

# 기초웹개발론

# CSS Part.1

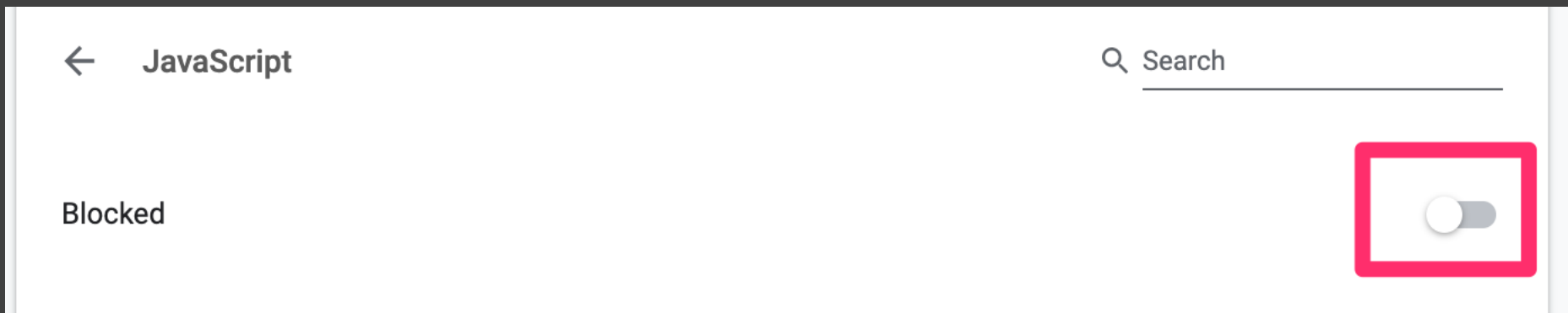
# CSS

- **C**ASCADING **S**TYLE **S**HEET
- 마크업 언어(HTML)가 실제 화면에 표현되는 방법을 기술하는 언어
- HTML 문서의 스타일과 레이아웃을 정의한다.
- W3C 에서 표준안을 제정하며 현재 CSS Level 3 까지 있음.
- 스타일 선언은 위에서 아래로 순차적으로 적용되며, 마지막 선언된 스타일이 우선순위를 갖게됨.

# CSS가 필요한 이유

- 웹문서의 내용과는 상관없이 디자인을 변경할수있음
- 문서와 디자인을 분리하여 다양한 환경에 유연하게 대처 할수 있음.

<chrome://settings/content/javascript?search=javascript>



kakao

# 인라인 스타일

- 간단한 스타일의 경우 태그에 직접 스타일을 작성
- `<tag style="속성:속성값;" >`

```
<!DOCTYPE html>
```

```
<html>  
<body>  
• <p style="color:red;"> 인라인 스타일 </p>
```

```
<h1 style="color: blue;text-align:center;">This is a heading</h1>  
<p style="color:red;">This is a paragraph.</p>
```

```
</body>  
</html>
```

# 스타일, 스타일 시트

- 스타일 시트 : 여러개의 스타일 규칙을 한군데에 묶어 놓은 것.

- 내부 스타일 시트

- 웹문서 안에 직접 스타일을 정리

- <head> 태그 안의 <style></style> 태그 사이에 내용을 작성(권장)

- 상황에 따라 <body> 안에도 선언이 가능함.

- 외부 스타일 시트

- <link href="외부 스타일 시트 파일 경로" rel="stylesheet">

- 파일을 여러개 로딩하는게 가능해서 공통 부분을 분리하여 사용하기에 편함.

kakao

# 스타일 형식

/\* 텍스트를 단락 중앙에 정렬하는 스타일 \*/

p { text-align: center; }

Selector

스타일 속성명

스타일 속성값

kakao

# 스타일 표기방식

- 스타일 속성은 세미콜론(;)으로 구분하여 중괄호 안에 나열한다.
- 각 속성은 여러줄에 나누어 표기(공백무시)
- 주석은 /\* 와 \*/ 사이에 작성



