NHN Coding Convention

for Markup Languages (HTML/CSS)

UIT센터 일반

저작권

문서 정보

문서 개요

이 문서는 NHN㈜의 마크업 개발자가 소스 코드 작성 시에 따라야 할 규칙을 기술한다.

1장. 개요

NHN 코딩 컨벤션의 필요성과 코딩 컨벤션의 요소, 용어를 소개한다.

2장. 네이밍 규칙

id/class, 이미지, 파일, 폴더의 네이밍 규칙을 설명한다.

3장. HTML 코드 작성 규칙

HTML 코드의 기본 작성 규칙과 들여쓰기, 빈 줄 사용, DTD 및 인코딩 선언, 주석 표기 규칙을 설명한다.

4장. HTML 엘리먼트 작성 규칙

HTML 엘리먼트 종류별 작성 규칙을 설명한다.

5장 CSS 코드 작성 규칙

CSS 코드의 기본 작성 규칙과 인코딩, 선택자, 속성 등의 작성 규칙을 설명한다.

6장. 웹 접근성 보장 방법

웹 접근성을 보장하도록 마크업하는 방법을 설명한다.

부록 A. 코딩 시 참고 사항

웹표준 기반의 마크업, 크로스 브라우징 범위, 플래시 사용 시 유의점, PNG 이미지 사용 방법을 설명한다.

부록 B. 예약어 목록

네이밍 예약어와 대체 텍스트 예약어 목록을 설명한다.

독자

이 문서는 사외 마크업 개발자를 대상으로 한다

표기 규칙

참고 표기

참고

독자가 참고해야 할 내용을 기술한다.

주의 표기

주의

독자가 반드시 알아야 할 사항, 시스템 에러를 유발할 수 있는 사항, 수행하지 않았을 때 재산상의 피해를 줄 수 있는 사항을 기술한다.

소스 코드 표기

이 문서에서 소스 코드는 회색 바탕에 검정색 글씨로 표기한다.

```
<dl class="error_list_type">
<dt>점검시간 :</dt>
<dd><span>2009.04.14(화) 오전 02:00~03:00</span>(총 6시간)</dd>
<dt>점검이유 :</dt>
<dd>방명록 UI 개선</dd>
</dl>
```

목차

표 및 그림 목록

표 목록

그림 목록

1. 개요

이 장에서는 NHN 마크업 코딩 컨벤션의 필요성, 요소 및 용어를 소개한다.

1.1 코딩 컨벤션 필요성

마크업 개발은 프런트-엔드 페이지의 기본 골격을 형성하기 때문에 디자인, 브라우저, 스크립트, 성능, 접근성 등과 긴밀한 관계가 있다. 즉, 마크업 개발을 잘 해야 모든 브라우저에서 콘텐츠를 손실 없이, 빠르고 쉽게 사용자에게 전달할 수 있다. 코딩 컨벤션은 이러한 조건을 만족시키기 위해 마크업 개발자가 지켜야 할 표준을 제시한다.

또한, 유지보수에 투자되는 비용을 최소화하기 위해 통일된 코드 작성법을 제시한다. 코드를 최초로 작성한 사람이 끝까지 유지보수할 확률은 매우 낮다. 따라서, 최초 개발자가 아닌 사람도 코드를 빠르고 정확하게 이해할 수 있도록 작성하는 것은 코드의 유지보수 비용을 절감하고 업무 효율을 높이는 데 결정적인 역할을 한다.

적어도 한 프로젝트의 마크업 코드는 같은 코딩 컨벤션에 따라 작성해야 한다. 코딩 컨벤션을 준수하면 프로젝트 멤버 간 코드 공유도 쉬워지고, 일관성 있게 코드를 작성할 수 있다. 어떤 코딩 컨벤션을 선택하느냐가 중요한 것이 아니라, 통일된 기준으로 소스 코드를 작성하는 것이 중요하다.

1.2 코딩 컨벤션 요소

1.2.1 네이밍 규칙

네이밍 규칙은 레이아웃, 객체, 이미지, 폴더, 파일의 이름을 작성하는 규칙이다. 이해하기 쉬운 이름으로 작성해야 코드를 쉽게 파악할 수 있다.

1.2.2 HTML 코드 작성 규칙

A. 들여쓰기

HTML 코드를 작성할 때 코드의 가독성을 높이기 위하여 왼쪽 첫 번째 열부터 오른쪽으로 일정한 간격만큼 띄어 쓴다. 들여쓰기를 하면 전체 HTML 구조를 쉽게 파악할 수 있다.

B. 빈 줄

HTML 코드의 빈 줄은 코드 그룹의 영역을 표시하기 위하여 사용한다.

C. DTD 및 인코딩

DTD(Document Type Definition)는 SGML(Standard Generalized Markup Language) 계열 마크업 언어의 문서 타입을 정의하는 것으로서, 해당 HTML 문서가 어떤 버전의 HTML로 작성되었는지, 어떤 규칙으로 내용을 기술하고 어떤 엘리먼트와 애트리뷰트, 애트리뷰트값을 지정할 수 있는지 정의한다. 또한 인코딩을 선언하여 문서에서 사용되는 문자 코드 세트를 지정한다. DTD와 인코딩 선언은 HTML 문서가 브라우저에서 바르게 해석될 수 있도록 한다.

D. 주석

HTML 코드의 주석은 코드 그룹을 구분하거나, 코드의 수정과 삭제를 표시하거나, 개발자가 참고해야 하는 사항을 기술한다.

1.2.3 HTML 엘리먼트 작성 규칙

HTML 엘리먼트 작성 규칙은 반드시 선언해야 하는 애트리뷰트와 선택적 사용이 가능한 애트리뷰트에 대한 내용을 기술하고 선언 순서를 제시하여 코드 품질을 유지한다.

1.2.4 CSS 코드 작성 규칙

A. 들여쓰기

CSS 코드는 들여쓰기를 하지 않는다.

B. 공백

CSS 코드는 공백을 최소화한다.

C. 빈 줄

CSS 코드의 빈 줄은 코드 그룹의 영역을 표시하기 위하여 사용한다.

D. 줄 바꿈

CSS 코드의 가독성을 위한 줄 바꿈은 하지 않는다.

E. 인코딩

CSS의 인코딩은 HTML의 인코딩과 동일하게 선언하여 HTML 문서가 브라우저에서 바르게 해석될 수 있도록 한다.

F. 선택자

선택자 버그가 발생하지 않도록 사용 규칙을 준수한다.

G. 속성

속성 선언 순서를 준수하여 개발자가 코드를 쉽게 파악할 수 있도록 하며, CSS 코드 최적화를 위해 속성값을 축약하여 사용하고 약식 속성을 허용 범위에 맞게 사용한다.

H. z-index

z-index 속성값을 범위에 맞게 사용하여 객체가 브라우저에서 바르게 표현되도록 한다.

I. 핵

크로스 브라우징을 위해 제시된 핵(hack)에 한해 최소한의 사용을 허용한다.

J. CSS 선언 타입

상황에 알맞은 CSS 선언 타입을 선택한다.

K. 주석

CSS 코드의 주석은 코드 그룹을 구분하거나, 코드의 수정과 삭제를 표시하거나, 개발자가 참고해야 하는 사항을 기술한다.

1.2.5 웹 접근성 보장 방법

모든 사람이 환경의 제약 없이 웹 콘텐츠에 접근할 수 있도록 보장하는 마크업 방법을 기술한다.

1.3 코딩 컨벤션 용어

A. 엘리먼트(Element)

HTML 문서를 구성하는 요소를 의미한다. 일반적으로 시작 태그, 내용, 종료 태그로 구성된다.

B. 애트리뷰트(attribute)

엘리먼트에 부여할 수 있는 특성을 의미한다. 기본값이 설정되어 있으나 애트리뷰트를 선언하여 다른 값으로 설정할 수 있다.

C. 선택자(selector)

엘리먼트에 CSS 스타일을 적용하기 위한 패턴이다. 이 문서에서 언급하는 선택자의 종류는 다음 표와 표 1-1같다.

표 1-1 선택자 종류

이름	패턴 예	설명
공통 선택자	*	어떤 엘리먼트와도 일치함.
타입 선택자	div	해당 엘리먼트와 일치함.
하위 선택자	div p	해당 엘리먼트의 하위의 모든 엘리먼트와 일치함.
자식 선택자	div > p	해당 엘리먼트의 자식 엘리먼트와 일치함.
class 선택자:	div.class	해당 엘리먼트의 class 애트리뷰트값과 일치함.
id 선택자	div#id	해당 엘리먼트의 id 애트리뷰트값과 일치함.

D. 속성(property)

엘리먼트에 부여할 CSS 스타일 특성을 의미한다. 기본값이 설정되어 있으나 속성을 선언하여 다른 값으로 설정할 수 있다.

E. 예약어

이 문서에서 사용하는 예약어는 일반적인 의미와 다르게 사용된다. 이 문서에서 예약어는 객체, 이미지, 파일 및 폴더의 네이밍 시 의미를 일관되게 표현하기 위해 미리 지정해 놓은 일종의 언어 규칙을 의미한다. 예를 들어, '검색'을 표현하는 예약어가 'srch'일 경우, 검색 영역을 위한 객체의 class 이름은 'srch'라는 예약어를 반드시 포함해야 한다.

2. 네이밍 규칙

이 장에서는 id/class, 이미지, 파일, 폴더의 네이밍 규칙을 설명한다.

2.1 기본 규칙

다음과 같은 기본 네이밍 규칙을 준수한다.

A. 일반 규칙

이름은 영문 소문자, 숫자, 언더스코어(_)로 작성한다.

B. 시작 이름

이름은 영문 대문자, 숫자로 시작할 수 없다.

표 2-1 공통 네이밍 규칙 예

잘못된 예	올바른 예
Btn	btn
2btn	btn2

C. 예약어

- 예약어가 있는 경우 예약어를 사용한다.
- 예약어는 표 B-1 공통 네이밍 예약어에 근거하여 작성한다.
- 예약어가 없으면, 종류와 특성을 나타내도록 네이밍한다.

D. 영문 소문자, 숫자, 언더스코어 조합

- 영문 소문자, 숫자, 언더스코어(_)만 사용할 수 있다.
- 언더스코어(_)는 2개 이상의 단어를 조합할 때만 사용한다.
- 단어와 숫자를 조합하는 경우 언더스코어(_)를 생략한다.
- 언더스코어(_)가 포함된 예약어는 숫자, 영문 소문자와 조합하여 사용할 수 있다.
- 언더스코어(_)를 이용하여 3단계를 초과하여 조합할 수 없다.
- 숫자가 없으면 '1'이라는 숫자가 생략된 것으로 간주한다. 단, 숫자 정보이면 '1'을 표기한다.

표 2-2 네이밍 조합 예

기본형	잘못된 예	올바른 예
section	sectionLst	section_lst
	section_list_style_type1	section_lst_type1
	section, section1, section2	section, section2, section3
no	-	no1, no2, no3 …. no10

2.2 id 및 class 네이밍 규칙

다음 그림은 HTML의 기본 템플릿 구조를 나타낸 것이다.

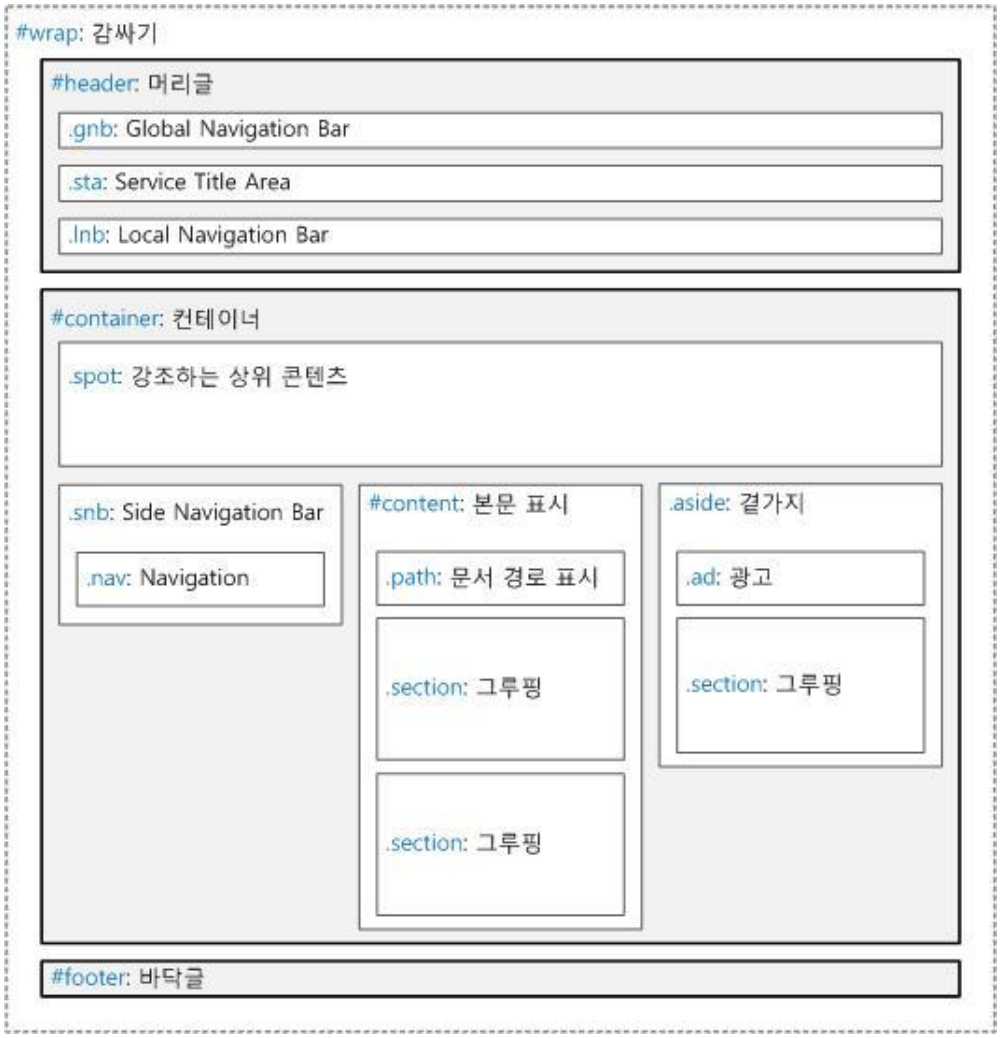


그림 2-1 HTML 기본 템플릿 구조

A. id, class

- id는 문서에서 한 번만 사용한다.
- class는 문서에서 여러 번 사용할 수 있다.

