CSS

1. CSS 개요

(1) CSS(Cascading Style Sheets) :

웹페이지에 자주 적용되는 부분들은 미리 정리하고 조합하여 정의한 후
페이지 내부에서 필요할 때마다 해당 부분을 적용하는 스타일시트 중 하나.
HTML로는 부족한 레이아웃이나 폰트 등에 다양성을 부여한다.

- 폰트 크기는 지정하는 수치에 따라 마음대로 조절 가능
- 자간/행간 배치가 자유로워 가독성을 높인다.
- 링크 상의 밑줄 변형이 자유롭다.
- 페이지의 여백을 원하는 만큼 만들어 줄 수 있다.

2. CSS 기본동작

(1) CSS 사용하는 방법 3가지

① <head></head> 사이 기술
② <body> 안에서 직접기술
③ *.css(파일)로 따로 저장

(2) 기본구성

```
<HEAD>
 <STYLE type="text/css">  ---> 스타일의 유형이 텍스트이고 그 파일은 css 이라는 뜻.
    선택자 {속성1: 값1; 속성2: 값2;}  선언문(속성과 값)간의 구분은 ;(세미콜론)이다.
 </STYLE>
</HEAD>
```

(3) CSS의 특성

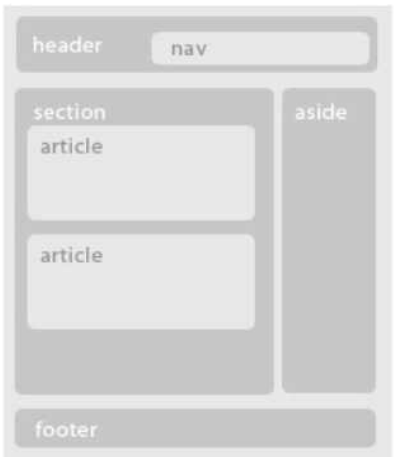
① 기존의 HTML 기능을 확장해서 사용할 수 있다.
② 한 번의 지정으로 웹 문서 모든 곳에 적용할 수 있다.
③ CSS는 상위의 기능을 상속받을 수 있다.
④ 복잡하지 않고 쉽게 파악할 수 있어 관리가 용이하다.

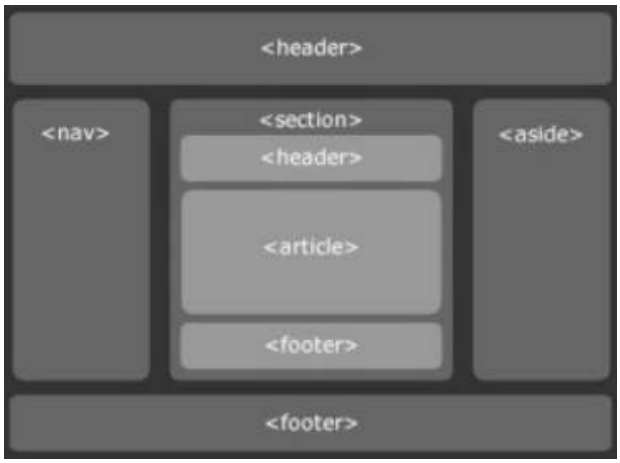
(4) 길이의 단위

- 1em = 기본 1개 글자 크기
- 부모의 길이가 정해져 있지 않을 때.
  1em = 16px
- 부모의 길이가 정해져 있으면 그 값이 기준
  1em = 12px