# 스타일 명시도(Specificity)


- 인라인 스타일 : 태그안의 style 속성으로 해당 태그에만 적용됨
- ID 스타일 : 지정한 부분에만 적용되는 스타일, 한문서안에 한개만 적용
- 클래스 스타일 : class 속성과 일치하는 부분에만 적용
- 태그 스타일 : 특정 태그에 똑같이 적용되는 스타일


인라인 > ID > 클래스 > 태그

# 우선순위 변경

- **!important**

- 다른 어떤 스타일 보다 최우선적으로 적용되는 스타일
- 중복되어 있을 경우 더 큰 우선순위를 갖는 선언이 적용됨
- 꼭 필요한 경우에만 사용

```
.css {  
  color: green;  
  font-size:2px !important;  
};
```



# 스타일의 상속

- 스타일시트는 자식요소의 별도의 스타일을 지정하지 않는 경우 부모의 스타일 속성이 자식요소로 전달됨
- 상속을 이용해 스타일 시트를 효과적으로 만들수 있음.
  - EX) 사이트 폰트나 컬러등 공통으로 사용 되는 요소를 body 태그에 적용
- 상속 예외 되는 속성도 있음
  - 배경색, 배경 이미지등..

kakao

```

-webkit-user-select: none; /* Chrome all / Safari all */
-moz-user-select: none; /* Firefox all */
-ms-user-select: none; /* IE 10+ */
user-select: none; /* Likely future */

```

# CSS Vendor Prefix

- 표준 규약이 아닌 속성들은 브라우저에 따라 다른 방식으로 구현
- 속성 앞에 접두어(prefix) 를 붙여 브라우저 별로 구별함.

표준 규약으로 변경 되었어도 이전 버전의 브라우저를 고려해 접두어를 붙여서 사용하기도함.

접두사	설명
-webkit-	웹킷을 사용하는 브라우저들(크롬, 사파리)
-moz-	gecko 엔진을 사용하는 브라우저(모질라 파이어폭스)
-o-	오페라 브라우저
-ms-	MS 인터넷 익스플로러

# SELECTOR

- 스타일을 적용하고자 하는 HTML 을 지정하는 방법.
- 복수개의 selector 를 연속하여 지정가능하며 콤마(,) 로 구분

```
h1 { text-align: center; }  
h2 { text-align: center; }  
h3 { text-align: center; }
```



```
h1, h2, h3 { text-align: center; }
```

# SELECTOR 의 종류

- 전체 선택자 (\*)
- 태그 선택자
- ID 선택자
- Class 선택자
- Attribute 선택자
- 복합 선택자
- 가상 선택자(Pseudo-Class Selector)
- 가상 요소 선택자(Pseudo-Element Selector)

Selector	Example
<u>.class</u>	.intro
<u>#id</u>	#firstname
<u>*</u>	*
<u>element</u>	p
<u>element,element,..</u>	div, p

