

# HTML5 기초

# 1. HTML5의 탄생

## ❖ HTML5의 탄생 History

- HTML5가 탄생하기 이전까지 HTML의 최초 버전은 1993년에, 최신 버전은 1999년에 발표 되었다.
- W3C가 XHTML1.0을 구체화 하기 위해 XHTML2.0 작업을 진행 중이었으나 하위 호환상에 치명적인 문제가 있었다.
- 2004년 애플, 모질라 재단, 오페라 소프트웨어가 공동으로 설립한 공개 그룹인 WHATWG(Web Hypertext Application Technology Working Group)가 W3C와 별개로 Web Application 1.0과 Web Forms 2.0을 만들었다.
- W3C가 2007년 3월 공개적으로 XHTML 2.0의 실패를 인정한 후 새롭게 HTML을 개발하기로 결정하면서 WHATWG의 표준안을 대부분 수용하여 HTML5가 탄생하게 되었다.

## 2. 작성 규칙

### ❖ HTML 명령 태그(Tag)

- HTML 문서는 요소(element)와 태그(tag) 그리고 콘텐츠로 구성되어 있다.
- 태그는 시작 태그와 종료 태그로 나눌 수 있으며 "<"와 ">"로 둘러싸여 있다.
- 기본 형식은 시작 태그의 경우 <tag>의 형태를 가지며, 종료 태그의 경우에는 </tag> 형태를 가진다.
- 일부 태그의 경우, 종료 태그를 가지지 않는 경우도 있는데, 이를 '빈 요소(empty element)'라고 한다.

`<p>` HTML과 XHTML의 미래! `<br>` HTML5 `</p>`

시작 태그                      빈 요소                      종료 태그

## 2. 작성 규칙

### ❖ HTML 속성과 값

- 시작 태그는 필요에 따라 정해진 속성을 가질 수 있으며, 사용할 수 있는 속성은 태그마다 다를 수 있다.
- 시작 태그 내에 여러 개의 속성을 선언할 수도 있는데, 이 경우 속성과 속성은 공백으로 구분하여 지정해야 한다.
- 속성에는 값을 가지지 않는 논리 속성도 있으며 논리 속성의 경우 XHTML1.0에서는 값을 생략할 수 없기 때문에 속성명을 값으로 지정해야 한다.

```
<input type="text" id="user" required>
```

속성                  속성                  논리 속성

## 2. 작성 규칙

### ❖ HTML 요소(Element)

- 시작 태그와 종료 태그 모두를 포함하여 '요소(element)'라고 한다.
- 요소는 다음과 같은 항목으로 구성되어 있다.

`<p> HTML과 XHTML의 미래 HTML5 </p>`

요소(element)

### 3. Rule Set(문법)

#### ❖ 하위 호환을 위한 HTML5

- HTML5는 HTML4.01이나 XHTML1.0 문법을 모두 허용하기 때문에 기존에 사용하던 마크업 문법을 그대로 사용할 수 있다.
- 가령 HTML4.01에서 종료 태그의 생략을 허용했던 방식이나 XHTML1.0에서 빈 요소 선언 시 <요소명 /> 형식으로 기술했던 방식 모두를 허용한다.
- 이는 하위 호환성을 위한 정책으로, 과거 HTML4.01로 제작된 문서가 문법적인 느슨함으로 인해 발생했던 문제가 되풀이될 수 있기 때문에 좀 더 엄격한 규칙을 정하고 마크업 문서를 작성하는 것이 바람직할 것이다.

### 3. Rule Set(문법)

#### ❖ 종료 태그의 생략

- HTML5는 종료 태그를 생략할 수도 있다. 그러나 모든 요소가 종료 태그를 생략해도 되는 것은 아니기 때문에 종료 태그가 생략할 수 있는 요소를 사전에 확인해야 한다.
- 그렇지만 HTML5에서 종료 태그를 생략하는 것이 가능하다고 하더라도 기존 XHTML1.0의 규칙처럼 시작과 종료 태그를 정확히 명시하여 정형식(Well-Formed) 구조로 마크업할 것을 권장한다.

<code>&lt;p&gt;&lt;img src="images/back.gif" alt="뒤로"&gt;</code>	○
<code>&lt;p&gt;&lt;img src="images/back.gif" alt="뒤로" /&gt;&lt;/p&gt;</code>	○

- HTML5는 시작 및 종료 태그는 물론 속성에 대문자 또는 소문자를 사용할 수도 있다. 그러나 기존 XHTML1.0 규칙처럼 소문자를 사용할 것을 권장한다.

