Home

2022-07-09

XHTML 방식이 아닌 HTML을 HTML의 방식으로 작성하세요.

> 원문: https://css-tricks.com/write-html-the-html-way-not-the-xhtml-way/

여러분은 XHTML을 (더 이상) 사용하지 않을 것이지만, HTML을 작성할 때 생각보다 XHTML의 영향을 많이 받을 수 있습니다. 여러분은 XHTML 방식으로 HTML을 작성하고 있을 수 있습니다.

HTML을 XHTML 방식으로 작성하는 것은 무엇이며, HTML을 HTML 방식으로 작성하는 방식은 무엇일까요? 함께 알아봅시다.

HTML, XHTML, HTML

1990년대에는 HTML이 있었습니다. 2000년대에는 XHTML이 있었고, 2010년대에는 HTML로 다시 돌아왔습니다. 간단한 이야기입니다.

또한 사양의 대략적인 날짜로도 알 수 있습니다. (HTML “1” 1992, HTML 2.0 1995, HTML 3.2 1997, HTML 4.01 1999; XHTML 1.0 2000, XHTML 1.1 2001; “HTML5” 2007).

XHTML은 XML과 XML에서 파생된 것들이 미래라고 모두가 믿었을 때 인기를 끌었었습니다. "XML이 전부 였습니다." 우리가 XHTML 방식으로 HTML을 작성하는 것을 배웠다는 사실로, HTML에게 많은 영향을 끼쳤다는 것을 알 수 있습니다.

HTML을 작성하는 XHTML 방식

XHTML 방식은 문서화가 매우 잘되어있습니다. “HTML4와의 차이”에서 XHTML 1.0에 대해 자세하게 설명하고 있습니다.

- 문서들은 잘 형식화 되어 있어야 합니다.
- 요소 및 속성 이름은 소문자이어야 합니다.
- 비어 있지 않은 요소의 경우 끝 태그가 필요합니다.
- 속성의 값은 항상 따옴표로 묶어야 합니다.
- 속성 최적화는 지원되지 않습니다.
- 빈 요소는 닫아야 합니다.
- 속성 값의 공백 처리는 XML에 따라 수행됩니다.

- 스크립트 및 스타일 요소에는 CDATA 섹션이 필요합니다.
- SGML 제외를 사용할 수 없습니다.
- `a`, `applet`, `form`, `frame`, `iframe`, `img` 및 `map` 과 같은 `id` 및 `name` 속성을 가진 요소에는 `id` 만 사용해야 합니다.
- 미리 정의된 값 집합이 있는 속성은 대소문자를 구분합니다.
- 16진수 값으로 참조하는 엔티티는 소문자여야 합니다.

위 내용이 익숙하게 보이나요? CDATA 콘텐츠 표시는 물론 SGML 제외 사항을 처리하는 것을 제외하면 다음과 같은 모든 규칙을 따를 수 있습니다. 전부요.

비록 XHTML은 죽었지만, 이러한 규칙들 중 많은 것들이 다시 반문되어지진 않았습니다. 일부는 HTML을 위한 "모범 사례"로 격상되기도 했습니다.

이 내용들이 HTML을 XHTML 방식으로 작성하는 방법이고, 지속적으로 이 분야에 영향을 미치고 있습니다.

HTML을 작성하는 HTML 방식

우리를 되돌리는 한 가지 방법은 XHTML에 의해 만들어진 규칙들을 부정하는 것입니다. (HTML이 더 이상 SGML을 기반으로 하지 않으므로 SGML은 제외) 자세한 내용은 아래와 같습니다.

