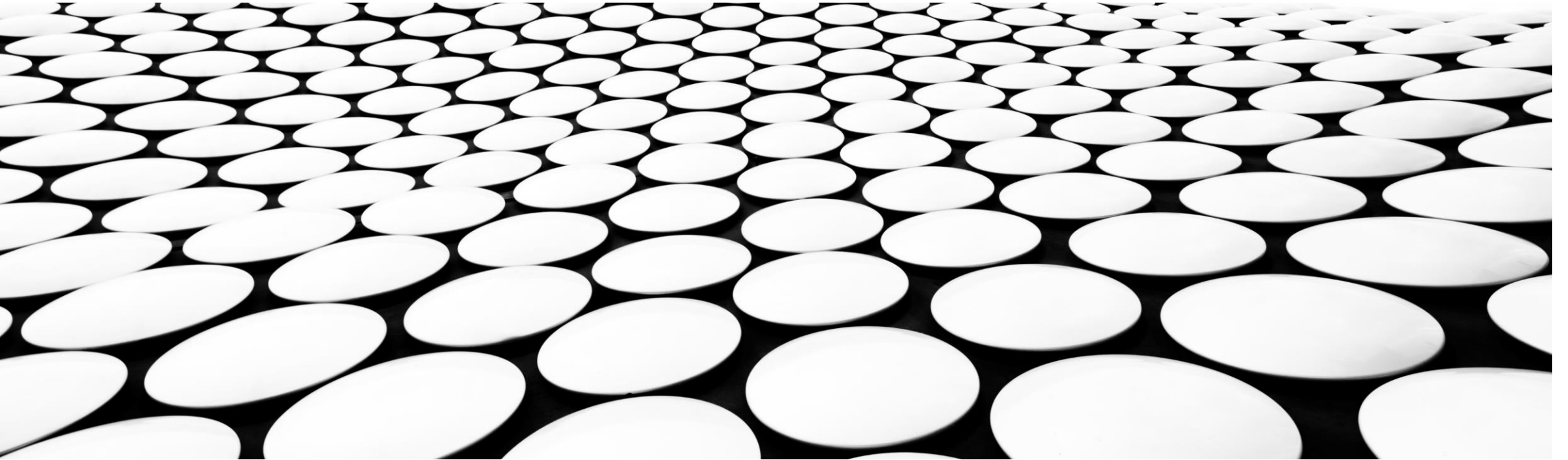

HTML5 + CSS

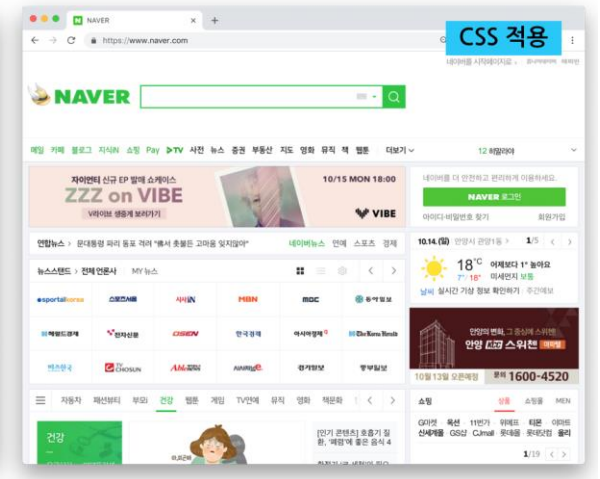
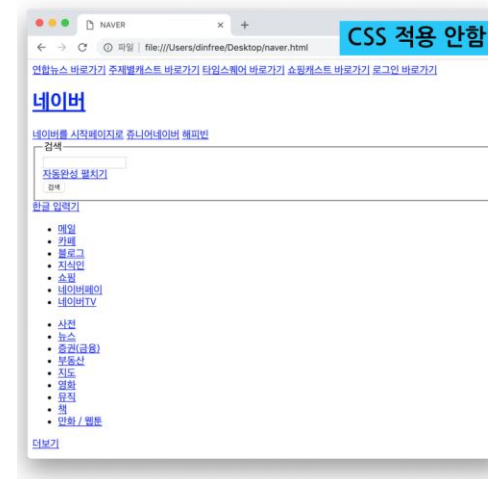


CSS 란?

- CSS는 HTML로 만들어진 콘텐츠에 레이아웃과 디자인요소를 정의하는 기술로 잘 설계된 css 는 재활용이 가능하며 나아가 테마, 템플릿의 형태로 확장할 수 있다. 또한 자바스크립트와 연계해 콘텐츠의 내용이나 디자인을 동적으로 처리할 경우에도 유용하게 사용된다.

CSS(CASCADING STYLE SHEET) 개요

- CSS는 HTML과 함께 웹을 구성하는 기본 프로그래밍 요소이다.
- HTML이 텍스트나 이미지, 표와 같은 구성 요소를 웹 문서에 넣어 뼈대를 만드는 것이라면 CSS는 색상이나 크기, 이미지 크기나 위치, 배치 방법 등 웹 문서의 **디자인** 요소를 담당한다.
 - CSS는 Cascading Style Sheet의 약어이다.
 - CSS는 HTML로 부터 디자인적인 요소를 분리해 정의할 수 있다.
 - 잘 정의된 css 는 서로 다른 여러 웹페이지에 적용할 수 있다. -> 템플릿/테마
 - 자바스크립트와 연계해 동적인 콘텐츠 표현이나 디자인 적용 가능하다.



CSS(CASCADING STYLE SHEET) 개요

- CSS를 사용해야만 하는 이유
 - 웹 문서의 내용과 상관없이 디자인만 바꾸거나 디자인은 그대로 두고 웹 문서의 내용 변경이 용이하다.
 - 다양한 기기(PC, 스마트폰, 태블릿 등)에 맞게 탄력적으로 바뀌는 콘텐츠에 용이하다. -> 반응형 디자인(Responsive Design)
 - 동일한 문서 구조를 가지고 서로 다른 CSS 테마 적용이 가능 하다.

CSS 기본 문법

- CSS는 선택자와 선언부로 구성된다. 선택자는 스타일을 지정할 HTML 요소(태그, 아이디 등)를 가리킨다. 선언부에는 CSS 속성 이름과 값이 포함된다. 속성이 여러 개일 경우, 한 줄로 나열해도 상관없지만 여러 줄에 걸쳐 작성하는 것이 좋다.
 - CSS 구문은 선택자(selector)와 선언부(declaration)로 구성된다.
 - 선택자는 디자인을 적용하고자 하는 HTML요소. -> 선택자 정의가 중요하다
 - 선언부는 콜론(:)으로 구분 되어진 다수의 항목을 포함한다.
 - 각 선언은 항상 세미콜론(;)으로 끝나며, 선언블록은 중괄호({ })로 묶는다.
 - /* comment */은 코드를 설명하는 데 사용된다.

선택자 {속성:값; 속성:값....}

예)

```
/* h1태그의 색상을 빨간색으로 크기는 15px로 지정합니다. */  
h1 {color:red; font-size:15px;}
```

선택자의 중요성

CSS의 핵심은 적절한 선택자를 사용하는 것이며 복잡한 문서 구조에서 특정 부분을 선택하기 위한 선택자 지정은 어려울 수 있으며 html 구조를 처음부터 잘 설계하는 것이 중요함.

CSS 적용 방법

- HTML 문서에 CSS를 적용하는 방법에는 내부 스타일시트, 외부 스타일시트, 인라인 스타일 등 총 3가지 방법이 있다.



CSS 적용 방법

내부 스타일시트

- html 파일에 스타일을 기술하는 방법으로 `<head></head>` 태그 사이에 `<style></style>` 태그 부분에 작성한다.
- html 과 css 가 한 파일에 있으므로 작업이 쉽고 간편하지만 css의 재활용이 안되는 문제가 있어 특별한 경우가 아니면 외부 스타일시트가 권장 된다.

```
<head>
<style>
body {
    background-color: red;
}

h1 {
    color: maroon;
    margin-left: 40px;
}
</style>
</head>
<body>
    ...
</body>
```

CSS 적용 방법

외부 스타일시트

- css 를 작성하는 가장 기본적인 방법 이다.
- 별도의 파일에 CSS 문서를 작성하고 해당 CSS를 필요로 하는 html 문서에서 불러와 사용하는 형식 이다.
- 이때 css는 동일한 서버에 있어도 되고 url을 통해 다른 서버의 css를 불러오는 것도 가능 하다.

```
<link rel="stylesheet" type="text/css" href="mystyle.css">  
<link rel="stylesheet" type="text/css" href="http://cdn.site.com/css/mystyle.css">
```


CSS 적용 방법

인라인 스타일

- html 태그에 필요한 디자인 속성을 직접 작성하는 형식 이다.
- 그때 그때 필요한 디자인을 바로 적용할 수 있다는 편리함이 있지만 일관된 디자인 체계를 유지하는 데에는 방해가 되기 때문에 꼭 필요한 경우가 아니라면 사용하지 않도록 한다.

```
<h1 style="color:blue; margin-left:30px;">This is a heading</h1>
```

CASCADING과 우선순위

Cascading 의미

- css 에서 Cascading 은 사전적 의미로 폭포처럼 떨어져 내리는 과 같은 의미를 가지고 있다. css 에서는 디자인 속성이 html 문서의 구조 즉 DOM(Document Object Model) Tree 구조에서 상위 요소에서 정의한 디자인 속성이 하위 요소로 전달되는(상속 개념) 의미에서 유래 되었다고 할 수 있다.
 - 상위 태그에서 정의된 디자인 속성은 하위 태그로 상속.
 - 하위 태그에서 상위 태그에 정의된 디자인 속성을 변경할 수 있음.

```
<!-- body 태그 안에 있는 모든 태그 요소들은 빨간색 글자로 표시됨 -->  
<body style="font-color:red">  
  <h1>Hello</h1>  
</body>
```

CASCADING과 우선순위

우선 순위

- 동일한 디자인 속성이 외부 스타일시트, 내부 스타일시트, 인라인 스타일시트에 적용 되어 있는 경우 우선순위는 가장 나중에 정의되는 스타일에 있다. 따라서 인라인 스타일시트가 가장 높은 우선순위로 적용되고 외부 스타일시트와 내부 스타일시트는 문서상 정의된 순서에 따라 우선순위가 결정되는 형식이다.
- 웹 브라우저 자체도 html 구성요소에 대한 내부적인 css 를 가지고 있다고 볼 수 있다. 브라우저에 따라 사용자 정의 css를 사용할 수 있는 것도 그 때문이다. 일반적인 우선순위(낮은순 -> 높은순)는 다음과 같다.