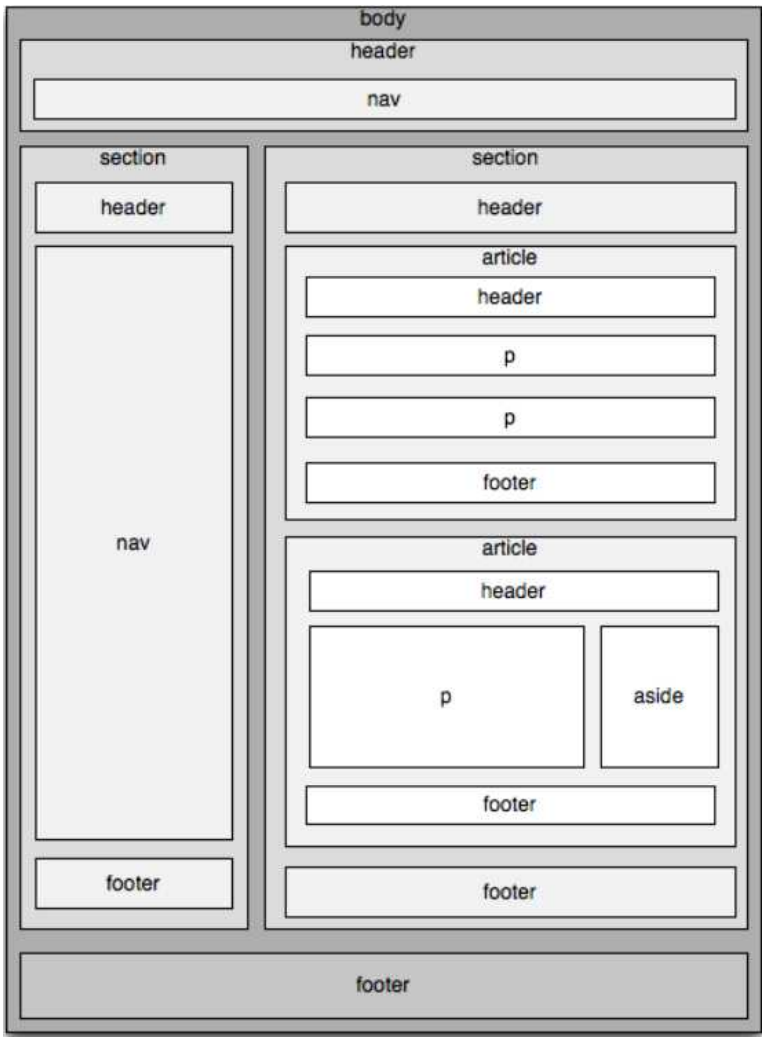
```
body { font-size : 12px; }
p { font-size : 1.1em; }
```

(5) HTML 기본 화면 구조





-



HTML5 기본 문법

HTML (HyperText Markup Language)은 웹페이지를 기술하기 위한 마크업 언어이다.

- HTML5 문서는 반드시 <!DOCTYPE html>으로 시작하여 문서 형식(document type)을 HTML5로 지정한다.
- 실제적인 HTML document은 2행부터 시작되는데 <html>과 </html> 사이에 기술한다.
- <head>와 </head> 사이에는 document title, 외부 파일의 참조, 메타데이터의 설정 등이 위치하며 이 정보들은 브라우저에 표시되지 않는다.
- 웹브라우저에 출력되는 모든 요소는 <body>와 </body> 사이에 위치한다.

HTML 요소는 시작 태그(start tag)와 종료 태그(end tag), 태그 사이에 위치한 content로 구성된다.

HTML5의 템플릿은 이전에 비해 매우 간소화 되었다. 기존의 방식이 에 의존하여 콘텐츠를 구조화 시켰다면 HTML5에서는 구조화를 위한 요소들이 등장하여 좀 더 시멘틱한 웹이 가능해졌습니다

```
<body>
    <header></header> <!-- 첫머리 -->
    <nav></nav> <!-- 메뉴 -->
    <article></article> <!-- 본문 -->
    <footer></footer> <!-- 끝머리 -->
</body>
```

HTML을 공부하기 위해서는 기본적으로 시맨틱 태그의 화면구조를 이해해야 한다.
크게는 head와 body로 body영역의 header / section / footer가 있다.
HTML4는 <div id="footer>형식으로 모든 구조를 만들지만
HTML5에서는 <header><footer><nav><article>등으로 시맨틱 태그를 쓰고 div는 css를 넣을 때 사용된다

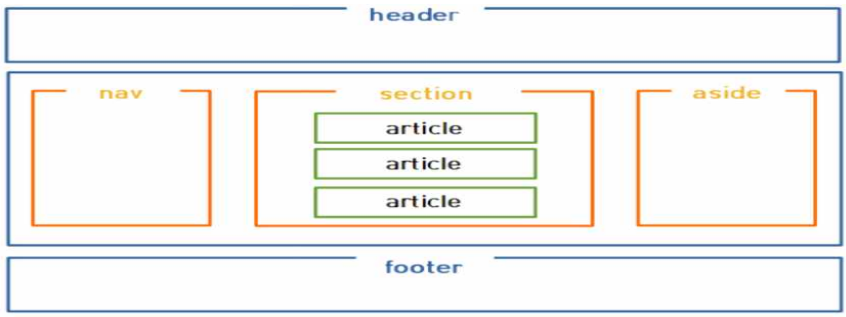
<!doctype>
문서 유형을 지정해주는 태그이다. 현재 문서가 HTML5 언어로 작성된 웹 문서라는 뜻이다

<html>
웹 문서의 시작과 끝을 알려준다. 태그 사이의 소스를 읽어 HTML 문법에 맞추어 브라우저에 표시한다.

<head>
웹브라우저가 웹 문서를 해석하기 위해 필요한 정보를 입력하는 부분이다.
실제 문서의 내용이 아니기 때문에 제목만 표시되고 나머지는 브라우저에 표시되지 않는다.
스타일이나 스크립트 등도 이곳에 포함된다.