B. 레이아웃 예약어

레이아웃에는 다음 표에 예약된 id만 사용한다.

표 2-3 레이아웃 예약어 범위

예약어	범위
#wrap	페이지 전체 영역
#header	머리글 영역

예약어	범위
#container	본문 영역
#content	주요 콘텐츠 영역
#footer	바닥글 영역

C. 팝업 레이아웃 예약어

- 팝업 문서의 레이아웃 지정 범위는 동일하다.
- 레이아웃 예약어 앞에 'pop_'를 조합하여 사용한다.

표 2-4 팝업 레이아웃 예약어 범위

예약어	범위
#pop_wrap	페이지 전체 영역
#pop_header	머리글 영역
#pop_container	본문 영역
#pop_content	주요 콘텐츠 영역
#pop_footer	바닥글 영역

D. 레이아웃 네이밍 조합

- 레이아웃 id의 네이밍은 조합하여 사용할 수 없다.
- 레이아웃에 디자인 속성을 추가/변경하려면 class를 사용한다.

표 2-5 레이아웃 네이밍 조합 예

잘못된 예	올바른 예
#wrap2, #wrap_type	#warp

```
<div id="wrap">
  <div id="header"></div>
  <div id="container" class="container_type1">
     <div id="content"></div>
  </div>
  </div>
  <div id="footer"></div>
</div>
```

E. 객체 네이밍

- 객체 예약어는 표 B-2 객체 예약어에 근거하여 작성한다.
- 팝업, iframe에도 동일한 규칙을 적용한다.

F. class 네이밍 확장

- 종속 확장 class: 기본형 class에 종속되어 여백, 색깔, 행간 등의 몇 가지 속성을 부여하고자 할 때 사용하는 class.
- 독립 확장 class: 기본형 class의 변형이 타입으로 분류할 만큼 클 경우 사용하는 class.

표 2-6 class 네이밍 확장 예

기본형	확장형	설명
mykin_lst	mykin_lst_v1, mykin_lst_v2...	종속 확장 class
	mykin_lst2, mykin_lst3	독립 확장 class

2.3 이미지 네이밍 규칙

다음과 같은 이미지 네이밍 규칙을 준수한다.

A. 이미지 네이밍

- 이미지 예약어는 표 B-3 이미지 예약어에 근거하여 작성한다.
- 같은 분류의 이미지가 두 개 이상이면 파일 이름 마지막에 숫자를 추가하여 구분한다.
- 임시 이미지에 한해 특수문자를 사용할 수 있다.
- 이미지 네이밍은 이미지 확장자와 관계 없이 순차적으로 적용한다. 예) bu_dot.gif, bu_dot2.jpg, bu_dot3.png
- 이미지 네이밍을 확장해야 할 때는 B. 이미지 네이밍 조합 규칙을 준수한다.
- 임시 이미지에 한해 @네이밍을 허용한다. 예) @tmp_

B. 이미지 네이밍 조합

이미지 이름은 '형태_의미_상태' 순서로 조합한다.

표 2-7 이미지 네이밍 조합 예

잘못된 예	올바른 예	설명
on_recommend_tab1	tab1_recomm_on	'형태_의미_상태' 순서로 조합한다.
bnm.gif	btn_naver_mail.gif	임의로 축약하지 않는다.
btn_search_naver_mail.gif	btn_srch_mail.gif	3 단계 이하로 조합한다.
btn_Search.gif	btn_srch.gif	영문 소문자를 사용한다.
1btn_search.gif	btn_srch.gif	숫자로 시작하지 않는다.

2.4 파일 및 폴더 네이밍 규칙

다음과 같은 파일 및 폴더 네이밍 규칙을 준수한다.

A. 파일 및 폴더 네이밍

- 산출되는 모든 HTML 파일과 폴더를 분류해야 할 경우, 파일 이름, 폴더 이름 앞에 '숫자+언더스코어'를 사용하여 그루핑한다.
- HTML 파일과 폴더를 그루핑하기 위해 '숫자+언더스코어'를 사용한 경우에만 4단계의 네이밍 분류를 허용한다.
- 그루핑할 때 부여하는 숫자는 00~99의 범위 내에서 1씩 증가한다.

표 2-8 파일 및 폴더 네이밍 규칙 예

분류	예제	설명
HTML	00_mykin_.html	'00~99_메뉴영문이름.html'로 사용한다.
	pop_.html	팝업 파일을 사용
	ifr_.html	iframe 파일을 사용
CSS	news_.css	'서비스영문이름.css'로 사용한다.
Folder	p_yymmdd_프로젝트이름	신규 프로젝트 작업 시 사용
	yymmdd_서스테이닝이름	서스테이닝 작업 시 사용
	img, css, js	Image, css, javascript 폴더 사용
	p_yymmdd_프로젝트명₩00_메뉴명	HTML 파일의 폴더 분류가 필요한 경우 사용

B. 파일 및 폴더 네이밍 조합

- HTML 파일은 '그루핑숫자_메뉴이름_의미_상태' 순서로 조합한다.
- CSS 파일은 '서비스이름_메뉴이름' 순서로 조합한다.

표 2-9 HTML/CSS 조합 예

분류	조합 예	설명
HTML	00_mykin_nboard.html	'그루핑숫자_메뉴이름_의미_상태' 순서로 조합한다.
	00_mykin_nboard_view.html	
CSS	news_home.css	'서비스이름_메뉴이름' 순서로 조합한다.
	news_admin.css	

3. HTML 코드 작성 규칙

이 장에서는 HTML 코드의 기본 작성 규칙과 들여쓰기, 빈 줄 사용, DTD 및 인코딩 선언, 주석 표기 규칙을 설명한다.

3.1 기본 규칙

HTML 코드를 작성할 때 다음과 같은 기본 규칙을 준수한다.

A. W3C Validation

HTML은 해당 DTD의 명세에 맞게 작성하며, W3C validation을 통과해야 한다.

B. 영문 소문자 사용

DTD를 제외한 모든 엘리먼트와 애트리뷰트는 소문자로 작성한다.

표 3-1 소문자 작성 예

잘못된 예	올바른 예
<SPAN Class="desc">간단한 설명</SPAN>	간단한 설명

C. 애트리뷰트값 표기

애트리뷰트값은 큰따옴표(" ")로 묶는다.

표 3-2 애트리뷰트값 표기 예

잘못된 예	올바른 예
<img src='test.gif' width='100' height='100' alt='테스트'>	<img src="test.gif" width="100" height="100" alt="테스트">

D. entity 코드 사용

특수기호는 entity name을 사용하여 entity 코드로 변환한다. entity number는 사용하지 않는다.

entity 코드는 ISO-8859-1을 기준으로 하며, 아래 경로에서 확인할 수 있다.

- HTML ISO-8859-1 Reference: http://w3schools.com/tags/ref_entities.asp

표 3-3 entity 코드 사용 예

잘못된 예	올바른 예
<a href="...&nid=2">	<a href="... &nid=2">

참고

<, >, ", ', & 등의 특수기호를 entity 코드로 변환하지 않으면, 브라우저가 이를 시작/종료 엘리먼트나 애트리뷰트로 잘못 해석할 수 있다. 자동으로 생성되는 링크의 경로나 이미지의 alt값에도 entity 코드가 바르게 적용되도록 한다.

E. 종료 태그 사용

종료 태그를 반드시 선언한다. 단, XHTML을 제외한 DTD에서는 다음의 엘리먼트 10개에 대해서 종료 태그를 선언하지 않는다.

```
<area>, <br>, <col>, <frame>, <hr>, <img>, <input>, <link>, <meta>, <param>
```

표 3-4 XHTML을 제외한 나머지 DTD에서의 종료 태그 사용 예

잘못된 예	올바른 예
<img src="test.gif" width="100" height="100" alt="테스트"></img>	<img src="test.gif" width="100" height="100" alt="테스트">
	

3.2 들여쓰기 규칙

들여쓰기를 하면 코드의 가독성이 높아지고 전체 HTML 구조를 쉽게 파악할 수 있다. 다음과 같은 들여쓰기 규칙을 준수한다.

마크업의 중첩이 깊어질 때마다 자식 엘리먼트는 1탭 들여 쓰고, 1탭의 크기는 공백 4칸으로 설정한다. 단, 다음의 경우에는 들여 쓰지 않는다.

- <html>의 자식 엘리먼트인 <head>, <body>
- <head>의 자식 엘리먼트
- <body>의 자식 엘리먼트
- <table>의 자식 엘리먼트인 <caption>, <colgroup>, <col>, <thead>, <tbody>, <tfoot>, <tr>,<th>, <td>
- , <ol>, <dl>의 자식 엘리먼트인 <li>, <dt>, <dd> 엘리먼트

표 3-5 들여쓰기 사용 예

잘못된 예	올바른 예
<table> <caption>...</caption> <colgroup> <col>...</col> </colgroup> <thead> <tr> <th>...</th> <th>...</th> </tr> </thead> <tbody> <tr> <td>...</td> <td>...</td> </tr> </tbody> </table>	<table> <caption>...</caption> <colgroup> <col>...</col> </colgroup> <thead> <tr> <th>...</th> <th>...</th> </tr> </thead> <tbody> <tr> <td>...</td> <td>...</td> </tr> </tbody> </table>
     	 ... ...
    	 ... ...

잘못된 예	올바른 예
<dl> <dt>...</dt> <dd>...</dd> </dl>	<dl> <dt>...</dt> <dd>...</dd> </dl>

참고

Dreamweaver와 Editplus에서 Tab Size 설정하는 법

- Dreamweaver: [Preferences] > [Category] > [Code Format] > [Indent & Tab Size] > Indent with 1 Tabs, Tab size 4
- Editplus: [Tools] > [Preferences] > [Categories] > [Files] > [Settings and syntax] > [Tab/Indent]

3.3 빈 줄

빈 줄을 사용하려면 다음과 같은 사용 규칙을 준수한다. 의미 있는 객체를 구분하기 위하여 코드 그룹 간 1줄씩 빈 줄을 만드는 것은 허용한다. 빈 줄의 간격은 1줄을 초과하지 않는다.

빈 줄을 사용하는 것은 선택 사항이다.

표 3-6 빈 줄 사용 예

잘못된 예	올바른 예
<head>	<head>
내용...	내용...
</head>	</head>
<!-- 빈 줄 -->	<!-- 빈 줄 -->
<!-- 빈 줄 -->	<body>
<body>	내용...
내용...	</body>
</body>	

3.4 DTD 및 인코딩

3.4.1 DTD 선언

HTML 문서는 반드시 DTD를 선언한다.

A. 기본 DTD

신규 HTML 문서를 작성할 때 기본 DTD는 'HTML 4.01 Transitional'을 사용한다. 작성 예는 다음과 같다.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

참고

XHTML DTD는 다음과 같은 이유로 사용하지 않는다.

- HTML 4.01 역시 XHTML 만큼 표현 규칙에 맞는(well-formed) 언어임.
- HTML 5의 확장성은 HTML 4.01이 더 좋음.
- XHTML에서 VML(Vector Markup Language)이 바르게 표현되지 않음.

B. 기타 DTD

사용 중인 서비스를 부분 개편하거나 완전히 개편하더라도 기존의 HTML/CSS 데이터 의존도가 높다면, 기존과 동일한 DTD를 사용할 수 있다. 작성 예는 다음과 같다.

XHTML 1.0 Transitional

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Quirks Mode

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
```

주의

아래와 같은 경우 DTD 설정별 표준 문법으로 마크업 하더라도 브라우저에서 Quirks Mode로 인식하여 바르게 해석되지 않으므로 주의한다.

- DTD가 선언되지 않은 경우(html 태그로 문서 시작)

```
<html>
```

- 선언한 DTD 앞에 다른 문자가 오는 경우

```
<!-- //html 문서 시작 -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.
org/TR/html4/loose.dtd">
```

3.4.2 인코딩 선언

HTML 문서는 반드시 인코딩 정보를 선언한다. 인코딩 설정은 DB의 인코딩 방식과도 관련이 있으므로 반드시 담당 개발자와 협의하여 정해야 한다.

A. 기본 인코딩

신규 HTML 문서를 작성할 때 기본 인코딩은 utf-8을 원칙으로 한다. 다음은 인코딩 방식으로 utf-8을 선언한 예이다.

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
```

utf-8은 다국어 지원이 가능하며, euc-kr보다 표현 가능한 한글(고어, 음절 등)이 더 많다.

참고

한글은 utf-8에서 3바이트, euc-kr에서 2바이트를 차지하기 때문에, utf-8이 euc-kr에 비하여 DB 저장 용량이나 트래픽이 많을 수 있다.

