NHN Coding Convention

for Markup Languages (HTML/CSS)

NTS UIT개발실

일반

NHN
TECHNOLOGY SERVICES

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 요소 작성 규칙

HTML 요소 종류별 작성 규칙을 설명한다.

5장 CSS 코드 작성 규칙

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

부록 A. 코딩 시 참고 사항

웹표준 기반의 마크업, 크로스 브라우징 범위, 폴더 생성 방법, 플래시 사용 시 유의점을 설명한다.

부록 B. 약속어 목록

네이밍 약속어를 설명한다.

독자

이 문서는 마크업 언어를 다루는 NHN㈜의 개발자를 대상으로 한다.

문의처

이 문서의 내용에 오류가 있거나 내용과 관련한 의문 사항이 있으면 아래의 연락처로 문의한다.

연락처: dl_markup@nhn.com

문서 버전 및 이력

버전	일자	이력사항	작성자	승인자

0.8	2009-01-22	초안 작성	웹표준 1 팀 박상혁, 주재승, 조진주	웹표준 1 팀 박태준
1.0	2010-02-09	검토 및 재작성	기술문서팀 김붕미	기술문서팀 유영경
1.1	2010-02-26	수정사항 • 내용 오류 수정	웹표준 1 팀 주재승	웹표준 1 팀 박태준
1.2	2010-06-28	수정사항 • CSS 속성값 축약 • CSS 핵 선언 방법 • 삭제/숨김 주석 처리 • tbody 선언 방법 • iframe 접근성 보장 방법 • 공통 네이밍 예약어 표	웹표준 1 팀 주재승	웹표준 1 팀 박태준
1.5	2010-09-10	수정사항 • 모바일 컨벤션 추가 • 표 2-5 오타 수정 • 3.5 D 내용 추가 • 5.1 C 내용 추가 • 5.3 내용 추가 • 5.8 내용 추가 • 올바르지 않은 URI 수정	웹표준 1 팀 박상혁, 주재승, 조진주 웹표준 2 팀 문지애, 노찬현, 윤미진	웹표준 1 팀 박태준
2.0	2010-02-07	수정사항 • 2.2 내용 추가 • 2.4 오타 및 내용 보완 • 3.1 내용 추가 및 일부 삭제 • 3.4 내용 추가 및 일부 삭제 • 3.5 내용 일부 삭제 및 수정 • 3.6 내용 추가 • 4.2 내용 추가 • 4.3 내용 추가 • 4.4 내용 삭제 • 4.5 내용 추가 • 5.1 내용 추가 • 5.5 잘못된 표기 변경 • 5.8 오타 및 내용 수정 • 5.12 잘못된 표기 변경 • 5.14 내용 일부 삭제 및 수정 • 5.16 잘못된 표기 변경 • 6.4 내용 보완 • 6.7 잘못된 표기 변경 • 부록 B 내용 추가	웹표준 1 팀 박상혁, 주재승, 조진주	웹표준 1 팀 박태준
2.5	2011-06-01	수정사항 소속팀,카피라이트,문의처 수정	웹표준개발 1 팀 김동우, 조진주	웹표준개발 1 팀 박태준

		• 1.3 내용 일부 수정 • 2.3 내용 일부 추가 • 2.4 내용 일부 수정 및 삭제 • 3.1 내용 일부 추가 • 3.3 내용 일부 수정 • 3,4 내용 일부 수정 및 추가 • 3.6 내용 수정 • 4.2 내용 일부 수정 • 4.3 내용 일부 수정 및 추가 • 4.4 내용 일부 수정 • 4.5 내용 일부 추가 • 5.3 내용 일부 수정 • 5.4 내용 일부 수정 • 5.8 내용 일부 수정 및 추가 • 5.9 내용 삭제 • 5.11 내용 일부 수정 • 5.12 내용 일부 수정 • 5.15 내용 일부 수정 및 삭제 • 5.16 내용 수정 • 6.2 일부 내용 삭제 • 6.4 일부 내용 추가 • 6.5 내용 추가 • 6.6 내용 추가 • 6.7 내용 일부 수정 및 추가 • 6.8 내용 일부 수정 및 삭제 • 부록 A 내용 수정 및 추가 • 부록 B 오타 수정 및 추가		
2.75	2012-01-02	수정사항 소속팀,카피라이트,문의처 수정 • 2.4 내용 일부 수정 • 3.1 내용 일부 수정 • 3.2 내용 일부 수정 • 3.4 내용 일부 수정 및 삭제 • 3.6 내용 수정 • 4.2 내용 일부 수정 • 4.3 내용 일부 수정 • 4.4 내용 일부 수정 • 4.5 내용 일부 수정 • 5.1 내용 일부 수정 • 5.9 내용 일부 수정 • 5.10 내용 일부 수정 • 5.13 내용 일부 수정 • 5.14 내용 수정	웹표준개발 1 팀 김동우	웹표준개발 1 팀 박태준

		• 5.15 내용 수정 • 6.2 내용 일부 수정 • 6.4 내용 일부 수정 • 6.5 내용 일부 수정 • 6.6 내용 일부 수정 • 6.7 내용 일부 수정 • 6.8 내용 일부 수정 • 6.9 내용 일부 수정 • 6.10 내용 일부 수정 • 6.11 내용 일부 수정 • A.1 내용 일부 수정 • A.2 내용 수정 • A.3 내용 수정 • A.6 내용 삭제 • A.8 내용 추가		
2.8	2012-11-09	수정사항 소속팀,문서개요,문의처 수정 • 목록 수정 • 1.2.1 내용 일부 삭제 • 1.2.2 B. 내용 일부 삭제 • 1.2.2 D. 내용 일부 삭제 • 1.2.3 내용 일부 삭제 • 1.2.4 A 내용 일부 수정 • 1.2.4 D 내용 일부 수정 • 1.2.4 E 내용 일부 수정 • 1.2.4 G 내용 일부 삭제 • 1.2.4 J 미디어타입 추가 • 1.2.4 L 내용 일부 삭제 • 1.3 A 제목 수정 • 1.3 C 내용 일부 삭제 • 표 1-1 내용 일부 수정 • 1.3 D 내용수정 • 1.3 E 내용수정 • 1.3 F 내용 일부 삭제 • 2.1 C 내용 일부 수정 • 2.1 D 내용 일부 수정 • 표 2-2 내용 일부 삭제 • 2.2 B 제목 수정 • 표 2-3 내용 일부 삭제 • 2.2 C 내용 일부 수정 • 표 2-4 내용 일부 수정 • 2.2 D 내용 일부 삭제 • 2.2 E 내용 일부수정	UX 디자인실 김용원, 서미연	UX 디자인실 소지훈

- 2.2 F 내용추가
- 그림 2-3 추가
- 그림 2-4 추가
- 표 2-5 내용 일부 수정
- 표 2-6 내용 일부 수정
- 2.3 A 내용 일부 삭제
- 표 2-7 내용일부 삭제
- 2.4 A 내용 일부 삭제
- 표 2-8 내용 일부 삭제
- 표 2-9 모바일 삭제
- 표 2-9 참고 내용 수정
- 2.4 B 내용 일부 삭제
- 표 2-10 내용 일부 삭제 및 표 번호 수정
- 3.1 B 내용 수정
- 표 3-3 내용 일부 삭제
- 3.1 E 내용 일부 삭제 및 추가
- 표 3-4 내용 일부 수정
- 3.2 내용 일부 수정
- 3.2 참고 삭제
- 3.3 내용 일부 삭제
- 3.4.1 A 내용 일부 삭제
- 3.4.2 A 내용 일부 수정
- 3.4.3 모바일 삭제
- 표 3-7 삭제
- 3.4.3 참고 삭제
- 3.5 A 내용 일부 삭제
- 표 3-8 내용 일부 삭제 및 표 숫자 변경
- 3.5 B 내용 일부 삭제
- 3.5 C 내용 일부 삭제
- 표 3-9 내용 일부 삭제 및 표 숫자 변경
- 3.6 일부 내용 수정 및 삭제
- 4. 제목수정
- 4.1 내용 추가
- 4.2 제목수정
- 4.2 A 내용 일부 수정
- 4.2 B 내용 일부 수정
- 4.2 C 삭제
- 4.2 D 내용 일부 수정 및 제목 번호 수정
- 4.2 E 내용 일부 삭제 및 제목 번호 수정
- 4.2 F 내용 일부 삭제 및 제목

