclass">headline
is <em>very</em> important</span></h1>
```

아래의 셀렉터:

```
div * p
```

는 div 요소의 손자 이하 후손인 p 요소를 나타낸다. "*" 양쪽 공백은 유니버셜 셀렉터의 일부라는 점에 유념할 것; 이 공백은 div가 일부 요소의 조상이 될 것이고, 그 요소는 p의 조상이 될 것임을 나타내는 일종의 콤비네이터이다.

후손 콤비네이터와 어트리뷰트 셀렉터를 결합한 아래의 셀렉터는 (1) href 속성 세트를

가지고 있고 (2) 그 자체가 안에 있는 p의 내부 요소를 나타낸다:

```
div p *[href]
```

11.2. 자식 콤비네이터

자식 콤비네이터는 두 요소들 간의 자식hood 관계를 나타낸다. 자식 콤비네이터는 "greater-than sign" (U+003E, T) 글자로 이루어지며 두 개의 단순 셀렉터 시퀀스를 구분해준다.

예시:

아래 셀렉터는 body의 자식인 p 요소를 나타낸다:

```
body > p
```

아래 예시는 후손 콤비네이터와 자식 콤비네이터를 결합시킨다.

```
div ol>li p
```

이것은 li 요소의 후손인 p 요소를 나타낸다; li 요소는 ol 요소의 자식이 될 것이다; ol 요소는 div 요소의 자식이 될 것이다. ">" 콤비네이터 주변의 공백 옵션은 방치되지 않았다는 데 유념할 것.

특정 요소의 첫 번째 자식 선택에 관한 정보는 상기 :first-child 슈도 클래스 항목을 참조할 것.

11.3. 형제 콤비네이터

인접 형제 콤비네이터와 일반 형제 콤비네이터 두 가지의 상이한 형제 콤비네이터가 있다. 두 경우 모두 요소의 인접성을 고려할 때 요소 외의 노드(예를 들면 요소 사이 텍스트)는 무시된다.

11.3.1. 인접 형제 콤비네이터

인접 형제 콤비네이터는 두 개의 단순 셀렉터 시퀀스를 구분하는 "플러스 기호" (U+002B, +) 글자로 이루어진다. 두 시퀀스로 표현되는 요소들은 문서 트리에서 같은 부모를 공유하며 첫 번째 시퀀스로 표현되는 요소는 두 번째 시퀀스로 표현되는 요소보다 바로 앞선다.

예시:

아래 셀렉터는 math 요소 바로 다음의 p 요소를 나타낸다:

```
math + p
```

아래 셀렉터는 어트리뷰트 셀렉터를 추가한다는 점을 제외하고는 개념적으로 앞서의 예시와 유사하다— class="opener"가 반드시 있어야 한다는 제약을 h1요소에 추가한다:

```
h1.opener + h2
```

11.3.2. 일반 형제 콤비네이터

일반 형제 콤비네이터는 두 개의 단순 셀렉터 시퀀스를 구분하는 "tilde" (U+007E, ~) 글자로 이루어진다. 두 시퀀스로 표현되는 요소들은 문서 트리에서 같은 부모를 공유하며 첫 번째 시퀀스로 표현되는 요소는 두 번째 시퀀스로 표현되는 요소보다 바로 앞선다 (단, 반드시 바로 앞서지는 않는다).

예시:

```
h1 ~ pre
```

는 h1다음의 pre 요소를 나타낸다. 올바르고 유효하지만, 다음의 부분 설명에 불과하다:

```
<h1>Definition of the function a</h1>
<p>Function a(x) has to be applied to all figures in the table.</p>
<pre>function a(x) = 12x/13.5</pre>
```

12. 셀렉터의 특이성 계산

셀렉터의 특이성은 다음과 같이 계산된다:

셀렉터의 ID 셀렉터 수를 센다 (= a)

셀렉터에서 클래스 셀렉터, 어트리뷰트 셀렉터 및 슈도 클래스의 수를 센다(= b).

셀렉터에서 타입 셀렉터와 의사 요소의 수를 센다(= c).

유니버셜 셀렉터는 무시.

여타와 마찬가지로 부인 슈도 클래스 내의 셀렉터를 세되, 부인 자체는 슈도 클래스로 간주하지 않는다.

(방대한 베이스의 숫자 체계에서) 세 개 숫자 a-b-c 를 연결하면 특이성이 주어진다.

예시:

```
* /* a=0 b=0 c=0 -> specificity = 0 */
LI /* a=0 b=0 c=1 -> specificity = 1 */
UL LI /* a=0 b=0 c=2 -> specificity = 2 */
UL OL+ LI /* a=0 b=0 c=3 -> specificity = 3 */
H1 + *[REL=up] /* a=0 b=1 c=1 -> specificity = 11 */
UL OL LI.red /* a=0 b=1 c=3 -> specificity = 13 */
LI.red.level /* a=0 b=2 c=1 -> specificity = 21 */
#x34y /* a=1 b=0 c=0 -> specificity = 100 */
#s12:not(FOO) /* a=1 b=0 c=1 -> specificity = 101 */
```

주: 단순 셀렉터의 반복적인 발생이 허용되며 특이성을 가중시킨다.

주: HTML style 속성에 지정된 스타일의 특이성은 CSS 2.1에 설명되어 있다. [CSS21]

13. 셀렉터 문법

13.1. 문법

아래 문법은 셀렉터의 구문을 정의한다. 일괄적으로 LL(1)이고 국지적으로 LL(2)이 될

수 있다(단, 구문 분석 규약을 표현문하지 않으므로 대부분의 UA들은 직접 사용해서는 안 된다는 점에 유념할 것). 프로덕션 포맷은 인간의 소비에 최적화되어 있으며 Yacc ([YACC] 참조) 외의 일부 단축 표기법도 사용된다:

*: 0 이상

+: 1 이상

?: 0 또는 1

|: 대안 구분

[ ]: 그룹핑

프로덕션은 다음과 같다:

```
selectors_group
 : selector [ COMMA S* selector ]*
 ;

selector
 : simple_selector_sequence [ combinator simple_selector_sequence ]*
 ;

combinator
 /* combinators can be surrounded by whitespace */
 : PLUS S* | GREATER S* | TILDE S* | S+
 ;

simple_selector_sequence
 : [ type_selector | universal ]
 [ HASH | class | attrib | pseudo | negation ]*
 | [ HASH | class | attrib | pseudo | negation ]+
 ;

type_selector
 : [ namespace_ prefix]? element_name
 ;
```

```
namespace_ prefix
 : [ IDENT | '*' ]? '|'
 ;

element_name
 : IDENT
 ;

universal
 : [ namespace_ prefix]? '*'
 ;

class
 : '.' IDENT
 ;

attrib
 : '[' S* [ namespace_ prefix]? IDENT S*
 [ [PREFIX MATCH |
 SUFFIX MATCH |
 SUBSTRINGMATCH |
 '=' |
 INCLUDES |
 DASHMATCH ] S* [ IDENT | STRING ] S*
 ]? ']'
 ;

pseudo
 /* '::' starts a pseudo element, ':' a pseudo class */
 /* Exceptions: :first-line, :first-letter, :before and :after. */
 /* Note that pseudo element are restricted to one per selector and */
 /* occur only in the last simple_selector_sequence. */
 : ':' ':'? [ IDENT | functional_pseudo ]
 ;

functional_pseudo
```

```
 : FUNCTION S* expression ')'
 ;

expression
 /* In CSS3, the expressions are identifiers, strings, */
 /* or of the form "an+b" */
 : [ [ PLUS | '-' | DIMENSION | NUMBER | STRING | IDENT ] S* ]+
 ;

negation
 : NOT S* negation_arg S* ')'
 ;

negation_arg
 : type_selector | universal | HASH | class | attrib | pseudo
 ;
```

13.2. 어휘 스캐너

다음은 Flex 표기법([FLEX] 참조)으로 작성된 tokenizer이다. Tokenizer는 케이스에 둔감하다.

"₩377"의 2회 발생은 현재 Flex 버전이 처리할 수 있는 최고의 글자 수를 나타낸다(십진수 255). Unicode/ISO-10646에서 가능한 최대 코드 포인트인 "₩4177777" (십진수 1114111)로 읽어야 한다. [UNICODE]