## • 전체 selector

패턴	Description
*	HTML 문서내의 모든 문서 요소를 선택한다. HTML 요소를 포함한 모든 요소가 선택된다(head 요소도 포함됨)

```
<!DOCTYPE html>
<html>
<head>
  <style>
    * { color: red; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div>
    <p>paragraph 1</p>
    <p>paragraph 2</p>
    <p>paragraph 3</p>
    <p>paragraph 4</p>
  </div>
  <p>paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

paragraph 2

paragraph 3

paragraph 4

paragraph 5

- TAG selector

패턴	Description
태그명	지정된 태그명을 가지는 요소를 선택한다.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    p { color: red; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div>
    <p>paragraph 1</p>
    <p>paragraph 2</p>
    <p>paragraph 3</p>
    <p>paragraph 4</p>
  </div>
  <p>paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

paragraph 2

paragraph 3

paragraph 4

paragraph 5



- ID selector

패턴	Description
#id 어트리뷰트 값	id 어트리뷰트 값을 지정하여 일치하는 요소를 선택한다. id 어트리뷰트 값은 중복될수 없는 유일한 값이다.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    #p1 { color: red; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div>
    <p id="p1">paragraph 1</p>
    <p id="p2">paragraph 2</p>
    <p>paragraph 3</p>
    <p>paragraph 4</p>
  </div>
  <p>paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

paragraph 2

paragraph 3

paragraph 4

paragraph 5

- class selector

패턴	Description
.class 어트리뷰트 값	.class 어트리뷰트 값을 지정하여 일치하는 요소를 선택한다. id 어트리뷰트 값은 중복될수 있다.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    .container {color: blue;}
    #p1 { color: red; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div class="container">
    <p id="p1">paragraph 1</p>
    <p id="p2">paragraph 2</p>
    <p>paragraph 3</p>
    <p>paragraph 4</p>
  </div>
  <p>paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

paragraph 2

paragraph 3

paragraph 4

paragraph 5

- Attribute selector

패턴	Description
셀렉터[어트리뷰트]	지정된 어트리뷰트를 갖는 모든 요소를 선택한다.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    p { color: red; }
    p[title] { color: green; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div class="container">
    <p title="paragraph 01 KR">paragraph 1</p>
    <p title="paragraph 02 EN">paragraph 2</p>
    <p title="paragraph 03 KR">paragraph 3</p>
    <p>paragraph 4</p>
  </div>
  <p>paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

paragraph 2

paragraph 3

paragraph 4

paragraph 5

- Attribute selector

패턴	Description
셀렉터[어트리뷰트="값"]	지정된 어트리뷰트 와 값이 정확히 일치하는 모든 요소를 선택한다.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    p { color: red; }
    p[title="paragraph 01 KR"] { color: green; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div class="container">
    <p title="paragraph 01 KR">paragraph 1</p>
    <p title="paragraph 02 EN">paragraph 2</p>
    <p title="paragraph 03 KR">paragraph 3</p>
    <p>paragraph 4</p>
  </div>
  <p>paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

paragraph 2

paragraph 3

paragraph 4

paragraph 5

- Attribute selector

패턴	Description
셀렉터[어트리뷰트~="값"]	지정된 어트리뷰트 와 값이 정확히 일치하는 모든 요소를 선택한다. 공백으로 구분한 여러개의 값을 가질수 있음.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    p { color: red; }
    p[title~="KR"] { color: green; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div class="container">
    <p title="paragraph 01 KR">paragraph 1</p>
    <p title="paragraph 02 EN">paragraph 2</p>
    <p title="paragraph 03 KR">paragraph 3</p>
    <p>paragraph 4</p>
  </div>
  <p>paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

paragraph 2

paragraph 3

paragraph 4

paragraph 5

- Attribute selector

패턴	Description
셀렉터[어트리뷰트 ="값"]	지정된 어트리뷰트의 값이 일치하거나 “값-(하이픈)” 으로 연결된 요소를 선택한다.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    p { color: red; }
    p[title="paragraph"] { color: green; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div class="container">
    <p title="paragraph 01 KR">paragraph 1</p>
    <p title="paragraph 02 EN">paragraph 2</p>
    <p title="paragraph 03 KR">paragraph 3</p>
    <p title="paragraph-03-KR">paragraph 4</p>
  </div>
  <p title="paragraph">paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

paragraph 2

paragraph 3

paragraph 4

paragraph 5

- Attribute selector

패턴	Description
셀렉터[어트리뷰트^="값"]	지정된 어트리뷰트의 값으로 시작하는 모든 요소를 선택한다.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    p { color: red; }
    p[title^="paragraph"] { color: green; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div class="container">
    <p title="paragraph 01 KR">paragraph 1</p>
    <p title="paragraph 02 EN">paragraph 2</p>
    <p title="paragraph 03 KR">paragraph 3</p>
    <p title="paragraph-03-KR">paragraph 4</p>
  </div>
  <p title="paragraph">paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

paragraph 2

paragraph 3

paragraph 4

paragraph 5

- Attribute selector

패턴	Description
셀렉터[어트리뷰트\$="값"]	지정된 어트리뷰트의 값으로 끝나는 모든 요소를 선택한다.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    p { color: red; }
    p[title$="KR"] { color: green; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div class="container">
    <p title="paragraph 01 KR">paragraph 1</p>
    <p title="paragraph 02 EN">paragraph 2</p>
    <p title="paragraph 03 KR">paragraph 3</p>
    <p title="paragraph-03-KR">paragraph 4</p>
  </div>
  <p title="paragraph">paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

paragraph 2

paragraph 3

paragraph 4

paragraph 5



- Attribute selector

패턴	Description
셀렉터[어트리뷰트*="값"]	지정된 어트리뷰트의 값을 포함하는 요소를 선택한다.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    p { color: red; }
    p[title*="0"] { color: green; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div class="container">
    <p title="paragraph 01 KR">paragraph 1</p>
    <p title="paragraph 02 EN">paragraph 2</p>
    <p title="paragraph 03 KR">paragraph 3</p>
    <p title="paragraph-03-KR">paragraph 4</p>
  </div>
  <p title="paragraph">paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

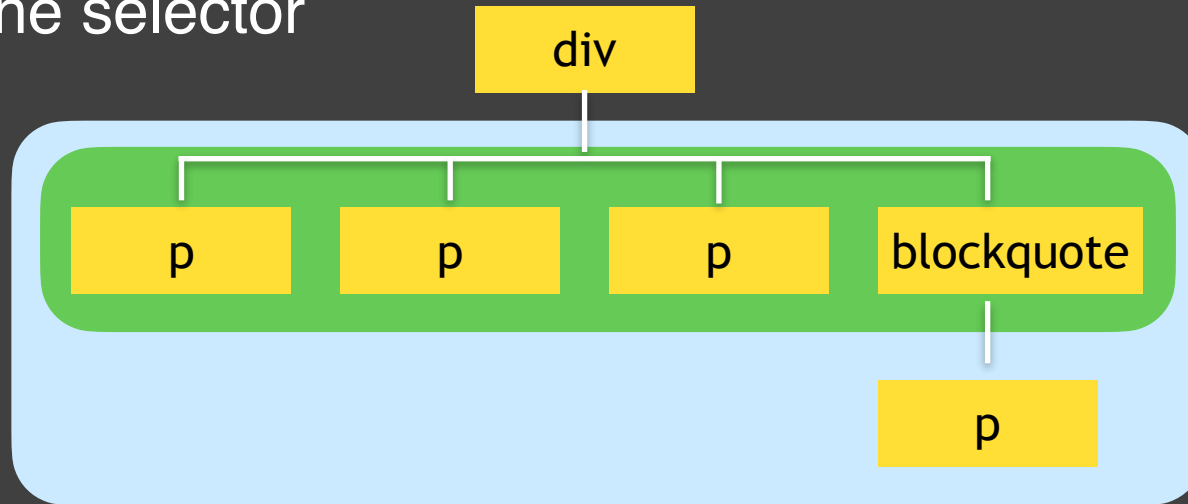
paragraph 2

paragraph 3

paragraph 4

paragraph 5

- Combine selector



- 자신의 상위는 부모요소 (Parent)
- 바로 하위는 자식요소 (Child)
- 하위에 속하는 모든 요소는 하위(후손)요소 (Descendant)
- 자신과 같은 level 의 요소를 형제 요소 (Sibling)