번호 수정

- 4.2 G 내용 일부 수정 및 제목 번호 수정
- 4.2 H 내용 일부 수정 및 제목 번호 수정
- 4.3 제목수정
- 4.3 A 내용 일부 수정
- 4.3 B 내용 일부 수정
- 4.3 C 내용 일부 삭제 및 추가
- 4.3 D 내용 일부 수정
- 4.3 E 내용 일부 수정
- 4.3 F 내용 일부 삭제
- 4.4 내용 일부 추가
- 4.4 A 내용 일부 삭제
- 4.4 B 내용 일부 수정
- 4.4 C 내용 일부 수정
- 4.4 D 내용 일부 삭제
- 4.4 E 내용 일부 삭제
- 4.4 F 내용 일부 삭제
- 4.4 G 내용 일부 수정
- 4.4 G 참고 내용 일부 수정
- 4.5 제목 수정
- 4.5 A 내용 일부 삭제
- 4.5 B 내용 일부 삭제
- 4.5 C 내용 일부 삭제
- 4.5 D 내용 일부 수정
- 4.5 E 내용 일부 삭제
- 5.1 A 내용 일부 삭제
- 5.2 내용 일부 추가
- 5.4 내용 일부 삭제
- 5.5 내용 일부 삭제 및 추가
- 5.7 A 내용 일부 삭제 및 수정
- 5.7 C 내용 일부 수정
- 5.7 참고 내용 일부 수정
- 5.8.1 내용 일부 삭제
- 표 5-11 내용 일부 수정
- 표 5-12 삭제
- 표 5-13 표 번호 수정
- 5.8.3 B 오타수정
- 5.10 핵 모바일 삭제
- 표 5-17 삭제
- 5.11 내용 일부 수정
- 5.11 참고 내용 일부 추가
- 5.12 A 내용 일부 수정

- 5.12 B 제목 수정
- 5.12 C 내용 일부 수정
- 5.13.1 내용 일부 추가
- 5.13.2 내용 일부 추가
- 5.13.3 A 내용 일부 수정
- 5.13.3 B 내용 일부 수정
- 5.14 내용 일부 수정
- 5.14 모바일 삭제
- 5.15 내용 일부 수정
- 6. 전체 삭제
- 부록 A.1.1 삭제
- 부록 A.1.2 내용 일부 삭제 및 부록 번호 수정 A.1.1
- 부록 A.1.3 내용 일부 삭제 및 부록 번호 수정 A.1.2
- 부록 A.1.3 추가
- 부록 A.2 내용 일부 수정
- 표 A-2 삭제
- 표 A-3 삭제
- 그림 A-2 수정
- 부록 A.3.2 내용 일부 수정
- 부록 A.4 내용 일부 삭제
- 부록 A.5 삭제
- 부록 A.6 번호 수정 A.5
- 그림 A-3 수정
- 부록 A.8 삭제
- 부록 B 제목수정
- 부록 B.1 내용 일부 수정 및 삭제
- 표 B-1 삭제
- 표 B-2 내용 추가
- 표 B-3 내용 추가
- 부록 B.2 삭제

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

C. HTML 코드의 빈줄은 의미 있는 객체를 구분하기 위하여 사용한다.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 요소와 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, 참고해야 하는 사항을 기술한다.

1.2.3 HTML 요소 작성 규칙

HTML 요소 작성 규칙은 반드시 선언해야 하는 애트리뷰트와 선택적 사용이 가능한 애트리뷰트에 대한 내용을 기술한다.

1.2.4 CSS 코드 작성 규칙

A. 들여쓰기

CSS 코드는 들여쓰기를 하지 않는다. 단, 중괄호"{}"가 중첩되는 경우 예외

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 속성 줄 바꿈은 하지 않는다.

E. 인코딩

HTML 문서가 브라우저에서 바르게 해석될수 있도록 CSS의 인코딩은 HTML의 인코딩과 동일하게 선언 한다.

F. 선택자

선택자 버그가 발생하지 않도록 사용 규칙을 준수한다.

G. 속성

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

H. z-index

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

I. 핵

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

J. 미디어타입

미디어타입 상황에 맞게 추가해서 사용한다.

K. CSS 선언 타입

상황에 알맞은 CSS 선언 타입을 선택한다.

L. 주석

CSS 코드의 주석은 코드 그룹을 구분하거나참고해야 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. 요소(Element)

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

B. 애트리뷰트(Attribute)

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

C. 선택자(Selector)

요소에 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 요소와도 일치함.
타입 선택자	div	해당 요소와 일치함.
하위 선택자	div p	해당 요소의 하위의 모든 요소와 일치함.
자식 선택자	div > p	해당 요소의 자식 요소와 일치함.
class 선택자:	div.class	해당 요소의 class 애트리뷰트값과 일치함.
id 선택자	div#id	해당 요소의 id 애트리뷰트값과 일치함.

D. 속성(Property)

요소에 부여할 CSS 스타일 특성을 의미한다. 기본값이 설정되어 있으나 속성을 선언하여 다른 값으로 설정할 수 있다. 각 속성 단위는 세미콜론(;)으로 구분지어 선언한다.

E. 속성 값(Value)

속성에 부여하는 값을 의미한다.

F. 약속어

이 문서에서 약속어는 객체, 이미지, 파일 및 폴더의 네이밍 시 의미를 일관되게 표현하기 위해 미리 지정해 놓은 일종의 언어 규칙을 의미한다.

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 소문자로만 시작할 수 있다.

단, css 주석문은 영문 대문자를 허용한다. (예: /* NHN WSD1Team KDW 120101 */)

표 2-1 공통 네이밍 규칙 예

잘못된 예	올바른 예
Btn	btn
2btn	btn2
_btn	btn
btn_	btn

C. 약속어

- 약속어는 레이아웃 약속어, 객체 약속어, 이미지 약속어에 근거하여 작성한다.(표 2-3 레이아웃 약속어 범위,표 2-4 팝업 레이아웃 약속어 범위,표 B 2-1 객체 약속어,표 B 2-2 이미지 약속어에 근거하여 작성한다.)
- 약속어가 없으면, 종류와 특성을 나타내도록 네이밍하거나 공통 네이밍 약속어 Guide를 참고하여 작성한다.
- 공통요소 약속어(.u_)는 공통업무 담당자에 한에서만 사용 한다.

참고

공통 네이밍 약속어 Guide는 권장 약속어로 공통 네이밍 약속어 Guide는 아래 링크주소를 참고한다.

http://devcafe.nhncorp.com/ws/1151467

-

D. 영문 소문자, 숫자, 언더스코어 조합

- 영문 소문자, 숫자, 언더스코어(_)만 사용할 수 있다.
- 언더스코어(_)는 단어와 단어 조합할 때만 사용한다.
- 언더스코어(_)가 포함된 약속어는 숫자, 영문 소문자와 조합하여 사용할 수 있다.

표 2-2 네이밍 조합 예

기본형	잘못된 예	올바른 예
section	sectionList	section_list

no	-	no1, no2, no3 …. no10

2.2 id 및 class 네이밍 규칙

다음 그림은 HTML의 기본 템플릿 구조를 나타낸 것이다.