```
%option case-insensitive

ident [-]?{nmstart}{nmchar}*
name {nmchar}+
nmstart [_a-z]|{nonascii}|{escape}
nonascii [^₩0-₩177]
unicode ₩₩[0-9a-f]{1,6}(₩r₩n|[ ₩n₩r₩t₩f])?
escape {unicode}|₩₩[^₩n₩r₩f0-9a-f]
nmchar [_a-z0-9-]|{nonascii}|{escape}
```

```
num [0-9]+|[0-9]*₩.[0-9]+
string {string1}|{string2}
string1 ₩"([^₩n₩r₩f₩₩"]|₩₩{nl}|{nonascii}|{escape})*₩"
string2 ₩'([^₩n₩r₩f₩₩']|₩₩{nl}|{nonascii}|{escape})*₩'
invalid {invalid1}|{invalid2}
invalid1 ₩"([^₩n₩r₩f₩₩"]|₩₩{nl}|{nonascii}|{escape})*
invalid2 ₩'([^₩n₩r₩f₩₩']|₩₩{nl}|{nonascii}|{escape})*
nl ₩n|₩r₩n|₩r|₩f
w [ ₩t₩r₩n₩f]*

D d|₩₩0{0,4}(44|64)(₩r₩n|[ ₩t₩r₩n₩f])?
E e|₩₩0{0,4}(45|65)(₩r₩n|[ ₩t₩r₩n₩f])?
N n|₩₩0{0,4}(4e|6e)(₩r₩n|[ ₩t₩r₩n₩f])?|₩₩n
O o|₩₩0{0,4}(4f|6f)(₩r₩n|[ ₩t₩r₩n₩f])?|₩₩o
T t|₩₩0{0,4}(54|74)(₩r₩n|[ ₩t₩r₩n₩f])?|₩₩t
V v|₩₩0{0,4}(58|78)(₩r₩n|[ ₩t₩r₩n₩f])?|₩₩v

%%

[ ₩t₩r₩n₩f]+  return S;

"~=" return INCLUDES;
"|=" return DASHMATCH;
"^=" return PREFIXMATCH;
"$=" return SUFFIXMATCH;
"*=" return SUBSTRINGMATCH;
{ident} return IDENT;
{string} return STRING;
{ident}"(" return FUNCTION;
{num} return NUMBER;
"#"{name} return HASH;
{w}"+" return PLUS;
{w}">" return GREATER;
{w}"," return COMMA;
{w}"~" return TILDE;
":"{N}{O}{T}"(" return NOT;
```

```
@{ident} return ATKEYWORD;
{invalid} return INVALID;
{num}% return PERCENTAGE;
{num}{ident} return DIMENSION;
"<!--" return CDO;
"-->" return CDC;

₩/₩*[^*]*₩*+([^₩*][^*]*₩*+)*₩/ /* ignore comments */

. return *yytext;
```

14. 프로파일

셀렉터를 사용하는 각 규격서는 그것이 허용하고 제외시키는 셀렉터의 서브세트를 반드시 정의하고, 그 서브세트의 모든 구성요소에 대하여 로컬 의미를 설명해야 한다.

비 규범적 예시:

셀렉터 프로파일	
규격	CSS level 1
수락	타입 셀렉터 클래스 셀렉터 ID 셀렉터 :link, :visited 및 :active 슈도 클래스 후손 콤비네이터 ::first-line 및 ::first-letter 의사 요소
제외	유니버셜 셀렉터 어트리뷰트 셀렉터 :hover & :focus 슈도 클래스 :target 슈도 클래스

	:lang() 슈도 클래스 모든 UI 요소 상태 슈도 클래스 모든 구조적 슈도 클래스 부인 슈도 클래스 ::before 및 ::after 의사 요소 자식 콤비네이터 형제 콤비네이터 네임스페이스
추가 제약	단순 셀렉터 시퀀스 당 단 한 개의 클래스 셀렉터만 허용
셀렉터 프로파일	
규격	CSS level 2
Accepts	타입 셀렉터 유니버셜 셀렉터 속성 존재 및 값 셀렉터 클래스 셀렉터 ID 셀렉터 :link, :visited, :active, :hover, :focus, :lang() 및 :first- child 슈도 클래스 후손 콤비네이터 자식 콤비네이터 인접 형제 콤비네이터 ::first-line & ::first-letter 의사 요소 ::before 및 ::after 의사 요소
제외	부속 문자열 매칭 어트리뷰트 셀렉터 :target 슈도 클래스 모든 UI 요소 상태 슈도 클래스 :first-child 외의 모든 구조적 슈도 클래스

	부인 슈도 클래스 일반 형제 콤비네이터 네임스페이스
추가 제약	단순 셀렉터 시퀀스 당 하나 이상의 클래스 셀렉터 (CSS1 제약) 허용

CSS에서, 셀렉터는 문서 트리의 요소에 적용되는 스타일 규칙을 결정하는 패턴 매칭 규칙을 표현한다.

아래의 셀렉터(CSS level 2)는 모든 앵커들을 항목 1 헤더 h1: 안의 속성 name 세트와 매칭한다:

```
h1 a[name]
```

모든 해당 셀렉터에 첨부되는 모든CSS 선언은 그것을 매칭하는 모든 요소들에 적용된다.

셀렉터 프로파일	
규격	STTS 3
수락	타입 셀렉터 유니버셜 셀렉터 어트리뷰트 셀렉터 클래스 셀렉터 ID 셀렉터 모든 구조적 슈도 클래스 모든 콤비네이터 네임스페이스
제외	미 수락 슈도 클래스 의사 요소
추가 제약	STTS 선언 오른쪽의 단락 설명에서 일부 셀렉터와 콤비네이터는 허용되지 않음.

셀렉터는 STTS 3에서 두 가지 상이한 메서드로 활용할 수 있다:

CSS 선택 메커니즘에 상응하는 선택 메커니즘: 주어진 셀렉터에 첨부되는 선언이 그 셀렉터를 매칭하는 요소에 적용된다.

선언 오른쪽에 나타나는 단락 설명.

15. 적합성 요구 사항

본 항목에서는 현재 규격과의 적합성 만을 정의한다.

사용자 에이전트가 특정 장치의 한계로 인해 일부 규격을 구현하지 못한다고 해서 (예를 들어, 상호작용 없이는 앞뒤가 맞지 않으므로, 비 상호적인 사용자 에이전트는 동적 슈도 클래스를 구현하지 못할 것이다) 적합하지 않는 것은 아니다.

셀렉터를 재사용하는 모든 규격에는 반드시 셀렉터가 수락하거나 제외시키는 서브세트를 수록하고, 현재 규격에 추가되는 제약을 설명한 프로파일이 들어가야 한다.

무효는 예를 들면 현재 구문 분석 포인트에서 인식되지 않거나 허용되지 않는 구문 분석 오류에서 비롯된다.

사용자 에이전트는 반드시 구문 분석 오류 처리 규칙을 관찰해야 한다:

선언되지 않은 네임스페이스 접두사가 들어있는 단순 셀렉터는 무효이다.

단순 셀렉터, 무효 콤비네이터, 또는 무효한 토큰이 들어있는 셀렉터는 무효이다.

무효한 셀렉터가 들어있는 셀렉터 그룹은 무효이다.

셀렉터를 재사용하는 모든 규격은 반드시 구문 분석 오류를 처리할 메서드를 정의해야 한다. (CSS의 경우, 사용되는 셀렉터가 드롭되는 전체 규칙.)

16. 테스트

본 규격에는 사용자 에이전트가 기본적인 규격과의 적합성을 검증할 수 있는 테스트 항목가 있다. 이 테스트 항목은 포괄적인 체 하지 않으며, 모든 가능한 셀렉터 결합 케이스를 다루지도 않는다.

부 록 Ⅰ

참조 문헌

Ⅰ.1. 규범 참조

[CSS21]

W3C, Bert Bos; et al., 'Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification', 2011. 06. 07., http://www.w3.org/TR/2011/REC-CSS2-20110607/

[CSS3NAMESPACE]

W3C, Elika J. Etemad; Anne van Kesteren., 'CSS Namespaces Module', 2011. 9.29., http://www.w3.org/TR/2011/REC-css3-namespace-20110929/

[FLEX]

Flex: The Lexical Scanner Generator. Version 2.3.7, ISBN 1882114213

[UNICODE]

The Unicode Consortium. The Unicode Standard, Version 6.0.0, (Mountain View, CA: The Unicode Consortium, 2011. ISBN 978-1-936213-01-6) and as updated from time to time by the publication of new versions. (See http://www.unicode.org/unicode/standard/versions/ for the latest version and additional information on versions of the standard and of the Unicode Character Database).

Available at http://www.unicode.org/versions/Unicode6.0.0/

[YACC]

S. C. Johnson. YACC - Yet another compiler compiler. Murray Hill. 1975. Technical Report.

Ⅰ.2. 정보 참조

[BCP47]

A. Phillips; M. Davis, 'Tags for Identifying Languages and Matching of Language Tags', 2009. 09. Internet Best Current Practice 47. URL:http://www.rfc-editor.org/rfc/bcp/bcp47.txt