메타데이터(metadata)
HTML 문서에 대한 정보(data)로 웹브라우저에는 표현되지 않는 정보를 의미한다
<title> <style> <meta> <link> <script> 태그 등이다

<body>
실제로 웹브라우저 화면에 표시되는 내용이다. 또 대부분의 시맨틱 태그들이 이 영역에서 사용된다.
<header> <section> <nav> <article> <footer> 태그들이 화면을 구성한다.



HTML5의 꽃이라 불리는 body 영역 안에 위 사진과 같이 화면을 나누기 위해 시맨틱 태그를 이용한다. 자주 방문하는 포탈 사이트인 네이버나 다음을 방문해봐도 이렇게 구성되어 있다.

<header>
위에서 언급한 <head>는 <html> 바로 밑에 쓰이지만 <header>는 <body> 안에 있기 때문에 둘은 전혀 다릅니다. <header>는 주로 머리말, 제목을 표현하기 위해 쓰인다

<nav>
HTML5에서 새롭게 생긴 시맨틱 태그이고 네비게이션이라고 불린다.
콘텐츠를 담고 있는 문서를 사이트간에 서로 연결하는 링크의 역활을 담당합니다. <nav>는 주로 메뉴에 사용되고 위치에 영향을 받지 않기 때문에 어디에서든 사용이 가능하다.

<section>
<body>영역은 콘텐츠를 <header> <section> <footer>의 3가지 공간에 콘텐츠를 저장하는데 그 중 <section>은 본문 콘텐츠를 담고 있습니다. <section>안에 <section>을 넣는 것도 가능하다.

<article>
<section>이 콘텐츠를 분류한다면 이 <article> 태그 안에는 실질적인 내용을 넣는다.
뉴스로 예를 들면 정치 / 연예 / 사회의 대분류는 <section>이고, 정치의 기사 내용과 연예의 기사 내용들을 <article>에 넣는 것이다.

<aside>
사이드바라고 부르기도 하며, 본문 이외의 내용을 담고 있는 시맨틱 태그이다.
주로 본문 옆에 광고를 달거나 링크들을 이 공간에 넣어 표현한다.

<footer>
화면의 구조 중 제일 아래에 위치하고, 회사소개 / 저작권 / 약관 / 제작정보 들이 들어간다.
연락처는 <address> 태그를 사용하여 표시한다.

** **
위 사진에는 없지만 는 HTML5에 와서 글자나 사진 등 콘텐츠들을 묶어서 CSS 스타일을 적용시킬 때 사용한다.

스타일시트(CSS) 파일을 로드 하는 방법

HTML 문서의 디자인을 담당할 스타일시트(CSS) 파일을 로드 하는 방법은 **link 태그**를 사용하는 방법과 **@import** 를 사용하는 방법 등 두 가지가 있다.

첫 번째 - link 태그

<link rel="stylesheet" type="text/css" href="css/style.css">
link 태그는 기본적으로 외부에 있는 리소스를 가져오는 태그이다.
head 섹션에 들어가 있어야 합니다.
rel 을 stylesheet 로 지정하고 href 에 해당 CSS 파일의 절대주소 혹은 상대주소를 넣어주면 CSS 파일을 로드 할 수 있다.

두 번째 - @import

```
<style type="text/css">
    @import url("css/style.css");
</style>
```

link 태그와 마찬가지로 head 섹션에 들어가 있어야 하며, head 섹션 내부에 style 태그를 만들고 그 안에서 @import 를 사용해야 한다.

셀렉터

(1) CSS 기본 셀렉터

- 태그 이름으로 접근한다.

예) h1 {.....}

=> h1 태그의 속성을 지정한다.

(2) CSS 자식 셀렉터

- ">" : 자식 셀렉터, HTML 태그의 계층 구조를 표현한다.

예) p > div > fieldset > #pwd {.....}

=> p태그 안의, div태그 안의, fieldset태그 안의, id가 pwd인 태그

```
<p>
    <div>
        <fieldset>
            <input type="text" >
            <input type="password" id="pwd">
            <input type="text" >
        </fieldset>
    </div>
</p>
```

(3) CSS 후손 셀렉터

- 후손 셀렉터는 태그의 계층 관계를 나타낸다.
- 자식 셀렉터와 다르게, 명시과정에서 중간단계를 생략해도 된다.