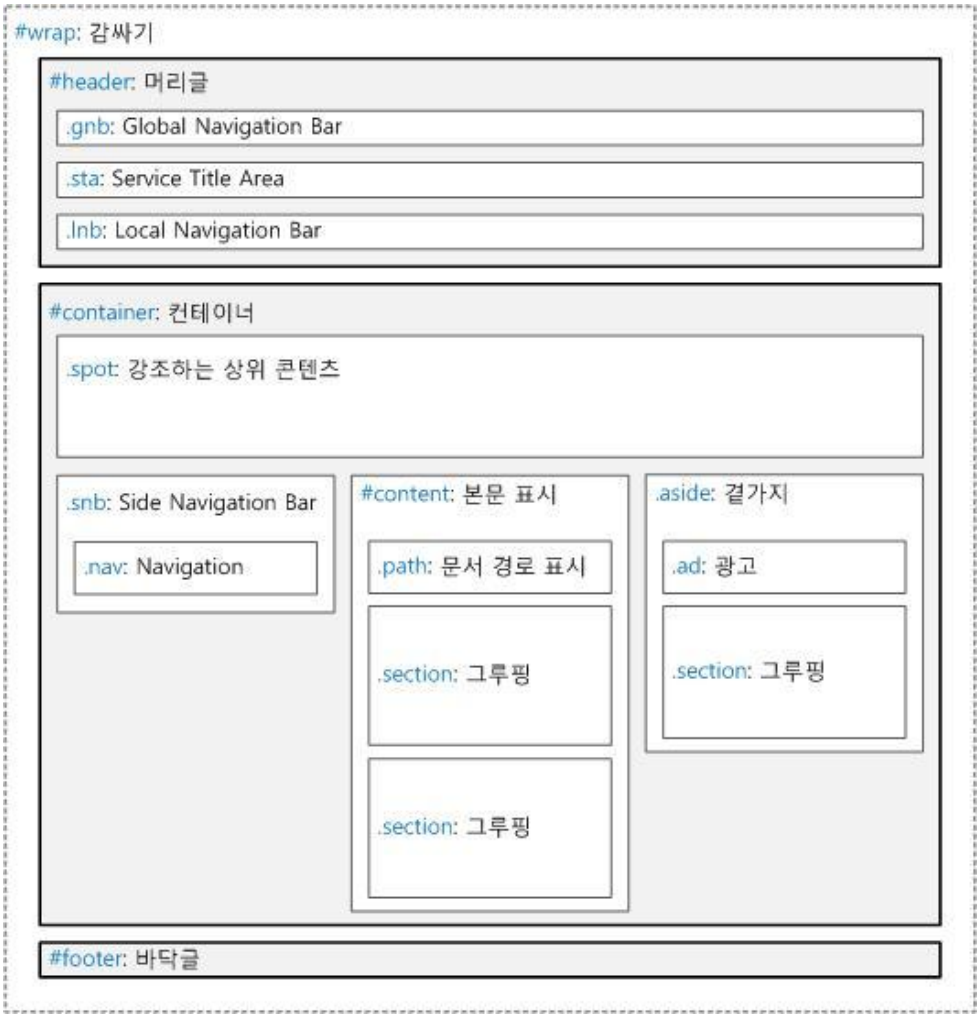

그림 2-1 PC HTML 기본 템플릿 구조

그림 2-2 모바일 HTML 기본 템플릿 구조

A. id, class

- id는 문서 전체의 고유 식별자 이므로 한 문서에서 동일한 id를 여러 번 사용하지 않는다.
- 레이아웃을 제외한 id는 스타일을 지정하지 않는다.
- class는 문서에서 여러 번 사용할 수 있다.

B. 레이아웃 약속어

레이아웃에는 다음 표에 예약된 id만 사용한다.

표 2-3 레이아웃 약속어 범위

약속어	범위
#wrap	페이지 전체 영역
#header	머리글 영역
#container	본문 영역
#content	주요 콘텐츠 영역
#footer	바닥글 영역

C. 팝업 레이아웃 약속어

- 팝업 문서의 레이아웃 지정 범위는 동일하다.

- 레이아웃 약속어 앞에 'pop_'를 조합하여 사용한다.

표 2-4 팝업 레이아웃 약속어 범위

약속어	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

- 레이아웃에 디자인 속성을 추가/변경하려면 class를 사용한다.

표 2-5 레이아웃 네이밍 조합 예

잘못된 예	올바른 예
#wrap2, #wrap_type	#wrap

```
<div id="wrap" class=”wrap type”>
  <div id="header"></div>
  <div id="container" class="container_type">
     <div id="content"></div>
  </div>
  </div>
  <div id="footer"></div>
</div>
```

E. 객체 네이밍

- 객체 약속어는 표 B-1 객체 에 근거하여 작성한다.
- 객체는 class만 사용할 수 있으며, 전사 공통 마크업 템플릿의 class와 중복되지 않아야 한다.
- 개발과 연동되는 동적 객체도 class만 사용한다.
- 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.

```
<style>
.mykin list a{font-size:12px;line-height:14px;font-weight:bold}
.mykin list v1 a{color:#f00}
.mykin list v2 a{color:#03f}
</style>

<div class="mykin_list"><a href="#">바로가기</a>
</div>

<div class="mykin_list mykin_list_v1"><a href="#">바로가기</a>
</div>

<div class="mykin_list mykin_list_v2"><a href="#">바로가기</a>
</div>
```

그림 2-3 종속 확장 class 예

- 독립 확장 class

  기본형 class의 변형이 타입으로 분류할 만큼 클 경우 사용하는 class.

```
<ul class="mykin list2">
  <li>리스트1</li>
  <li>리스트1</li>
  <li>리스트1</li>
</ul>

<ul class="mykin_list3">
  <li>리스트1</li>
  <li>리스트1</li>
  <li>리스트1</li>
</ul>
```

그림 2-4 독립 확장 class 예

표 2-6 class 네이밍 확장 예

기본형	확장형	설명
mykin_list	mykin_list_v1, mykin_list_v2...	종속 확장 class
	mykin_list2, mykin_list3	독립 확장 class

2.3 이미지 네이밍 규칙

다음과 같은 이미지 네이밍 규칙을 준수한다.

A. 이미지 네이밍

- 이미지 약속어는 표 B-2 이미지 에 근거하여 작성한다.
- 같은 분류의 이미지가 두 개 이상이면 파일 이름 마지막에 숫자를 추가하여 구분한다.
- 이미지 네이밍은 이미지 확장자와 관계 없이 순차적으로 적용한다. 예) bu_dot1.gif, bu_dot2.jpg, bu_dot3.png
- 임시 이미지는 tmp_를 조합한다. 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.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

- 표 2-8 파일 및 폴더 네이밍 규칙을 따른다.

표 2-8 파일 및 폴더 네이밍 규칙 예

분류	예제	설명
HTML	news.html	'서비스영문이름.html' 로 사용
	pop_.html	팝업 파일을 사용
	ifr_.html	iframe 파일을 사용
	Inc_.html	Include 파일
	tmp_.html	임시파일을 사용(QP 예외규칙)
CSS	news.css	'서비스영문이름.css'로 사용한다.
	news_e.css	모바일 예외 대응시 '모바일 CSS 파일 분기 규칙'에 따라 사용한다.
Folder	p_yymmdd_프로젝트이름	신규 프로젝트 작업 시 사용
	s_yymmdd_서스테이닝이름	서스테이닝 작업 시 사용
	img, css, js	image, css, javascript 폴더 사용
	p_yymmdd_프로젝트명₩메뉴명	HTML 파일의 폴더 분류가 필요한 경우 사용
	tmp_	임시파일의 폴더 분류가 필요한 경우 사용(QP 예외규칙)

참고모바일CSS파일 분기 규칙은 아래 링크주소를 참고한다.

http://devcafe.nhncorp.com/mobilewebsource/board_17

B. 파일 및 폴더 네이밍 조합

- HTML파일은 '메뉴이름_의미_상태' 순서로 조합한다.
- CSS 파일은 '서비스이름'을 맨 앞으로 하여 조합한다.