Internet Explorer 7 이상의 버전에서 아래와 같이 링크의 밑줄이 글자와 붙어 보이는 현상이 있다.

무궁화꽃이피었습니다

B. 기타 인코딩

utf-8 인코딩을 사용할 수 없으면 euc-kr을 사용한다. 다음은 인코딩 방식으로 euc-kr을 선언한 예이다.

```
<meta http-equiv="Content-Type" content="text/html; charset=euc-kr">
```

HTML, CSS 파일을 저장할 때 반드시 설정한 인코딩을 선택하여 저장한다.

3.5 주석

A. 기본 형식

HTML 주석의 시작과 종료는 아래와 같이 표기하며, 기본 형식에 맞게 작성한다.

```
시작 주석 <!-- 주석 내용 -->
종료 주석 <!-- //주석 내용 -->
```

- 주석 기호와 주석 내용 사이에는 반드시 공백 한 칸이 있어야 한다.
- 주석 기호와 주석 내용 사이에는 다른 기호가 올 수 없다.
- 주석 기호와 주석 내용 사이의 줄 바꿈은 허용하지 않는다. 단, 주석 내용 간 줄 바꿈은 허용한다.
- 시작과 종료 주석의 주석 내용은 동일해야 한다.

표 3-7 주석 기본 형식 예

잘못된 예	올바른 예
<!--GNB-->	<!-- GNB --> … <!-- //GNB-->
<!--* GNB *-->	
<!-- GNB -->	
<!-- GNB 시작--> … <!-- //GNB 끝 -->	

주의

마크업과 개발의 편의를 위해 작성한 주석은 실제 서비스를 적용할 때 반드시 삭제한다.

B. 레이아웃 및 콘텐츠 영역의 주석 표기

wrap을 제외한 레이아웃과 독립된 콘텐츠 영역의 시작과 끝에 주석을 표기하며, 레이아웃은 id 이름과 동일하게 주석을 넣는다. 독립된 콘텐츠 영역의 주석 표기는 선택 사항이다.

HTML 코드와 주석은 줄로 분리해야 한다.

```
<!-- content -->
<div id="content">
   <!-- 네임카드 -->
   <div class="namecard"> ... </div>
   <!-- //네임카드 -->
</div>
<!-- //content -->
```

C. 수정된 부분의 주석 표기

SVN 로그로 수정 내용을 전달하기 어려우면, 수정된 부분의 시작과 끝에 주석을 넣는다. 주석 내용 앞에는 YYMMDD의 형식으로 수정된 날짜 정보를 입력한다. 필요에 따라서 수정된 내용을 한 줄 이내로 입력할 수 있다.

수정된 부분의 주석 표기는 선택 사항이다.

```
<!-- 091120 -->
...
<!-- //091120 -->

<!-- 091120 수정된 내용 -->
...
<!-- //091120 수정된 내용 -->
```

D. 삭제 및 숨김 처리된 부분의 주석 표기

삭제되었거나 숨김 처리된 부분은 해당 영역의 시작과 끝에 주석을 표기하며, 주석 표기와 내용은 줄을 구분한다. 주석 작성 규칙은 3.5C. 수정된 부분의 주석 표기와 동일하다.

삭제된 부분의 주석 표기는 선택 사항이며 삭제 및 숨김 주석에 포함된 주석은 삭제한다.

```
<!-- 091120 삭제
...
//091120 삭제 -->

<!-- 091120 숨김 처리
...
//091120 숨김 처리 -->
```

E. 개발 적용과 관련된 주석 표기

개발 적용과 관련된 주석은 해당되는 영역 위에 표기하며, 종료 주석은 표기하지 않는다. 주석 앞에는 [D]라는 말머리를 사용하여 담당 개발자가 반드시 확인할 수 있도록 한다. 주석이 두 줄 이상이 되더라도 주석 기호는 한 번만 표기한다.

표 3-8 개발 적용 관련 주석 표기 예

잘못된 예	올바른 예
<!-- 케이스별 클래스 변화 --> <!-- 의사 : my_doctor 변호사 : my_lawyer -->	<!-- [D] 케이스별 클래스 변화 의사 : my_doctor 변호사 : my_lawyer -->

3.6 초기 파일

신규 HTML 문서를 작성할 때 아래 파일을 기본으로 한다.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html lang="ko">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta http-equiv="Content-Script-Type" content="text/javascript">
<meta http-equiv="Content-Style-Type" content="text/css">
<title>네이버 :: 세상의 모든 지식, 네이버</title>
<link rel="stylesheet" type="text/css" href="sevice name.css">
</head>
<body>
<div id="wrap">
   <!-- header -->
   <div id="header">
   </div>
   <!-- //header -->
   <!-- container -->
   <div id="container">
      <!-- content -->
      <div id="content">
      </div>
      <!-- //content -->
   </div>
   <!-- //container -->
   <!-- footer -->
   <div id="footer">
   </div>
   <!-- //footer -->
</div>
</body>
</html>
```

4. HTML 엘리먼트 작성 규칙

이 장에서는 HTML 엘리먼트 종류별 작성 규칙을 설명한다.

4.1 기본 규칙

특정 엘리먼트에 class, style을 선언할 때는 선언 순서를 준수한다. 다음과 같이 class와 style은 제일 뒷부분에 선언한다.

```
<input type="text" id="user_id" title="사용자ID" class="input_txt" style="width:100px;">
```

4.2 전역 구조화 엘리먼트

A. 〈html〉

다음과 같이 lang 애트리뷰트를 선언한다.

class 애트리뷰트는 선언하지 않는다.

```
<html lang="ko">
```

lang 애트리뷰트는 User Agent가 언어를 올바로 해석할 수 있게 도와주며, 검색과 음성 장치(speech synthesizers)에 활용된다. 언어 코드는 모든 엘리먼트에 사용할 수 있지만 HTML 엘리먼트에 해당 문서의 주 언어 코드만 선언한다.

HTML 4.01 Specification, XHTML 1.0 Specification에서 자연어는 RFC 1766에 명시된 언어 코드를 사용한다. RFC1766은 각각 ISO639, ISO3166을 참조하며, ISO Specification은 유료이므로 아래 웹 페이지를 참조한다.

- MSDN Language Codes: http://msdn.microsoft.com/en-au/library/ms533052(VS.85).aspx

자주 사용하는 국가코드는 en(영어), ja(일본어), zh-cn(중국어)이다.

참고

개발팀에서 XML namespace를 정의하고자 할 때에는 아래와 같이 lang 애트리뷰트 다음에 xmlns 애트리뷰트를 선언한다.

```
<html lang="ko" xmlns="http://www.w3.org/1999/xhtml">
```

B. 〈head〉

meta, title, link, script, style 순서로 엘리먼트를 선언한다.

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta http-equiv="Content-Script-Type" content="text/javascript">
<meta http-equiv="Content-Style-Type" content="text/css">
<title>속보 :: 뉴스</title>
<link rel="stylesheet" type="text/css" href="css/default.css">
<script type="text/javascript" src="js/default.js"></script>
<style type="text/css">
  [stuff]
</style>
</head>
```

C. 〈meta〉

문서의 기본 인코딩, 스크립트 형식, 스타일 형식 순서로 엘리먼트를 선언한다.

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta http-equiv="Content-Script-Type" content="text/javascript">
<meta http-equiv="Content-Style-Type" content="text/css">
```

D. 〈title〉

"메뉴 :: 서비스 이름"의 형식으로 작성한다.

```
<title>속보 :: 뉴스</title>
```

E. <link>

rel, type, href 순서로 애트리뷰트를 선언한다.

```
<link rel="stylesheet" type="text/css" href="css/default.css">
```

F. <script>

type, src 순서로 애트리뷰트를 선언한다.

```
<script type="text/javascript" src="js/default.js"></script>
```

script는 <head></head> 내에 선언하는 것을 원칙으로 한다. 단, 성능상의 이슈가 있으면 개발 부서와 협의하여 선언 위치를 변경한다.

참고

language 애트리뷰트는 HTML4 이전 버전의 하위 호환성을 위해 사용하는 비표준 애트리뷰트로, 사용하지 않는다.

G. <style>

type 애트리뷰트를 선언한다.

```
<style type="text/css">
  [stuff]
</style>
```

H. <body>

class 애트리뷰트는 웹페이지의 스킨셋을 변경해야 할 때 선택적으로 사용한다.

4.3 폼 엘리먼트

폼 컨트롤 엘리먼트를 마크업할 때 form, fieldset, legend 엘리먼트를 다음과 같이 선언한다.

```
<form action="">
<fieldset>
<legend>개인정보</legend>
   [form control element here]
</fieldset>
</form>
```

A. 〈form〉

action 애트리뷰트는 서버 사이드 폼 핸들러이나 필수 선언 애트리뷰트이기 때문에, 마크업 단계에서 다음과 같이 선언한다.

```
<form action="">
</form>
```

CSS로 디자인을 컨트롤하지 않는다.

B. 〈fieldset〉

form 엘리먼트의 자식 노드로 선언하여 폼 컨트롤 엘리먼트들을 그루핑하기 위해 선언한다.

```
<form action="">
<fieldset>
<legend>개인정보</legend>
   [stuff]
</fieldset>
</form>
```

C. 〈legend〉

폼 컨트롤 그룹인 fieldset의 자식 엘리먼트로서 폼 컨트롤 엘리먼트들의 그룹 이름을 표현하기 위해 선언한다.

```
<form action="">
<fieldset>
<legend>개인 정보</legend>
   [stuff]
</fieldset>
</form>
```

D. 〈input〉

다음과 같이 label 엘리먼트 또는 title 애트리뷰트를 선언한다(type="image"이면 alt 애트리뷰트 사용).

```
<label for="user_id">아이디</label> <input type="text" id="user_id" name="user_id">
<input type="image" src="btn_confirm.gif" alt="확인">

<label for="num1">유선전화</label>
<input type="text" id="num1" name="num1" title="지역번호">
```

label 엘리먼트, title 애트리뷰트, alt 애트리뷰트를 통해 스크린 리더 사용자는 주변 맥락에 대한 이해 없이 각 엘리먼트에 독립적으로 접근해도 폼을 이해할 수 있다.

다음과 같이 type값에 따라 애트리뷰트를 선언한다.

- type이 text인 경우: type, id, title, value, accesskey 순서로 애트리뷰트를 선언한다.

```
<input type="text" id="user_id" title="사용자ID" value="아이디를 입력하세요" accesskey="l">
```

- 동일한 스타일의 텍스트필드이나 너비값이 다를 경우 style 애트리뷰트를 이용하여 width값을 제어한다.
- 텍스트필드에 가이드를 위한 내용이 들어갈 경우 value 애트리뷰트를 선택적으로 사용할 수 있다.
- accesskey 애트리뷰트를 선언하여 검색 창에 단축키 "s", 로그인 창에 단축키 "l"을 부여하는 것을 권장한다.

- type이 radio, checkbox인 경우: type, name, id, checked 순서로 애트리뷰트를 선언한다.

```
<input type="radio" name="vt_align" id="align_lft" checked="checked"><label
for="align_lft">왼쪽정렬</label>
<input type="radio" name="vt_align" id="align_cnt"> <label
for="align_cnt">가운데정렬</label>
<input type="radio" name="vt_align" id="align_rgt"> <label
for="align_rgt">오른쪽정렬</label>

<input type="checkbox" name="sports" id="soccer" checked="checked"> <label
for="soccer">축구</label>
<input type="checkbox" name="sports" id="basketball"> <label
for="basketball">농구</label>
<input type="checkbox" name="sports" id="tennis"> <label for="tennis">테니스</label>
```

- 그루핑이 필요하면 선택적으로 name 애트리뷰트를 이용하여 항목들을 그루핑한다.
- 기본 선택에 대한 표현이 필요할 경우 checked="checked"라고 표기한다.
- title 애트리뷰트는 선언하지 않는다.

- type이 image인 경우: type, src, alt 순서로 애트리뷰트를 선언한다. 폼 전송 역할을 하는 디자인 버튼은 image 타입을 사용한다(디자인이 적용되지 않은 버튼은 <button type="submit">로 사용).

```
<input type="image" src="img/btn_confirm.gif" alt="확인">
```

E. <select>

동일한 스타일의 셀렉트 박스이나 너비값이 다르면 선택적으로 style 애트리뷰트를 이용하여 width값을 제어한다.

```
<select style="width:100px;">
<option>등급</option>
</select>
```

F. <label>

라디오 버튼, 체크 박스는 for 애트리뷰트를 부여하여 해당 엘리먼트의 id값과 동일한 이름으로 연결(coupling)한다.

```
<input type="radio" name="alignment" id="align_lft"> <label
for="align_lft">왼쪽정렬</label>
<input type="checkbox" name="sports" id="soccer"> <label for="soccer">축구</label>
```

G. <textarea>

cols, rows 순서로 애트리뷰트를 선언한다.

CSS를 정상적으로 불러오지 못하는 상황에서도 사용에 문제가 없도록 col, rows의 애트리뷰트값은 각각 최소 30, 5 이상이 되도록 선언한다.

```
<textarea cols="30" rows="5"></textarea>
```

H. <button>

type 애트리뷰트를 선언한다.

- type 애트리뷰트를 button으로 선언하여 열기/닫기, 접기/펼치기 등의 UI를 제어한다.
- 폼 전송 역할을 하는 디자인 미적용 버튼은 submit 타입을 사용한다(디자인 버튼은 <input type="image">로 사용).