[CSS1]

W3C, Håkon Wium Lie; Bert Bos., 'Cascading Style Sheets (CSS1) Level 1 Specification', 2008. 04.11. http://www.w3.org/TR/2008/REC-CSS1-20080411

[DOM-LEVEL-3-CORE]

W3C, Gavin Nicol; et al., 'Document Object Model (DOM) Level 3 Core Specification', 2004. 04.07. http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407

[HTML401]

W3C, David Raggett; Ian Jacobs; Arnaud Le Hors., 'HTML 4.01 Specification', 1999. 12. 24. http://www.w3.org/TR/1999/REC-html401-19991224

[MATHML]

W3C, Patrick Ion; Robert Miner., 'Mathematical Markup Language (MathML) 1.01 Specification', 1999. 07. 07. http://www.w3.org/1999/07/REC-MathML-19990707

[STTS3]

W3C, Daniel Glazman. 'Simple Tree Transformation Sheets 3. Electricité de France', 1998. 11.11. http://www.w3.org/TR/NOTE-STTS3

[SVG11]

W3C, Erik Dahlström et. al., 'Scalable Vector Graphics (SVG) 1.1 Specification', 2011. 08. 16. http://www.w3.org/TR/2011/REC-SVG11-20110816/

[UAX29]

Unicode Standard Annex #29, Mark Davis. Text Boundaries. 2005. 03. 25.,

http://www.unicode.org/unicode/reports/tr29/tr29-9.html

[XML-NAMES]

W3C, Tim Bray; et al. Namespaces in XML 1.0 (Third Edition). 2009. 08. 06., http://www.w3.org/TR/2009/PER-xml-names-20090806

[XML10]

W3C, C. M. Sperberg-McQueen; et al., 'Extensible Markup Language (XML) 1.0 (Fifth Edition)', 2008. 02. 05, http://www.w3.org/TR/2008/PER-xml-20080205

표준 작성 공헌자

표준 번호 : TTAK.OT-10.0356

이 표준의 제정·개정 및 발간을 위해 아래와 같이 여러분들이 공헌하였습니다.

구분	성명	위원회 및 직위	연락처 (E-mail 등)	소속사
표준(과제) 제안	전종홍	웹 프로젝트그룹 부의장	hollobit@etri.re.kr	한국전자통신연구원
표준 초안 작성자	전종홍	웹 프로젝트그룹 부의장	hollobit@etri.re.kr	한국전자통신연구원
표준 초안 에디터	전종홍	웹 프로젝트그룹 부의장	hollobit@etri.re.kr	한국전자통신연구원
표준 초안 검토	이승윤	웹 프로젝트그룹 의장	syl@etri.re.kr	한국전자통신연구원
		외 프로젝트그룹 위원		
표준안 심의	박승민	소프트웨어/콘텐츠 기술위원회 의장	minpark@etri.re.kr	한국전자통신연구원
		외 기술위원회 위원		
사무국 담당	이혜진	정보기술부 선임	031-724-0082 hjlee@tta.or.kr	TTA
	김영화	정보기술부 부장	ykim@tta.or.kr	TTA

정보통신단체표준(국문표준)

CSS 셀렉터 레벨 3
(CSS Selectors Level 3)

발행인 : 한국정보통신기술협회 회장
발행처 : 한국정보통신기술협회
463-824, 경기도 성남시 분당구 분당로 47
Tel : 031-724-0114, Fax : 031-724-0109
발행일 : 2014.12.