표 2-99 HTML/CSS 조합 예

분류	조합 예	설명
HTML	news_nboard_view.html	'메뉴이름_의미_상태' 순서로 조합한다.
CSS	news_home.css	'서비스이름'을 맨 앞으로 하여 조합한다.

	news_admin.css
	news_pop.css, news_nanum.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. 단, HTML5 DTD 선언 시 다음 오류 내용은 허용한다.

- <iframe>의 frameborder, marginwidth, marginheight, scrolling 애트리뷰트

B. 영문 소문자 사용

DTD를 제외한 모든 요소와 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. Character entity references 사용

특수 기호는 문자 엔티티 참조를 사용하여 코드로 변환한다. 자세한 사항은 아래 경로에서 확인할 수 있다. 단, 아포스트로피(')는 '로 선언한다.

HTML 4.01의 Character entity references: http://www.w3.org/TR/html4/sgml/entities.html

HTML 5의 Character references: http://www.w3.org/TR/html5/syntax.html#character-references

표 3-3 Character entity references 사용 예

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

참고

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

E. 스크립트 선언 지양

Javascript를 사용하되 태그에 inline방식으로 사용하는 이벤트 속성 및 스크립트는 HTML 산출물에 선언하지 않는다.(<head>태그 사이 또는 <body>태그 최하단에 선언할 수 있다)

표 3-4 스크립트 예

잘못된 예	올바른 예
<input type="text" id="user_id" name="user_id" onfocus="this.className='focus'">	<!-- [D] 입력 박스에 포커싱되었을 때, class="focus" 추가 --> <input type="text" id="user_id" name="user_id" class="focus"> …. <script> document.getElementById("user_id").addEventListener('focus',function(e){e.target.className="focus";},false); </script> </body>

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 요소는 1탭 들여 쓰고, 1탭의 크기는 공백 4칸으로 설정한다. 문서 내에서 반드시 탭을 이용하여 들여쓰기를 하며, 탭을 대신하여 공백으로 띄어 들여쓰지 않는다. 또한 다음의 경우에는 중첩 단계의 파악이 용이하도록 들여쓰지 않는다.

- <html>의 자식 요소인 <head>, <body>
- <head>의 자식 요소
- <body>의 자식 요소
- <table>의 자식 요소인 <caption>, <colgroup>, <col>, <thead>, <tbody>, <tfoot>, <tr>,<th>, <td>
- , <ol>, <dl>의 자식 요소인 <li>, <dt>, <dd> 요소
- <fieldset>의 자식 요소인 <legend>
- <map>의 자식 요소인 <area>

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>

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.

.

표 3-6 빈 줄 사용 예

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

3.4 DTD 및 인코딩

3.4.1 DTD 선언

HTML 문서는 반드시 DTD를 선언한다.

A. 기본 DTD

신규 HTML 문서를 작성할 때 'HTML5'를 사용한다. 작성 예는 다음과 같다.

```
<!DOCTYPE html>
```

참고

HTML5는 SGML에 기본으로 하지 않기 때문에, DTD의 참조를 요하지 않는다. 〈!DOCTYPE〉은 웹 브라우저가 HTML이 무슨 버전인지 알수 있게 해준다.

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.

HTML 4.01 Transitional

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

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 charset="utf-8">
```

HTML4.01 DTD 사용 시에는 아래와 같이 선언한다.

```
<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.
- 시작과 종료 주석의 주석 내용은 동일해야 한다.

표 3-7 주석 기본 형식 예

잘못된 예	올바른 예
<!--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, 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. 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.

표 3-8 개발 적용 관련 주석 표기 예

사용 예
<!-- content -- > <!-- [D] 케이스별 클래스 변화 의사 : my_doctor 변호사 : my_lawyer -->
<img src="btn.gif" width="100" height="100" alt="버튼"> <!-- [D] 활성된 버튼은 파일명에 _on 추가 <img src="btn_on.gif" width="100" height="100" alt="버튼"> -->

3.6 초기 파일

신규 HTML 문서를 작성할 때 아래 파일을 참고한다.

```
<!DOCTYPE html>
<html lang="ko">
<head>
<meta charset="utf-8">
<title>메뉴 : 브랜드명 서비스</title>
<link rel="stylesheet" type="text/css" href="sevice name.css">
</head>
<body>
<div id="wrap">
   <!-- u skip -->
   <div id="u skip">
   </div>
   <!-- //u_skip -->
   <!-- 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>
```

모바일 신규 HTML 문서를 작성할 때 아래 파일을 기본으로 한다.

```
<!doctype html>
<html lang="ko">
<head>
<meta charset="utf-8">
<meta name="viewport" content="..">
<title>상단: 헤더명</title>
<link rel="stylesheet" type="text/css" href="css/..">
</head>
<body class="..">
<!-- u_skip -->
<div id="u skip" >
</div>
<!-- //u skip -->
<!-- header -->
<header role="banner">
</header>
<!-- //header -->
<hr />
<!-- content -->
<div id="content" role="main">
</div>
<!-- //content -->
<hr />
<!-- footer -->
<div id="footer" role="contentinfo">
</div>
<!-- //footer -->
</body>
</html>
```

4. HTML 요소 작성 규칙

이 장에서는 HTML 요소 종류별 작성 규칙을 설명한다.

4.1 기본 규칙

특정 요소에 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 요소

A. <html>

다음과 같이 lang 애트리뷰트를 선언한다.

class 애트리뷰트는 선언하지 않는다.

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

XHTML DTD 선언 시에는 다음과 같이 lang 애트리뷰트를 선언한다.

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

lang 애트리뷰트는 User Agent가 언어를 올바로 해석할 수 있게 도와주며, 검색과 음성 장치(speech synthesizers)에 활용된다. 언어 코드는 모든 요소에 사용할 수 있지만 HTML 요소에 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.

B. <head>

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

```
<head>
<meta charset="utf-8">
<title>속보 : 뉴스</title>
<link rel="stylesheet" href="css/default.css">
<script src="js/default.js"></script>
<style>
…
</style>
</head>
```

HTML 4.01 DTD선언 시에는 다음과 같다.

```
<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">
…
</style>
</head>
```

C. <title>

"메뉴 : 브랜드명 서비스"의 형식으로 작성한다.

```
<title>속보 : 네이버 뉴스</title>
```

D. <link>

rel, type, href 애트리뷰트를 선언한다.

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

E. <script>

src 애트리뷰트를 선언한다.

```
<script src="js/default.js"></script>
```

F. <style>

```
<style >
 …
</style>
```

참고

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

HTML4.01 문서의 경우 <script>와 <style> 선언 시 type 애트리뷰트 를 선언한다.

G. <body>

모바일에서는 <body>에 브라우저별로 정의된 class명을 지정하고 이를 상속하여 브라우저별 CSS 렌더링 차이를 해결한다. 변경 사항이 잦으니 아래 경로에서 확인한다

http://devcafe.nhncorp.com/mobilewebsource/board_17

표 4-1 <body> 태그의 클래스 추가

브라우저	클래스
사파리	s, s2(iOS 4.2 미만)
안드로이드	a

```
<body class="s">
```

4.3 폼 요소

폼 컨트롤 요소를 마크업할 때 <form>, <fieldset>, <legend> 요소를 다음과 같이 선언한다. 단, 필요에 따라 개별적으로 사용할 수 있다.

```
<form>
<fieldset>
<legend>개인정보</legend>
   …
</fieldset>
</form>
```

A. <fieldset>

<form> 요소의 자식 노드로 선언하여 폼 컨트롤 요소들을 그루핑하기 위해 선언한다.

```
<form>
<fieldset>
<legend>개인정보</legend>
   …
</fieldset>
</form>
```

B. <legend>

폼 컨트롤 그룹인 <fieldset>의 자식 요소로서 폼 컨트롤 요소들의 그룹 이름을 표현하기 위해 선언한다.

```
<form>
<fieldset>
<legend>개인 정보</legend>
   …
</fieldset>
</form>
```

C. <input>

<label> 요소, title 애트리뷰트, alt 애트리뷰트를 통해 스크린 리더 사용자는 주변 맥락에 대한 이해 없이 각 요소에 독립적으로 접근해도 폼을 이해할 수 있다.