```
<button type="button">열기</button>
<button type="submit">전송</button>
```

4.4 표 엘리먼트

표 엘리먼트를 아래와 같이 배치한다.

```
<table cellspacing="0" border="1" summary="짬뽕은 자장면보다 500원 비싸고 열량이 50kcal 높다">
<caption>자장면과 짬뽕의 가격과 열량 비교</caption>
<col width="100" span="2">
<thead>
<tr>
<th>구분</th>
<th scope="col" abbr="가격">가격(won)</th>
<th scope="col" abbr="열량">열량(kcal)</th>
</tr>
</thead>
<tfoot>
<tr>
<th>계</th>
<td>6,500</td>
<td>650</td>
</tr>
</tfoot>
<tbody>
<tr>
<th scope="row">자장면</th>
<td>3,000</td>
<td>300</td>
</tr>
<tr>
<th scope="row">짬뽕</th>
<td>3,500</td>
<td>350</td>
</tr>
</tbody>
</table>
```

A. <table>

일반적으로 레이아웃을 표현하기 위해 사용하지 않고, 2차원의 격자형 데이터를 표현하기 위해 사용한다.

- cellspacing, border, summary 순서로 애트리뷰트를 선언한다.
- border="1" 애트리뷰트를 선언하여, CSS를 지원하지 않는 환경에서도 표의 테두리가 표시되도록 한다.
- cellpadding="0"은 선언하지 않고, CSS에서 padding:0으로 셀 여백을 초기화한다.
- 표의 요약 내용을 표기해야 할 때 summary 애트리뷰트를 선택적으로 사용한다.

```
<table cellspacing="0" border="1" summary="summary context here">
```

B. <caption>

표의 제목을 표현하기 위해 사용한다.

```
<caption>자장면과 짬뽕의 가격과 열량 비교</caption>
```

표 윗부분의 헤딩 엘리먼트가 표의 제목 역할을 하는 경우에는 생략할 수 있다.

C. <colgroup>

col 엘리먼트를 그루핑하여 디자인을 제어할 때 선언한다. 이 엘리먼트는 선택적으로 사용한다.

```
<colgroup>
   <col width="100">
</colgroup>
<colgroup>
   <col span="2" width="100">
```

```
</colgroup>
```

D. <col>

표 각 열의 너비 지정을 위해 선언한다.

width, span 애트리뷰트를 아래와 같은 순서로 선언한다.

```
<col width="100" span="2">
<col width="100"><col width="100">
<col width="100"><col>
<col width="50%"><col width="50%">
```

width, span 애트리뷰트는 필요에 따라 선택적으로 사용한다.

E. <thead>

표 머리글을 그루핑할 때 선언한다. 이 엘리먼트는 th 엘리먼트만으로 그루핑되지 않으면 선언하지 않는다.

```
<thead>
<tr>
<th>구분</th>
<th scope="col" abbr="가격">가격(won)</th>
<th scope="col" abbr="열량">열량(kcal)</th>
</tr>
</thead>
```

F. <tfoot>

표 바닥글을 그루핑할 때 선언한다. tfoot 애트리뷰트는 thead와 tbody 엘리먼트 사이에 위치해야 한다. 이 애트리뷰트는 선택적으로 사용한다.

```
<thead>
…
</thead>
<tfoot>
<tr>
<th>계</th>
<td>6,500</td>
<td>650</td>
</tr>
</tfoot>
<tbody>
…
</tbody>
```

G. <th>

scope, abbr 순서로 애트리뷰트를 선언한다.

- 표에 셀 제목이 명시되지 않은 경우에도 th 엘리먼트를 필수로 선언하여 의미에 맞는 제목을 명시한다(화면에 표시되지 않도록 CSS로 숨김 처리).
- scope 애트리뷰트는 반드시 선언해야 한다.
- abbr 애트리뷰트는 선택적으로 사용한다.

```
<th scope="col" abbr="가격">음식의 가격(won)</th>
```

참고

scope 애트리뷰트: th 엘리먼트에 동반되는 애트리뷰트로서 현재의 셀이 어느 셀의 제목인지 범위를 명시한다. scope 애트리뷰트의 값으로는 col, colgroup, row, rowgroup이 있다.

abbr 애트리뷰트: th 엘리먼트에 동반되는 애트리뷰트로서 포함하고 있는 콘텐츠를 축약하여 표

기할 수 있을 때 약어를 표기하는 데 사용한다. th 셀의 내용을 반복해서 음성으로 출력하는 도구들은 abbr 애트리뷰트에 표기된 약어를 읽는다.

H. <tbody>

표 본문을 그루핑하기 위해 선언한다. 테이블의 본문(body)이 하나이고 thead나 tfoot이 없을 경우 생략할 수 있다.

```
<tbody>
<tr>
<th scope="row">자장면</th>
<td>3,000</td>
<td>300</td>
</tr>
</tbody>
```

4.5 기타 엘리먼트

A. <a>

href, target, title 순서로 애트리뷰트를 선언한다.

- 새 창으로 페이지를 표시해야 할 때 target 애트리뷰트를 선택적으로 사용한다.
- class로 디자인을 제어할 경우 class 애트리뷰트를 선택적으로 사용한다.
- title 애트리뷰트는 예고 없이 새 창을 표시해야 하거나 이동 경로를 정확히 알 수 없을 때, 또는 브라우저에 독립적으로 툴팁을 표현하기 위해 사용한다.

```
<a href="print.nhn" target="_blank" title="새창">인쇄하기</a>
```

B. <iframe>

iframe은 페이지 성능에 영향을 주기 때문에 가급적 사용하지 않는다.

- 부득이하게 사용해야 할 경우 src, width, height, title, frameborder, marginwidth, marginheight, scrolling 순서로 애트리뷰트를 선언한다.
- 스크린 리더 사용자를 위해 title 애트리뷰트에 제목을 표기한다. 단, iframe 윗부분의 heading 엘리먼트가 iframe의 제목 역할을 할 때는 title 애트리뷰트를 생략할 수 있다.

```
<iframe src="nbox.html" width="410" height="800" title="공지사항 게시판" frameborder="0"
marginwidth="0" marginheight="0" scrolling="no"></iframe>
```

참고

iframe 사용 시 다음과 같은 문제점이 있다.

- iframe을 포함하는 페이지의 도메인과 iframe에서 불러오는 페이지의 도메인이 다르면, 크로스 도메인 설정을 위해 별도의 소스 코드가 추가되어 성능에 영향을 줄 수 있다.
- iframe의 높이값이 콘텐츠에 따라 유동적이어야 한다면 별도의 소스 코드가 추가되어 성능에 영향을 줄 수 있다.
- 검색 엔진에서 iframe의 내용만 추출하여 표시하면 전체 페이지의 레이아웃이 비정상적으로 보일 수 있으며, 주변 맥락(머리글, 바닥글, 메뉴)이 노출되지 않아 검색 엔진 최적화(SEO) 관점에서 적합하지 않다.
- iframe은 가장 마지막으로 로딩되기 때문에 페이지 로딩 초기에는 화면이 비어 보일 수 있다.

접근성 보장을 위해 유관부서와 협의 가능 여부를 판단하여 아래 안 중 하나를 선택할 수 있다.

- <iframe>iframe 미지원 장치에서는 내용을 확인할 수 없습니다.</iframe>
- <iframe></iframe>

C. <img>

src, width, height, title, alt, usemap 순서로 애트리뷰트를 선언한다.

- 이미지 내용과 동일한 값을 alt 애트리뷰트에 표기하여, 이미지를 볼 수 없는 환경(스크린 리더, 이미지 서버 장애, 이미지 표시 하지 않음 설정)에서도 내용을 확인할 수 있게 한다.
- title 애트리뷰트를 선언한 경우에도 alt 애트리뷰트를 함께 선언한다.
- title 애트리뷰트는 alt 애트리뷰트값을 축약하거나 브라우저에 독립적으로 툴팁을 표현하기 위해 사용한다.

```
<img src="logo.gif" wdith="30" height="10" title="고객센터" alt="고객센터, 모든 궁금증이 해결되는 곳" usemap="#help">
```

D. <map>

map 엘리먼트의 name 애트리뷰트를 선언하여 img 엘리먼트의 usemap 애트리뷰트와 같은 이름으로 연결(coupling)한다.

```
<img src="img_main.gif" width="696" height="196" title="한글날이벤트" alt="한글날이벤트, 한글을 글꼴로 나눕니다" usemap="#help">
<map name="help">
<area shape="rect" coords="506,48,608,139" href="#" target="_blank" title="고객센터"
alt="고객센터, 모든 궁금증이 해결되는 곳">
</map>
```

E. <area>

shape, coords, href, target, title, alt 순서로 애트리뷰트를 선언한다.

- title 애트리뷰트를 선언한 경우에도 alt 애트리뷰트를 함께 선언한다.
- target 애트리뷰트는 새 창으로 페이지를 표시해야 할 때 사용한다.
- title 애트리뷰트는 예고 없이 새 창을 표시해야 하거나 이동 경로를 정확히 알 수 없을 때, alt 애트리뷰트값을 축약하거나, 브라우저에 독립적으로 툴팁을 표현하기 위해 사용한다.

```
<area shape="rect" coords="506,48,608,139" href="#" target="_blank" title="새창"
alt="고객센터, 모든 궁금증이 해결되는 곳">
```

5. CSS 코드 작성 규칙

이 장에서는 CSS 코드의 기본 작성 규칙과 인코딩, 선택자, 속성 등의 작성 규칙을 설명한다.

5.1 기본 규칙

A. 영문 소문자 사용

모든 속성은 영문 소문자로만 작성한다.

표 5-1 소문자 작성 예

잘못된 예	올바른 예
.class{Font-Family:AppleGothic}	.class{font-family:AppleGothic}

B. 따옴표 사용 범위

한글 폰트 선언, <string> 데이터 타입, filter 속성의 파라미터값은 작은따옴표(' ')로 감싸며, 그 외의 경우에는 따옴표를 사용하지 않는다.

<url> 데이터 타입에는 작은 따옴표를 사용하지 않는다.

예)

```
filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(src='bg.png',
sizingMethod='scale')
background:url(bg.gif) no-repeat 0 0
content:'Chapter: '
```

표 5-2 따옴표 사용 예

잘못된 예	올바른 예
font-family:돋움	font-family:'돋움'
filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(src="bg.png", sizingMethod="scale")	filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(src='bg.png', sizingMethod='scale')
background:url('bg.gif') no-repeat 0 0	background:url(bg.gif) no-repeat 0 0
content:"Chapter: "	content:'Chapter: '

C. 마지막 세미콜론 사용 지양

마지막 속성의 세미콜론(;)은 삭제한다.

표 5-3 마지막 세미콜론 예

잘못된 예	올바른 예
.class{font-size:12px;color:#000;}	.class{font-size:12px;color:#000}

5.2 들여쓰기

CSS 코드를 작성할 때는 들여쓰기를 하지 않는다.

표 5-4 들여쓰기 사용 예

잘못된 예	올바른 예
#content{border:1px solid #000} #content .class{color:#000}	#content{border:1px solid #000} #content .class{color:#000}

5.3 공백

A. 선택자 간 공백 제거

쉼표로 구분되는 선택자 간 공백은 제거한다.

표 5-5 선택자 간 공백 제거 예

잘못된 예	올바른 예
a:hover,^a:active,^a:focus{text-decoration:underline;}	a:hover,a:active,a:focus{text-decoration:underline;}

B. 속성 간 공백 제거

속성 간 공백은 제거한다.

표 5-6 속성 간 공백 제거 예

잘못된 예	올바른 예
.class p{color:#000;^line-height:18px}	.class p{color:#000;line-height:18px}

C. 중괄호 좌우 공백 제거

중괄호 좌, 우의 공백은 제거한다.

표 5-7 중괄호 좌우 공백 예

잘못된 예	올바른 예
.class p^{color:#000}	.class p{color:#000}
.class p{^color:#000; line-height:18px^}	.class p{color:#000;line-height:18px}

5.4 빈 줄

의미 있는 객체를 구분하기 위하여 코드 그룹 간 1줄씩 빈 줄을 넣을 수 있다. 단, 빈 줄의 간격은 1줄을 초과하지 않게 한다. 빈 줄의 사용은 선택 사항이다.

표 5-8 빈 줄 사용 예

잘못된 예	올바른 예
/* 의미 있는 그룹 1 */ .class1{border:1px solid #000} .class1 img{border:1px solid #000} 빈 줄 빈 줄 /* 의미 있는 그룹 2 */ .class2{border:1px solid #000} .class2 img{border:1px solid #000}	/* 의미 있는 그룹 1 */ .class1{border:1px solid #000} .class1 img{border:1px solid #000} 빈 줄 /* 의미 있는 그룹 2 */ .class2{border:1px solid #000} .class2 img{border:1px solid #000}

5.5 줄 바꿈

선택자 간 줄 바꿈은 허용하지 않는다.

표 5-9 줄 바꿈 사용 예

잘못된 예	올바른 예
.class1, .class2, .class3{color:#000}	.class1,.class2,.class3{color:#000}

5.6 인코딩

폰트 이름이 영문이 아닐 때 이를 브라우저에서 바르게 표현하고, HTML에서 불러온 스타일을 제대로 렌더링하려면 반드시 CSS 인코딩을 선언해야 한다. HTML과 동일한 인코딩을 문서 첫 줄에 공백 없이 선언한다. 작성 예는 다음과 같다.