브라우저 디자인 정의 -> 외부 스타일시트 -> 내부 스타일시트 -> 인라인 스타일시트

CSS - 셀렉터와 스타일 속성

셀렉터란?

- 스타일은 적용 대상이 있어야 하는데 셀렉터가 바로 그 대상이다. 기본적으로 태그, 아이디, 클래스를 셀렉터로 사용하며 이들을 조합해서 특정 조건에 맞는 셀렉터를 정의해 사용하게 된다.
 - Html 문서에서 스타일의 적용 대상을 지칭.
 - 디자인이 적용되기를 원하는 특정 부분의 선택이 용이하도록 적절한 태그 구조를 사용.
 - Html 문서의 기본 구성요소인 태그는 가장 기본이 되는 셀렉터.
 - 태그에 사용할 수 있는 id 속성은 문서내 유일한 값으로 셀렉터로 사용할 수 있음.
 - 스타일 정의에 클래스를 사용하고 html 태그에 class 속성으로 스타일 지정.

셀렉터	사용예	사용예 설명
.class	.intro	html 태그에서 class=" intro" 로 된 모든 태그 영역 선택
#id	#banner	html 태그에서 id=" banner" 로 된 태그 영역 선택
*	*	문서내 모든 요소를 선택
태그	p	문서내 모든 <p> 태그 영역 선택
태그, 태그	div, p	모든 <div> 와 <p> 태그 영역 선택
태그 태그	div p	<div> 태그 안에 있는 모든 <p> 태그 영역 선택

CSS - 셀렉터와 스타일 속성

기본 셀렉터

- 기본 셀렉터에는 태그, 아이디, 클래스 3가지가 있다. 태그 셀렉터
 - 태그 셀렉터
 - id 셀렉터
 - class 셀렉터

CSS - 셀렉터와 스타일 속성

태그 셀렉터

- 태그 셀렉터는 태그 이름으로 요소를 선택.
- 문서내 임의의 태그를 선택자로 사용.
- 같은 디자인 속성을 가지는 여러 태그는 ,로 나열해 일괄적용.

```
p {  
  text-align: center;  
  color: red;  
}  
  
h1,h2,h3,h4 { color: blue; }
```

CSS - 셀렉터와 스타일 속성

태그 셀렉터

- 경우에 따라서는 태그의 특정 속성에 대해 셀렉터를 지정하는 것이 가능 하다. 예를 들면 `<input>` 태그는 `type` 속성에 따라 다양한 입력 양식을 제공하게 되어 있다. 이경우 특정 `type` 에만 배경색이나 크기를 지정하기 위해서는 다음과 같이 태그 셀렉터에 속성을 함께 사용한다.

```
input[type=text] {  
  background-color: blue;  
  color: white;  
}
```

CSS - 셀렉터와 스타일 속성

id 셀렉터

- HTML 요소의 id 속성을 사용해 특정 요소를 선택.
- id는 페이지 내에서 유일한 값이기 때문에 하나의 고유한 요소를 선택하는 데 사용.

```
#id_name { color: blue; }
```

```
---  
<div id="id_name">  
...  
</div>
```


CSS - 셀렉터와 스타일 속성

기본 셀렉터 조합

- 기본 셀렉터인 태그, 클래스, 아이디의 조합에 따라 다양한 셀렉터 정의가 가능하다.
- 예를 들어 하나의 태그에 대해 어떤 경우에는 적용이되고 어떤 경우에는 적용이 되지 않는 상황을 셀렉터 정의에 따라 만들 수 있다.

CSS - 셀렉터와 스타일 속성

동시 지정

- ,를 이용해 셀렉터를 나열하면 해당 셀렉터에 동일한 속성을 부여할 수 있다.
 - h1,h2 에 동일 속성 지정
 - box, note 클래스에 동일 속성 지정

```
h1, h2 {...}  
.box, .note {...}
```

CSS - 셀렉터와 스타일 속성

태그와 클래스 결합

- 태그와 클래스를 ,을 이용해 결합할 수 있다.
- 예를 들면 같은 클래스를 사용 하더라도 h1 과 h2 에 각각 다르게 적용하고 싶은 경우에 사용할 수 있다.
- 다음 예제에서는 .header 에 공통된 여러 속성을 적용해 두고 h1, h2에서는 각각 특정 속성만 변경하는 형식으로 많이 사용한다.
 - header클래스를 사용하는 모든 태그의 텍스트는 붉은색으로 출력 된다.
 - h1에서 사용할 경우 파란색, h2의 경우 녹색이 적용 된다.

```
.header { color: red; }  
h1.header { color: blue;}  
h2.header { color: green;}
```

CSS - 셀렉터와 스타일 속성

기본 셀렉터 조합 사용 예

- CSS 가 다음과 같이 작성되었다고 가정 합니다.

```
.header { color: red; }  
div.header { color: blue; }  
h1,h2 {color: green}
```

- html 파일은 다음과 같이 작성되어 있을때 각각의 코드는 셀렉터 조합에 따라 스타일이 적용 됩니다.

```
<p class="header">hello</p>    -> 붉은색 출력  
<div>hello</div>              -> 기본색 출력  
<div class="header">world</div> -> 파란색 출력  
<div><h1>hello</h1></div>      -> 녹색 출력  
<div class="header"><h1>world</h1></div> -> 녹색 출력  
<h2>hello world</h2>          -> 녹색 출력
```

CSS - 셀렉터와 스타일 속성

속성 활용하기

- 셀렉터로 스타일 적용을 원하는 html 문서의 영역을 선택 했다면 다음 단계는 적절한 속성을 활용해 스타일을 정의하는 것이다.
 - CSS 속성은 의미를 짐작 할 수 있는 간단한 영어단어로 구성
 - 개발 도구에서 지원하는 코드 완성 기능을 사용
 - 모든 css 속성을 한번에 학습하기 어렵기 때문에 참고문서를 활용해야 함
- 여기서는 대표적인 속성들을 유형별로 구분해 살펴보도록 한다.

CSS - 셀렉터와 스타일 속성

텍스트 속성

- 기본적으로 화면에 출력되는 글자와 관련된 속성들입니다. 대표적인 속성은 다음과 같다.
 - color: 글자색 지정
 - text-align: 주어진 영역에서 글자의 정렬 방식 지정 (left/right/center).

폰트 속성

- 기본적으로 화면에 출력되는 글꼴과 관련된 속성들이다.
- 폰트 설정의 경우 PC에 폰트가 설치되어 있어야 하며 설치된 폰트와 상관없이 폰트 적용을 위해서는 web font 라는 것을 사용해야 한다.

CSS - 셀렉터와 스타일 속성

font-family

- 폰트의 이름과 유형을 지정하는 속성 이다. 영문을 기준으로 폰트의 유형은 세가지로 구분하며 다음과 같다.
 - Serif: 바탕체 계열의 폰트. Times New Roman, Georgia
 - Sans-serif: 굴림 계열의 폰트. Arial, Verdana
 - Monospace: 고정폭 폰트. Courier New, Lucida Console
 - 폰트 이름이 두 글자 이상인 경우 반드시 " "로 감싸야 함.
 - 나열된 순서로 폰트가 적용됨.

```
h1 {  
  font-family: "Times New Roman", verdana, arial;  
}
```

CSS - 셀렉터와 스타일 속성

font-style

- 폰트의 스타일을 지정하는 속성 이다.
- normal, italic, oblique 가 있으면 oblique 는 italic 과 유사하며 잘 사용되지 않는다.

```
.text1 { font-style: normal}  
.text2 { font-style: italic}
```


CSS - 셀렉터와 스타일 속성

font-size

- 폰트의 크기를 지정하는 속성 이다.
- px, %, rem, em 등 여러 단위로 사용할 수 있다.

```
.text1 { font-size: 10px}  
.text2 { font-size: 2em}
```

CSS - 셀렉터와 스타일 속성

font-weight

- 폰트의 두께를 지정하는 속성 이다.
- normal, bold, bolder 등을 사용할 수 있다.

```
.text1 { font-weight: normal}  
.text2 { font-weight: bold}
```

CSS - 셀렉터와 스타일 속성

정렬 속성

- 일반적인 문서 작성 프로그램에서는 텍스트나 이미지를 가운데로 정렬하기가 쉽다.
- css 에서 가장 어려운 부분이 원하는 곳에 원하는 텍스트나 박스를 배치하는 것이다.
- 기본적인 css 정렬 속성은 다음과 같다.
 - text-align: 요소내 텍스트의 정렬
 - vertical-align: 인라인 혹은 테이블 셀에서 수직 정렬

CSS - 셀렉터와 스타일 속성

링크 속성

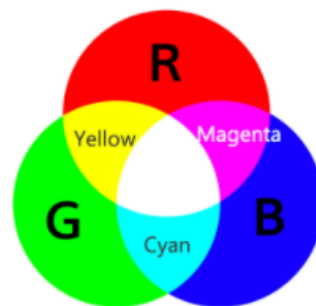
- 하이퍼링크를 만들기 위한 <a> 태그에 적용할 수 있는 속성 이다. Html 의 기본 특성상 하이퍼링크 텍스트는 기본텍스트와 다른 색상을 가지고 마우스가 올라가면 색상이 변경되고 기본적으로 밑줄이 그어져 있다. 방문했던 사이트의 색상은 다르게 나와 디자인적인 일관성을 유지하려면 귀찮지만 관련된 속성을 모두 지정해 주어야 한다.
- 다음은 <a> 태그에 적용되는 가상 셀렉터 이다.
 - a:link - 방문한적 없는 기본 링크
 - a:visited - 방문한 링크
 - a:hover - 마우스가 링크위에 올라갔을 때
 - a:active - 링크를 클릭 했을 때
- 각각의 가상 셀렉터에는 color, background-color, text-decoration 등의 속성이 사용 된다.

```
a:link {  
    color: red;  
    text-decoration: none;  
}  
  
a:visited {  
    color: blue;  
    text-decoration: none;  
}  
  
a:hover {  
    color: hotpink;  
    text-decoration: underline;  
}  
  
a:active {  
    background-color: blue;  
    text-decoration: underline;  
}
```