<code>&lt;P&gt;&lt;img SRC="images/back.gif" ALT="뒤로"&gt;&lt;/P&gt;</code>	○
<code>&lt;p&gt;&lt;img src="images/back.gif" alt="뒤로"&gt;&lt;/p&gt;</code>	○

### 3. Rule Set(문법)

#### ❖ 빈요소(Empty Element)와 논리 속성

- HTML에서 <meta>, <link>, <img>, <br>, <input> 등 종료 태그를 가지고 있지 않은 요소를 '빈 요소(empty element)'라고 하는데, 기존 HTML4.01에서는 <img> 형식으로, XHTML1.0에서는 <img /> 형식으로 선언해야 하며, HTML5에서는 두 가지 방식 모두 허용하고 있다.

<code>&lt;img src="images/back.gif" alt="뒤로"&gt;</code>	○
<code>&lt;img src="images/back.gif" alt="뒤로" /&gt;</code>	○

- 논리 속성의 경우 속성값을 지정 또는 생략할 수 있다.

<code>&lt;select multiple&gt;&lt;/select&gt;</code>	○
<code>&lt;select multiple="multiple"&gt;&lt;/select&gt;</code>	○



### 3. Rule Set(문법)

#### ❖ 속성값 및 태그의 중첩

- 속성 값은 인용 부호를 생략하거나 홑 따옴표와 겹 따옴표 등으로 구분할 수도 있다.

<code>&lt;img src=images/back.gif&gt;</code>	○
<code>&lt;img src='images/back.gif'&gt;</code>	○
<code>&lt;img src="images/back.gif"&gt;</code>	○

- 시작 태그와 종료 태그의 중첩에 문제가 발생하지 않도록 해야 한다.

<code>&lt;p&gt;&lt;strong&gt;HTML5&lt;/p&gt;&lt;/strong&gt;</code>	×
<code>&lt;p&gt;&lt;strong&gt; HTML5&lt;/strong&gt;&lt;/p&gt;</code>	○

### 3. Rule Set(문법)

#### ❖ 속성의 중복 선언 및 특수문자

- 시작 태그에 동일한 속성을 중복하여 선언할 수 없다.

<code>&lt;p style="color:red;" style="background:yellow"&gt;HTML5&lt;/p&gt;</code>	×
<code>&lt;p style="color:red; background:yellow"&gt;HTML5&lt;/p&gt;</code>	○

- "<", ">", "&"와 같은 특수 문자의 경우, Characters Entity Name이나 Characters Entity Code로 변환하여 마크업해야 한다.

<code>&lt;p&gt;웹 표준 &amp; 웹 접근성&lt;/p&gt;</code>	×
<code>&lt;p&gt;웹 표준 &amp;amp; 웹 접근성&lt;/p&gt;</code>	○
<code>&lt;p&gt;웹 표준 &amp;#38; 웹 접근성&lt;/p&gt;</code>	○

### 3. Rule Set(문법)

#### ❖ HTML5의 Doctype

- HTML4.01이나 XHTML1.0에서는 HTML 문서의 첫 줄에 문서 형 선언을 기술했으며 이러한 문서 형 선언은 해당 HTML 문서의 버전과 문서 타입을 명시했지만, HTML5에서는 문서의 버전 및 문서 타입이 생략된, 간소화된 형식을 가진다.  
그 이유는 기존 HTML 문서형 선언의 목적과 달리 모든 웹 브라우저에서 표준 모드 (Standards Mode)로 렌더링 될 수 있도록 하는 역할만을 담당하기 때문이다.

```
<!DOCTYPE html>
```

### 3. Rule Set(문법)

#### ❖ 간소화 된 인코딩 선언

- HTML5 문서에서 간소화된 문자 인코딩 지정 방법과 기존 HTML 방식의 비교.