```
@charset "utf-8";
```

파일을 저장할 때는 반드시 선언한 인코딩과 동일한 인코딩을 선택한다.

5.7 선택자

A. 공통 선택자 사용 지양

공통 선택자 '*'는 웹 페이지의 성능을 떨어뜨리고, Internet Explorer에서는 주석까지 영향을 받을 수 있으므로 사용하지 않는다.

B. id 선택자 범위

id 선택자는 2.2 id 및 class 네이밍 규칙에서 정의한 레이아웃(wrap, header, container, content, footer)에만 사용한다.

C. 교차속성 사용 지양

id와 class, class와 class 간의 교차속성을 사용하지 않는다.

표 5-10 교차속성 사용 예

잘못된 예	올바른 예
.class{width:980px}	.class{width:980px}
.class.bg{background:yellow}	.bg{background:yellow}
.class.bgv1{background:green}	.bgv1{background:green}

참고

교차속성이란 두 선택자의 조합에 의해 속성이 부여되는 경우를 의미한다.

아래 예에서 background:yellow 속성은 .class와 .bg가 동시에 선언될 때만 의미가 있으며, background:green 속성은 .class와 .bgv1이 동시에 선언될 때만 의미가 있다.

```
.class{width:980px}
.class.bg{background:yellow}
.class.bgv1{background:green}
```

Internet Explorer 6에서는 id와 class 간 교차속성을 사용할 경우 두 번째 선언된 교차속성은 무시하는 버그가 있으며, class와 class 간의 교차속성을 잘못 해석하는 버그가 있다. 이와 같은 버그를 방지하기 위해 교차속성을 사용하지 않도록 한다.

교차속성을 사용하지 않으면 코드양을 줄이고 각 class 간 독립성을 유지할 수 있다.

5.8 속성

5.8.1 속성 선언 순서

속성을 선언할 때는 그 쓰임새가 레이아웃과 관련이 큰 것에서 시작하여 레이아웃과 무관한 것 순서로 선언한다. 관련 속성은 대표되는 속성 다음으로 선언하며, 표 5-11에 표기된 순서대로 선언한다.

대표되는 속성 중 (그룹) 표기된 것은 약식속성으로, 약식속성을 선언했을 때 속성 값의 순서와 동일하게 전체속성을 선언하며 표 5-12에 표기된 순서를 참고한다.

표 5-11 속성 선언 순서

순서	의미	대표되는 속성(그룹)	관련 속성
1	표시	display	visibility
2	넘침	overflow	-
3	흐름	float	clear
4	위치	position	top, right, left, bottom, z-index
5	크기	width & height	-
6	간격	margin & padding (그룹)	-
7	테두리	border (그룹)	-
8	배경	background (그룹)	-
9	폰트	font (그룹)	color, letter-spacing, line-height, text-align, text-decoration, text-indent, vertical-align, white-spacing 등
10	기타	-	위에 언급되지 않은 나머지 속성들로 폰트의 관련 속성 이후에 선언하며, 기타 속성 내의 선언 순서는 무관함.

* [속성 선언 순서 기준: 1~6: 레이아웃, 7~8: 테두리/배경, 9: 폰트, 10: 기타]

표 5-12 약식속성의 전체속성 선언 순서

약식속성	전체속성 선언 순서
margin	margin-top, margin-right, margin-bottom, margin-left
padding	padding-top, padding-right, padding-bottom, padding-left
border	border-top, border-right, border-bottom, border-left, border-width, border-top-width, border-right-width, border-bottom-width, border-left-width, border-style, border-top-style, border-right-style, border-bottom-style, border-left-style, border-color, border-top-color, border-right-color, border-bottom-color, border-left-color
background	background-color, background-image, background-repeat, background-attachment, background-position
font	font-style, font-variant, font-weight, font-size, font-family

5.8.2 속성값 축약

CSS 최적화를 위해 다음과 같이 속성값을 축약한다.

표 5-13 속성값 축약 예

축약 전	축약 후	설명
#555555	#555	16 진수 컬러 코드값 동일한 값으로 이루어진 16 진수 컬러 코드값은 세 자릿수로 축약하여 사용한다. 단 IE 전용 속성인 CSS filter 를 사용했을 경우 축약 속성을 인식하지 못하는 오류가 있기 때문에 속성값을 축약하지 않는다.
#ff4400	#f40	
background-position:left center	background-postion:0 50%	위치값 문자로 표현한 위치값은 숫자로 변경한다.
top:0px	top:0	0 의 단위 속성값이 0 일 경우 단위를 표시하지 않는다.
padding:20px 20px 20px 20px	padding:20px	동일한 속성값 상, 우, 하, 좌의 속성값이 동일하면 축약한다.
margin:0 auto 0 auto	margin:0 auto	
padding:20px 30px 50px 30px	padding:20px 30px 50px	
border-color:#ccc #eee #ccc #eee	border-color:#ccc #eee	

참고

문자로 표현된 위치값을 숫자로 변환할 때 다음과 같이 한다.

top, left → 0

right, bottom → 100%

5.8.3 약식 속성 사용 범위

background와 border는 약식 속성을 우선적으로 사용하며, font 약식 속성은 사용하지 않는다.

A. background

속성값은 background-color, background-image, background-repeat, background-attachment, background-position 순서로 선언한다.

배경 스타일 속성을 초기 선언할 때는 반드시 background 단일 속성을 사용하며, 이후 배경의 부분적인 속성이 변경될 경우 background 관련 속성(background-attachment, background-color, background-image, background-position, background-repeat)을 선언한다.

작성 예는 다음과 같다.

```
.class {background:#ccc url(bg.gif) no-repeat 0 0}
.class_v1 {background-position:0 -50px}
.class_v2 {background-position:0 -100px}
```

B. border

속성값은 border-width, border-style, border-color 순서로 선언한다.

테두리 스타일 속성을 초기 선언할 때는 반드시 border 단일 속성을 사용하고, 이후 테두리의 부분적인 속성이 변경될 경우 border 관련 속성(border-width, border-style, border-color)을 선언한다.

작성 예는 다음과 같다.

```
.class {border:1px solid #ccc}
.class_v1 {border-color:#666}
.class_v2 {border-style:dotted}

.class2 {border-top:1px solid #ccc}
.class2_v1 {border-top-color:#666}
.class2_v2 {border-top-style:dotted}
```

테두리의 상, 우, 하, 좌 스타일이 2개 이상 다르면, 공통 스타일을 약식 속성으로 선언한 후 다른 부분은 관련 속성으로 선언한다.

표 5-14 테두리 스타일이 2개 이상 다를 경우 약식 속성 선언의 예

잘못된 예	올바른 예
.class{border:1px solid #ddd; border-bottom:1px solid #eee; border-left:1px solid #eee}	.class{border:1px solid; border-color:#ddd #ddd #eee #eee}
.class{border-top:1px solid #ddd; border-right:1px dotted #ddd;border-bottom:1px solid #eee; border-left:1px dotted #eee}	.class{border:1px; border-style:solid dotted; border-color:#ddd #ddd #eee #eee}

C. font

폰트 스타일은 약식 속성으로 선언하지 않는다.

폰트 스타일을 약식 속성으로 선언하면 다음과 같은 문제가 발생한다.

아래와 같이 선언하면, font-weight:bold는 상속되지 않고 font 속성의 디폴트값인 font-weight:normal로 변경되기 때문에 불필요한 속성값을 선언해야 하는 문제가 발생한다.

```
.class{font-weight:bold;font-size:12px; font-family:dotum}
.class p{font:15px gulim}
```

결국, class p의 폰트 스타일은 아래와 같아진다.

```
.class p{font-family:gulim;font-style:normal;font-variant:normal;font-weight:normal;
font-size:15px;line-height:normal}
```

5.9 z-index

z-index 속성의 속성값의 범위는 최소 10, 최고 1000이며, 10 단위로 증감한다. 단, 10 단위 사이의 예외 변수가 발생하면 1 단위 값을 지정할 수 있다.

다음은 기본 가이드로, 각 서비스의 z-index값은 상황에 맞게 선택적으로 선언한다.

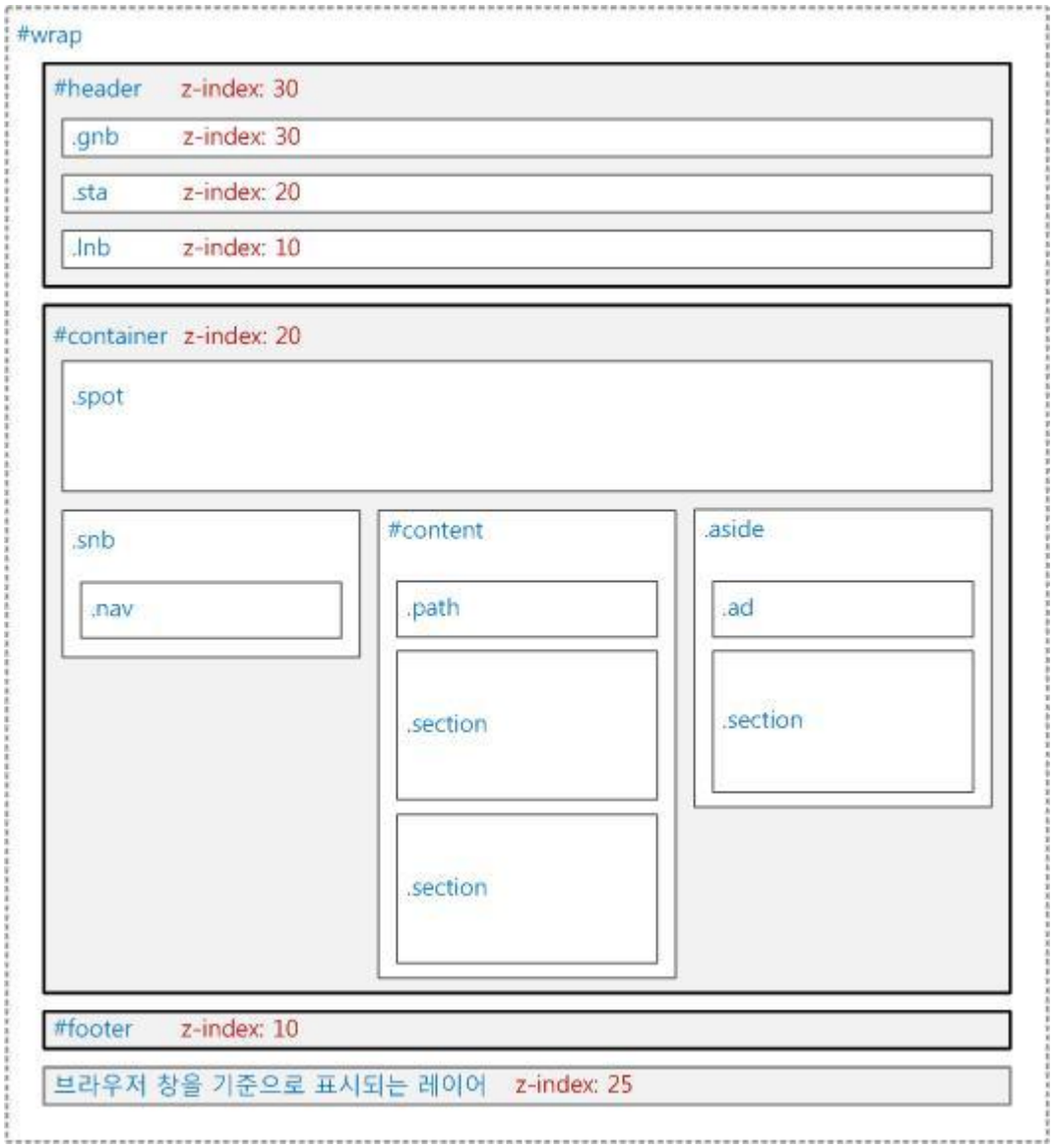


그림 5-1 z-index 권장 선언값

5.10 핵

핵(hack)은 가능한 한 사용하지 않는다. 불가피하게 사용해야 할 때는 표 5-15에 제시된 핵만 사용하는 것을 원칙으로 한다.

표 5-15 사용 가능한 핵

<table>
<tr><th>브라우저</th><th colspan="3">표준 DTD</th><th colspan="3">Quirks Mode</th></tr>
<tr><td>IE5.5</td><td rowspan="3">S{*P:V}</td><td rowspan="2">S{_P:V}</td><td>S{_P /**/:V}</td><td rowspan="4">S{_P:V}</td><td rowspan="3"></td><td rowspan="3"></td></tr>
<tr><td>IE6</td><td></td></tr>
<tr><td>IE7</td><td>S,x:-moz-any-link, x:default {P:V}</td><td>*+html S{P:V}</td></tr>
<tr><td>IE8</td><td></td><td></td><td></td><td rowspan="3">S,x:-moz-any-link, x:default{P:V}</td></tr>
<tr><td>Firefox</td><td rowspan="3">:root S{P:V}</td><td>S,x:-moz-any-link, x:default{P:V}</td><td rowspan="3"></td><td rowspan="3"></td><td>:root S{P:V}</td></tr>
<tr><td>Safari/ Chrome</td><td rowspan="2"></td><td rowspan="2"></td></tr>
<tr><td>Opera</td></tr>
</table>

[약어 표기 - S(Selector): 선택자, P(Property): 속성, V(Value): 속성값]

5.11 미디어 타입

미디어 타입은 기본 CSS 파일의 가장 아랫부분에 선언한다. 미디어 타입을 위한 별도의 CSS 생성은 허용하지 않는다.

미디어 타입 선언을 위한 명령문(중괄호 포함)은 다음과 같이 세부 스타일을 지정하는 명령문과 줄로 구분한다.