CSS - 셀렉터와 스타일 속성

컬러 속성

- 컴퓨터에서 사용하는 색상은 빛의 삼원색인 빨강색(Red), 초록색(Green), 파랑색(Blue)이다. 이를 보통 RGB Color라고 부르는데, 각각의 색상은 0 ~ 255까지의 단계로 표현할 수 있다.
- 0부터 255를 16진수로 표현하면 00 ~ FF로 표현된다.
- CSS는 140개 이상의 색상 이름, 16진수(HEX) 값, RGB 값, RGBA 값, 불투명도를 지원한다.
 - CSS에서 Color속성은 색상이름, HEX(#)코드, rgb, hsl로 나타낼 수 있다.
 - hsl 은 Hue(색조), Saturation(채도), Lightness(밝기) 의 조합을 나타낸다.
 - 색상에 투명도(alpha)를 적용시킬 때는 rgba, hsla를 사용하며, 0.0(완전투명)~1.0(완전불투명) 사이의 숫자로 나타낸다



Red = #FF0000 = RGB(255,0,0)

Cyan = #0000FF = RGB(0,0,255)

Green = #00FF00 = RGB(0,255,0)

Magenta = #FF00FF = RGB(255,0,255)

Blue = #0000FF = RGB(0,0,255)

Yellow = #FFFF00 = RGB(255,255,0)

Black = #000000 = RGB(0,0,0)

White = #FFFFFF = RGB(255,255,255)

CSS - 셀렉터와 스타일 속성

컬러 속성

- CSS에서 Color속성은 색상이름, HEX(#)코드, rgb, hsl로 나타낼 수 있다.
- hsl 은 Hue(색조), Saturation(채도), Lightness(밝기) 의 조합을 나타낸다.
- 색상에 투명도(alpha)를 적용시킬 때는 rgba, hsla를 사용하며, 0.0(완전투명)~1.0(완전불투명) 사이의 숫자로 나타낸다

CSS Color

```
#text1 { color: red; }  
#text2 { color: #FF0000; }  
#text3 { color: rgb(255, 0, 0); }
```

rgba, hsla

```
#text1 { color: rgba(255, 99, 71, 0.5) }  
#text2 { color: hsla(9,100%, 64%, 0.5) }
```

CSS - 셀렉터와 스타일 속성

배경 속성

- 색상이나 이미지를 배경으로 지정하기 위한 속성 이다.
- 대표적인 속성들은 다음과 같다.
 - background-color: 배경색 지정
 - Background-image: 배경 이미지 지정, 상대경로, 절대경로, url 사용가능
 - Background-repeat: 이미지 반복
 - Background-position: 이미지의 위치를 지정
 - background-attachment: 이미지의 스크롤이나 고정을 지정함

```
body {  
  background-image: url("back_img.png");  
  background-repeat: no-repeat;  
  background-position: right top;  
  background-attachment: fixed;  
}
```

CSS - 박스 모델(BOX MODEL)

- CSS 에서 화면 구성요소를 다루기 위한 기본적인 방법인 박스 모델에 대해 살펴 본다. 또한 박스 모델들의 화면 배치를 위한 position 속성과 float, display 속성들을 활용하는 레이아웃 기법을 배운다. 박스 모델은 구성요소를 배치해 웹 페이지 레이아웃을 구성할 때 매우 중요한 개념이다.
- 이 강의를 통해 박스모델을 정의하고 웹페이지 레이아웃을 설계할 수 있으며 원하는 위치에 구성요소를 자유롭게 배치할 수 있게 된다.

CSS - 박스 모델(BOX MODEL)

CSS 단위

- 단위는 크기를 결정하는 중요한 요소 이다. 일반적인 츠, px 등과 같은 단위 이외에도 CSS에는 여러 단위가 있다.
- 정확한 레이아웃 구성을 위해서는 어떤 단위가 있는지 명확하게 알고 있어야 하며 특히 여러 화면 크기에 대응하는 반응형 웹 디자인을 위해서는 단위에 대한 이해가 필수적이다.
 - 절대 단위
 - 상대 단위

CSS - 박스 모델(BOX MODEL)

절대 단위

- 절대적인 크기가 정해져 있는 단위 이다. 절대 단위는 화면 크기나 해상도에 따라 지나치게 크거나 작게 보일 수 있다.
- 예를 들어 1024x768 해상도에서 500px 의 크기는 화면의 50% 이상에 해당하지만 동일한 콘텐츠를 요즘 많이 사용하고 있는 Full HD 해상도인 1920x1080 에서 보게 될 경우 전체 화면의 26% 밖에 되지않아 글씨 크기가 작게 보이거나 많은 공간이 남게 되어 주의가 필요 하다.
- pc 는 파이카(pica)라고 하며 예전에 활자의 크기를 나타내던 단위로 인쇄및 출판시스템에 많이 사용되던 단위 이다. 1인치에 가로 열 자, 세로 여섯 자가 찍히던 크기로, 현재의 12포인트 활자 크기와 같다.

단 위	설 명
cm	centimeter
mm	millimeter
in	inch (1in = 2.54cm)
px	픽셀, pixel
pt	포인트, point (1pt = 1/72 inch)
pc	파이카, pica (1pc = 12pt)

CSS - 박스 모델(BOX MODEL)

상대 단위

- 부모 요소의 크기를 기준으로 상대적인 크기를 지정하는 단위 이다. 적절한 설계를 통해 상대 단위를 잘 활용하면 반응형 웹 등 다양한 화면 크기에 대응할 수 있는 웹 페이지 레이아웃 구현이 가능하다. 가장 많이 사용되는 단위는 %, em, rem 이다.
 - em : 부모 요소의 기본 크기를 1em 으로 상대적인 크기를 지정.
 - rem : rem에서 r은 루트(root)를 뜻하며, 부모가 아닌 최상위 root를 기준으로 하기 때문에 중간에 기본값이 바뀌지 않음.
- 앞에 v가 붙은 단위들은 뷰 포트(viewport)와 관련된 것으로 뷰 포트는 웹 페이지가 사용자에게 보여지는 영역을 말하는 것으로 PC의 경우 브라우저 크기를 줄이게 되면 스크롤 바를 통해 한 화면에 보이지 않는 콘텐츠를 볼 수 있다. 이때도 보여지는 영역이 뷰 포트가 된다. 반면 모바일 기기들의 경우 브라우저의 크기를 조정할 수 없으므로 콘텐츠가 브라우저 크기를 벗어나는 경우 한 화면에 담기 위해 크기를 줄이게 된다. 따라서 글자들이 너무 작게 되어 보기 어렵게 된다. Html 의 meta 태그 중에 viewport 설정은 모바일 화면의 콘텐츠를 뷰 포트 크기로 맞춰주는 설정 이다.

단 위	설 명
em	부모 요소의 글꼴에 비례. (2em은 현재 글꼴 크기의 2배를 의미.)
ex	현재 글꼴의 x 높이에 비례. (거의 사용되지 않음)
rem	루트 요소(<html>)의 글꼴 크기에 비례.
vw	뷰포트 너비의 1%에 비례.
vh	뷰포트 높이의 1%에 비례.
vmin	뷰포트의 너비와 높이 중 더 작은 치수 1%에 비례.
vmax	뷰포트의 너비와 높이 중 더 큰 치수 1%에 비례.
%	퍼센트, 100% 를 기준으로 하는 상대 크기

CSS - 박스 모델(Box Model)

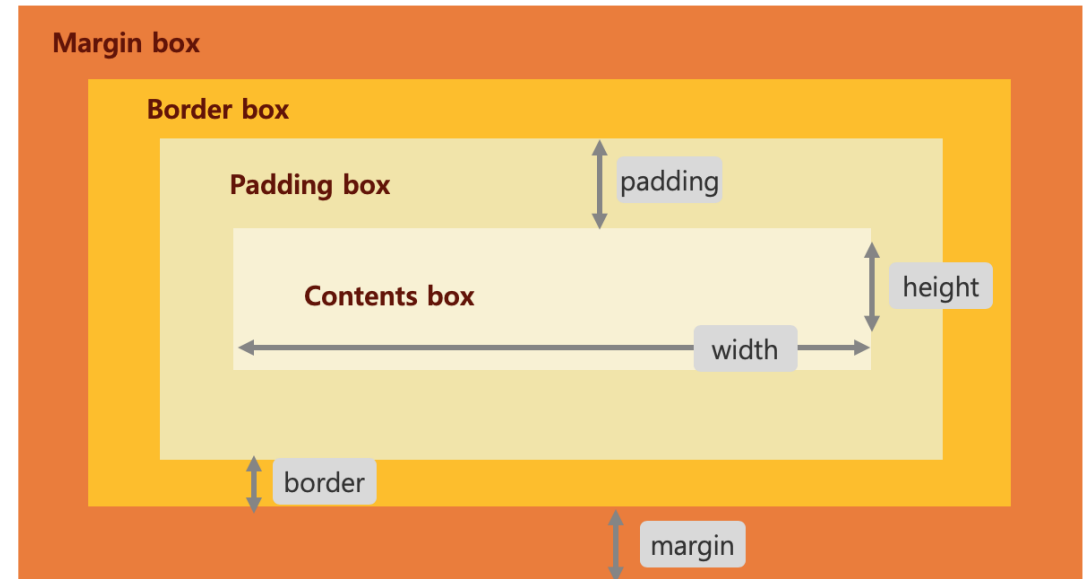
박스 모델

- html 문서의 구성요소들은 기본적으로 박스형태로 정의 됩니다. 쉽게 생각하면 웹페이지 레이아웃은 이러한 박스들을 위/아래, 좌/우로 적절하게 배치하는 것을 말합니다.
 - 박스 개요
 - 박스 크기
 - border 속성
 - Margin 속성

CSS - 박스 모델(BOX MODEL)

박스 개요

- 박스 모델은 테두리(border)와 내용(content) 그리고 안쪽 여백(padding)과 바깥쪽 여백(margin)의 네 가지 요소로 구성된다.
- 각 요소는 상, 하, 좌, 우 네 영역을 개별적으로 설정할 수 있다.