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

<input type="text" id="num1" name="num1" title="지역번호">
<input type="text" id="num2" name="num" title="국번">
<input type="text" id="num3" name="num" title="전화번호">
```

다음과 같이 type값에 따라 애트리뷰트를 선언한다.

- type이 text인 경우
  - 동일한 스타일의 텍스트필드이나 너비값이 다를 경우 style 애트리뷰트를 이용하여 width값을 제어한다.
  - type이 radio, checkbox인 경우그루핑이 필요하면 선택적으로 name 애트리뷰트를 이용하여 항목들을 그루핑한다.
  - 기본 선택에 대한 표현이 필요할 경우 checked="checked"라고 표기한다.
- type이 image인 경우
  - alt 애트리뷰트를 반드시 선언한다.

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

- type 이 file인 경우: type 애트리뷰트를 선언한다.

```
<input type="file">
```

- type 이 button, reset, submit 인 경우: type 애트리뷰트를 선언한다.

```
<input type="button">
```

D. 〈select〉

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

```
<label for="grade">등급</label>
<select id="grade" style="width:100px">
<option>1등급</option>
<option>2등급</option>
</select>
<select title="등급" style="width:100px">
<option>등급</option>
</select>
```

E. 〈label〉

〈input〉(text, checkbox, radio, file, password), 〈select〉, 〈textarea〉와 같은 폼 요소는 for 애트리뷰트를 부여하여 해당 요소의 id값과 동일한 이름으로 연결(coupling)한다. 단, 레이블 명이 위치할 공간이 없는 경우 title 애트리뷰트로 대체할 수 있다.

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

단, 〈label〉 안에 〈input〉 앨리먼트가 있는 경우 for와 id를 이용하여 연결하지 않아도 된다.

```
<label><input type="checkbox" name="sports">축구</label>
```

F. 〈textarea〉

cols, rows 애트리뷰트를 선언한다.

CSS를 정상적으로 불러오지 못하는 상황에서도 사용에 문제가 없도록 cols, rows의 애트리뷰트값은 각각 최소 30, 5 이상이 되도록 선언한다. 〈label〉로 연결(coupling) 할 수 없는 경우 title을 사용하여도 무방하며 id, title 애트리뷰트를 선언한다.

```
<textarea cols="30" rows="5" (id=" ") (title=" ")></textarea>
```

G. 〈button〉

type 애트리뷰트를 선언한다.

- type 애트리뷰트를 button으로 선언하여 열기/닫기, 접기/펼치기 등의 UI를 제어한다.
- 폼 전송 역할을 하는 버튼은 submit 타입을 사용한다.

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

4.4 표 요소

표 요소를 아래와 같이 배치한다.

```
<table summary="짬뽕은 자장면보다 500원 비싸고 열량이 50kcal 높다">
<caption>자장면과 짬뽕의 가격과 열량 비교</caption>
<col span="2" style="width:100%">
<thead>
<tr>
<th scope="col" >구분</th>
<th scope="col" abbr="가격">가격(won)</th>
<th scope="col" abbr="열량">열량(kcal)</th>
</tr>
</thead>
<tfoot>
<tr>
<th scope="row" >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>

2차원의 격자형 데이터를 표현하기 위해 사용한다.

표의 요약 내용을 표기해야 할 때 summary 애트리뷰트를 선택적으로 사용할 수 있다.

```
<table summary="summary context here">
```

..

B. <caption>

표의 제목을 표현하기 위해 사용한다. caption 요소는 반드시 선언한다

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

C. <colgroup>

<col> 요소를 그루핑하여 디자인을 제어할 때 선언한다. 이 요소는 선택적으로 사용한다.

```
<colgroup><col style="width:100px"></colgroup>

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

D. <col>

표 각 열의 너비 지정을 위해 선언한다.

width, span 애트리뷰트를 아래와 같이 선택적으로 선언한다.

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

E. 〈thead〉

표 머리글을 그루핑할 때 선언한다.

```
<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 선택적으로 선언한다. tfoot 요소는 thead와 tbody 요소 사이에 위치해야 한다.

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

G. 〈th〉

scope, abbr, id 애트리뷰트를 선언한다. abbr과 id애트리뷰느는 선택적으로 선언한다.

- 표에 셀 제목이 명시되지 않은 경우에도 〈th〉 요소를 선언하여 의미에 맞는 제목을 명시한다(화면에 표시되지 않도록 CSS로 숨김 처리).
- scope 애트리뷰트는 반드시 선언해야 한다.(id, Headers 사용시 필요없음)

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

참고

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

abbr 애트리뷰트: 〈th〉 요소에 동반되는 애트리뷰트로서 포함하고 있는 콘텐츠를 축약하여 표기할 수 있을 때 약어를 표기하는 데 사용한다. 헤딩 셀의 내용을 반복해서 음성으로 출력하는 도구들은 abbr 애트리뷰트에 표기된 약어를 읽는다.

H. 〈tbody〉

표 본문을 그루핑하기 위해 선언한다. 테이블의 본문(body)이 하나이고 〈thead〉나 〈tfoot〉이 없을 경우 생략할 수 있다.

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

```
<td>300</td>
</tr>
</tbody>
```

4.5 기타 요소

A. <a>

href, target, title 애트리뷰트를 선택적으로 선언한다.

- 새 창으로 페이지를 표시해야 할 때 target 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>
```

참고

HTML5 에서의 <a>

- HTML5에서의 <a> 요소 안에 블록 속성의 요소를 포함할 수 있다.

B. <iframe>

<iframe>은 페이지 성능에 영향을 주기 때문에 가급적 사용하지 않는다.

- 부득이하게 사용해야 할 경우 src, width, height, title, frameborder, marginwidth, marginheight, scrolling 애트리뷰트를 선언한다.
- 스크린 리더 사용자를 위해 title 애트리뷰트에 제목을 표기한다. 상단에 iframe의 heading 요소가 있는 경우더라도 반드시 title을 선언한다.
- Iframe의 내용이 빈 경우더라도 빈 아이프레임라는 것을 사용자에게 알려주도록 한다.

```
<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 자바스크립트 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.
- 모바일에서는 iframe 스크롤에 대한 렌더링이 브라우저별로 다르며, CSS 말 줄임이 동작하지 않는 브라우저가 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 애트리뷰트를 선택적으로 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 브라우저에 독립적으로 툴팁을 표현하기 위해 사용한다.

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

D. <map>

<map> 요소의 name 애트리뷰트를 선언하여 <img> 요소의 usemap 애트리뷰트와 같은 이름으로 커플링한다.

```
<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 애트리뷰트를 선택적으로 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. W3C Validation

CSS는 사용 가능한 Hack과 CSS3 속성을 제외하고 W3C Validation을 통과해야 한다.

B. 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}

C. 따옴표 사용 범위

공백이 포함된 폰트명, 한글 폰트명, 문자열 데이터 타입, filter 속성의 파라미터값은 작은따옴표(' ')로 감싸며, @charset 선언 시에는 속성 값을 큰따옴표(" ")로 감싼다. 그 외의 경우에는 따옴표를 사용하지 않는다. filter 속성의 파라미터값에 따옴표를 사용하는 것은 선택사항이다.

url 데이터 타입에는 작은 따옴표를 사용하지 않는다.

예)