<code>&lt;meta charset="utf-8"&gt;</code>	HTML5
<code>&lt;meta http-equiv="content-type" content="text/html; charset=utf-8"&gt;</code>	기존 HTML

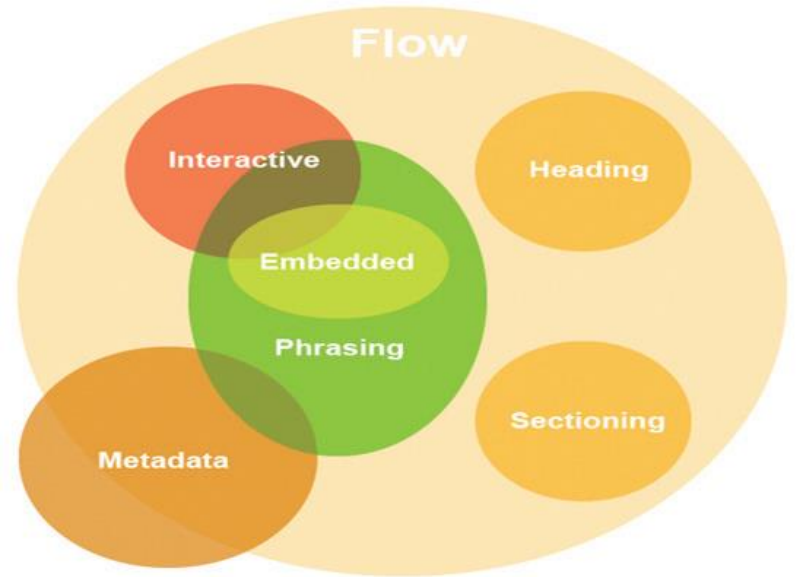
## 4. 콘텐츠 모델과 카테고리

### ❖ 콘텐츠 모델(Content Models)

- 명확한 정보 구조 설계 및 구성을 위해 카테고리를 정의하여 각 요소 별로 비슷한 성격을 가지고 있는 것끼리 그룹화하였는데, 이를 HTML5의 콘텐츠 모델이라고 한다.

### ❖ 카테고리(Category)

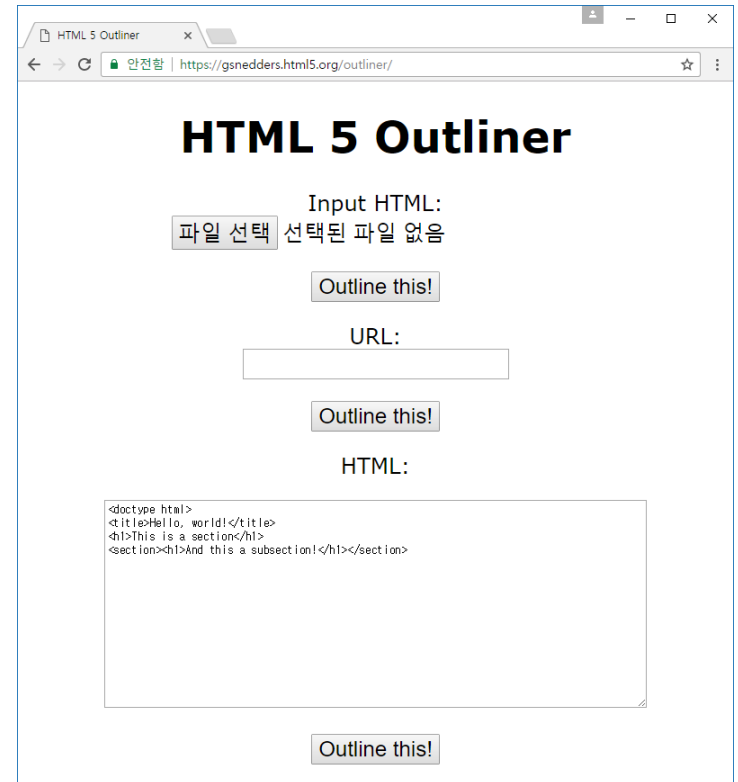
- HTML5의 카테고리(Category)에는 Sectioning Root, Metadata Content, Flow Content, Sectioning Content, Heading Content, Phrasing Content, Embedded Content, Interactive Content, Palpable Content, Script-supporting Elements, Transparent Content 등의 그룹이 있으며, 하나의 요소가 여러 그룹에 속해 있을 수도 있고, 그렇지 않을 수도 있다.