- 문서의 형식이 올바르지 않을 수도 있습니다.
- 요소 및 속성은 소문자가 아닐 수도 있습니다.
- 비어 있지 않은 요소의 경우 끝 태그가 항상 필요한 것은 아닙니다.
- 속성 값이 항상 따옴표로 묶이는 것은 아닙니다.
- 특성 최소화가 지원됩니다.
- 빈 요소는 닫을 필요가 없습니다.
- 속성 값의 공백처리가 XML에 따라 수행되지 않습니다.
- 스크립트 및 스타일 요소에는 CDATA 섹션이 필요하지 않습니다.
- `id` 및 `name` 속성이 있는 요소는 `id` 만 사용하지 않아도 됩니다.
- 미리 정의된 값 집합이 있는 속성은 대소문자를 구별하지 않습니다.
- 16진수 값으로 참조하는 엔티티는 소문자만 사용하지 않을 수 있습니다.

난해한 것들, 관련이 없어 보이는 것들을 제거해 봅시다. 여기에는 XML 공백 처리, CDATA 섹션, 이름 속성 값 중복, 사전 정의된 값 집합의 경우 및 16진수 엔티티 참조가 포함됩니다.

- 문서의 형식이 올바르지 않을 수도 있습니다.
- 요소 및 속성은 소문자가 아닐 수도 있습니다.
- 비어 있지 않은 요소의 경우 끝 태그가 항상 필요한 것은 아닙니다.
- 속성 값이 항상 따옴표로 묶이는 것은 아닙니다.
- 특성 최소화가 지원됩니다.
- 빈 요소는 닫을 필요가 없습니다.

이러한 규칙에서 벗어나면 XML로 작업하는 것 보다 HTML로 작업하는 것에 가깝습니다. 하지만 아직 끝나지 않았습니다.

"문서의 형식이 올바르지 않을 수도 있습니다."는 HTML 코드가 유효하지 않아도 괜찮다는 것을 암시합니다. XHTML의 엄격한 오류 처리로 인해 XHTML이 올바른 형식을 가지게 되는 것은 좋은 점이었습니다. 그러나 HTML 문서는 심각한 구문 및 올바른 형식에 대한 문제가 발생해도 동작합니다. 좀 더 전문적으로 사용하는 경우에 이러한 탄력적인 기능을 사용하고 남용하는 것은 유용하지 않습니다. (저는 이전에 "프론트엔드 개발에 대한 비판적 방어"라는 글에서 이 사례를 논의한 적이 있습니다.)

따라서 HTML 방식은 "문서의 형식이 올바르지 않을 수도 있습니다."는 것을 제안하지 않을 것입니다. 또한 끝 태그 뿐만 아니라 시작 태그도 항상 필요한 것은 아닙니다. 구문을 재정비하고 재정렬 하는 것이 핵심입니다.

- 시작 및 끝 태그가 항상 필요한 것은 아닙니다.
- 빈 요소는 닫을 필요가 없습니다.
- 요소 및 속성 이름은 소문자 또는 대문자일 수 있습니다.
- 속성 값이 항상 따옴표로 묶이는 것은 아닙니다.
- 특성 최소화가 지원됩니다.

예제

실제로 이것은 사례에선 어떻게 작성되나요? 시작 태그와 끝 태그의 경우 많은 태그들은 선택 사항입니다. 예를 들어 XHTML에서는 다음과 같이 단락과 목록이 작성됩니다.

```
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
<ul>
  <li>Praesent augue nisl</li>
  <li>Lobortis nec bibendum ut</li>
  <li>Dictum ac quam</li>
</ul>
```

하지만 HTML에서는 이 코드로도 작성할 수 있습니다. (유효한 코드입니다)

```
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.
<ul>
  <li>Praesent augue nisl
  <li>Lobortis nec bibendum ut
  <li>Dictum ac quam
</ul>
```

개발자들은 빈 요소들을 작성할 때 이렇게 작성하라고 배웠었습니다.

```
<br />
```

이것은 XHTML이 HTML로 가져온 것이지만, 슬래시가 void 요소에 영향을 미치지 않기 때문에, 여러분은 아래처럼 해도 됩니다.

```
<br>
```

HTML에서는 모든 내용을 대문자로 쓸 수도 있습니다.

```
<a href="https://css-tricks.com/">CSS-Tricks</a>
```

이것은 여러분을 소리를 지르게 만들며 마음에 들지 않게할 수도 있지만, 이렇게 작성해도 괜찮습니다.