```
font-family:'돋움'
filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(src='bg.png',
sizingMethod='scale')
background:url(bg.gif) no-repeat
content:'Chapter:'
charset "utf-8";
```

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	background:url(bg.gif) no-repeat
content: "Chapter: "	content: 'Chapter: '
@charset 'utf-8';	@charset "utf-8";

D. 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.단, 중괄호"{}"가 중첩되는 경우 예외

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}

@media all and (min-width:480px){.vod_wrp2{height:278px}}	@media all and (min-width:480px){
	.vod_wrp2{height:278px}
	}

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 공백 및 속성 값 사이 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}	.class p{color:#000;line-height:18px}
font-family:'돋움',^dotum;	font-family:'돋움',dotum;

C. 중괄호 좌우 공백 제거

중괄호 좌, 우 공백은 제거한다.

표 5-7 중괄호 좌, 우 공백 제거 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.

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

속성, 속성 값 사이 줄 바꿈은 허용하지 않는다.단, 선택자와 중괄호"{}"가 중첩되는 경우 예외

표 5-9 줄 바꿈 사용 예

잘못된 예	올바른 예

.class1,
.class2,
.class3{
width:
100px;
color:
#000
}
	.class1,.class2,.class3{width:100px;color:#000}

@media print{#header{display:none}}	.class1, .class2, .class3{width:100px;color:#000}

	@media print{
	#header{display:none}
	}

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

최 상위 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. 다중선택자 사용

대응 범위에 인터넷 익스플로러 6.0 브라우저 환경이 포함될 경우 다중선택자 오류가 발생하므로 id와 class, class와 class 간의 다중선택자를 사용하지 않는다.

표 5-10 다중선택자 사용 예

잘못된 예	올바른 예
.class{width:980px}	.class{width:980px}
.class.bg{background:yellow}	.bg{background:yellow}
.class.bg_v1{background:green}	.bg_v1{background:green}

참고

다중선택자란 두 선택자의 조합에 의해 속성이 부여되는 경우를 의미한다.

아래 예에서 background:yellow 속성은 .class와 .bg가 동시에 선언될 때만 의미가 있으며, background:green 속성은 .class와 .bg_v1이 동시에 선언될 때만 의미가 있다.

```
<style>
.class{width:980px}
.bg{background:yellow}
.bg v1{background:green}
</style>
<p class="class bg">다중선택자</p>
<p class="class bg_v1">다중선택자</p>
```

인터넷 익스플로러 6.0에서는 id와 class 간 다중선택자를 사용할 경우 두 번째 선언된 다중선택자를 무시하는 버그가 있으며, class와 class 간의 다중선택자를 잘못 해석하는 버그가 있다. 이와 같은 버그를 방지하기 위해 다중선택자를 사용하지 않도록 한다.

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 의미 순서대로 선언한다

.

표 5-11 속성 선언 순서

순서	의미	대표되는 속성
1	레이아웃	display, visibility, overflow, float, clear, position, top, right, bottom, left, z-index,
2	BOX	width, height, margin, padding, border
3	배경	background
4	폰트	font,color, letter-spacing, text-align, text-decoration, text-indent, vertical-align, white-space
5	기타	위에 언급되지 않은 나머지 속성들로 폰트의 관련 속성 이후에 선언하며, 기타 속성 내의 선언 순서는 무관함

* [속성 선언 순서 기준: 1: 레이아웃, 2: BOX, 3: 배경, 4: 폰트, 5: 기타]

밴더속성과 핵속성은 해당 속성 뒤에 선언한다.

```
.btn{border:1px solid #f60;*border:2px solid #f60;border-radius:2px;-webkit-boder-radius:2px}
```

5.8.2 속성 값 축약

CSS 최적화를 위해 다음과 같이 속성 값을 축약한다.

표 5-12 속성 값 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 인터넷 익스플로러 전용 속성인 CSS filter 를 사용했을 경우 축약 속성을 인식하지 못하는 오류가 있기 때문에 속성 값을 축약하지 않는다.
#ff4400	#f40	
background-position:left center	background-position:0 50%	위치값 문자로 표현한 위치값은 숫자로 변경한다.
top:0px	top:0	0 의 단위 속성 값이 0 일 경우 단위를 표시하지 않는다.

축약 전	축약 후	설명
padding:20px 20px 20px 20px	padding:20px	동일한 속성 값 상, 우, 하, 좌의 속성 값이 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

border 와 background 는 약식 속성을 우선적으로 사용하며, font 약식 속성은 사용하지 않는다.

A. border

속성 값은 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 5-13 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}

B. background

속성 값은 background-color, background-image, background-repeat, background-position, background-size, background-attachment, background-origin, background-clip 순서로 선언한다.

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

작성 예는 다음과 같다.

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

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 속성 값을 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}
```

5.8.4 한글 폰트 선언

영문폰트만 선언할 경우 특정 브라우저에서 폰트를 올바르게 출력하지 못하는 경우가 있으므로 한글 폰트 선언 시 한글, 영문 폰트를 모두 선언한다.

표 5-14 폰트 선언 예

잘못된 예	올바른 예
font-family:'돋움'	font-family:'돋움',dotum

5.9 z-index

z-index 속성의 속성 값의 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.

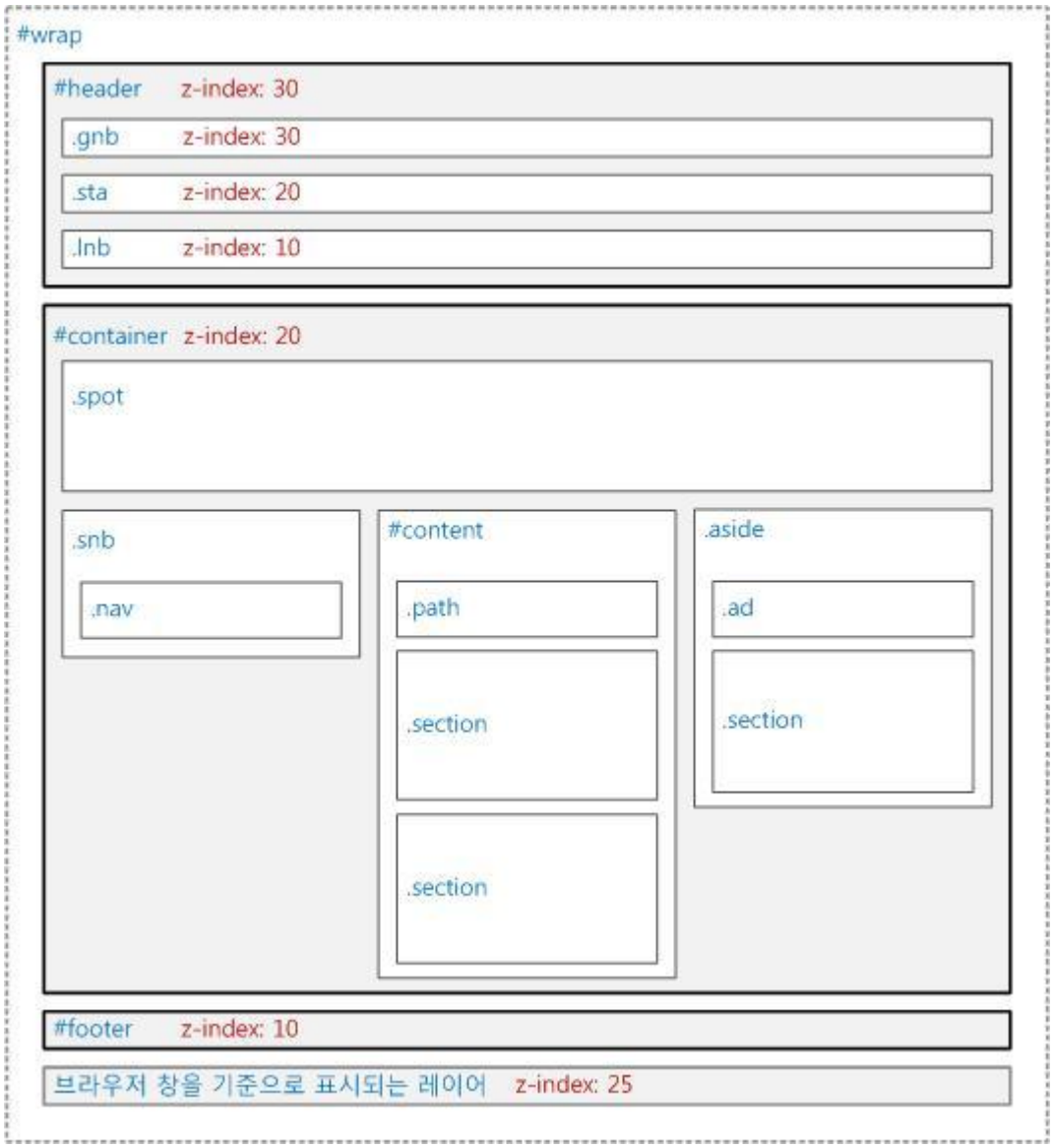

그림 5-1 z-index 권장 선언값

5.10 핵(hack)