```
...
...
/* For Print */
@media print{
#header,
.snb,.aside{display:none}
}
```

참고

미디어 타입은 각종 미디어(프린터, 모바일, 보조공학기기 등)에 대응하는 별도의 CSS 코드를 작성할 수 있게 한다. CSS 2.1 Specification에 따라 대응 가능한 미디어 타입은 아래와 같으며, 가장 범용적으로 사용하는 타입은 print 타입이다.

- all (모든 출력 장치)
- aural (음성 출력)
- braille (점자 출력)
- handheld (포켓, 모바일)
- print (인쇄)
- projection (투사 장치)

5.12 CSS 선언 타입

CSS 선언 타입은 크게 세 가지로 분류하며, 용도에 맞게 사용한다.

A. External type

CSS를 선언하는 가장 기본적인 방식으로, CSS 파일이 별도로 존재하는 형태이다. link 엘리먼트를 통해 HTML과 CSS 파일을 연결한다. 작성 예는 다음과 같다.

```
<link rel="stylesheet" type="text/css" href="../css/service_name.css">
```

B. Internal(embedded) type

HTML 파일의 head 안에 스타일을 선언하는 방식으로, 단발성 페이지의 CSS 분량이 작을 경우 사용한다. 작성 예는 다음과 같다.

```
<head>
…
<style type="text/css">
…
</style>
</head>
```

C. Inline type

HTML의 개별 엘리먼트에 style 속성을 이용하여 스타일을 선언하는 방식으로, 속성값이 동적으로 변경되어야 하는 경우 사용한다. 속성값에 사용되는 %와 같은 특수기호는 entity 코드로 변환하지 않는다. 작성 예는 다음과 같다.

```
<div style="top:0; left:50%;">
…
</div>
```

5.13 주석

5.13.1 기본 형식

CSS 주석의 시작과 종료 주석은 아래와 같이 표기하며, 기본 형식에 맞게 작성한다.

```
시작 주석 /* 주석 내용 */
종료 주석 /* //주석 내용 */
```

- 주석 기호와 주석 내용 사이에는 반드시 공백 한 칸이 있어야 한다.
- 주석 기호와 주석 내용 사이에는 다른 기호가 올 수 없다.
- 주석 기호와 주석 내용 사이의 줄 바꿈은 허용하지 않는다. 단, 주석 내용 간 줄 바꿈은 허용한다.
- 시작과 종료 주석의 주석 내용은 동일해야 한다.

주의

마크업과 개발의 편의를 위해 작성한 주석은 실제 서비스를 적용할 때 반드시 삭제한다. 단, 작성자 정보는 삭제하지 않는다.

5.13.2 작성자 정보 표기

작성자 정보에는 소속 부서, 영문 이름 이니셜, CSS 파일 생성 날짜(YYMMDD 형식)를 작성하며, 유지보수만 담당하는 경우라도 모두 기입한다. 작성자 정보 밑에는 빈 줄(한 줄)을 두어 스타일을 선언하는 문장과 구분되도록 한다.

```
/* NHN Web Standard 1Team JJS 090707, JJJ 090815 */
빈 줄
```

5.13.3 의미 있는 그룹 영역의 주석 표기

의미 있는 객체를 구분하기 위한 주석은 영역의 윗부분에 시작 주석만 표기한다. 초기화와 레이아웃 스타일 그룹을 제외한 의미 있는 그룹 영역의 주석 표기는 선택 사항이다.

작성 예는 다음과 같다.

```
/* 마이지식 SNB */
.my_snb{width:182px;}
.my_snb li .num{padding-left:4px;color:#919190;font-size:11px;letter-spacing:0;}
.my_snb li.on a,.my_snb li.on .num{color:#259e0b;}
.my_snb li a{color:#424242;}
```

A. 초기화 스타일 그룹

CSS 초기 파일에 따라 초기화 속성은 /* Common */으로 그루핑한다.

```
/* Common */
body,p,h1,h2,h3,h4,h5,h6,ul,ol,li,dl,dt,dd,table,th,td,form,fieldset,legend,input,textare
a,button,select{margin:0;padding:0}
body,input,textarea,select,table{font-family:'돋움',Dotum,AppleGothic,sans-serif;font-
size:12px}
```

B. 레이아웃 스타일 그룹

레이아웃을 위한 스타일 선언 시 /* Layout */으로 그루핑한다.

```
/* Layout */
#wrap{…}
```

```
#header{…}
#container{…}
#footer{…}
```

5.13.4 수정 또는 삭제된 부분의 주석 표기

SVN 로그로 수정/삭제 내용 전달이 어려울 때, 수정 또는 삭제된 부분의 시작과 끝에 주석을 표기한다. 주석 작성 방법은 3.5 주석을 참조한다.

```
/* 091120 */
...
/* //091120 */


/* 091120 수정된 내용 */
...
/* //091120 수정된 내용 */
```

```
/* 091120 삭제
...
//091120 삭제 */
```

이때, 주석으로 포함된 영역 안에 다른 주석이 들어가지 않도록 주의한다.

표 5-16 삭제된 부분의 주석 표기 예

잘못된 예	올바른 예
/* 091120 삭제	/* 091120 삭제
…	…
… /* 090701 수정 */	…
…	…
//091120 삭제 */	//091120 삭제 */

수정 또는 삭제된 부분의 주석 표기는 선택 사항이다.

5.14 파일 분기

파일은 독립된 한 서비스 내에 반드시 하나의 파일을 생성하며, 분기를 허용하는 경우는 다음과 같다.

- 한 서비스가 사용자 화면/어드민으로 구성되어 전체 레이아웃이 다를 경우
- 개편 시 개발상의 이슈로 이전에 분리되어 있던 CSS를 합칠 수 없는 경우
- 한 서비스 내에 단발성 페이지로 존재하나 CSS를 embed하기 어려운 경우

5.15 초기 파일

CSS를 새로 작성할 때는 아래 파일을 기본으로 한다. 초기 파일에는 스타일 속성 초기화 문장이 포함되어 있으며, 초기화 문장은 변경할 수 없다.

```
@charset "utf-8";
/* NHN Web Standard 1Team JJS 090707, JJJ 090815 */

/* Common */
body,p,h1,h2,h3,h4,h5,h6,ul,ol,li,dl,dt,dd,table,th,td,form,fieldset,legend,input,textare
a,button,select{margin:0;padding:0}
body,input,textarea,select,button,table{font-family:'돋움',Dotum,AppleGothic,sans-
serif;font-size:12px}
img,fieldset{border:0}
ul,ol{list-style:none}
em,address{font-style:normal}
a{text-decoration:none}
a:hover,a:active,a:focus{text-decoration:underline}
```

6. 웹 접근성 보장 방법

이 장에서는 웹 접근성을 보장하도록 마크업하는 방법을 설명한다.

6.1 웹 접근성 정의

웹 접근성의 정의는 기관별로 약간씩 차이가 있으나, 이 문서에서는 보편적 접근성(Universal Accessibility)으로 정의한다.

보편적 접근성이란, 모든 사람이 다양한 조건의 환경에서 다양한 장치를 이용하여 웹 콘텐츠에 접근할 수 있도록 보장하는 것을 의미한다. 보편적 접근성을 보장하게 되면 장애를 지닌 사람부터 일시적으로 불리한 조건을 갖게 된 정상인에 이르기까지 편리하게 웹 콘텐츠에 접근할 수 있다. 또한, 네트워크 속도가 느리거나 모바일 기기를 사용하는 경우, 다양한 운영체제와 웹 브라우저를 사용하는 경우 등의 환경 조건에 관계없이 웹 콘텐츠에 접근할 수 있어 웹의 사용성을 증가시킨다.

6.2 의미에 맞는 마크업

의미에 맞게 마크업을 하는 것은 웹표준 기반의 마크업을 진행할 때 숙지해야 하는 내용임과 동시에 웹 접근성을 보장하는 가장 기본적이고 중요한 방법에 해당한다. 4장에서 다루고 있는 HTML 엘리먼트 작성 규칙에 따라 각 엘리먼트가 쓰임에 맞게 사용되어야 하며, 이 장에서는 특별히 강조되어야 하는 부분만 언급한다.

6.2.1 적절한 헤딩 엘리먼트 사용

문서 구조 파악이 용이하도록 h1~h6의 헤딩 엘리먼트를 순차적으로, 적절한 위치에 사용한다.

적절하게 사용된 헤딩 엘리먼트를 통해 문서의 목차를 자동으로 생성할 수도 있으며, 스크린 리더에서 지원하는 헤딩 간 이동을 통해 사용자가 도달하고자 하는 정보에 더 빠르게 접근할 수 있다.

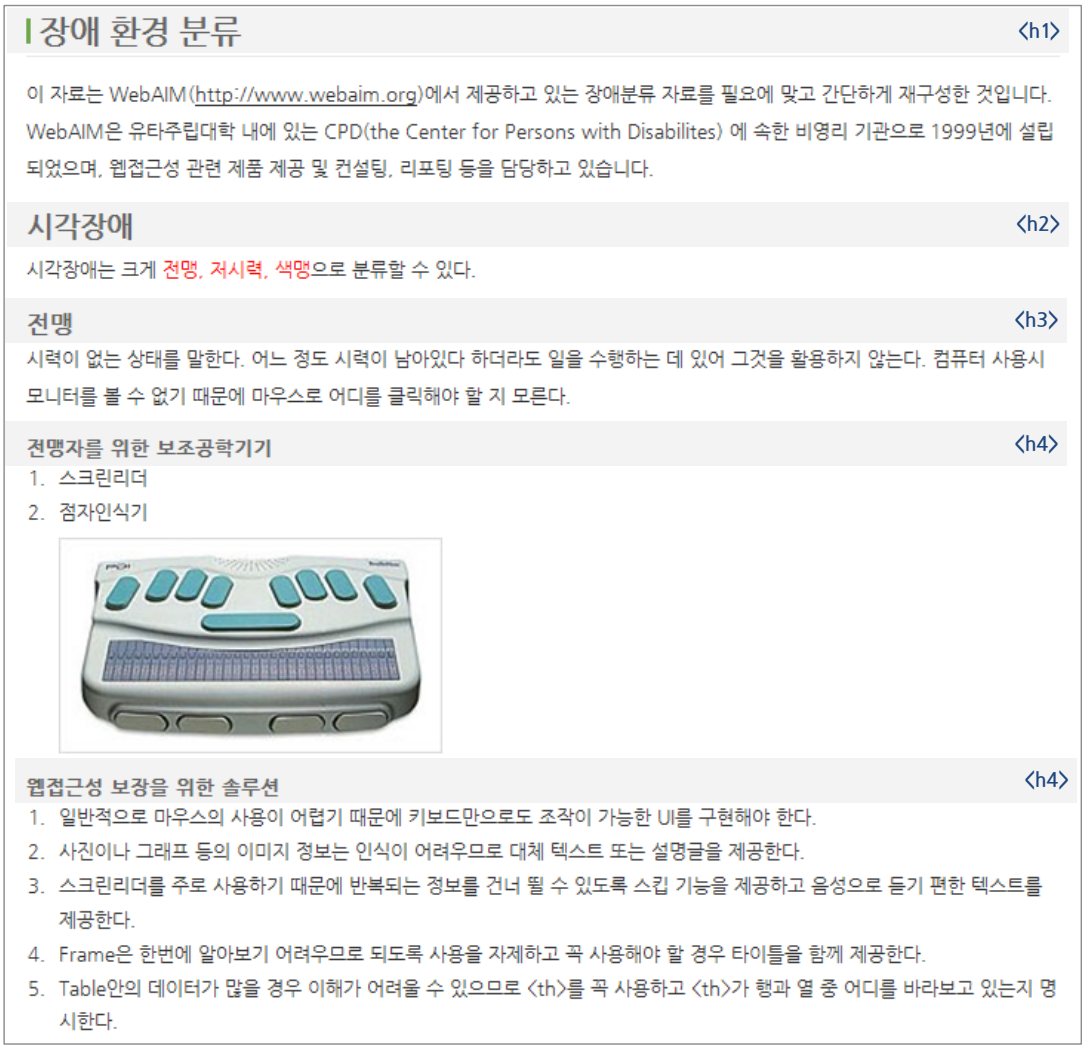


그림 6-1 알맞은 헤딩 엘리먼트를 사용한 예

6.2.2 적절한 목록 엘리먼트 사용

순차/비순차/정의 목록을 쓰임새에 맞게 사용하면 CSS가 적용되지 않은 화면에서도 각 목록의 의미를 확실히 전달할 수 있으며 스크린 리더에서는 전체 목록의 개수, 시작과 끝을 파악할 수 있다.

표 6-1 목록 형태에 따른 출력 형태

실제 화면	마크업	CSS 미적용	음성 출력
·비순차 목록1 ·비순차 목록2	 - 비순차 목록 1 - 비순차 목록 2 	• 비순차 목록1 • 비순차 목록2	목록 시작 개수 2(1/1) 비순차 목록 1 비순차 목록 2 목록 끝
1 순차 목록1 2 순차 목록2	 <li class="num1">순차 목록 1 <li class="num2">순차 목록 2 	1. 순차 목록1 2. 순차 목록2	목록 시작 개수 2(1/1) 일 순차 목록 1 이 순차 목록 2 목록 끝
정의란? 정의내용 정의 내용	<dl> <dt>정의란?</dt> </dd>정의내용 정의 내용</dd> </dl>	정의란? 정의내용 정의 내용	목록 시작 개수 1(1/1) 정의 정의내용 정의 내용 목록 끝

6.2.3 thead, tfoot, tbody 엘리먼트 사용

thead, tfoot, tbody 엘리먼트를 사용하여 머리글과 바닥글을 분리한다. 본문이 길면 스크롤이 가능하게 하고 인쇄할 때 머리글과 바닥글은 반복적으로 인쇄할 수 있다.