이 링크를 압축시키고 싶다면, HTML에서 특정 따옴표를 생략할 수 있는 옵션을 제공합니다.

```
<A HREF=https://css-tricks.com/>CSS-Tricks</A>
```

> 일반적으로 속성 값에 공백이나 등호가 포함되지 않은 경우 따옴표를 삭제해도 됩니다.

마지막으로, XHTML-HTML이 아닌 HTML-HTML은 속성을 최소화할 수 있습니다. 즉, 입력 요소를 필수 및 읽기 전용으로 표시하는 대신 다음과 같이 합니다.

```
<input type="text" required="required" readonly="readonly" />
```

속성을 최소화할 수 있습니다.

```
<input type="text" required readonly />
```

따옴표가 필요하지 않다는 사실뿐만 아니라 `text` 가 여기서 `type` 속성에 대한 기본값(더 많은 불필요한 속성-값 조합이라는 사실을 활용하는 경우 HTML의 미니멀한 아름다움에 대한 예를 볼 수 있습니다.

```
<input required readonly />
```

HTML을 HTML 방식대로 작성하세요.

위의 내용은 HTML이 90년대에는 어디에 있었는지를 말하고자 하는 것이 아닙니다. 그 당시 HTML은 레이아웃을 위한 `<table>` 요소들로 구성되어 있었고, 표현을 위한 코드들로 채워져 있었으며, 매우 다양한 사용자 에이전트 지원으로 인해 대부분 유효하지 않았습니다(현재도 여전히 그렇습니다). 그러나 XML과 XHTML이 등장하지 않았더라면 우리가 유지하고자 했던 것의 본질을 말합니다.

좀 더 포괄적이고 현대적인 HTML 작성 방법이 어떤 것인지 제안해 주시면 감사하겠습니다. (HTML은 제가 주요하게 집중하고 있는 영역입니다. 그래서 제 글들 중 일부에 대한 링크를 통해 이 내용을 보충합니다.)

1. 구문과 시맨틱을 준수합니다.

- HTML을 검증하고 유효한 HTML을 제공합니다.

2. HTML이 제공하는 옵션을 일관성 있게 사용합니다.

- 요소 및 속성 이름은 소문자 또는 대문자일 수 있다는 것을 기억하세요.

3. HTML 사용을 가장 최소로 유지하세요.

- 표현하거나 동작과 관련된 마크업은 CSS와 자바스크립트에 의해 처리하는 것으로 대체된다는 것을 기억하세요.
- 시작 및 끝 태그는 항상 필요하지는 않다는 것을 기억하세요.
- 빈 요소는 닫을 필요가 없다는 것을 기억하세요.
- 일부 속성은 속성-값 쌍을 생략할 수 있는 기본값을 허용한다는 것을 기억하세요.
- 속성 값들은 항상 따옴표로 묶이지 않아도 된다는 것을 기억하세요.
- 속성 최소화가 지원된다는 것을 기억하세요.

이것이 HTML을 위한 세 가지 기본 규칙과 유사하다는 것은 우연이 아니며, 더 작은 페이로드가 더 빠른 사이트로 이어진다는 전제 하에 있으며, 최소한의 웹 개발의 내용을 따르고 있습니다. 이 모든 것이 새로운 것은 아닙니다. 우리 분야는 그저 그것을 재발견하는 것을 결정할 수 있을 뿐입니다. 툴링도 사용할 수 있습니다. html-minifier는 아마도 가장 저명하며 모든 HTML 최적화를 처리할 수 있을 것입니다.

여러분은 HTML을 XHTML 방식으로 배웠습니다. HTML은 XHTML이 아닙니다. HTML을 재발견하고, HTML을 인식하는 새로운 현대적 작성 방법을 형성하는데 도움을 준다는 것은 인정합니다. 하지만 HTML은 반드시 XML에 기반 하는 것은 아닙니다.