예) p div #pwd {.....}

=> p 태그하위의, div 태그안의, id가 pwd인 태그

```
<p>
    <div>
        <fieldset>
            <input type="text" >
            <input type="password" id="pwd">
            <input type="text" >
        </fieldset>
    </div>
</p>
```

(4) 특정 요소를 구체적으로 서술하기

- HTML 태그와 class 또는 id 속성과 함께 명시된다.
- 특정 요소를 더욱 정교하게 명시하고자 할 때 사용한다.
- "A#B" : id 속성이 B인 A태그
- "A.B" : class 속성이 B인 A태그

예) p div input#pwd {.....}
=> p태그 안의, div태그 안의, id가 pwd인 input태그

```
<p>
    <div>
        <fieldset>
            <input type="text" >
            <input type="password" id="pwd">
            <input type="text" >
        </fieldset>
    </div>
</p>
```

(5) HTML 태그의 속성에 따른 구분

- CSS 셀렉터가 적용되는 대상에게 특정 속성이 있는지 여부와 특정 속성의 값이 적용되어 있는지 여부에 따라서 태그 요소를 좀 더 구체적으로 가리킬 수 있다.
- selector[속성] : []안에 명시된 태그 속성을 갖는 요소를 의미한다.
- selector[속성=값] : []안에 명시된 태그 속성이 지정된 값을 갖는 요소를 의미한다.

예) input[type='password'] {.....}
=> input 태그 중에서 type이 'password'인 input 태그

```
<p>
    <div>
        <fieldset>
            <input type="text" >
            <input type="password" id="pwd">
            <input type="text" >
        </fieldset>
    </div>
</p>
```

(6) 가상 클래스 기법

가상 클래스	의미
:first-child	여러 개의 반복되는 태그들중에서 첫 번째 요소
:last-child	여러 개의 반복되는 태그들중에서 마지막 요소
:link	링크의 기본 상태
:active	요소 위에서 마우스가 눌려져 있는 상태
:hover	요소에 마우스가 올라가 있는 상태
:visited	방문했던 경험이 있는 링크
:focus	요소에 포커스가 지정된 상태
:checked	체크박스나 라디오버튼 등이 체크된 상태
:selected	드롭다운에서 선택된 요소

예) 링크의 글자 색상 적용
a:link {color : #555555; }
a:visited {color : #555555; }
a:active {color : #555555; }
a:hover {color : #555555; }

예) focus 유무에 따른 배경색 설정
input {background-color: #ffffff; }
input:focus {background-color: #555555; }

예) 마우스 위치에 따른 배경 이미지 설정
.menu_item {background-image: url(기본이미지); }
.menu_item:hover {background-image: url(롤오버이미지); }

(7) CSS 선택자 우선순위

특정 태그에 스타일 속성이 중복될 경우, 어떤 속성이 적용될까?
즉 스타일에 우선순위를 두어서 HTML의 요소가 어떤 CSS 스타일에 영향을 받을지 결정하는 것이다

- 1순위. 속성 값 뒤에 !important를 붙인 속성
- 2순위. HTML에서 style을 지정한 속성 (inline으로 style 속성 지정)
  인라인 스타일 속성을 쓰면 코드 유지가 어려우므로 권장하지 않는다.
- 3순위. id로 지정된 속성
- 4순위. class로 지정된 속성
- 5순위. 태그 이름으로 지정된 속성
- 6순위. 상위 객체에 상속된 속성

색과 텍스트 꾸미기

1. 배경(background) 스타일

(1) background 속성

- 배경에 관련된 여러 가지 속성을 포함한다.
- background 속성은 아래의 여러 속성 들을 지정할 수 있다.
- 한 요소에 여러 개의 배경 이미지를 적용할 수 있다.

```
.bg{
    width: 600px;
    height: 400px;
    border: 3px solid;
    background:
        url(img/bg_a.png) no-repeat 0 0,
        url(img/bg_b.png) no-repeat 100% 0,
        url(img/bg_c.png) no-repeat 0 100%,
        url(img/bg_d.png) no-repeat 100% 100%,
        url(img/bg_e.png) no-repeat 40px 40px;
}
```

- background-color : 색상 [red, rgba(0, 255, 0, 0.4)]
- background-image : 이미지 경로 [url(img/star.png)]
- background-repeat : 이미지의 반복 지정
  [repeat : 반복, no-repeat : 반복안함, repeat-x : x축으로 반복, repeat-y : y축으로 반복]
- background-attachment : 이미지의 고정
  [scroll : 기본값, 내용이 스크롤될 때 같이 움직인다, fixed : 내용이 스크롤 되더라도 안 움직인다.]
- background-position : 이미지의 위치
  [가로 : left, center, right, 세로 : top, center, bottom, 숫자 : px, %]
  background-position : left top; /* 가로 세로 */
- background-size : 이미지의 크기 조절
  background-size : 100%; /* width 100% */
  background-size : 100% 100%; /* width 100%, height 100%*/
- background-clip : 배경을 'content' 영역에만 보이게 하는 속성
  => 배경은 padding 영역까지 보이지만, 이 속성으로 padding 영역의 배경을 보이지 않게 지정한다.
  => padding 영역의 배경은 잘린다.
  background-clip : content-box;
- background-origin : 배경을 'content' 영역에 맞추어 보이게 하는 속성
  => padding 영역의 배경에 잘리지 않는다
  => 배경이미지가 content-box 영역에 잘리지 않고 정확히 나타난다.
  background-origin : content-box;