```
<table summary="올림픽 종목별 메달 성적">
<caption>종목별 성적</caption>
<thead>
<tr>
<th>종목</th> <th>금</th> <th>은</th> <th>동</th>
</tr>
</thead>
<tfoot>
<tr>
<th>합계</th> <td>9</td> <td>4</td> <td>2</td>
</tr>
</tfoot>
<tbody>
<tr>
<th>태권도</th> <td>2</td> <td>1</td> <td>1</td>
</tr>
<tr>
<th>양궁</th> <td>4</td> <td>1</td> <td>1</td>
</tr>
</tbody>
</table>
```

tfoot은 표에 바닥글 정보가 있을 때만 사용한다.

6.2.4 th 엘리먼트 사용

th 엘리먼트를 사용하여 표의 데이터값이 무엇을 의미하는지 명확하게 전달한다. 디자인에 머리글 정보가 표현되어 있지 않더라도 반드시 넣어 준다.

WTI 10-0 (02.04)	73.14	▼ 3.84
두바이유현물 (02.04)	74.78	▼ 1.22
금 (온스) (02.05)	1,066.53	▼ 15.56
필라델피아 반도 (02.04)	312.53	▼ 15.22

그림 6-2 머리글 정보가 표현되어 있지 않은 표

다음과 같이 머리글 정보를 넣는다.

```
<table>
<caption>해외 증시 현황</caption>
<thead class="blind">
<tr>
<th scope="col">종목명</th>
<th scope="col">현재가</th>
<th scope="col">전일비</th>
</tr>
</thead>
</tbody>
<tr>
<td>WTI 10-0 (12.17)</td>
<td>72.65</td>
<td><img src="..." width="..." height="..." alt="하락">0.01</td>
</tr>
...
```

6.2.5 scope 애트리뷰트 사용

scope 애트리뷰트는 지정된 셀의 머리글 정보를 제공하여 행과 열의 데이터값을 쉽게 매칭하고 이해할 수 있도록 한다. 아래의 예제 코드에서 th에 선언된 scope="col" 애트리뷰트는 같은 열의 셀에 대한 머리글 정보를 제공하며, td에 선언된 scope="row" 애트리뷰트는 같은 행의 셀에 대한 머리글 정보를 제공한다.

```
<table summary="현재개봉 영화들의 예매순위입니다. 순위변동은 지난 1시간 대비 결과를 나타냅니다." >
<caption>개봉중 영화들의 예매순위</caption>
<thead>
<tr>
<th scope="col">순위</th>
<th scope="col">영화이름</th>
<th scope="col">순위변동</th>
</tr>
</thead>
</tbody>
<tr>
<td scope="row">1위</td>
<td>디워</td>
<td>2단계상승</td>
</tr>
...
```

6.3 논리적인 순서 보장

논리적인 순서에 맞게 마크업을 한다는 것은 문서의 흐름과 동일하게 마크업 순서를 결정하는 것이다. 키보드 내비게이팅을 하거나 스크린 리더를 통해 웹을 접근하는 사용자는 마크업 순서가 논리적이지 못하면 원하는 콘텐츠에 도달하지 못하거나 문서를 이해하는 데 어려움을 겪게 된다.

표 6-2 논리적인 순서의 마크업 예

잘못된 예	올바른 예
1 수준 메뉴 1	1 수준 메뉴 1
1 수준 메뉴 2	2 수준 메뉴 1-1
1 수준 메뉴 3	2 수준 메뉴 1-2
2 수준 메뉴 1-1	1 수준 메뉴 2
2 수준 메뉴 1-2	1 수준 메뉴 3

6.4 대체 텍스트 제공

6.4.1 이미지 대체 텍스트 제공

이미지 내용과 동일한 값을 alt 애트리뷰트에 표기하여, 이미지를 볼 수 없는 환경(스크린 리더, 이미지 서버 장애, 이미지 표시하지 않음 설정)에서도 내용을 확인할 수 있게 한다.

다음은 이미지 대체 텍스트를 제공하는 예이다.

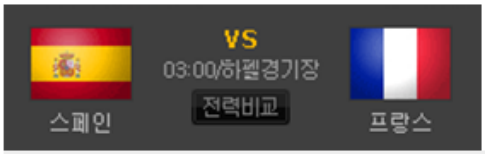
〈img〉


```
<img src="h2_recom.gif" width="83" height="16" alt="네이버 추천 정보">
```

〈input type="image"〉


```
<input type="image" src="btn_sch.gif" alt="검색">
```

〈map〉




```
<img src="vs.gif" width="244" height="74" alt=”스페인과 프랑스의 전력비교” usemap="#vs">
<map name="vs" id="vs">
<area shape="rect" coords="9,7,69,69" href="spa.nhn" alt="스페인" >
<area shape="rect" coords="173,8,232,69" href="fra.nhn" alt="프랑스" >
<area shape="rect" coords="95,43,146,61" href="vs.nhn" alt="전력비교" >
</map>
```

6.4.2 텍스트가 없는 정보성 이미지의 대체 텍스트 제공

텍스트가 없는 정보성 이미지에도 이미지가 나타내는 정보를 alt 애트리뷰트값에 표기한다. 정보성 이미지와 매칭되는 대체 텍스트값은 표 B-4 공통 대체 텍스트 예약어를 참조한다.

6.4.3 텍스트가 많은 이미지의 대체 텍스트 제공

텍스트가 많은 이미지의 경우, 이미지를 배경으로 처리하고 텍스트는 배경으로 처리된 이미지 뒤에 숨긴다.

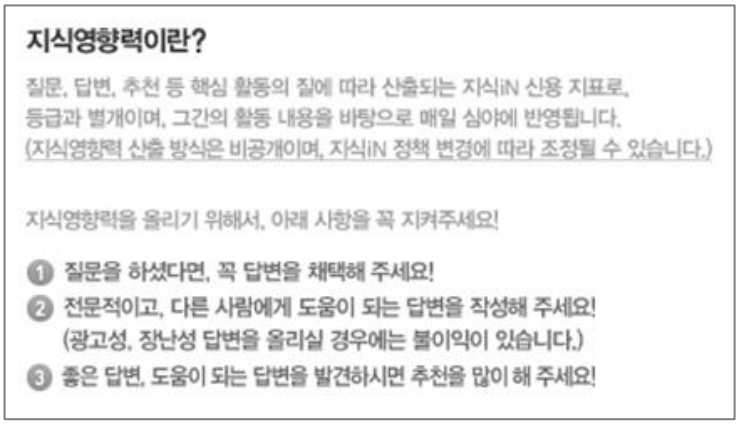


그림 6-3 텍스트가 많은 이미지

```
<style type="text/css">
.kin{position:relative; width:375px; height:207px;}
.kin .kin_img{position:absolute; left:0; top:0; z-index:10; width:375px; height:207px;
background:url(tx_kin.gif) no-repeat 0 0;}
.kin .kin_cont{overflow:auto; width:100%; height:207px;}
</style>

<div class="kin">
   <div class="kin_img"></div>
   <div class="kin_cont">
      <h3>지식 영향력이란?</h3>
      <p>질문, 답변, 추천 등 핵심 활동의 질에 따라 산출되는 지식iN의 신용 지표로, 등급과 별개이며 그간의
      활동 내용을 바탕으로 매일 심야에 반영됩니다. (지식 영향력 산출 방식은 비공개이며, 지식iN 정책
      변경에 따라 조정될 수 있습니다.)</p>
      <p>지식영향력을 올리기 위해서, 아래 사항을 꼭 지켜주세요! </p>
      <ol>
      <li>질문을 하셨다면, 꼭 답변을 채택해주세요!</li>
      <li>전문적이고, 다른 사람에게 도움이 되는 답변을 작성해주세요! (광고성, 장난성 답변을 올리실 경우
      불이익이 있습니다.)</li>
      <li>좋은, 도움이 되는 답변을 발견하시면 추천을 많이 해주세요!</li>
      </ol>
   </div>
</div>
```

6.4.4 추가 정보 필요 시 title 애트리뷰트 사용

title 애트리뷰트는 html, head, title, base, basefont, meta, script, param을 제외한 모든 엘리먼트에 사용할 수 있으며, 일반적으로 브라우저에서 툴팁으로 표시된다.

A. 폼 엘리먼트

input, select, textarea와 같은 폼 엘리먼트에 기본적으로 label 엘리먼트가 부여되지 않은 경우, title 애트리뷰트를 이용하여 해당 엘리먼트에 대한 추가 정보를 표기할 수 있다.

```
<input type="password" value="" title="비밀번호를 입력해주세요" >
<input type="checkbox" name="2" id="check_id2" title="빨간박스" >
<textarea id="babo5" rows="2" cols="20" title="약관" >제 1조 1항 </textarea>
<select title="포탈선택" >
<option value="1">선택하세요</option>
<option value="2">네이버</option>
<option value="3">다음</option>
```

```
</select>
```

B. iframe

부득이하게 iframe을 사용해야 한다면 title에 iframe을 설명하는 내용을 표기한다. 단, iframe 내용을 설명하는 헤딩 엘리먼트가 별도로 있으면 title 애트리뷰트를 선언하지 않는다.

```
<iframe src="sbox.html" title="검색창 스타일 미리보기"></iframe>
```

참고

Internet Explorer 7 이하의 브라우저에서는 alt 애트리뷰트에 입력한 텍스트가 툴팁으로 표시되는 버그가 있다. 대체 텍스트의 문장이 길면 툴팁 내용이 길어져서 사용성을 해칠 우려가 있으므로 title 애트리뷰트를 이용해 간단한 툴팁으로 변경한다.

Internet Explorer 7 이하에서 alt 애트리뷰트만 있는 경우

아래와 같이 alt 애트리뷰트에 설정된 긴 문장이 툴팁에 그대로 드러난다.

```
<img src="h_deditor.gif" width="800" height="307" alt="지식iN을 더욱 풍요롭고 유익한
지식공유 공간으로 만드는 자발적인 사용자 그룹 디렉토리 에디터. 지식iN은 디렉토리 에디터분들의
```


```
열정과 애정이 있기에 특별합니다.">
```

Internet Explorer 7 이하에서 alt와 title을 동시에 넣은 경우

아래와 같이 title 애트리뷰트를 사용하여 툴팁을 간단하게 만든다.

```
<img src="h_deditor.gif" width="800" height="307" title="디렉토리 에디터"
alt="지식iN을 더욱 풍요롭고 유익한 지식공유 공간으로 만드는 자발적인 사용자 그룹 디렉토리 에디터.
지식iN은 디렉토리 에디터분들의 열정과 애정이 있기에 특별합니다.">
```

6.5 키보드 내비게이팅 환경 보장

키보드 내비게이팅은 마우스를 사용할 수 없는 환경(장치 없음, 시각 장애, 지체 장애)에서 웹을 이용할 수 있는 가장 기본적인 방법이다. 따라서, 모든 콘텐츠는 키보드로 접근이 가능해야 하며 마우스 의존적인 이벤트 핸들러는 사용하지 않는다.

A. accesskey 애트리뷰트를 이용한 단축키 제공

accesskey 애트리뷰트는 지정된 단축키를 실행할 경우 해당 엘리먼트로 포커스를 이동한다. 중요한 기능에 단축키를 제공하면 키보드 접근성이 좋아진다. accesskey 애트리뷰트의 사용은 선택 사항이다.

accesskey 애트리뷰트가 사용할 수 있는 엘리먼트는 a, area, button, input, label, legend, textarea이다. accesskey 사용 예는 다음과 같다.

