

CSS 관리 규칙_v1.3_060818

1. CSS 관리 규칙의 개요와 목적

CSS Based Design 은 본래 HTML 코드를 간결하게 만들고 유지보수가 용이 하도록 설계 된다. 그러나 코딩업무(디자인, 퍼블리셔)를 담당하고 있는 사람에 따라서 CSS의 네이밍 규칙이 다르게 부여되는 경우가 많고 팀 내의 다른 담당자는 이것을 처리하기 위하여 이전 작업자의 CSS 네이밍 규칙에 대한 이해를 필요로 하게 된다.

CSS 파일 작성에 대한 일관된 관리 지침을 정하고 이것을 활용하면 문서의 스타일 구조를 파악하기 쉬워지고 유지 보수작업을 더욱 빠르게 처리할 수 있다.

태그 재정의 규칙과 네이밍 규칙에서는 보편적으로 사용이 권장되는 방식의 스타일을 사전정의 하고 있다. 동일한 ID, Class를 지속적으로 부여하여 팀 내의 다른 담당자가 문서의 구조를 쉽게 파악하고 ID, Class 규칙을 이해하는데 소요되는 작업기간과 노력을 단축할 수 있도록 한다.

2. CSS 파일 생성 및 파일 네이밍 규칙

CSS 코딩 시 태그 재정의 스타일(default.css)을 가장 먼저 작성하도록 한다. 즉, 가장 보편적인 속성에 대한 정의로부터 시작하여 특수한 경우에 이르는 순서대로 작성하면 된다. 인라인 방식의 CSS는 모바일과 시각장애인용 페이지로 전환하고자 할 때 동적으로(서버사이드 측에서) CSS를 제거할 수 없으므로 사용하지 않도록 한다.

모든 페이지에서 참조하는 공통 파일은 태그 재정의 내용 위주로 작성하여 /open_content/common/css/default.css 으로 설정한다. 가능한 default.css 파일은 프로젝트 개시 전 미리 작성하도록 하고 중간에 수정되는 일이 없도록 한다. 화면 분할을 기준으로 하여 영역별(layout.css, navigation.css, content.css)로 CSS 코드를 분리하고 각각 별도의 파일로 관리하며, 각각의 섹션 별로 콘텐츠 영역에 대한 별도의 CSS 파일(content_섹션명.css)을 생성하여 관리 하도록 한다. 콘텐츠 영역에 대한 섹션별 CSS 파일의 명칭은 content_섹션명.css 형태로 생성하도록 하고 해당 섹션의 디렉토리 구조가 깊어지더라도 CSS 파일의 디렉토리 구조는 깊어지지 않도록 1섹션 1개 파일 처리 원칙을 준수한다. (예: /open_content/common/css/content_life.css)

스타일시트	/open_content/common/css/
공통(태그재정의)	/open_content/common/css/default.css
레이아웃	/open_content/common/css/layout.css
네비게이션	/open_content/common/css/navigation.css
콘텐츠영역 공통	/open_content/common/css/content.css
콘텐츠영역 섹션별	/open_content/common/css/content_섹션명.css
게시판	/open_content/common/css/board.css
프린트	/open_content/common/css/print.css

3. 외부 CSS 파일 불러오기 규칙

웹 사이트 전체를 관장하는 CSS 파일이 1개인 경우에만 Link 방식을 사용하고, 2개 이상으로 증가 하는 경우 Import 방식으로 대체 한다. CSS Import 는 반드시 웹 문서 헤더에서만 이루어 져야 하며 CSS 파일 내부에서 다른 CSS를 Import 방식은 취하지 않는다. 이는 현재 페이지에서 Import 되는 CSS 파일을 가시화 하고, CSS 관리방식을 일원화 하며, 불필요한 CSS 파일이 관계없는 페이지로 로드 되는 것을 방지하기 위한 목적이 있다.

4. ID, Class 네이밍 기본 규칙

1. ID, Class 모두 카멜 케이스(ex: globalBiz) 규칙을 기본으로 사용하되 예외(ex: D2M1)를 적용할 수 있다.
2. 영문 대/소문자와 숫자 그리고 “_” 만 사용 가능하며 가급적 “_” 의 사용은 자제한다.
3. 한 개의 단어로 구성된 경우 영문 소문자 작성을 기본으로 하며 숫자는 맨 앞에 나올 수 없다.
4. 2개 이상의 단어의 조합일 때 두 번째 이후에 연결되는 단어의 머리글자는 반드시 대문자 표기한다.
5. 2개 이상의 단어를 조합하는 방식은 큰 범주 + 작은 범주 순으로 한다.(ex: linkOn, linkOff)
6. 단어와 단어의 조합은 가능하지만 이를 연결하는 목적으로 “_” 를 사용하지 않는다.