```
<!DOCTYPE html>
<html>
<head>
  <style>
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div class="container">
    <p>paragraph 1</p>
    <p>paragraph 2</p>
    <p>paragraph 3</p>
    <blockquote><p>paragraph 4</p></blockquote>
  </div>
  <p title="paragraph">paragraph 5</p>
</body>
</html>
```

- Combine selector - 후손 셀렉터 (Descendant Combinator)

패턴	Description
A B	A의 모든 후손 요소중 B 와 일치하는 요소를 찾는다.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    div p { color: red; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div class="container">
    <p>paragraph 1</p>
    <p>paragraph 2</p>
    <p>paragraph 3</p>
    <blockquote><p>paragraph 4</p></blockquote>
  </div>
  <p title="paragraph">paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

paragraph 2

paragraph 3

paragraph 4

paragraph 5

- Combine selector - 자식 셀렉터 (Child Combinator)

패턴	Description
A > B	A의 모든 자식 요소중 B 와 일치하는 요소를 찾는다.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    div > p { color: red; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div class="container">
    <p>paragraph 1</p>
    <p>paragraph 2</p>
    <p>paragraph 3</p>
    <blockquote><p>paragraph 4</p></blockquote>
  </div>
  <p title="paragraph">paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

paragraph 2

paragraph 3

paragraph 4

paragraph 5

- Combine selector - 인접형제 셀렉터 (Adjacent Sibling Combinator)

패턴	Description
A + B	A의 형제 요소중 A 바로 뒤에 위치 하는 B 요소를 찾는다.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    p + p { color: red; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div class="container">
    <p>paragraph 1</p>
    <p>paragraph 2</p>
    <p>paragraph 3</p>
    <blockquote><p>paragraph 4</p></blockquote>
  </div>
  <p title="paragraph">paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

paragraph 2

paragraph 3

paragraph 4

paragraph 5

- Combine selector - 일반형제 셀렉터 (General Sibling Combinator)

패턴	Description
A ~ B	A의 형제 요소중 A 뒤에 위치 하는 B 요소를 모두 선택한다.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    p ~ p { color: red; }
  </style>
</head>
<body>
  <h1>Selector</h1>
  <div class="container">
    <p>paragraph 1</p>
    <p>paragraph 2</p>
    <p>paragraph 3</p>
    <blockquote><p>paragraph 4</p></blockquote>
  </div>
  <p title="paragraph">paragraph 5</p>
</body>
</html>
```