2. 서체(font)와 글(text) 스타일

(1) 서체
  1) font-family 속성
    - 서체를 지정
    - 예) body { font-family : '돋움', dotum, helvetica, sans-serif; }
  2) @font-face
    - 서체를 같이 올려두고 연결하는 방법
    - 예) <style type="text/css">
```
@font-face{
    font-family: 'Nanum Gothic';
    src:url(fonts/NanumGothic.eot);
    src:url(fonts/NanumGothic.eot?#iefix) format('embedded-opentype'),
        url(fonts/NanumGothic.woff) format('woff'),
        url(fonts/NanumGothic.ttf) format('truetype')
}
p{
    font-family: 'Nanum Gothic', sans-serif;
}
</style>
```

(2) 글자 크기 지정
  1) font-size
    - 예) p { font-size: 2em; }

(3) 글자 두께 지정
  1) font-weight
    - 기본 두께 (normal)
```
font-weight: normal;
font-weight: lighter;
font-weight: 100;
…
font-weight: 500;
```
    - 볼드체 (bold)
```
font-weight: bold;
font-weight: bolder;
font-weight: 600;
…
font-weight: 900;
```

(4) 글자 장식
  1) text-decoration
    - none : 장식이 없다.
    - underline : 밑줄
    - overline : 윗줄
    - line-through : 가운데 지나가는 선
    - blink : 깜빡거림 => 더 이상 지원 안함

(5) 글자 스타일 지정
  1) font-style

- italic : 기울임 - oblique : 비스듬하게 - normal : 기울어진 글자를 바로 세움
- italic의 경우 italic체로 디자인된 폰트를 사용하는 것이고
  oblique는 normal폰트를 단순히 기울기만 한 것이다.

(6) 글자색 지정
1) color
- 예) p { color: red; }

(7) 글 정렬 지정
1) text-align
- left : 좌측정렬 - right : 우측정렬 - center : 중앙정렬 -justify : 양쪽정렬

(8) 수직 정렬 지정
1) vertical-align
- 인라인 요소끼리의 위, 아래 간격을 맞출 수 있다.
- <img> 요소는 블록 요소 안에 있을 경우 약간의 공백이 발생할 수 있는데,
  vertical-align을 이용해서 그 공백을 없앨 수 있다.
  예) img { vertical-align: top; }
- vertical-align: baseline; /* 글자의 baseline에 맞춤 */
- vertical-align: sub; /* 부모요소의 아래 첨자 위치로 맞춤 */
- vertical-align: super; /* 부모요소의 위 첨자 위치로 맞춤 */
- vertical-align: top; /* 부모요소의 상단에 맞춤 */
- vertical-align: text-top; /* 부모요소의 글꼴 요소의 상단에 맞춤 */
- vertical-align: middle; /* 부모요소의 소문자를 기준으로 중간에 맞춤 */
- vertical-align: bottom; /* 부모요소의 아래쪽에 맞춤 */
- vertical-align: text-bottom; /* 부모요소의 글꼴의 아래쪽에 맞춤 */

(9) 글의 간격 조절
1) letter-spacing
- 예) p { letter-spacing: 0.1em; }
2) word-spacing
- 예) p { word-spacing: 0.3em; }
3) line-height
- 줄이 여러 줄일 때 줄 간격 지정
- 예) p { line-height: 1.6; } /* 1.6배 */

(10) 글자 들여쓰기
1) text-indent
- 예) p { text-indent: 1em; }

(11) 영문의 대소문자 지정
1) text-transform
- 예) h1.uppercase{text-transform: uppercase;} /* 대문자로 */
  h1.lowercase{text-transform: lowercase;} /* 소문자로 */
  h1.capitalize{text-transform: capitalize;} /* 첫 글자를 대문자로 */
2) text-variant
- 예) h1.smallcaps{font-variant: small-caps;} /* 대소문자 섞여 있을 때,
  소문자를 대문자로 변경하고, 크기는 소문자 크기 */

(12) 글의 줄 바꿈 처리

=> 인라인 요소인 '글(text)'은 자신을 감싸고 있는 상위 블록 요소의 'width'를 넘어서게 되면 줄을 바꾸게 된다.