CSS - 박스 모델(BOX MODEL)

박스 개요

- Contents box - 콘텐츠 영역으로 텍스트 및 이미지의 실제 영역.
- Padding box - 테두리와 콘텐츠 사이의 안쪽 여백.
- Border box - 박스를 둘러싼 테두리 영역.
- Margin box - 박스의 외부 영역으로 바로 앞 박스와의 여백.

```
div {  
  width: 320px;  
  padding: 10px;  
  border: 5px solid gray;  
  margin: 0;  
}
```

CSS - 박스 모델(BOX MODEL)

박스 크기

- 박스 요소에 크기를 지정하면 기본적으로 콘텐츠 영역에 적용 된다. 그러나 실제 박스의 크기는 border, margin, padding을 모두 더해야 하기 때문에 각각의 박스를 적절하게 배치 하려면 이들 크기가 함께 고려 되어야 한다.
- `<div width="300px">` 은 콘텐츠 영역이 300px 임을 의미 한다.
- 전체 박스 크기는 border, margin, padding 에 콘텐츠 영역 크기를 더해야 한다.
- 박스 크기 계산을 편하게 하기 위해서는 border 를 기준으로 하는 것이 편하다. 이경우 box-sizing 속성을 border-box 로 지정하면 된다. 기본은 content-box 이다.

CSS - 박스 모델(BOX MODEL)

border 속성

- border 영역을 지정하기 위한 여러 속성이 있다. width 와 같이 크기를 지정하는 것 이외에 테두리 선을 지정하기 위한 border-style과 border-color 속성 등이 있다.
- border-width
 - 테두리 두께를 지정하는 속성으로 상하좌우 네 영역을 개별적으로 설정하거나 상하, 좌우를 묶어서 설정할 수 있다.
- border-style
 - 테두리의 모양을 지정하는 속성으로 실선, 점선, 이중 선 등을 사용할 수 있다. 4곳의 테두리를 각각 다르게 지정할 수도 있다. border-top-style 처럼 사용한다.
- Border-color
 - 테두리의 색상을 지정하는 속성으로 일반적인 컬러 속성을 이용해 색상을 지정 한다.
- Border-radius
 - 테두리의 모서리를 둥글게 만들기 위한 속성 이다. 반지름의 크기를 px 이나 % 등의 단위를 이용해 지정할 수 있다.

```
div {  
  border-width: 2px 10px 4px 20px; /* top, right, bottom, left */  
  border-width: 2px 10px; /* top bottom, right left */  
}
```


CSS - 박스 모델(BOX MODEL)

border 속성

- 단축형(Shorthand)
 - css 의 많은 속성들이 여러 속성들을 묶어 단축 형으로 사용할 수 있도록 지원하고 있다. border 의 경우에도 width, style, color 순으로 나열하면 된다. 또한 4곳의 테두리를 다르게 설정해 다양한 용도로 활용 가능한 박스를 만들 수 있다.

```
div {  
  border: 5px solid red;  
  border-left: 5px solid red;  
}
```

CSS - 박스 모델(BOX MODEL)

margin 속성

- margin 은 박스와 인접 요소간의 여백을 말한다. 박스 간의 적적할 배치를 위해 사용할 수 있으며 경우에 따라서는 박스를 가운데 정렬하기 위해 사용할 수도 있다.

```
div {  
  margin: 140px;  
  border: 1px solid #4CAF50;  
}
```

CSS - 레이아웃

레이아웃

- 레이아웃은 화면의 배치를 말하는 것으로 css 는 웹 화면의 디자인적인 요소와 함께 구성요소들을 적절한 위치에 배치하기 위한 방법을 제공하고 있다.
- 기본적으로는 박스들의 배치 방법을 결정하는 position 속성이 있으며 나란히 배치되는 박스 콘텐츠들을 지정하기 위해 float 이나 display 속성 등이 사용 된다.
- 그리드 시스템(Grid System)
- 컨테이너(Container)

CSS - 레이아웃

그리드 시스템(Grid System)

- 그리드 시스템은 화면을 테이블과 유사하게 가로, 세로의 격자로 나누는 것을 말하며 널리 사용하는 css 라이브러리인 bootstrap의 경우 한 줄을 최대 12개의 컬럼으로 분할할 수 있도록 되어 있다
 - 화면의 가로 폭을 100%로 두고 분할하고자 하는 영역을 원하는 비율로 크기를 지정할 수 있다.
 - 반응 형 웹의 경우 화면 크기에 따라 구성이 달라지기 때문에 좀 더 복잡한 설정이 필요 하다.
 - 하나의 컬럼 안에서 가로 혹은 세로 분할이 필요한 경우 해당 컬럼을 100% 로 가정하고 분할 할 수 있다.



CSS - 레이아웃

컨테이너(Container)

- 컨테이너는 다른 물건을 담을 수 있는 그릇의 의미를 가지고 있다. css 에서는 다른 구성요소를 포함 할 수 있는 박스영역으로 이해할 수 있다.
- 실제 구현 상에는 <div> 속에 다른 <div> 가 들어가는 형태로 이해할 수 있다.
 - container class는 header, contents, footer 를 가지는 컨테이너 이다.
 - Contents class는 box 클래스 요소를 가지는 컨테이너 이다.

```
<div class="container">  
  <div class="header">  
  </div>  
  <div class="contents">  
    <div class="box"></div>  
    <div class="box"></div>  
    <div class="box"></div>  
  </div>  
</div>  
<div class="footer">  
</div>  
</div>
```

CSS - POSITION

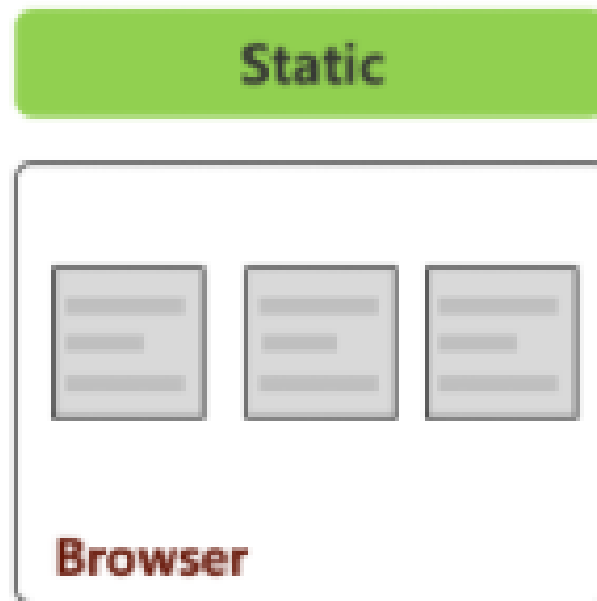
Position

- position 속성은 박스 구성요소들을 배치하기 위한 속성다. 어떤 position 속성을 사용 하느냐에 따라 위치가 달라질 수 있으므로 전체적인 내용을 잘 이해하고 익숙하게 사용할 수 있도록 해야 한다.
- Static
- Relative
- Absolute
- Fixed
- z-index

CSS - POSITION

Static

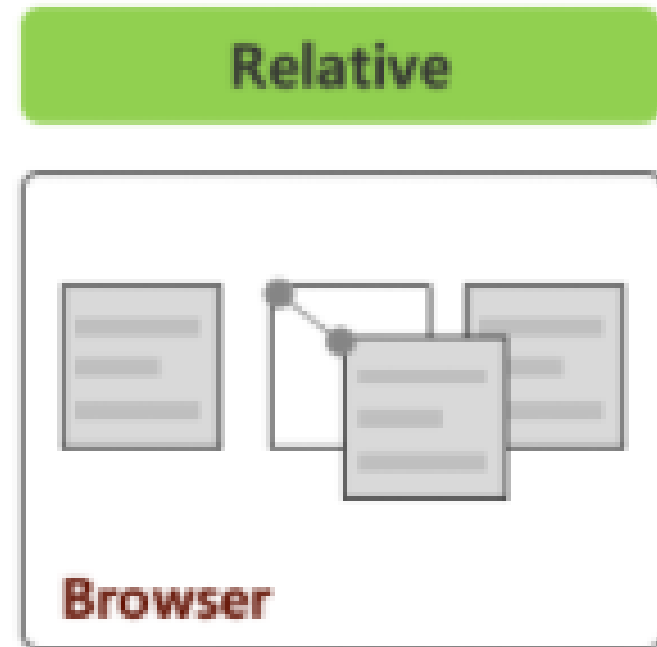
- position 속성의 기본값으로, 요소를 나열한 순서대로 배치하며 원하는 위치에 콘텐츠를 배치할 방법은 없다.
- 순서대로라는 의미는 콘텐츠를 왼쪽에서 오른쪽으로 추가해 나가고 오른쪽에 공간이 없을 경우 다음 줄로 넘겨 배치하는 것을 의미 한다.
- 뒤에 나오는 float 속성을 이용해 좌우로 배치할 수 있다.



CSS - POSITION

Relative

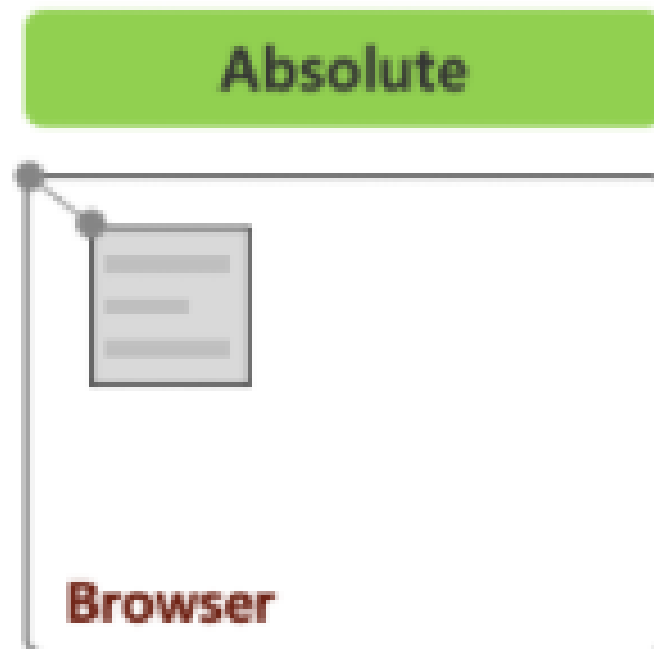
- static과 같이 나열한 순서대로 배치되지만 top, right, bottom, left 속성을 사용해 원하는 위치를 지정할 수 있다.
- 이때 좌표 값은 원래 있던 위치 즉, static 기준으로 원래 위치해야 할 곳이 기준이 되며 지정한 속성에 따라 상/하/좌/우 원하는 자리에 배치하는 것이 가능하다.