7. “_” 는 문자와 숫자의 조합만을 위하여 사용하고 문자와 숫자의 조합일 때는 반드시 사용 한다.
8. 직관적으로 어떤 목적인지 알 수 있도록 의미 있는 단어를 사용해야 한다. 특히 top, bottom, left, right 와 같이 특정 위치나 방향을 지시하는 단어는 앞부분에 나타날 수 없고 오직 뒤에 붙어서 선행되는 단어의 의미를 보조하는 용도로만 사용한다. 그러나 원칙적으로 특정 위치나 방향을 의미하는 단어보다는 콘텐츠의 성질에 따라서 이름을 부여할 수 있다면 그것을 사용하는 방법이 바람직하다. 특정 위치나 방향을 지시하는 네이밍을 사용하는 경우 해당 콘텐츠의 위치가 변경될 때 ID 또는 Class 의 네이밍과 콘텐츠의 위치가 불일치 하여 혼동이 올 수 있기 때문이다.

5. 마크업 재 정의 규칙 (default.css)

기본 태그에 대하여 권장하는 방식으로 재 정의 되어있다. 아래 코드는 default.css 파일 안에 정의한다. 스타일 ‘속성:값’ 은 상황에 알맞게 수정/추가/삭제 하여 사용 할 수 있다. 모든 상황에 이것을 강제적용 할 수 없지만 가능하면 현재 방식을 최대한 응용하여 사용하기를 권장한다. default.css 의 경우 프로젝트 개시 전에 작성하여 프로젝트 중간에는 가급적 수정되는 일이 없도록 하여야 공동작업 시 문제가 발생하지 않는다. 가상 선택자의 경우 순서에 유의한다. L.V.H.A.F. 순 “링크 방문 시 지나치게 활동하면 주목 받습니다.”

Tag	속성 재정의	설명
*	{ margin:0; padding:0; font-size:small; font-family:돋움, Dotum, AppleGothic, sans-serif; }	/*전체 태그에 대하여 마진, 패딩, 서체크기, 서체종류 초기화*/
html, body	{ height:100%; }	/*브라우저 높이 값 초기화*/
body	{ margin: 20px; }	/*페이지 여백 초기화*/
a:link	{ text-decoration:none; }	/*기본 링크 초기화*/
a:visited	{ text-decoration:none; }	/*방문한 링크 초기화*/
a:hover, a:active, a:focus	{ text-decoration:underline; }	/*활성 링크 초기화*/
h1	{ display:block; }	/*CI 또는 사이트 로고를 정의*/
h2	{ }	/*h2 제목태그의 스타일 재정의*/
h3	{ }	/*h3 제목태그의 스타일 재정의*/
h4	{ }	/*h4 제목태그의 스타일 재정의*/
p	{ margin-bottom:1em; }	/*문단의 기본 마진*/
img	{ border:0; }	/*이미지 보더 초기화*/
label	{ cursor:pointer; }	/*라벨 텍스트에 대한 커서모양 설정*/
input, textarea	{ border:1px solid #CCC; padding:1px; cursor:pointer; }	/*input, textarea 보더 및 패딩 초기화*/
input.etc	{ border:none; padding:0; }	/*checkbox, radiobutton, image 타입에 적용*/
input.button	{ border:1px solid #999; padding:0; }	/*submit, button, reset 타입에 적용*/
select	{ border:1px solid #CCC; cursor:pointer; }	/*select 요소의 보더 및 패딩 초기화*/
ol, ul	{ margin-left:25px; }	/*기본 목록의 초기화된 마진을 블릿이 보이도록 재설정*/
hr	{ display:none }	/*분리선 초기화*/
table	{ border-top:1px solid #CCC; border-left:1px solid #CCC; }	/*기본 table 보더 초기화*/
table th, table td	{ padding:3px; border-bottom:1px solid #CCC; border-right:1px solid #CCC; }	/*기본 th, td 보더 초기화*/
table th	{ background:#EEE; }	/*기본 th 배경색 초기화*/

6. 레이아웃용 ID 네이밍 규칙 (layout.css)

아래 규칙은 공통 레이아웃을 위하여 사용하되 불필요한 ID는 생략하거나 추가로 필요한 ID는 생성 및 확장 할 수 있지만 기존의 ID 이름을 변경하는 것은 허락하지 않는다. XHTML 요소와 같은 이름의 ID는 만들 수 없다.