1) word-break
- 단어를 깨뜨려 줄바꿈을 지정
- 예) <style>

```
body{
    width: 400px;
    font-family: '돋움',dotum,helvetica,sans-serif;
    font-size: 12px;
    text-align: justify;
}
.area{
    background-color: #ddd;
    word-break: break-all;      /* text-align: justify;와 같이 지정 */
}
</style>
```

2) white-space
- 줄바꿈을 금지하거나, <pre> 요소의 특성을 부여
- nowrap : 줄바꿈 금지
- pre : <pre> 요소를 지정한 것처럼 띄어쓰기나 줄바꿈 등이 작성한 그대로 표현된다.
- pre-wrap : 앞의 pre의 효과와 비슷하지만, 지정한 영역을 넘어가지 않는다.
- pre-line : pre-wrap과 비슷하지만, 띄어쓰기한 공백은 한 칸만 표현된다.
- 예) <style>

```
.nowrap{
    white-space: nowrap;
}
.pre{
    white-space: pre;
}
.pre-wrap{
    white-space: pre-wrap;
}
.pre-line{
    white-space: pre-line;
}
</style>
```

(13) 글꼴 여러 개 설정 시
글꼴을 여러 개 설정할 때는 쉼표로 구분한다.

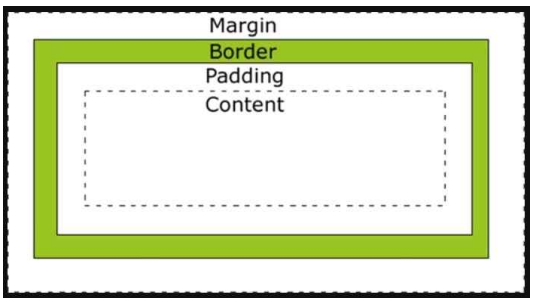
font-family: Georgia, "Times New Roman", serif;
로 설정했을 때, 제일 먼저 Georgia 글꼴을 찾는다. 해당 글꼴이 있다면 사용하고, 없다면 Times New Roman 글꼴을 사용한다. 그 글꼴도 없다면 웹브라우저에서 설정한 명조 계열의 글꼴을 사용합니다.

글꼴은 접속한 기기에 설치되어 있는 글꼴을 사용한다. 따라서 CSS로 설정한 글꼴이 없을 수도 있으므로, 마지막은 generic-family로 정해두는 것이 좋다.

박스 모델

1. 박스 모델

- HTML의 각 요소들은 사각의 박스 형태를 가진다.
- 이 박스를 표현하기 위해 지정하게 되는 여러 요소들을 한데 묶어 '박스 모델'이라고 한다.
- content : 내용
- border : 테두리
- margin : 테두리를 기준으로 바깥쪽 여백
- padding : 테두리와 content 사이의 안쪽 여백



(1) width, height 속성
  - width, height 속성이 지정하는 크기는 content의 크기를 나타낸다.

(2) border 속성
  - 테두리 표현
  - width(두께), style(형태), color(색상)를 적용할 수 있다.
  - 색상을 지정하지 않으면, 해당 요소의 글자색과 같은 색이 적용된다.

(3) margin 속성
  - 테두리 바깥 여백
  - margin: 10px;
    => border-width와 같이 top, right, bottom, left 방향으로 지정
  - margin을 활용한 가운데 정렬 방법
    - margin속성에서 right와 left의 값을 'auto'로 설정하면, 중앙의 '가운데' 정렬을 지정할 수 있다.
    - top과 bottom은 auto로 지정해도 0과 같이 출력되므로, 'margin: auto;'라고 지정해도 된다.
    - 해당 영역에는 width값이 지정되어 있어야 한다.

(4) padding 속성
  - 테두리의 안쪽 여백
  - border-width와 같이 top, right, bottom, left 방향으로 지정

배치

1. display, visibility 속성

(1) display 속성

1) 기본 속성

- HTML 요소를 블록 혹은 인라인으로 지정한다.
- 인라인 요소는 width를 가질 수 없고, 블록 요소는 텍스트 정렬이 적용되지 않는다.
- inline-block은 width 지정도 가능하고, text-align 속성을 지정하면, 정렬도 적용된다.
  게시판 하단의 페이징(1, 2, 3 ...) 부분을 꾸미는데 유용하다.
- display : inline /* 인라인 지정 */
- display : block /* 블록 지정 */
- display : inline-block /* 인라인 블록 지정 */
- display : none /* 존재하지 않는 것으로 만든다. */

2) 목록 표현

- 지정하는 요소가 인라인 요소라도 목록과 같이 블록 형태로 나타낸다.
- display : list-item /* 해당 요소가 목록의 형태로 나타난다. */

3) 테이블 형태 표현

- display : table /* table 지정 */
- display : table-caption /* caption 지정 */
- display : table-cell /* td 지정 */
- display : table-column /* col 지정 */
- display : table-column-group /* colgroup 지정 */
- display : table-footer-group /* tfoot 지정 */
- display : table-header-group /* thead 지정 */
- display : table-row /* tr 지정 */
- display : table-row-group /* tbody 지정 */

(2) visibility 속성

- 기본값은 'visible'이며 'hidden'을 지정할 경우에는 지정된 요소가 보이지 않게 된다.
- 'display: none;'은 원래 없었던 것처럼 표현되지만, 'visibility: hidden'은 보이지만 않고 공간은 유지된다.