CSS - POSITION

Absolute

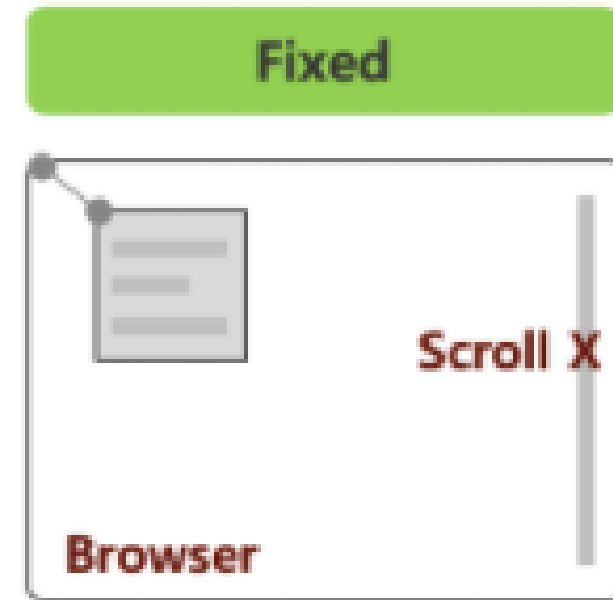
- top, right, bottom, left 속성값을 이용해 요소를 원하는 위치에 배치할 수 있다.
- 이때 기준 위치는 가장 가까운 상위 요소 중 position 속성이 relative인 요소.
- 따라서 absolute를 사용하는 경우 콘텐츠 박스를 감싸는 컨테이너를 만들고 position을 relative로 지정해놓고 사용.
- 상위 요소가 없다면 브라우저 화면의 좌측 상단을 기준으로 설정.



CSS - POSITION

Fixed

- absolute 속성처럼 좌표로 위치를 결정하지만 기준이 부모 요소가 아닌 브라우저 창(Browser Window)임.
- 페이지를 스크롤하더라도 계속 고정되어 표시됨. 즉, 항상 같은 위치를 유지.



CSS - POSITION

z-index

- 박스들이 중첩되는 경우 박스들의 수직 위치를 조정하기 위한 속성.
- 파워포인트에서 박스들을 맨 위로 보내거나 맨 뒤로 보내는것과 같다.
- z-index 값이 높을수록 위, 작을수록 아래에 배치되며 -인 경우 기준 콘텐츠의 아래쪽을 의미.
- position 속성이 적용된 경우에만 의미가 있음.

CSS – FLOAT/DISPLAY

Float 속성

- float 속성은 속성의 명칭과 같이 요소를 화면 위에 떠있는 형식으로 배치 한다.
- float 속성을 이용하며 좌측 혹은 우측부터 정렬되는 박스 콘텐츠를 배열할 수 있다.
- float
 - float: left; 혹은 float: right;를 지정하면 width는 콘텐츠를 표시할 때 필요한 만큼만 차지하고 다른 요소가 들어올 만큼의 공간을 비워 둔다.
 - Float 속성이 더 이상 동작하지 않도록 원할 경우 다음에 나오는 clear 속성을 반드시 지정해 주어야 한다.
- Clear
 - Clear 속성은 float 속성이 더 이상 유용하지 않다고 알려 주는 속성이다.
 - 만약 float: left;로 왼쪽으로 배치했다면 clear: left;로 무효화시킬 수 있다.
 - 무조건 기본 상태로 되돌리고 싶다면 clear: both;라고 하면 된다.

CSS – FLOAT/DISPLAY

display 속성

- display 속성은 요소를 보여주는 방식을 지정하는 속성 이다. 콘텐츠를 보이지 않게 할수도 있고 float 을 대체해 콘텐츠를 나란히 배치하는데 사용되기도 한다.
- none : 보이지 않음 , visibility:hidden 속성과 유사하나 영역 자체가 없어짐.
- block : 블록 박스
- inline : 인라인 박스
- inline-block : block과 inline의 중간 형태

CSS – FLOAT/DISPLAY

display 속성

- block과 inline

- html 을 배우면서 block 태그와 inline 태그의 차이점에 대해 배웠다. block 태그는 width=100% 인 태그들로 요소를 나란히 배치할 수 없다. inline 은 콘텐츠의 크기 만큼만 자리를 차지하기 때문에 다른 콘텐츠와 나란히 배치될 수 있다. 다만 width와 height 를 사용할 수 없어 주변 콘텐츠와 균형을 맞추기가 어려운 문제가 있다.

- inline-block

- display 속성을 사용하면 block 태그에 inline 속성을 가지도록 변경할 수도 있고 그 반대도 가능하다. 또한 inline 속성을 가지면서 즉, 다른 콘텐츠와 나란히 배치되면서 block 요소의 width, height 등의 속성이 적용되도록 하는 inline-block 속성도 있다.

CSS – FLOAT/DISPLAY

display 속성 - block

```
<!DOCTYPE html>
<html>
<head>
<style>

span.c {
  display: block;
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;
}
</style>
</head>
<body>

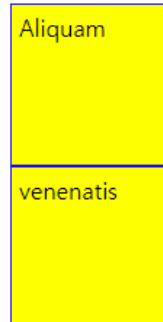
<h1>The display Property</h1>
|
<h2>display: block</h2>
<div>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat
scelerisque elit sit amet consequat. Aliquam erat volutpat. <span
class="c">Aliquam</span> <span class="c">venenatis</span> gravida nisl sit amet
facilisis. Nullam cursus fermentum velit sed laoreet. </div>

</body>
</html>
```

The display Property

display: block

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat.



gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet.

CSS – FLOAT/DISPLAY

display 속성 - inline

```
<!DOCTYPE html>
<html>
<head>
<style>
span.a {
  display: inline; /* the default for span */
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;
}

</style>
</head>
<body>

<h1>The display Property</h1>

<h2>display: inline</h2>
<div>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat
scelerisque elit sit amet consequat. Aliquam erat volutpat. <span
class="a">Aliquam</span> <span class="a">venenatis</span> gravida nisl sit amet
facilisis. Nullam cursus fermentum velit sed laoreet. </div>

</body>
</html>
```

The display Property

display: inline

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat. Aliquam venenatis gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet.

CSS – FLOAT/DISPLAY

display 속성 – inline-block

```
<!DOCTYPE html>
<html>
<head>
<style>
span.b {
  display: inline-block;
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;
}
</style>
</head>
<body>

<h1>The display Property</h1>

<h2>display: inline-block</h2>
<div>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat
scelerisque elit sit amet consequat. Aliquam erat volutpat. <span
class="b">Aliquam</span> <span class="b">venenatis</span> gravida nisl sit amet
facilisis. Nullam cursus fermentum velit sed laoreet. </div>

</body>
</html>
```

The display Property

display: inline-block

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat.

Aliquam

venenatis

gravida nisl sit

amet facilisis. Nullam cursus fermentum velit sed laoreet.

CSS – BOX ELEMENT

박스 요소 정렬

- 기본적으로 문서내 영역을 정렬하기 위해서는 박스 요소들로 배치가 되어야 한다.
- 앞에서는 박스요소들을 차례로 혹은 임의로 원하는 위치에 배치하기 위한 방법들을 살펴 보았다. 여기서는 일반적으로 필요한 좌-우, 중앙 정렬 방법을 살펴본다.

CSS – FLOAT/DISPLAY

블록 요소 가운데 정렬

- `margin:auto`는 블록 속성 요소를 가운데 정렬하기 위한 일반적인 방법이다.
- `<div>` 등으로 블록 박스를 만들었을 경우 `margin:auto` 속성을 사용하면 가운데 정렬이 된다.
- `width` 속성이 100% 이거나 정의되지 않은 경우에는 정렬이 이루어지지 않음.

```
.box-center {  
  margin: auto;  
  width: 50%;  
  border: 3px solid green;  
  padding: 10px;  
}
```

CSS – FLOAT/DISPLAY

블록 요소 좌우 정렬

- 블록 요소의 position 속성을 absolute 로 두고 right, left 속성을 이용해 원하는 위치에 정렬 시킨다.

```
.box-right {  
  position: absolute;  
  right: 0px;  
  width: 300px;  
  border: 3px solid green;  
  padding: 10px;  
}
```

CSS – FLOAT/DISPLAY

블록 요소 수직 정렬

- 블록 요소를 수직 정렬하는 방법은 여러 개가 있으며 가장 기본적인 방법은 padding 속성으로 박스의 안쪽 위/아래 여백을 동일하게 지정하는 방법 이다.

```
.box-vcenter {  
  padding: 50px 0; /* 위/아래 50px, 좌/우 0 */  
  border: 3px solid green;  
}
```

CSS – FLOAT/DISPLAY

이미지 가운데 정렬

- 이미지를 가운데 정렬하는 방법은 크게 두가지이다.
 - 방법1: 부모 요소의 속성에 text-align:center 를 사용.
 - 방법2: img 태그만 사용한다면 블록 속성으로 변경한 다음 margin:auto 적용.

```
// 방법1
div {
  text-align:center;
}

// 방법2
img {
  display: block;
  margin: auto;
}

...
<div>
  
</div>
```