ID	적용대상	설명
#wrap	div	문서 전체를 감싼다.
#header	div	문서의 상단, 로고, 글로벌 네비게이션 따위.
#footer	div	문서의 하단, 저작권 따위.
#center	div	헤더와 풋터 사이에 존재하는 분할된 컬럼을 감싸준다.
#visual	div	비주얼 이미지 영역을 감싼다.
#content	div	페이지의 실제 본문영역을 감싼다.
#global	div	Global Navigation 영역.
#local	div	Local Navigation 영역.
#link***	div	*** 용도의 바로 가기 링크.

7. 네비게이션용 ID 네이밍 참조 (navigation.css)

네비게이션 ID 네이밍 작업에는 변수가 많기 때문에 현재의 룰을 모든 경우에 강제적용 할 수 없다. 참조하기 위한 규칙으로서 가능하면 따르되 따르지 않아도 무방하다. 현재의 ID 네이밍 규칙은 웹사이트 정보구조가 4단계 Depth 까지 있는 경우를 고려하여 제작 되었다. 메뉴는 하나의 페이지에 딱 한번만 출력하여 중복되는 것을 방지하고 상단에서 1~2Depth가 표현 되었다면 좌측에는 2~3Depth가 아닌 3~4Depth 를 표현하는 방식을 취한다. 글로벌 네비게이션의 경우 복잡성을 회피하고자 1Depth 메뉴에는 태그를 사용하고 2Depth 메뉴그룹에는 태그를 사용하였음에 유의할 것. 하지만 또는 태그를 사용하는 것은 순차적으로 표현할 필요가 있는지 또는 아닌지에 따라 결국 퍼블리셔의 주관적 판단에 따른다. 기본 규칙은 1Depth 는 순차적 목록으로 표현하고 2Depth 는 비 순차 목록으로 표현함을 원칙으로 하였다. ID 네이밍 뒤에 Tag 요소의 이름이 붙은 것은 현재 ID의 적용 대상이 무엇인지 직관적으로 판단하는 것을 돕기 위함이다.

D = Depth 를 의미하며 수직적 상하구조를 표현

M = Menu 를 의미하며 수평적 순서구조를 표현

MG = Menu Group 을 의미하며 수평적 메뉴구조의 묶음을 표현. ol, ul 에 적용됨.

ID (*)=숫자	적용대상	설명
#global	ol, ul	글로벌 메뉴의 또는 태그에 적용한다. 메뉴의 영역 과 위치 를 잡아준다.
#D1M*_li ~ #D1M**_li	li	1 Depth 메뉴의 목록 순서로서 *_li ~ **_li까지 사용할 수 있다. 메뉴의 영역 과 위치 를 잡아준다.
#D1M*_a ~ #D1M**_a	a	1 Depth 메뉴 태그인 의 자식태그 <a>에 적용한다. 메뉴의 영역 을 잡아준다. 마우스, 키보드에 반응 하여야 하고 서브메뉴 을 제어 한다.
#D2MG*_ul ~ #D2MG**_ul	ul	2 Depth 메뉴의 그룹인 에 부여된다. 메뉴의 위치 와 영역 을 잡아준다. 상위메뉴 <a>에 대한 마우스, 키보드 조작 으로 제어된다.
#D2M*_*_li ~ #D2M**_**_li	li	2 Depth 메뉴의 목록 순서로서 *_*_li ~ **_**_li 까지 사용할 수 있다. 첫 번째 숫자는 1 Depth의 메뉴 순서를 나타내고 두 번째 숫자는 2 Depth의 메뉴 순서를 나타낸다. 메뉴의 위치 와 영역 을 잡아준다.
#D2M*_*_a ~ #D2M**_**_a	a	2 Depth 메뉴 태그인 의 자식태그 <a>에 적용한다. 2 Depth 메뉴의 목록 순서로서 *_*_a ~ **_**_a 까지 사용할 수 있다. 첫 번째 숫자는 1 Depth의 메뉴 순서를 나타내고 두 번째 숫자는 2 Depth의 메뉴 순서를 나타낸다. 메뉴의 영역 을 잡아준다. 마우스, 키보드에 반응 하여야 한다.
#local	ol, ul	로컬 메뉴의 또는 태그에 적용한다. 메뉴의 영역 과 위치 를 잡아준다.