## 5. 아웃라인 알고리즘

### ❖ 아웃라인 알고리즘(Outline Algorithm)이란?

- HTML5에서는 정보 구조를 명확히 할 수 있도록 '아웃라인 알고리즘'(Outline Algorithm)이라는 개념이 도입되었다.
- 아웃라인 알고리즘은 웹 페이지의 정보 구조를 판별할 수 있는 개념으로, 책의 목차와 비슷하다.
- HTML 5에서 추가된 많은 요소들은 대부분 아웃라인 알고리즘과 관련이 있으며 그 중 직접적으로 아웃라인을 구성하는 요소에는 헤딩 콘텐츠, 섹셔닝 콘텐츠 그리고 섹셔닝 루트 요소 등이 있다.
- HTML5 문서의 아웃라인은 HTML5 Outliner(<http://gsnedders.html5.org/outliner>)에서 확인할 수 있다.



## 5. 아웃라인 알고리즘

### ❖ 아웃라인 알고리즘에 의한 제목의 계층구조 분석

- 앞서 살펴본 HTML5 Outliner 사이트에 국내 포털 웹사이트의 URL을 입력 후 분석을 한 결과는 다음과 같다.

HTML 5 Outliner

← → ↻ 안전함 | <https://gsnedders.html5.org/outliner/>

# HTML 5 Outliner

Input HTML:

파일 선택 선택된 파일 없음

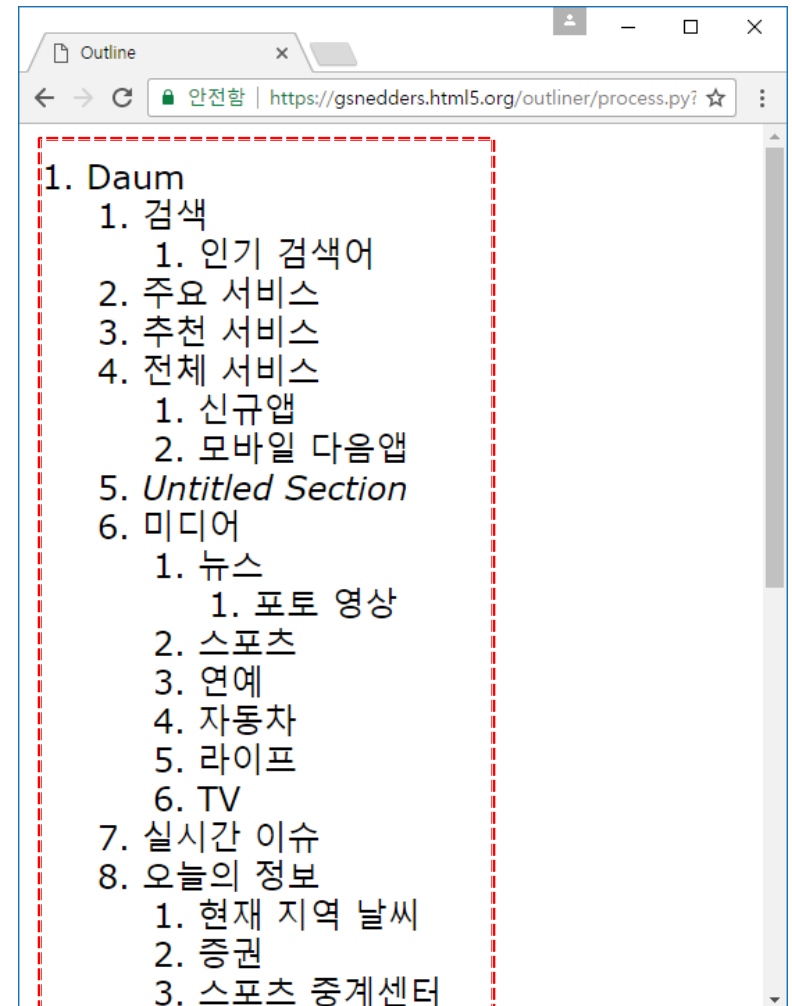
Outline this!

URL:

<http://www.daum.net>

Outline this!