CSS – 복합 셀렉터

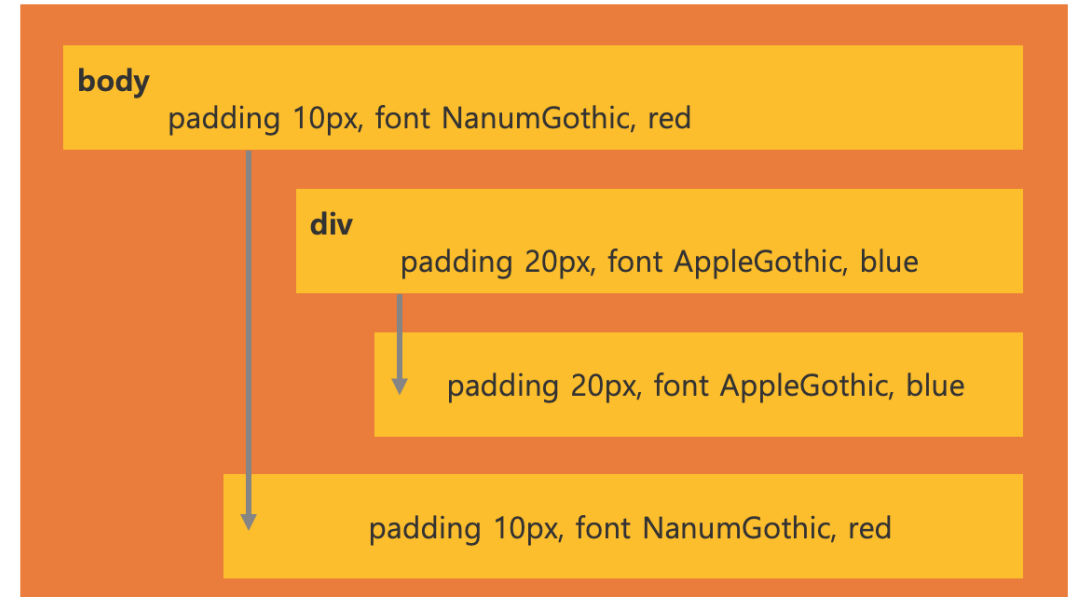
복합 셀렉터

- 이번 강의에서는 좀 더 복잡한 셀렉터를 배우게 된다. 이를 위해 html 계층 구조 안에서 부모 요소와 자식 요소간의 관계를 파악하고 상속을 통해 CSS 속성 적용이 어떻게 이루어지는지 학습한다. 또한 사용자 동작과 UI 요소 상태에 스타일을 적용하는 가상클래스에 대해 배우고 어떤 속성들이 존재하는지 살펴본다.
- 또한 CSS 상속의 개념과 후손, 자손, 형제 선택자등의 선택자 조합과 가상 선택자를 이해하고 사용할 수 있게 된다.

CSS – 복합 셀렉터

복합 셀렉터 - 상속

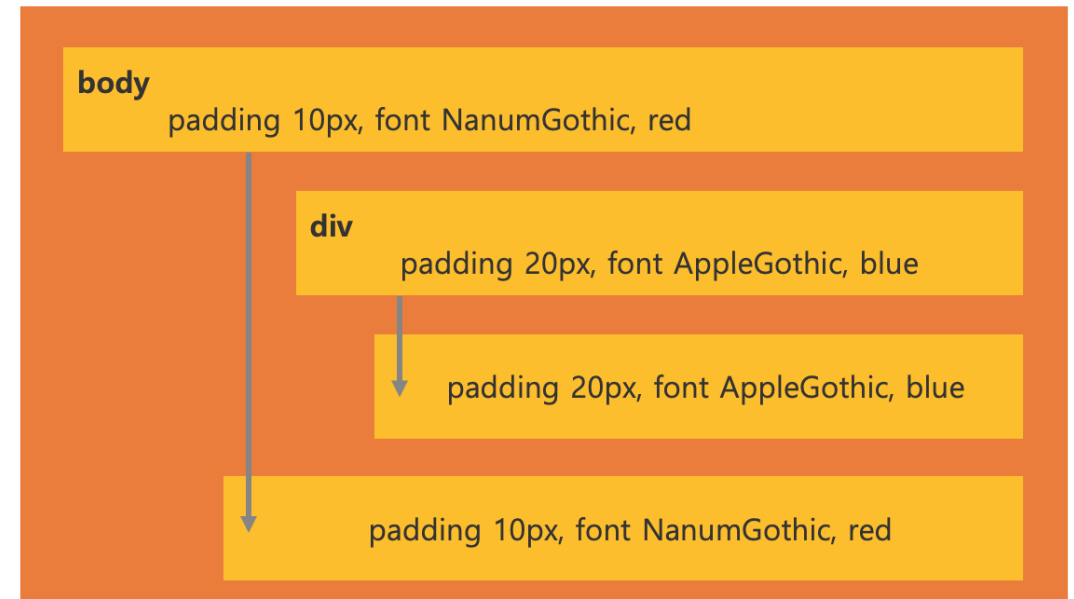
- CSS에서 상속이란 html 계층구조에서 특정 속성들이 부모요소로부터 자식 요소로 전달되는 개념이다.
- 부모 요소의 색상이 red라면, 자식도 red라는 속성을 물려받는 것이다.
- 그러나 모든 CSS 속성들이 상속되는 것은 아니다. 예를 들어 일반적으로 자식 요소가 부모와 동일한 margin을 가지는 경우는 거의 없기 때문에 margin 속성은 상속이 되지 않는다.



CSS – 복합 셀렉터

복합 셀렉터 - 상속

- 프로그래밍에서의 상속(inheritance)은 상위 객체(object)와 하위 객체(object)의 관계형 구조를 의미.
- 상속은 부모(parent) 요소의 속성을 자식(child) 요소가 물려받는 것을 의미.
- 상속을 이용하면 코드의 중복 성을 줄여주기 때문에 생산성을 높일 수 있음.
- 유지 보수가 편하고 재활용이 용이.



CSS – 복합 셀렉터

복합 셀렉터 - 셀렉터 조합(Combinators)

- 기본적으로 html 요소들의 상속 구조 관계에서 원하는 요소를 선택하기 위해 셀렉터를 결합하는 것을 셀렉터 조합 이라고 하고 한다.
- 셀렉터 조합에는 여러 유형이 있는데 어떤 유형을 사용하느냐에 따라 결과가 달라지기 때문에 잘 살펴보아야 한다.

CSS – 복합 셀렉터

후손 선택 자(Descendant Selector)

- selector A Selector B
- Selector A의 후손(Descendant)인 Selector B를 선택.
- 즉, A 요소 아래에 있는 모든 B 요소가 해당.
- B 는 A 아래에 있는 다른 요소의 자식 이어도 됨.

```
<!DOCTYPE html>
<html>
<head>
<style>
div p {
background-color: yellow;
}
</style>
</head>
<body>

<h2>Descendant Selector</h2>

<p>The descendant selector matches all elements that are descendants of a specified element.</p>

<div>
<p>Paragraph 1 in the div.</p>
<p>Paragraph 2 in the div.</p>
<section><p>Paragraph 3 in the div.</p></section>
</div>

<p>Paragraph 4. Not in a div.</p>
<p>Paragraph 5. Not in a div.</p>

</body>
</html>
```

Descendant Selector

The descendant selector matches all elements that are descendants of a specified element.

Paragraph 1 in the div.

Paragraph 2 in the div.

Paragraph 3 in the div.

Paragraph 4. Not in a div.

Paragraph 5. Not in a div.

CSS – 복합 셀렉터

자식 선택자(Child Selector)

- Selector A > Selector B
- Selector A의 직접적인 자식(Child)인 Selector B를 선택.
- 즉, A 요소 아래에 있는 모든 B를 선택하되 B가 다른 요소의 자식이면 안된다.

```
<!DOCTYPE html>
<html>
<head>
<style>
div > p {
background-color: yellow;
}
</style>
</head>
<body>

<h2>Child Selector</h2>

<p>The child selector (>) selects all elements that are the children of a specified element.</p>

<div>
<p>Paragraph 1 in the div.</p>
<p>Paragraph 2 in the div.</p>
<section>
<!-- not Child but Descendant -->
<p>Paragraph 3 in the div (inside a section element).</p>
</section>
<p>Paragraph 4 in the div.</p>
</div>

<p>Paragraph 5. Not in a div.</p>
<p>Paragraph 6. Not in a div.</p>

</body>
</html>
```

Child Selector

The child selector (>) selects all elements that are the children of a specified element.

Paragraph 1 in the div.

Paragraph 2 in the div.

Paragraph 3 in the div (inside a section element).

Paragraph 4 in the div.

Paragraph 5. Not in a div.

Paragraph 6. Not in a div.

CSS – 복합 셀렉터

인접 형제 선택자(Adjacent Sibling Selector)

- Selector A + Selector B
- A와 가장 인접한 형제 요소(Sibling) B에 속성 적용.
- 즉, A와 B는 같은 부모 요소를 가지고 있어야 하며 여러 B 요소 중에서 A에 가장 인접한 B만 선택된다.

```
<!DOCTYPE html>
<html>
<head>
<style>
div + p {
background-color: yellow;
}
</style>
</head>
<body>

<h2>Adjacent Sibling Selector</h2>

<p>The + selector is used to select an element that is directly after another
specific element.</p>
<p>The following example selects the first p element that are placed immediately
after div elements:</p>

<div>
<p>Paragraph 1 in the div.</p>
<p>Paragraph 2 in the div.</p>
</div>

<p>Paragraph 3. After a div.</p>
<p>Paragraph 4. After a div.</p>

<div>
<p>Paragraph 5 in the div.</p>
<p>Paragraph 6 in the div.</p>
</div>

<p>Paragraph 7. After a div.</p>
<p>Paragraph 8. After a div.</p>

</body>
</html>
```

Adjacent Sibling Selector

The + selector is used to select an element that is directly after another specific element.

The following example selects the first p element that are placed immediately after div elements:

Paragraph 1 in the div.

Paragraph 2 in the div.

Paragraph 3. After a div.

Paragraph 4. After a div.

Paragraph 5 in the div.

Paragraph 6 in the div.

Paragraph 7. After a div.

Paragraph 8. After a div.