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>IE6</td><td rowspan="4">S{*P:V}</td><td>S{_P:V}</td><td></td><td rowspan="3">S{_P:V}</td><td></td><td rowspan="3"></td></tr>
<tr><td>IE7</td><td>S,x:-moz-any-link, x:default {P:V}</td><td>*+html S{P:V}</td><td rowspan="4">S,x:-moz-any-link, x:default{P:V}</td></tr>
<tr><td>IE8</td><td>S{P:V₩0}</td><td></td><td></td></tr>
<tr><td>IE9</td><td>:root S{P:V₩0 !important}</td><td rowspan="4">:root S{P:V}</td><td></td><td>:root S{P:V}</td></tr>
<tr><td>Firefox</td><td rowspan="3"></td><td>S,x:-moz-any-link, x:default{P:V}</td><td rowspan="3"></td><td rowspan="3"></td></tr>
<tr><td>Safari/ Chrome</td><td></td></tr>
<tr><td>Opera</td><td></td><td></td></tr>
</table>

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

5.11 미디어 타입

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

인쇄를 위한 미디어 타입은 기본 CSS 파일의 가장 아랫부분에 선언한다.

```
...
...
/* 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)
- screen(컴퓨터회면)
- tty(고정폭 글자를 사용하기위한 미디어타입)
- tv(텔레비전 타입)

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 요소를 통해 HTML과 CSS 파일을 연결한다. 작성 예는 다음과 같다.

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

B. Internaltype

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 요소에 style 속성을 이용하여 스타일을 선언하는 방식으로, 속성 값이 동적으로 변경되어야 하는 경우 사용한다. 속성 값에 사용되는 %와 같은 특수기호는 Character entity references로 변환하지 않는다. 작성 예는 다음과 같다.

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

5.13 주석

5.13.1 기본 형식

CSS 주석은 아래와 같이 표기하며, 기본 형식에 맞게 작성한다.

```
/* 주석 내용 */
```

- 주석 기호와 주석 내용 사이에는 반드시 공백 한 칸이 있어야 한다.
- 주석 기호와 주석 내용 사이의 줄 바꿈은 허용하지 않는다. 단, 주석 내용 간 줄 바꿈은 허용한다.
- 종료 주석은 사용하지 않는다

주의

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

5.13.2 작성자 정보 표기

작성자 표기는 css인코딩 선언 다음줄에 1회만 작성하며, 작성자 정보에는 소속 부서, 영문 이름 이니셜, CSS 파일 생성 날짜(YYMMDD 형식)를 작성한다. 유지보수만 담당하는 경우라도 모두 기입한다. 작성자 정보 밑에는 빈 줄(한 줄)을 두어 스타일을 선언하는 문장과 구분되도록 한다. 동일 파일을 유지보수 하는 경우, 작성자 바로 아래 줄에 동일한 형식으로 작성한다.

```
@charset "utf-8";
/* NHN WSD1Team JJS 090707, NHN WSD2Team JJJ 090815 */
/* NHN WSD1Team KDW 120101 */
빈 줄
```

5.13.3 의미 있는 그룹 영역의 주석 표기

의미 있는 객체를 구분하기 위한 주석은 영역의 윗부분에 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{…}
```

5.14 파일 분기

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

- 한 서비스가 사용자 화면/어드민으로 구성되어 전체 레이아웃이 다를 경우
- 나눔글꼴을 적용할 경우(ex. 서비스명_nanum.css)
- 개편 시 개발상의 이슈로 이전에 분리되어 있던 CSS를 합칠 수 없는 경우
- 한 서비스 내에 단발성 페이지로 존재하나 CSS를 임베드하기 어려운 경우
- 태블릿 PC를 대응할 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, Helvetica,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}
.blind{visibility:hidden;overflow:hidden;position:absolute;top:0;left:0;width:1px;height:
1px;font-size:0;line-height:0}
```

모바일에서 초기 파일은 브라우저에 따라 서비스영문이름.css, 서비스영문이름_e.css 파일을 작성한다.

서비스영문이름.css