## Selector

paragraph 1

paragraph 2

paragraph 3

paragraph 4

paragraph 5

- Pseudo-Class selector - 가상클래스 셀렉터
- 수도 Class
- 가상 Class
- 의사 Class

직접 확인해 보기

[https://www.w3schools.com/css/css\\_pseudo\\_classes.asp](https://www.w3schools.com/css/css_pseudo_classes.asp)

- Link pseudo-classes, User action pseudo-classes

패턴	Description
:link	방문하지 않은 링크
:visit	방문한 링크
:hover	마우스가 올라가 있을때
:active	클릭된 상태 일때
:focus	포커스가 들어와 있을때.



- UI 요소 상태 가상 클래스 - UI element states pseudo-classes

패턴	Description
:checked	체크 상태일때
:enabled	사용가능한 상태일때
:disabled	사용 불가능한 상태일때

- 구조 가상 클래스 - Structural pseudo classes

패턴	Description
:first-child	셀렉터의 요소중 첫번째 자식 요소를 선택한다.
:last-child	셀렉터의 요소중 마지막 자식 요소를 선택한다.
:nth-child(n)	셀렉터에 해당하는 요소 중 앞에서 n 번째 자식인 요소를 선택한다.
:nth-last-child(n)	셀렉터에 해당하는 요소 중 뒤에서 n 번째 자식인 요소를 선택한다.

- 부정셀렉터 - Negation pseudo classes

패턴	Description
:not	해당하지 않는 모든 요소를 선택한다.

- 정합성 셀렉터 - validity pseudo classes

패턴	Description
:valid	정합성 검증이 성공한 input 요소 또는 form 요소
:invalid	정합성 검증이 실패한 input 요소 또는 form 요소

- 가상요소 - pseudo element Selector

패턴	Description
::first-letter	컨텐츠의 첫글자를 선택한다.
::first-line	컨텐츠의 첫줄을 선택한다.(블럭요소에만 적용가능)
::after	컨텐츠의 뒤에 위치 하는 공간을 선택하며, content 어트리뷰트와 함께 사용된다.
::before	컨텐츠의 앞에 위치 하는 공간을 선택하며, content 어트리뷰트와 함께 사용된다.
::selection	드래그한 콘텐츠를 선택한다.