CSS – 복합 셀렉터

일반 형제 선택자(General Sibling Selector)

- Selector A ~ Selector B
- A 요소의 형제인 모든 B를 선택합니다.
- 즉, A+B와 달리 A 이후에 오는 모든 B가 선택됩니다.

```
<!DOCTYPE html>
<html>
<head>
<style>
div ~ p {
background-color: yellow;
}
</style>
</head>
<body>

<h2>General Sibling Selector</h2>

<p>The general sibling selector (~) selects all elements that are next siblings of
a specified element.</p>

<p>Paragraph 1.</p>

<div>
<p>Paragraph 2.</p>
</div>

<p>Paragraph 3.</p>
<code>Some code.</code>
<p>Paragraph 4.</p>

</body>
</html>
```

General Sibling Selector

The general sibling selector (~) selects all elements that are next siblings of a specified element.

Paragraph 1.

Paragraph 2.

Paragraph 3.

Some code.

Paragraph 4.

CSS – 가상 셀렉터

가상 셀렉터

- 가상 셀렉터에는 가상 클래스와 가상 엘리먼트가 있으며 선택된 요소에 특별한 상태 혹은 특정 부분을 선택할 수 있는 셀렉터 이다.
- :virtual selector 와 같이 사용 한다.

CSS – 가상 셀렉터

가상 클래스(pseudo class)

- 가상 클래스는 선택된 요소의 특정 상태에 동작하는 셀렉터로 예를 들어 하이퍼링크에서 마우스가 링크에 올라 갔을 때 혹은, 체크박스에서 선택이 되었을 때와 같은 상황에 적용될 스타일을 정의하기 위해 사용한다.

```
<style>
/* 하이퍼 링크의 링크 텍스트 색상 지정 */
a:link {
    color: red;
}
/* 하이퍼 링크에 마우스가 올라 갔을때의 색상 지정 */
a:hover {
    color: hotpink;
}
</style>

<body>
<p><a href="default.asp">This is a link</a></p>
</body>
```


CSS – 가상 셀렉터

가상 클래스(pseudo class)

셀렉터	사용 예	동작 설명
:active	a:active	링크를 마우스로 클릭 했을때
:hover	a:hover	마우스가 해당 요소의 위로 올라간 경우
:link	a:link	방문하지 않은 모든 링크를 선택
:focus	input:focus	<input> 태그에서 해당 요소가 마우스 포커스를 가진경우(선택된 경우)
:checked	input:checked	<input> 태그에서 체크박스가 체크 되었을때
:disabled	input:disabled	<input> 태그의 상태가 disabled 인 경우, disabled 속성 적용시
:enabled	input:enabled	<input> 태그의 상태가 enabled 인 경우
:first-child	p:first-child	해당 요소의 첫번째 자식 요소를 선택
:last-child	p:last-child	해당 요소의 마지막 자식 요소를 선택
:nth-child(n)	p:nth-child(2)	해당 요소의 n번째 자식 요소를 선택

CSS – 가상 셀렉터

가상 엘리먼트(pseudo element)

- 가상 엘리먼트는 선택된 요소의 특정 위치에 동작하는 셀렉터로 예를 들어 특정 요소의 앞,뒤,첫글자,첫 줄 등에 동작하는 스타일을 지정하기 위해 사용 한다.
- ::pseudo element 와 같이 사용 한다.
- 다음은 <h1> 태그 앞에 Title - 을 추가하는 예제 이다.

```
<style>
  h1::before {
    content: "Title - ";
  }
</style>

<body>
  <h1>This is a heading</h1>
</body>
```

Title - This is a heading

CSS – 가상 셀렉터

가상 엘리먼트(pseudo element)

셀렉터	사용 예	동작 설명
::after	p::after	선택된 요소 다음 위치
::before	p::before	선택된 요소 앞 위치
::first-letter	p::first-letter	선택된 요소의 텍스트 내용중 첫번째 글자
::first-line	p::first-line	선택된 요소의 텍스트 내용중 첫번째 줄
::selection	p::selection	선택된 요소에서 텍스트가 선택(마우스로 클릭해서 영역 지정)된 영역

CSS

고급 활용

- 이번 강의에서는 CSS에서 박스요소들을 효과적으로 배치하기 위한 flexbox, 화면 크기에 따라 최적의 레이아웃을 제공하기 위한 반응형 웹, 웹 페이지를 보다 풍성하게 꾸며주는 웹 폰트와 아이콘 사용 등 CSS의 고급 활용을 위해 필요한 내용들을 배운다.

CSS - FLEXBOX

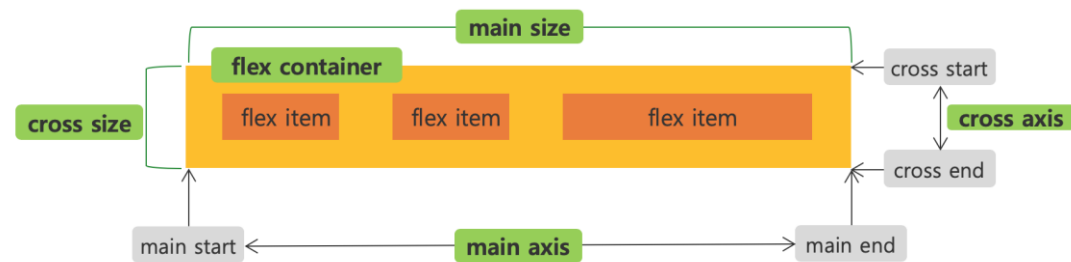
Flexbox 시작

- Flexbox 는 Flexible Box module로 박스요소들에 대한 효과적인 배치를 위해 개발되었다. 일반적으로 웹 페이지의 레이아웃은 display, float, position 등과 같은 속성을 사용해 구현한다. 하지만 이 속성을 사용하면 구현 방법이 복잡하고 레이아웃을 표현하는 데 많은 한계가 있다.
- flexbox를 사용하면 복잡한 계산 없이 요소의 크기와 순서를 유연하게 배치할 수 있게 해주며 정렬, 방향, 순서, 크기 등을 유연하게 조절할 수 있으며 기존에 불가능하거나 복잡한 구현이 필요했던 레이아웃 구성도 비교적 쉽게 구현이 가능하다.

CSS - FLEXBOX






Flexbox 시작

- 기본 구성 요소는 다음과 같다.
 - flex container: 정렬하려는 요소의 부모 요소를 말함.
 - flex item: flex container 의 아이템들로 정렬하려는 대상.
 - main axis: 메인 축으로 왼쪽에서 오른쪽으로 움직이는 inline 방향.(역방향도 가능)
 - cross axis: 메인과 교차하는 축으로 위에서 아래로 움직이는 block 방향.(역방향도 가능)
- 일반적으로 flexbox 는 작은 영역의 정밀한 레이아웃을 조정하는데 유용하고 보다 큰 레이아웃의 경우 grid 를 사용하는 것이 좋다.



CSS - FLEXBOX

flexbox 지원 현황

속성					
single-line flex box	11.0	29.0 21.0 -web kit-	22.0 18.0 -moz -	6.1 -webkit-	12.1 -web kit-
multi-line flex box	11.0	29.0 21.0 -web kit-	28.0	6.1 -webkit-	17.0 15.0 -web kit- 12.1

- IE 를 제외한 대부분의 PC 및 모바일 브라우저에서 지원됨.
- 대부분의 모바일 브라우저는 문제 없음.
- PC의 경우 크롬, 파이어폭스, 사파리, 엣지등은 문제 없음.
- Internet Explorer 10이상에서 지원되나 아직까지 많은 버그로 도입에 신중해야 함.

CSS - FLEXBOX

flexbox 개념

- 플렉스 박스(flex box)는 플렉스 컨테이너(flex container)와 플렉스 요소(flex item)로 구성된다.
- 플렉스 컨테이너(flex container)는 해당 HTML 요소의 display 속성을 설정하는 것으로 정의할 수 있다.
- 해당 요소를 블록 타입으로 정의하려면 display 속성값을 flex로, 인라인 타입으로 정의하려면 inline-flex로 설정한다.
- 플렉스 컨테이너는 언제나 하나 이상의 플렉스 요소를 포함해야 한다.
- 플렉스 컨테이너의 외부와 플렉스 요소의 내부의 모든 것들은 평소처럼 동작한다.
- 플렉스 박스(flex box)는 오직 플렉스 요소가 플렉스 컨테이너의 내부에서 어떻게 위치 하는가를 정의한다.
- 플렉스 요소는 플렉스 컨테이너 안에서 플렉스 라인(flex line)이라는 가상의 선을 따라 위치하게 된다.
- 기본적으로 하나의 플렉스 컨테이너는 오직 단 하나의 플렉스 라인만을 가지고 있다.

CSS - FLEXBOX

flexbox 적용

- 정렬하려는 요소의 부모 요소 display 속성을 flex 로 지정한다.
- flexbox 속성은 부모 요소인 flex container에 정의하는 속성과 자식 요소인 flex item에 정의하는 속성으로 구분되어 있다.

```
.flex_container {  
  display: flex;  
}
```

- flex: flex container는 block요소와 같이 수직 정렬.
- inline-flex: flex container는 inline 요소와 같이 수평 정렬.

CSS - FLEXBOX

flex container 속성

- 전체적인 정렬이나 흐름에 관련된 속성은 flex container에서 정의 한다.

CSS - FLEXBOX

flex container 속성 - flex-direction

- flex-direction 속성은 플렉스 컨테이너 안에서 플렉스 요소가 배치될 방향을 설정한다.
- 이 속성은 다음과 같은 속성값을 가질 수 있다.
 - row : 기본 설정으로, 플렉스 요소는 왼쪽에서 오른쪽으로, 그리고 위쪽에서 아래쪽으로 배치된다.
 - Row-reverse : 만약에 direction 속성값이 ltr(left-to-right)이면, 플렉스 요소는 반대로 오른쪽에서 왼쪽으로 배치된다.
 - Column : 만약에 쓰기 방식이 수평이면, 플렉스 요소는 수직 방향으로 위쪽에서 아래쪽으로 배치된다.
 - Column-reverse : 만약에 쓰기 방식이 수평이면, 플렉스 요소는 수직 방향으로 아래쪽에서 위쪽으로 배치된다.

```
- row: left -> right  
- row-reverse: left -> right  
- column: top -> bottom  
- column-reverse: bottom -> top
```

CSS - FLEXBOX

flex container 속성 - flex-wrap

- flex-wrap 속성은 플렉스 라인에 더 이상의 여유 공간이 없을 때, 플렉스 요소의 위치를 다음 줄로 넘길지를 설정한다.
- 이 속성은 다음과 같은 속성값을 가질 수 있다.
 - nowrap : 기본 설정으로, 플렉스 요소가 다음 줄로 넘어가지 않는다. 대신에 플렉스 요소의 너비를 줄여서 한 줄에 모두 배치시킨다.
 - wrap : 플렉스 요소가 여유 공간이 없으면 다음 줄로 넘어가서 배치된다.
 - wrap-reverse : 플렉스 요소가 여유 공간이 없으면 다음 줄로 넘어가서 배치된다. 단, 아래쪽이 아닌 위쪽으로 넘어간다.