```
<input type="text" name="search" id="search" title="검색어입력" accesskey="S">
<input type="text" name="uid" id="uid" title="로그인ID" accesskey="L">
```

B. 브라우저별 accesskey 키 조합

브라우저에서 accesskey로 지정된 단축키를 사용하려면 다음과 같이 브라우저별 키 조합에 맞게 사용한다.

표 6-3 accesskey 인식을 위한 브라우저별 키 조합

운영체제	브라우저	키 조합
Windows	Internet Explorer	alt + accesskey
	Firefox	alt + shift + accesskey
	Safari	alt + accesskey
	Opera	shift + esc + accesskey
Macintosh	Firefox	command + accesskey
	Safari	command + accesskey
	Opera	shift + esc + accesskey

C. 웹 브라우징 도구와의 충돌

웹 사이트에서 제공하는 접근키와 웹 브라우징 도구(브라우저, 스크린 리더 포함)의 단축키가 충돌하면 웹 사이트에서 제공하는 접근키가 우선권을 가진다. 따라서 겹치도록 설정되어 있다 하더라도 사실상 충돌하지는 않는다. 하지만 겹치는 설정은 피해야 한다. 이들이 겹치는 상황에서 사용자가 웹 브라우징 도구의 단축키를 사용하려고 하면 의도하지 않은 상황(웹 사이트 접근키가 실행되는)이 발생하기 때문이다.

웹 브라우징 도구와 겹치지 않아서 단축키로 사용하기에 안전한 것으로 판단되는 권장 접근키 목록은 다음 표와 같다.

표 6-4 단축키 사용이 가능한 권장 접근키

분류	키

영문키	C, J, K, L, M, O, P, Q, R, S, U, X, Y
숫자키	1, 2, 3, 4, 5, 6, 7, 8, 9, 0
기타	` - [ ] ; ' , . /

D. 권장 단축키

현재 네이버 메인에 적용 중인 단축키를 기준으로 검색 창과 로그인의 단축키 제공을 권장한다.

표 6-5 사용을 권장하는 단축키

분류	접근키	의미
검색 창	S	Search
로그인	L	Login

6.6 스킵 내비게이션 제공

스킵 내비게이션은 스크린 리더를 통해 웹에 접근할 때 서비스 전체에 공통으로 사용되는 콘텐츠를 매번 반복해서 읽지 않도록 한다. 다음과 같이 콘텐츠의 가장 윗부분에 스킵 내비게이션을 사용한다.

그림 6-4 스킵 내비게이션 사용 예

```
<style type="text/css">
.blind{position:absolute;top:0;left:0;width:1px;height:1px;overflow:hidden;visibility:hid
den;font-size:0; line-height:0}
</style>

<div id="header">
   <a href="#content" class="blind">본문 바로가기</a>
</div>
<div id="content">
   본문
</div>
```

6.7 생략된 제목 표시

문서를 올바르게 이해할 수 있도록 생략된 제목을 표시한다. 일반 사용자는 활성화된 디자인을 통해 제목을 인식할 수 있지만, 이미지를 볼 수 없는 상황, 특히 스크린 리더를 사용하는 시각 장애인은 생략된 제목을 인식할 수 없다. 따라서 HTML 구조만으로 제목을 파악할 수 없는 내용일 때는 반드시 제목을 표시한다.

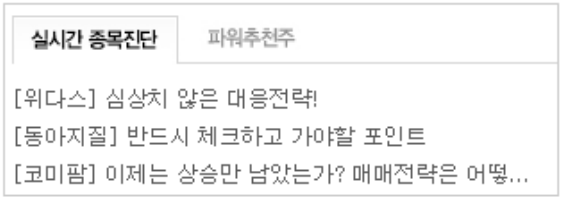


그림 6-5 탭 메뉴와 목록으로 구성된 콘텐츠

```
<style type="text/css">
.blind{position:absolute;top:0;left:0;width:1px;height:1px;overflow:hidden;visibility:hid
den;font-size:0; line-height:0}
</style>

<ul>
<li class="on"><a href="…">실시간 종목 진단</a></li>
<li><a href="…">파워추천주</a></li>
</ul>
<div id="…">
   <h3 class="blind"> 실시간 종목 진단 </h3>
   <ul>
   <li><a href="…">[위다스] 심상치 않은 대응전략!</a></li>
   <li><a href="…">[동아지질] 반드시 체크하고 가야할 포인트</a></li>
   <li><a href="…">[코미팜] 이제는 상승만 남았는가? 매매전략은 어떻게...</a></li>
   </ul>
</div>
<div id="…">
…
```

부록 A. 코딩 시 참고 사항

부록 A에서는 웹표준 기반의 마크업, 크로스 브라우징 범위, 폴더 생성 방법, 플래시 사용 시 유의점, 나눔글꼴 웹폰트 사용 방법, PNG 이미지 사용 방법을 설명한다.

A.1 웹표준 기반의 마크업

A.1.1 table 기반의 마크업

웹표준 기반의 마크업이 도입되기 이전에는 구조와 표현을 모두 HTML 파일에 표현했다. 이를 위해 아래와 같은 방법을 사용했다.

- 구조와 표현을 모두 넣기 위해 Table 엘리먼트를 사용
- 표현을 위한 Table 엘리먼트를 추가하거나 중첩 사용
- 정밀한 표현을 위해 의미 없는 투명 이미지를 사용

이런 마크업 방법을 사용하면 HTML 파일의 용량이 커져서 유지보수 비용이 증가하고 효율성도 떨어진다. 사용자는 의미 없는 콘텐츠를 전달받아야 하며, 다양한 환경에서 접근할 수도 없다. 이러한 문제점들을 해결하기 위해 제안된 마크업 방법이 웹표준 기반의 마크업이다.

A.1.2 웹표준 기반의 마크업

웹표준 기반 마크업이 Table 기반의 마크업과 차별화되는 점은 다음과 같다.

W3C 표준에 근거한 마크업

표준에 근거한 HTML과 CSS 마크업은 향후 웹 브라우저 호환성을 보장받을 수 있다.

의미에 맞는 HTML 엘리먼트를 사용하여 문서 구조 마크업

웹 문서의 내용을 HTML 엘리먼트의 의미만으로 구조화, 선형화하여 정보를 전달할 수 있다. 따라서 다양한 웹 브라우저와 장치에서 읽을 수 있으며, 화면 크기 등에 따라 디자인 정보를 가진 CSS 파일만 수정하면 One-Source Multi Device가 가능하다. 또한 웹 접근성이 높아져 언제, 어디서, 누구나, 어떤 디바이스, 어떤 응용 프로그램을 이용하더라도 동일한 정보를 제공받을 수 있다.

구조와 표현을 분리하여 마크업

기존에는 구조와 표현이 분리되어 있지 않아서 디자인 정보만 수정하고 싶을 때도 전체를 수정해야 했다. 하지만 웹표준 기반의 마크업에서는 HTML은 문서의 메타데이터 정보를, CSS는 문서의 디자인 정보를 포함하도록 분리되어 있다. 디자인 정보를 수정할 때는 CSS 파일만 수정하면 되므로 유지보수가 한결 쉬워졌다. 또한, HTML 문서의 table 중첩 사용이 없어져 용량이 현저히 줄어들기 때문에, 로딩 시간을 단축할 수 있고 HTML 소스 코드의 재사용성이 높아진다.

A.2 크로스 브라우징 범위

크로스 브라우징 범위는 표 A-1과 같으며, Internet Explorer를 제외한 나머지 브라우저는 최신 버전을 기준으로 한다.

표 A-1 크로스 브라우징 범위

운영체제	브라우저
Windows XP	Internet Explorer 6.x
	Internet Explorer 7.x
	Internet Explorer 8.x
	Firefox
	Safari
	Chrome
	Opera
Macintosh	Firefox
	Safari

A.3 플래시 사용 시 유의점

웹 페이지에 플래시가 삽입될 때 다른 콘텐츠가 영향을 받지 않도록 플래시의 wmode를 확인한다.

표 A-2 wmode별 특징

wmode	설명
window	모든 HTML 객체보다 우선순위가 높다. swf 배경을 표현하며, wmode 중 애니메이션 성능이 가장 뛰어나다.
opaque	HTML 객체와의 z-index 를 조절할 수 있고, swf 배경을 표현한다.
transparent	HTML 객체와의 z-index 를 조절할 수 있고, swf 배경을 투명하게 표현한다.

wmode가 'window'로 설정되어 있을 때는 다른 HTML 객체들보다 항상 위에 존재하기 때문에 플래시 화면 위에서 동작하는 HTML 객체가 있는지 확인해야 한다. 플래시의 성능이 중요하여 wmode를 변경하기 어려울 때는 iframe을 사용하여 플래시에 콘텐츠가 가리지 않도록 마크업한다. 작성 예는 다음과 같다.

```
<style type="text/css">
.layer_highest {overflow:hidden; position:absolute; top:0px; left:0; z-index:-1;}
</style>

<div class="over flash">
   <p>플래시 위에 뜨는 레이어</p>
   <iframe frameborder="0" scrolling="no" class=" layer highest "></iframe>
</div>
```

A.4 PNG 이미지 사용

PNG 형식의 이미지는 GIF보다 표현할 수 있는 색이 많고 투명하게도 표현할 수 있다는 장점이 있으나, 용량이 크고 Internet Explorer 6 이하에서는 바르게 표시되지 않는 버그가 있어 사용하지 않는다. 부득이하게 PNG 이미지를 사용해야 할 때는 아래의 내용을 준수한다.

A.4.1 배경 이미지로 사용

Internet Explorer 6 이하 버전에서 회색으로 표시되는 버그를 해결하기 위해 핵(hack)과 Internet Explorer 전용 filter를 동시에 사용하여 아래와 같이 코드를 작성한다.

```
.selector{background:url(test.png);  _background:none;
_filter:progid:DXImageTransform.Microsoft.AlphaImage Loader(src='test.png',
sizingMethod='crop');}
```

filter 속성으로 표현된 배경 이미지는 background-repeat이나 background-position 속성을 포함할 수 없다. 또한, PNG 배경 이미지 처리가 된 엘리먼트에 a 엘리먼트가 포함될 때, 마우스가 정상적으로 접근할 수 없는 문제가 발생하므로 a 엘리먼트와 PNG 배경이 적용된 엘리먼트는 별도로 마크업한다.

A.4.2 전경 이미지로 사용

전경 이미지에 PNG 형식을 사용할 때는 PNG를 정상적으로 표시하도록 작성된 Internet Explorer 전용 자바스크립트 'iepngfix.htc' 파일을 사용하여 아래와 같이 작성한다. 이 파일은 Internet Explorer 5.5~6 브라우저에서 동작하며, 크로스 도메인이 되지 않으므로 같은 도메인 이름으로 된 서버에 있어야 한다.

```
.png{_behavior:url(iepngfix.htc);}
```

부록 B. 예약어 목록

부록 B에서는 네이밍 예약어와 대체 텍스트 예약어 목록을 설명한다.

B.1 네이밍 예약어

표 B-1 공통 네이밍 예약어

의미	예약어	의미	예약어	의미	예약어	의미	예약어
가기	_go_	별	_star_	오버	_ovr_	취소	_cncl_
갱신/업데이트	_updt_	불릿	bu_	완료/예	_ok_	켜짐	_on_
검색	_srch_	사진	_pht_	왼쪽	_lft_	탭 메뉴	tab_
경로	_path_	삭제	_del_	왼쪽 아래	_lb_	텍스트 입력필드	input_txt
광고	ad_	새로고침	_rfsh_	왼쪽 위	_lt_	파일	_file_
그루핑 1	section_	선	line_	왼쪽 플로트	fl	페이징	paginate
그루핑 2	group_	설명	dsc_	원	_circ_	펼치기	_unfd_
그루핑 3	_area	설치/인스톨	_inst_	위	_up_	하단	_btm_
꺼짐	_off_	섬네일	_thmb_	이미지	img_	한국어	_kr_
내비게이션	nav_	수정	_edt_	이전	_prev_	해제	clear, _rels_, _unlc_
다운로드	_dnld_	수직	_vr_	일어	_jp_	허용	_pmit_
다음	_next_	수평	_hr_	읽기	_read_	화살표	_arr_
닫기	_clse_	숨김	blind	임시	@tmp_	확대	_zin_
답변/회신	_repl_	숫자	_num_	잠그기	_lc_	확인	_cfm_
더보기	_more_	스프라이트	_sp_	재생	_play_	aside	aside_
동영상	_mov_	슬래시	_sl_	전송	_smit_	gnb	gnb
등록	_reg_	신규	_new_	점	_dot_	lnb	lnb_
라디오버튼	input_rdo	쓰기	_wrt_	접기	_fd_	snb	snb_
레이어	ly_	아래	_dn_	정지/멈춤	_stp_	spot	spot_
마스크	_mask_	아이콘	_ico_	제목	h_	sta	sta
마지막	_last_	압축	_zip_	주의/유의	_noti_	tahoma	thm
맨 위	_top_	업로드	_upld_	중국어	_cn_	verdana	vdn
목록	_lst_	열기	_opn_	중앙(수직)	_mid_		
문장	p_	영어	_en_	중앙(수평)	_cen_		
배경/박스	_bg_	오른쪽	_rgt_	찾기	_fnd_		
백슬래시	_bsl_	오른쪽 위	_rt_	처음	_frst_		
버튼	btn_	오른쪽 플로트	fr	체크박스	input_chk		
베스트	_best_	오른쪽 하단	_rb_	축소	_zout_		

[조합 기호: "_"]

표 B-2 객체 예약어

분류	예약어	영역/객체
공통	.gnb	최상위 전역 내비게이션 영역
	.sta	서비스 이름, 연관 서비스, 검색 영역
	. lnb_	현재 서비스의 지역 내비게이션 영역
	.snb_	측면 내비게이션 영역
	.aside_	문서의 주요 부분을 표시하고 남은 콘텐츠 영역
	.spot_	강조하는 상위 콘텐츠 영역
	.path_	현재 페이지의 경로
	.nav_	내비게이션 엘리먼트
	.ad_	광고 엘리먼트
그루핑	.section_	heading 태그(h1~h6)를 포함한 엘리먼트들의 그루핑
	.group_	section 보다 낮은 단계의 heading 태그를 포함한 엘리먼트들의 그루핑
	. _area	위치에 제한이 없는 특정 기능을 수행하는 엘리먼트들의 그루핑

[조합 기호: "_"]

표 B-3 이미지 예약어

예약어	분류
h_	제목
p_	문장
gnb_ , lnb_ , snb_	내비게이션
tab_	탭
btn_	버튼
bu_	불릿
ico_	아이콘
line_	선
bg_	배경, 박스
img_	이미지
_off , _over , _on	상태 변화
ad_	광고
@tmp_	임시

[조합 기호: "_"]

B.2 대체 텍스트 예약어

표 B-4 공통 대체 텍스트 예약어

그룹	이미지	대체 텍스트(alt)값	그룹	이미지	대체 텍스트(alt)값
페이징		이전	메일		안 읽은 메일
		다음			읽은 메일
주식		상승	트리		폴더
		하락			펼치기
		보합			접기
랭킹		상승	기타		더보기
		하락			업데이트
		동일			NEW
기능		확대			동영상
		축소			주의
플레이어		일시 정지			닫기
파일		한/글			달력
		Word			검색
		Excel			삭제
		PowerPoint			위로
		GIF			아래로
		JPG			인쇄
		기타			