- 가상요소는 특정부분에 스타일을 적용하기 위해 사용된다.
  - 요소 컨텐츠의 첫글자 또는 첫줄
  - 요소 컨텐츠의 앞 또는 뒤
- 가상요소는 두개의 콜론(::)을 사용한다.

# 실습(과제)

## ☐ ROOT

- html
- css
- ☐ javascript
  - react
  - vue

## ROOT

- html
- css
- javascript
  - react
  - vue

## ROOT

- html
- css
- javascript

- ROOT, javascript를 클릭하면 체크박스가 선택되도록 label 코딩을 합니다.
- 체크박스를 안보이게 합니다.
- ROOT나 javascript를 클릭하면 형제요소의 ul이 안보이게 합니다.
- {display:none}, ~, :checked 등을 이용합니다.

# TEXT 관련 스타일

# 글꼴 관련 스타일

- font-family - 글꼴지정
  - font-family:<글꼴이름>[,<글꼴 이름>, <글꼴 이름>]
  - 지정된 폰트가 시스템에 설치 되어 있지 않으면 기본폰트로 보여짐

\* {font-family:AppleSDGothicNeo-Regular,sans-serif}

# 웹폰트

- 시스템에 폰트가 설치 되어 있지 않은 사용자에게도 동일한 폰트를 보여 주기 위한 방법
- 구글웹폰트
  - <https://fonts.google.com/earlyaccess>
  - <https://fonts.google.com/>
  - <https://fonts.googleapis.com/css?family=Indie+Flower|Londrina+Outline|Open+Sans+Condensed:300>

# 웹폰트

```
@font-face {
```

```
font-family: 'Indie Flower';
```

```
font-style: normal;
```

```
font-weight: 400;
```

```
src: local('Indie Flower'),
```

```
local('Indie Flower');
```

- 직접 업로드하여 사용가능함(저작권에 주의)

```
url(https://fonts.gstatic.com/s/indieflower/v1/mUvjHnKknhhQmKkfcZVaUuH99GUDg.woff2) format('woff2');  
unicode-range: U+0000-00FF, U+0131, U+0152-0153, U+02BB-02BC, U+02C6, U+02DA, U+02DC, U+2000-206F, U+2074, U
```

```
@font-face {
```

```
font-family: <a-remote-font-name>;
```

```
src: <source> [,<source>]*;
```

```
[font-weight: <weight>];
```

```
[font-style: <style>];
```

```
}
```

```
@font-face {
```

```
font-family: 'trana';
```

```
src: local('trana'),
```

```
url('trana.woff') format('woff'),
```

```
url('trana.ttf') format('truetype');
```

```
}
```

```
.w-font { font-family: 'trana', sans-serif; }
```



# 글자크기

- font-size: <절대크기> | <상대크기> | <크기> | <백분율>

속성값	설명
<절대크기>	브라우저에서 지정한 글자크기. xx-small, x-small, small, medium, large, x-large, xx-large
<상대크기>	부모요소 글자크기 기준으로 표시. larger, smaller
<크기>	글자크기 직접 지정
<백분율>	부모요소의 글자 크기를 기준으로 해당하는 %를 계산해 표시

# 글자크기

단위	설명
em	해당 글꼴의 대문자 M 의 너비를 기준으로 크기를 조절
ex	x-height. 해당 글꼴의 소문자 x의 높이를 기준으로 크기를 조절
px	픽셀, 모니터에 따라 상대적 크기가 됨. 고정된값
pt	포인트. 일반 문서에서 많이 사용되는 단위.
rem	html 태그의 폰트 크기에 따른 상대적 크기.

# 글자 굵기 지정

- font-weight: normal | bold | bolder | lighter | 100 ~ 900

단위	설명
normal	일반적인 형태, 기본값
bold   lighter   bolder	굵게, 가늘게, 더 굵게
100 ~ 900	400 == normal, 700 == bold 숫자값을 조절해 세밀하게 두께를 조절가능함.