생략	li	ID가 필요 없음. 케스케이딩을 이용하여 Tag 요소에 직접 스타일 적용. 메뉴의 위치와 영역을 잡아준다.
#D3M*_a ~ #D3M**_a	a	3 Depth 메뉴의 목록 순서로서 * ~ **까지 사용할 수 있다. 메뉴의 영역을 잡아준다. 마우스, 키보드에 반응 하여야 하고 서브메뉴 을 제어한다.
#D4MG*_ul ~ #D4MG**_ul	ul	4 Depth 메뉴의 그룹인 에 부여된다. 메뉴의 위치와 영역을 잡아준다. 상위메뉴 <a>에 대한 마우스, 키보드 조작으로 제어된다.
생략	li	ID가 필요 없음. 케스케이딩을 이용하여 Tag 요소에 직접 스타일 적용. 메뉴의 위치와 영역을 잡아준다.
#D4M*_*_a ~ #D4M**_**_a	a	4 Depth 메뉴의 목록 순서로서 *_*_a ~ **_**_a 까지 사용할 수 있다. 첫 번째 숫자는 3 Depth의 메뉴 순서를 나타내고 두 번째 숫자는 4 Depth의 메뉴 순서를 나타낸다. 메뉴의 영역을 잡아준다. 마우스, 키보드에 반응 하여야 한다.

8. 콘텐츠용 Class 네이밍 참조 (content.css)

네이밍 기본 규칙을 따른다. 하나의 페이지에 두 번 이상 사용 되어 ID로 정의할 수 없는 디자인 패턴을 정의한다. 클래스는 필요한 만큼 추가로 확장해서 사용 가능하지만 XHTML 요소와 같은 이름의 Class는 만들 수 없다. 스타일 가이드를 참조하여 일관된 디자인 패턴이 있는 경우 Class로 정의해 두고 특정 페이지만을 위한 패턴도 재활용 가능성을 위하여 Class로 정의 하되 가급적 일관된 네이밍 룰을 스스로 만들어 적용하고 주석을 이용하여 간략하게 설명해 주도록 한다. 주석을 이용하여 설명하지 않으려면 최소한 어떤 클래스 인지 이름 만으로도 추측할 수 있도록 의미 있는 단어를 사용하고 일관된 네이밍 룰을 지켜야 한다.

특정 위치나 방향을 의미하는 이름을 부여하는 것은 가급적 피하도록 하고 가능하면 콘텐츠의 내용과 관련된 이름으로 네이밍 하는 것이 좋다. 특정 위치나 방향을 의미하는 이름을 짓는 경우 콘텐츠의 위치가 변경될 때 Class 이름 때문에 공동작업자가 혼란에 빠질 수 있다.

아래 예제는 경험상 자주 필요로 하던 Class 정의 예제들로서 content.css 파일에 정의하는 것이 적당하지만 특정 위치나 방향 또는 한가지 속성만 재정의 되어 있기 때문에 Class를 남발할 우려가 있으므로 되도록 사용하지 않는 것이 좋고 문서의 디자인 패턴이 면밀하게 분석/정의 되어 있을 수록 사용할 확률은 줄어들게 된다.

CLASS	속성 재정의	설명
.color666	{ color:#666; }	/*진한 회색*/
.color999	{ color:#999; }	/*중간 회색*/
.colorCcc	{ color:#CCC; }	/*흐린 회색*/
.alignLeft	{ text-align:left; }	/*좌측정렬*/
.alignRight	{ text-align:right; }	/*우측정렬*/
.alignCenter	{ text-align:center; }	/*수평 중앙정렬*/
.alignMiddle *	{ vertical-align:middle; }	/*자식 요소의 수직 중앙정렬*/
.noBgImg	{ background-image:none; }	/*배경이미지 제거*/
.noBorderL	{ border-left:none; }	/*좌측 보더 제거*/
.noBorderR	{ border-right:none; }	/*우측 보더 제거*/
.noBorderT	{ border-top:none; }	/*상단 보더 제거*/
.noBorderB	{ border-bottom:none; }	/*하단 보더 제거*/

9. 프린트용 CSS 코드 작성요령 (print.css)

글로벌 네비게이션을 포함한 헤더 영역, 로컬 네비게이션 영역, 풋터 영역은 출력하지 않도록 #header, #footer, #local { display:none; } 으로 처리하고, 문서에서 print.css 파일을 링크할 때 media="print" 으로 설정한다. print.css 는 인쇄하는 경우에만 유효한 코드가 되어 불필요한 영역은 숨기고 #content 영역만 출력할 수 있게 된다.