- wrap: 기본값으로 화면 크기에 따라 여러라인에 배치
- nowrap: 하나의 라인에 모든 item 을 배치
- wrap-reverse: 아래쪽에서 위쪽으로 여러라인에 배치

CSS - FLEXBOX

flex container 속성 - justify-content

- justify-content 속성은 플렉스 요소의 수평 방향 정렬 방식을 설정한다.
- 이 속성은 다음과 같은 속성값을 가질 수 있다.
 - flex-start : 기본 설정으로, 플렉스 요소는 플렉스 컨테이너의 앞쪽에서부터 배치된다.
 - Flex-end : 플렉스 요소는 플렉스 컨테이너의 뒤쪽에서부터 배치된다.
 - Center : 플렉스 요소는 플렉스 컨테이너의 가운데에서부터 배치된다.
 - Space-between : 플렉스 요소는 요소들 사이에만 여유 공간을 두고 배치된다.
 - Space-around : 플렉스 요소는 앞, 뒤, 그리고 요소들 사이에도 모두 여유 공간을 두고 배치된다.

CSS - FLEXBOX

flex container 속성 - align-items

- align-items 속성은 플렉스 요소의 수직 방향 정렬 방식을 설정한다.
- 이 속성은 한 줄만을 가지는 플렉스 박스에서는 효과가 없으며, 두 줄 이상을 가지는 플렉스 박스에서만 효과가 있다.
- 이 속성은 다음과 같은 속성값을 가질 수 있다.
 - stretch : 기본 설정으로, 플렉스 요소의 높이가 플렉스 컨테이너의 높이와 같게 변경된 뒤 연이어 배치된다.
 - flex-start : 플렉스 요소는 플렉스 컨테이너의 위쪽에 배치된다.
 - flex-end : 플렉스 요소는 플렉스 컨테이너의 아래쪽에 배치된다.
 - center : 플렉스 요소는 플렉스 컨테이너의 가운데에 배치된다.
 - baseline : 플렉스 요소는 플렉스 컨테이너의 기준선(baseline)에 배치된다.

- stretch: 기본값으로 컨테이너를 모두 채워 정렬
- flex-start, flex-end: 축의 시작 혹은 종료 위치로 부터 정렬
- center: 수직축의 중간에 요소들을 배치(각 요소 크기별로 중심을 잡아 정렬)
- baseline: 베이스라인 정렬(차이점 추가 설명 필요)

CSS - FLEXBOX

flex container 속성 - align-content

- align-content 속성은 flex-wrap 속성의 동작을 변경할 수 있다.
- 이 속성은 align-items 속성과 비슷한 동작을 하지만, 플렉스 요소를 정렬하는 대신에 플렉스 라인을 정렬한다.
- 이 속성은 다음과 같은 속성값을 가질 수 있다.
 - stretch : 기본 설정으로, 플렉스 라인의 높이가 남는 공간을 전부 차지하게 된다.
 - flex-start : 플렉스 라인은 플렉스 컨테이너의 앞쪽에 뭉치게 된다.
 - flex-end : 플렉스 라인은 플렉스 컨테이너의 뒤쪽에 뭉치게 된다.
 - center : 플렉스 라인은 플렉스 컨테이너의 가운데에 뭉치게 된다.
 - space-between : 플렉스 라인은 플렉스 컨테이너에 고르게 분포된다.
 - space-around : 플렉스 라인은 플렉스 컨테이너에 고르게 분포됩니다. 단, 양쪽 끝에 약간의 공간을 남겨둔다.

- stretch: 기본값으로 컨테이너 전체 높이에 맞게 늘어나 모든 item 들이 배치
- flex-start, flex-end: 전체 item을 축의 시작 혹은 종료 위치로 부터 정렬
- center: 전체 item을 하나의 묶음으로 봤을때 중앙 정렬
- space-between: 첫번째 item은 위쪽, 마지막 item은 아래쪽에 정렬되고 나머지 item 을 사이에 정렬.
- space-around: item들의 위쪽 및 아래쪽 공간을 균등하게 배분.

CSS

반응형 웹(Responsive Web)

- 웹사이트 레이아웃을 설계할 때는 사용자의 모니터 화면 해상도를 고려해야 한다. 너무 크게 가로 폭을 만들면 작은 해상도의 모니터로 접속했을 때 가로 스크롤이 생겨 콘텐츠를 보는 게 불편하고 특히 스마트폰이나 태블릿 등 모바일 기기로 접속할 경우 화면이 작기 때문에 가독성을 신경을 써야 한다.
- 반응형 웹이라는 표현은 Ethan Marcotte에 의해 2010년 처음 사용된 것으로 알려져 있다. 기본적인 정의는 다음과 같다.
- 콘텐츠는 사용자 기기의 화면크기, 해상도 등 요구사항에 맞게 반응.
- 레이아웃은 기기의 크기와 기능에 따라 변함.
- 예를 들어 휴대폰에서는 콘텐츠가 단일 열 뷰로 표시될 수 있고 태블릿에서는 동일한 콘텐츠가 두 개의 열로 표시될 수 있음.
- 반응형 웹은 프로그램요소가 아니며 HTML, CSS만으로 구성.

CSS

뷰포트(Viewport)

- 뷰포트는 사용자에게 보여지는 웹 페이지 영역을 말한다. 데스크탑에서는 브라우저 크기가 뷰포트가 되지만 모바일의 경우 윈도우 크기를 조절할 수 없기 때문에 디바이스의 크기가 뷰포트가 된다. 최근 들어서는 모바일기기들의 크기가 다양해진 관계로 특정 크기를 표준으로 정하기는 어렵다.
- 반응 형 웹 디자인의 기본은 뷰포트를 설정하는 것으로 앞에서 배운 것 처럼 <meta> 태그에 뷰포트 설정을 하게 된다.

CSS

뷰포트(Viewport)

- width=device-width : 페이지의 너비를 기기의 스크린 너비로 설정. 즉 콘텐츠가 보여지는 영역을 기기의 뷰포트의 크기와 동일하게 함.
- initial-scale=1.0 : 페이지 로딩 시 확대/축소가 되지 않은 원래 크기를 사용하도록 한다. 화면 회전 방향에 상관없이 기기 독립적 픽셀과 css픽셀이 1:1 이 되도록 함.

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

CSS

뷰포트(Viewport)

- 추가적인 속성은 다음과 같다.
 - minimum-scale : 줄일 수 있는 최소 크기. 범위는 0~10.
 - maximum-scale : 늘릴 수 있는 최대 크기. 범위는 0~10.
 - user-scalable : yes or no, 사용자가 화면을 확대/축소 (핀치 투 줌, 두손가락으로 확대) 할 수 있는지는 지정.
- 다음은 적절한 뷰포트 활용을 위한 몇가지 원칙이다.
 - 크기가 큰 fixed 요소를 사용하지 말 것.
 - 특정 뷰포트 크기에 최적화된 콘텐츠를 두지 말 것.
 - 미디어쿼리를 이용해 화면 크기가 다른 경우에 최적화된 스타일을 적용.

CSS

미디어 쿼리(Media Query)

- CSS2에서는 @media 규칙을 통해 서로 다른 매체 유형(media type)을 위한 맞춤 식 스타일 시트(style sheet)를 지원한다.
- 예를 들면, HTML 문서가 스크린에 표현될 때와 프린트할 때 서로 다른 스타일을 적용할 수 있다.
- CSS3에서는 @media 규칙을 더욱 발전시켜 매체 유형(media type)과 하나 이상의 표현식(expression)으로 구성된 미디어 쿼리(media query)를 사용할 수 있다.
- 미디어 쿼리(media query)는 width, height, color 속성과 같은 미디어 관련 속성을 이용한 표현식을 통해 스타일이 적용되는 범위를 조절할 수 있다.
- 미디어 쿼리를 사용하면 콘텐츠(content)를 별도로 변경하지 않아도 웹 페이지에 접속하고 있는 기기에 알맞은 형태로 스타일이 조정된다.

CSS

미디어 쿼리(Media Query) - 속성

속성	설명
width	화면의 너비
height	화면의 높이
device-width	매체 화면의 너비
device-height	매체 화면의 높이
device-aspect-ratio	매체 화면의 비율
orientation	매체 화면의 방향
color	매체의 색상 비트 수
color-index	매체에서 표현 가능한 색상의 개수
monochrome	흑백 매체에서의 픽셀당 비트 수
resolution	매체의 해상도

CSS

Mobile first

- 작은 가로 폭부터 큰 가로 폭 순서로 만드는 것을 모바일 우선(Mobile First)이라고 한다.
- Bootstrap 등 대부분의 css 라이브러리는 모바일 우선 이다.

```
/* 기본적인 css 속성 */  
...  
@media ( min-width: 768px ) {  
    /* 768px 이상의 경우 적용 */  
}  
@media ( min-width: 1024px ) {  
    /* 1024px 이상인 경우 지정 */  
}
```

CSS

Desktop first

- 큰 가로 폭부터 작은 가로 폭 순서로 만드는 것을 데스크탑 우선(Desktop First)이라고 한다.

```
/* 기본적인 css 속성 */  
...  
@media ( max-width: 1023px ) {  
    /* 1023px 이하인 경우 적용 */  
}  
@media ( max-width: 767px ) {  
    /* 767px 이하인 경우 적용 */  
}
```