# 글자 스타일

- font-style: normal | italic | oblique

단위	설명
normal	일반적인 형태, 기본값
italic	이탤릭체
oblique	기울어진 서체(화면에서 보여 지기에는 italic과 같음)

# 텍스트 스타일

- 글자 색 지정
  - color: <색상>
- 색상표현 방법
  - 16진수 표기법(#RRGGBB)
  - RGB, RGBA (rgba(255, 0, 0, 0.5))
  - HSL, HSLA (hsla(240, 100%, 25%, 0.3))
  - 색상이름표기법(red, blue, yellow, black.....)
  - [https://www.w3schools.com/colors/colors\\_names.asp](https://www.w3schools.com/colors/colors_names.asp)

# 텍스트 줄 표시

- text-decoration: none | underline | overline | line-through

속성 값	설명
none	줄표시안함 기본값.
underline	밑줄표시
overline	영역위로 선을 표시
line-through	취소선 표시

# 그림자 효과

```
/* offset-x | offset-y | blur-radius | color */
```

```
text-shadow: 1px 1px 2px black;
```

```
/* color | offset-x | offset-y | blur-radius */
```

```
text-shadow: #fc0 1px 0 10px;
```

```
/* offset-x | offset-y | blur-radius | color */
```

• **text-shadow: none | <가로거리> <세로거리> <번짐정도> <색상>**

```
text-shadow: 5px 5px #558abb;
```

```
/* color | offset-x | offset-y | blur-radius */
```

```
text-sh
```

```
/* offs
```

```
/* Use
```

```
text-sh
```

```
/* Glob
```

```
text-shadow: inherit;
```

```
text-shadow: initial;
```

```
text-shadow: unset;
```

속성 값	설명
<가로거리>	텍스트로 부터 그림자까지의 가로 길이. 양수는 오른쪽, 음수는 왼쪽
<세로거리>	텍스트로 부터 그림자까지의 세로 길이. 양수는 아래쪽, 음수는 위쪽
<번짐정도>	그림자가 번지는 정도. 양수는 퍼져나감, 음수는 좁혀짐
색상	그림자 색상을지정한다, 공백으로 구분해 여러색상 지정가능

# 공백 처리하기

	개행 문자	스페이스, 탭	자동 줄 바꿈	줄 끝의 공백
<b>normal</b>	병합	병합	예	제거
<b>nowrap</b>	병합	병합	아니오	제거
<b>pre</b>	유지	유지	아니오	유지
<b>pre-wrap</b>	유지	유지	예	넘침
<b>pre-line</b>	유지	병합	예	제거
<b>break-spaces</b>	유지	유지	예	유지

• white-space: normal | nowrap | pre | pre-line | pre-wrap

속성 값	설명
normal	여러개의 공백을 하나로 표시
nowrap	여러개의 공백을 하나로 표시, 영역 너비를 넘어가는 내용을 한줄로 표시
pre	여러개의 공백을 그대로 표시, 영역 너비를 넘어가는 내용을 한줄로 표시
pre-line	여러개의 공백을 하나로 표시, 영역 너비를 넘어가는 내용은 자동 줄바꿈, 줄바꿈은 그대로 표시
pre-wrap	여러개의 공백을 그대로 표시, 영역 너비를 넘어가는 내용은 자동으로 줄바꿈



# 텍스트 간격 조절

- 글자 사이의 간격을 조절
  - letter-spacing:normal | <크기>
- 단어와 단어 사이의 간격을 조절
  - word-spacing:normal | <크기>

# 실습

## 최신 웹 디자인 트렌드

**반응형 웹 디자인** - 다양한 화면 크기에 최적화하다

**플랫 디자인** - 입체에서 평면으로

**풀스크린 배경** - 콘텐츠에 집중

**원 페이지 사이트** - 한 페이지에 모든 내용을 담다

**패럴랙스 스크롤링** - 동적인 효과로 강한 인상을!

**웹 폰트** - 웹 타이포그래피를 받쳐주는 기술

1. SPAN 태그 사용
2. Class 속성 사용
3. 텍스트 굵기 700
4. 굵은 텍스트의 크기는 기본 폰트의 1.2배