```
@charset "utf-8";
/* NHN Web Standard 2Team MJA 111025 */

/* 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-size:14px;line-height:1.25em}
body.s,.s input,.s textarea,.s select,.s button,.s table{font-family:Helvetica}
body{position:relative;-webkit-text-size-adjust:none}
img,fieldset{border:0}
ul,ol{list-style:none}
em,address{font-style:normal}
a{text-decoration:none}
table{border-collapse:collapse}
hr{display:none !important}
.blind,#u_skip{visibility:hidden;overflow:hidden;position:absolute;left:-
999em;width:0;height:0;font-size:0;line-height:0}
#content{clear:both;width:100%;}
#content::after{display:block;clear:both;height:1px;margin-top:-1px;content:''}
```

부록 A. 코딩 시 참고 사항

부록 A에서는 웹표준 기반의 마크업, 크로스 브라우징 범위, 폴더 생성 방법, 플래시 사용 시 유의점을 설명한다.

A.1 웹표준 기반의 마크업

A.1.1 웹표준 기반의 마크업

- **W3C 표준에 근거한 마크업**

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

- **의미에 맞는 HTML 요소를 사용하여 문서 구조 마크업**

  웹 문서의 내용을 HTML 요소의 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 Use가 가능하다. 또한 웹 접근성이 높아져 어떤 디바이스, 어떤 응용 프로그램을 이용하더라도 동일한 콘텐츠를 제공받을 수 있다.

- **구조와 표현을 분리**

  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.1.2 웹표준 기반의 마크업 프로세스

웹표준 기반의 마크업은 다음 그림과 같은 프로세스로 진행한다.

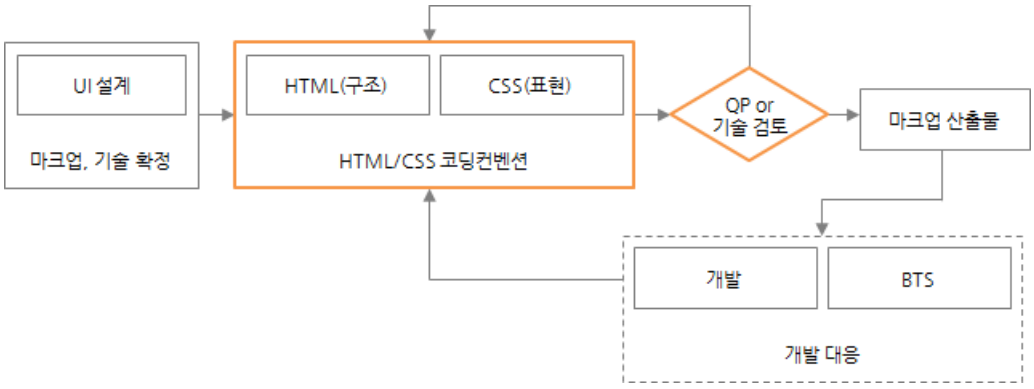

그림 A-1 웹표준 기반의 마크업 프로세스

A.1.3 웹 접근성 보장 방법

웹 접근성 보장하는 방법으로 아래 링크를 참고한다.

http://a11y.nhncorp.com/nwcag/index.html

A.2 크로스 브라우징 범위

크로스 브라우징 지원은 다음과 같이 3단계로 분류된다.

- 마크업 개발 범위: 마크업 개발 단계에 반드시 적용해야 하는 범위이다.
- QA 결함 대응 범위: QA에서 결함이 발생할 때 대응하는 범위로 마크업 스펙 질의/확정 단계에서 가감이 가능하다. 마크업 개발 범위를 포함한다.
- 서비스 예외 범위: 서비스 특성상 좀 더 확장된 사용자 층을 고려해야 하는 경우 적용해야 하는 범위이다. 네이버 메인과 고객센터가 이에 해당되며, 마크업 개발 범위, QA 결함 대응 범위를 모두 포함한다.

다음 표는 크로스 브라우징의 범위를 나타낸 것이다. 인터넷 익스플로러를 제외한 나머지 브라우저는 최신 버전을 기준으로 한다.

표 A-1 크로스 브라우징 범위

운영체제	브라우저	해당 범위		
윈도우 7	인터넷 익스플로러 8.x~9.x	마크업 개발 범위	QA 결함 대응 범위	서비스 예외 범위
	파이어폭스			
	사파리			
	크롬			
OS X(Mac)	파이어폭스			
	사파리			
윈도우 XP	인터넷 익스플로러 7.x			
윈도우 8	인터넷 익스플로러 10.x			
윈도우 비스타	인터넷 익스플로러 7.x~9.x			
윈도우 XP 윈도우 비스타	파이어폭스			
	사파리			
	크롬			
iOS	Safari			
	인터넷 익스플로러 6.0 이하 오페라			

A.3 폴더 생성

A.3.1 마크업 폴더 구조

마크업을 위한 폴더를 생성할 때 폴더 구조는 다음 그림과 같다. 필요에 따라 js(자바스크립트), flash(플래시) 폴더도 생성할 수 있다.

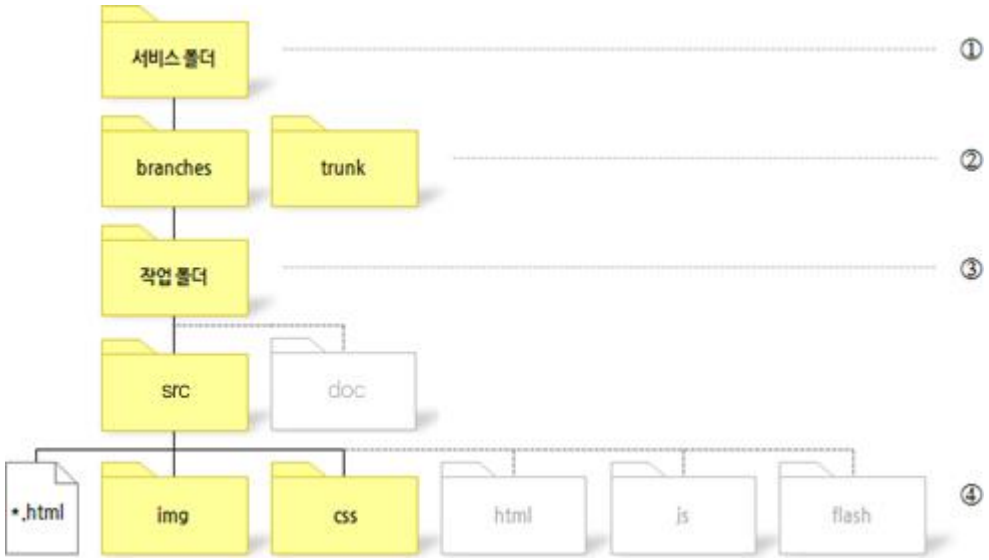

그림 A-2 마크업 폴더 구조

A.3.2 폴더 생성 시 고려사항

작업 폴더를 생성할 때는 아래와 같은 사항을 고려한다.

서비스 폴더 확인

서비스 폴더(그림 A-1 웹표준 기반의 마크업 프로세스의 ①에 해당)는 서비스 이름으로 된 폴더이다.

SVN 최상위 폴더(예: blog/m.blog)가 이에 해당

branches/ trunk 폴더 확인

모든 작업 폴더는 branches 폴더의 하위 폴더가 된다(신규 프로젝트, 서스테이닝, 프로모션 일반). 프로젝트, 서스테이닝 업무가 종료되면 기존에 있는 trunk 폴더를 백업(trunk_yymmdd)하고 작업 폴더를 ②에 복제하고 폴더 명을 trunk 로 변경한다.

작업 폴더 생성

작업 폴더(그림 A-1 웹표준 기반의 마크업 프로세스의 ③에 해당)는 업무 유형에 따라 프로젝트형, 서스테이닝형으로 구분한다. 네이밍 규칙(**오류! 참조 원본을 찾을 수 없습니다. 오류! 참조 원본을 찾을 수 없습니다.** 참조)에 맞는 폴더 이름을 정하여 서비스 폴더 아래에 작업 폴더를 생성한다.

작업 폴더 내 구성

이미지 폴더와 CSS 폴더는 기본으로 생성한다. HTML 파일의 개수가 많거나, 별도의 폴더로 그루핑하여 관리하는 것이 편리할 경우, HTML을 위한 하위 폴더(그림 A-1 웹표준 기반의 마크업 프로세스의 ④에 해당)를 생성할 수 있다.

A.4 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.

A.5 CSS Sprites 사용 방법

화면이 확대, 축소될 때 CSS 스프라이트의 배경 이미지 간 여백이 없으면 사방에 위치한 이미지가 보이는 경우가 있다. 웹 페이지의 구성요소가 모두 일정한 비율로 변경 되기 때문에, 이 과정에서 화면에 오류가 발생한다. 따라서 이미지 간 사방간격을 최소 1px 띄우도록 한다.

그림 A-3 CSS 스프라이트 배경 이미지 배치의 잘못된 예

그림 A-4 CSS 스프라이트 배경 이미지 배치의 예

A.6 단위 변환

표 A-3 단위 변환

px	em	px	em
1px	0.07em	21px	1.5em
2px	0.14em	22px	1.57em
3px	0.21em	23px	1.64em
4px	0.29em	24px	1.71em
5px	0.36em	25px	1.79em
6px	0.43em	26px	1.86em
7px	0.5em	27px	1.93em
8px	0.57em	28px	2em
9px	0.64em	29px	2.07em
10px	0.71em	30px	2.14em
11px	0.79em	31px	2.21em
12px	0.86em	32px	2.29em
13px	0.93em	33px	2.36em
14px	1em	34px	2.43em
15px	1.07em	35px	2.5em
16px	1.14em	36px	2.57em
17px	1.21em	37px	2.46em
18px	1.29em	38px	2.71em
19px	1.36em	39px	2.79em
20px	1.43em	40px	2.86em

※ 참고 웹 사이트 : http://pxtoem.com/

부록 B. 약속어 목록

부록 B에서는 네이밍 약속어를 설명한다.

B.1 네이밍 약속어

표 B-1 객체 약속어

분류	약속어	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 요소
	.ad_	광고 요소
	.paginate	페이지 목록 요소
	.ly	레이아웃 요소
	#u_skip	스킵네비게이션
	.blind	숨김영역
	.u_	공통요소
그루핑	.section_	heading 태그(h1~h6)를 포함한 요소들의 그루핑
	.group_	section 보다 낮은 단계의 heading 태그를 포함한 요소들의 그루핑
	. _area	위치에 제한이 없는 특정 기능을 수행하는 요소들의 그루핑

[조합 기호: "_"]

- 공통요소 약속어(.u_)는 공통업무 담당자에 한에서만 사용 한다.

표 B-2 이미지 약속어

약속어	분류
h_	제목
p_	문장
gnb_ , lnb_ , snb_	내비게이션
tab_	탭
btn_	버튼
bu_	불릿
ico_	아이콘

약속어	분류
line_	선
bg_	배경, 박스
img_	이미지
_off , _over , _on	상태 변화
ad_	광고
tmp_	임시
sp_	스프라이트 이미지

[조합 기호: "_"]