2. 포지셔닝(positioning) 스타일

(1) float 속성

1) left, right

- float를 지정하면 다음에 오는 컨텐츠가 float를 지정한 블록 주위를 감싸게 된다.
- html 요소 중에서 인라인 요소나 블록 요소는 그 결과가 무척 다르다.
  그러나, float를 지정하면 두 요소는 같은 결과를 보여준다.
  인라인 요소가 블록 요소처럼 크기를 지정할 수 있게 된다.

2) clear 속성

- float를 해지한다.
- clear: left; /* float: left; 해지 */
- clear: right; /* float: right; 해지 */
- clear: both; /* float: left;, float: right; 둘 다 해지 */

3. position 속성

- 값으로는 absolute, relative, fixed의 3가지가 있다.
- 이렇게 설정된 후에는 left, right, top, bottom 속성으로 그 위치를 지정하게 된다.

(1) position: absolute;

- HTML 문서에 2개 이상의 요소가 나열되면, 뒤에 배치되는 요소는 앞에 놓인 요소를 기준으로 배치된다. 하지만, 'position: absolute'으로 지정하면, 다음 요소가 이 블록을 인지하지 못한 채 오로지 주어진 절대값을 기준으로 배치된다.
- 'position: absolute' 지정한 후, left, right, top, bottom 속성으로 그 위치를 지정할 수 있다.

(2) position: relative;

- 기준값 기준으로 상대적으로 배치된다.
- 'position: relative'로 지정된 후, left, right, top, bottom의 위치 값들은 relative가 지정된 블록 그 자신의 위치에서 지정한 값만큼 이동하게 된다.
- 'position: absolute'를 지정하는 경우, 이를 감싸는 블록에는 'position: relative'로 지정해야 한다.

(3) position: fixed;

- 위치는 항상 <body>를 기준으로 위치된다.
- 위치는 고정이다.

(4) z-index

- position을 지정한 여러 블록이 겹치게 되면, 나중에 작성한 블록이 위로 올라가게 된다.
- z-index 속성으로 블록의 위, 아래의 위치를 조정할 수 있다.
- z-index의 큰 값이 위로 올라간다.
- 기본값이 0, 음수 ~ 양수까지 줄 수 있다

Position Property

Position이란 문서상에 요소를 배치하는 방법을 지정한다

- static : default값
  top, right, bottom, left, z-index 속성이 아무런 영향을 주지 않는다
- relative : **자기 자신을 기준**으로 top, right, bottom, left의 값에 따라 오프셋을 적용한다
- absolute : 현재 위치랑 무관하게 **부모 tag의 위치를 기준**으로 상대적으로 배치한다.
- fixed : **뷰포트를 기준**으로 상대적으로 위치가 지정된다.
  화면이 스크롤을 하더라도 항상 그 위치에 고정된다.
  주로 헤더, 플로팅 배너나 우측 하단에 스크롤 탑 버튼에 활용된다.
- sticky : 문서 흐름에서 유지하고 마치 포스트 잇처럼 붙어 다닌다
  가장 가까운 스크롤 되는 부모에 귀속된다.

inline

- 줄바꿈 없이 순서대로 한 줄에 다른 엘리먼트들과 나란히 배치된다
- 컨텐트의 크기 만큼만 공간을 차지하므로 width, height 속성을 지정해도 무시된다
- padding, margin 속성은 좌우 간격만 반영이 되고 상하 간격은 반영되지 않는다
- , <a>, 등

block

- 혼자 한 줄을 차지한다
- 매번 줄바꿈이 되어 여러 줄에 보이게 된다
- width, height, padding, margin 속성이 모두 반영된다
- , <p>, <h1> 등

inline-block

- inline 엘리먼트처럼 전후 줄바꿈 없이 한 줄에 다른 엘리먼트들과 나란히 배치된다.
- inline에서 불가능하던 width, height, padding, margin 속성의 상하 간격 지정이 가능하다.
- inline-block을 이용하면 여러 개의 엘리먼트를 한 줄에 정확히 원하는 너비만큼 배치할 수 있기 때문에 레이아웃에 활용할 수 있다.
- <button>, <select> 등

visibility 속성

- 기본값은 'visible'이며 'hidden'을 지정할 경우에는 지정된 요소가 보이지 않게 된다.
- 'display: none;'은 원래 없었던 것처럼 표현되지만,
  'visibility: hidden'은 보이지만 않고 공간은 유지된다.

리스트, 표, 폼 꾸미기

1. 목록(List) 스타일

(1) list-style-type

1) 비순차적 목록 타입 지정

- list-style-type: disc;
- list-style-type: circle;
- list-style-type: square;
- list-style-type: none; /* 마커 삭제 */

2) 순차적 목록 타입 지정

- css로는 ul, ol 관계없이 적용 가능하며, 로마숫자, 그리스, 알파벳, 넘버링 형태를 제공
- list-style-type: decimal
- list-style-type: lower-roman
- list-style-type: upper-roman
- list-style-type: lower-greek
- list-style-type: lower-alpha
- list-style-type: upper-alpha
- li.hangul { /* 가, 나, 다, 라 ... */
  list-style-type: -moz-hangul;
  list-style-type: hangul;
  }
- li.hangul-consonant { /* ㄱ, ㄴ, ㄷ, ㄹ, ㅁ ... */
  list-style-type: -moz-hangul-consonant;
  list-style-type: hangul-consonant;
  }

(2) list-style-position

- 목록 위치 지정
- 목록이 지정된 블록 영역의 '안쪽(inside)' 혹은 '바깥쪽(outside)'에 블릿(점)을 표현한다.
- 예) .inside { list-style-position: inside; }
  .outside { list-style-position: outside; }

(3) list-style-image

- 블릿을 이미지로 표현할 수 있다.
- 예) .list { list-style-image: url(img/star.png); }

2. <table> 요소와 관련된 속성들

(1) table-layout 속성

- table-layout: auto; /* 테이블의 셀이 텍스트의 길이에 자동조절 */
- table-layout: fixed; /* 테이블의 셀의 너비를 고정 */

(2) border-collapse 속성

- border-collapse: separate; /* 셀 사이를 분리 */
- border-collapse: collapse; /* 공백을 없애고 셀의 경계를 합친다. */

(3) border-spacing 속성

- 셀 사이의 공백을 지정
- 'border-collapse: collapse;'가 지정되지 않아야 한다.
- border-spacing: 5px; /* 상, 하, 좌, 우 5px */
- border-spacing: 5px 10px; /* 가로(좌, 우) 5px, 세로(상, 하) 10px */

지도

- Javscript API 는 키 발급을 받아야 사용할 수 있다.
- 키를 발급받기 위해서는 카카오 계정이 필요하다.

카카오 APIKI 발급 및 적용

1. 카카오 개발자 페이지 접속하기

1) 카카오 개발자사이트 (https://developers.kakao.com) 접속
2) 오른쪽 상단 로그인 버튼을 클릭해 카카오 계정으로 로그인 합니다.
   카카오 계정이 없는 경우 새롭게 계정을 생성합니다.

2. 내 애플리케이션 만들기

1) 상단 메뉴에서 내 애플리케이션을 클릭합니다.
2) 서비스 이용 동의 페이지가 나타나는 경우 필수 항목에 동의하고, 개발자 이름을 입력한 다음 회원가입 버튼을 클릭해 다음 단계로 넘어갑니다.
   (기존에 서비스 이용 동의한 경우 표시되지 않을 수 있습니다.)
3) 전체 애플리케이션 페이지가 나타나면, 애플리케이션 추가하기를 클릭합니다.
4) 앱 아이콘(로고)을 업로드하고, 앱 이름, 회사 이름을 작성합니다.
5) 저장 버튼을 클릭합니다.

3. 플랫폼 설정하기

1) 전체 애플리케이션 목록으로 돌아가면, 새로 추가한 앱을 클릭합니다.
2) 플랫폼 항목에서 플랫폼 설정하기를 클릭합니다.
3) Web 항목의 Web 플랫폼 등록 버튼을 클릭합니다.
4) 운영 중인 사이트 주소(URL)를 입력합니다.
   여러 개의 도메인을 사용하고 계신 경우 모두 입력합니다.
   (예: http://localhost:8080)
5) 저장 버튼을 클릭해 적용합니다.

4. 발급 받은 Javascript 키 복사하기

1) 왼쪽 탭 메뉴에서 앱 키를 클릭합니다.
2) JavaScript 키 오른쪽 복사 버튼을 클릭합니다.

GNB (Global Navication bar)

어느 페이지에 들어가든 공통적으로 사용할 수 있는 메뉴를 가리킨다.
최상위 메뉴로 보통 상단에 위치하고 있다.
메인메뉴라고도 부른다.

LNB (Local Navigation Bar)

GNB를 누를 경우 소제목 형식으로 나오는 메뉴를 가리킨다.
네비게이션을 통해 특정 지역으로 가는 네비게이션 바이다

SNB(Side Navigation Bar)

일반적으로 왼쪽에 많이 있기 때문에 LNB(Left Navigation Bar)라고 부르기도 한다.
메인메뉴와 서브메뉴를 제외한 나머지 사이드 메뉴이다.

FNB (Foot Navigation Bar)

하단 메뉴를 가리킨다.