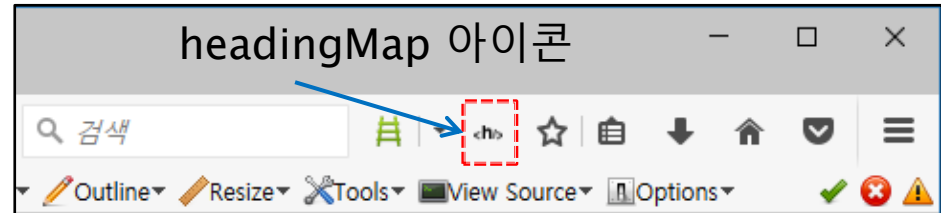
HTML:



## 5. 아웃라인 알고리즘

### ❖ headingMap을 활용한 아웃라인 분석

- 앞서 파이어폭스에 설치 한 headingMap을 활용하여 웹사이트의 아웃라인을 보다 쉽게 확인해 보자.
- 파이어폭스를 실행 한 후 아웃라인 분석을 위해 웹사이트로 접속한다.
- 상단에 headingMap 버튼을 클릭한다.
- HTML5의 Outline 구조를 분석해 보고 문제가 있는 제목 구조에 대한 해결책을 고민해 보자.





## 6. HTML 기본 요소

### ❖ <html> 요소

- 문서의 루트 요소로, 모든 HTML 요소는 루트 요소 내에 포함되어야 한다.

<html lang="ko"> ... </html>



문서의 기본 언어를 지정하기 위한 속성

언어명	국가 언어 코드
한국어	ko
영어	en
일본어	jp
중국어(간체)	zh-cn
중국어(번체)	zh-tw
프랑스어	fr
스페인어	es
포르투갈어	pt
러시아어	ru
독일어	de

## 6. HTML 기본 요소

### ❖ <head> 요소

- <head> 요소는 <html> 요소 내부의 첫 번째 요소로 사용되며, 해당 문서의 정보, 스타일 시트와 스크립트 등 메타 데이터의 집합을 나타낸다.

<head> ... </head>

```
<!DOCTYPE html>
<html lang="ko">
<head>
```

```
    <meta charset="utf-8"> .....
    <title>웹카페에서 웹 표준을</title>
    <link rel="stylesheet" href="default.css">
    <script src="default.js"></script> .....
```

메타데이터의 집합

```
</head>
```

## 6. HTML 기본 요소

### ❖ <meta> 요소

- 빈 요소(Empty Element)로 <head> 요소 안에 위치하며, 문서의 다양한 문서 정보를 나타낼 때 사용한다. 해당 요소로 지정한 정보들은 화면에 표시되지 않지만, 기기가 해당 정보를 이해하고 분석할 수 있도록 정보(데이터)를 제공한다.
- <meta> 요소에 사용할 수 있는 name 속성 값은 다음과 같다.
  - 문서의 요약 설명문 제공(description)
  - 문서의 키워드 지정(keyword)
  - 페이지 새로 고침(refresh)
  - 웹 문서 제작자(author)
  - 문서의 유효 기간 지정(expires)
  - 브라우저 문서 캐시 확인(pragma)
  - 검색 엔진 제어 여부(robot)
  - 뷰 포트 영역 조절(viewport)

<meta charset="utf-8">

↑  
문서의 인코딩 지정

## 6. HTML 기본 요소

### ❖ <link> 요소

- <link> 요소는 외부 문서를 연결하거나 다른 문서와의 관계를 지정할 수 있다.
- <link> 요소를 사용하여 스타일 시트를 HTML 문서에 연결하는 경우가 대표적인 사용 예라고 할 수 있다.

`<link rel="stylesheet" href="css/layout.css">`

↑  
연결하고자 하는 파일의  
종류가 스타일시트임을  
명시적으로 선언

↑  
href 속성을 사용하여 연결하고자 하는  
스타일시트 파일을 지정

### ❖ <style> 요소

- 특정 HTML 문서에만 CSS를 적용하고자 할 때 사용할 수 있는 요소로, HTML5에서는 scoped 속성이 새롭게 추가되었다. scoped 속성은 특정 문서 내 모든 콘텐츠가 아닌 특정 콘텐츠 블록에만 CSS를 적용할 수 있도록 한다.

`<style> ... </style>`

↑  
CSS 코드

## 6. HTML 기본 요소

### ❖ <script> 요소

- <script> 요소를 사용하면 HTML 문서에 자바스크립트 파일을 삽입할 수 있고, <script> 요소 안에 직접 자바스크립트 코드를 기술할 수도 있다.

```
<script src="file-name.js"></script>
```

← 자바스크립트 파일 삽입

```
<script>
```

```
    document.write('<h3>안녕하세요</h3>');
```

```
</script>
```

← 자바스크립트 코드  
직접 기술

### ❖ <body> 요소

- <body> 요소는 HTML 문서의 본문에 해당하는 모든 콘텐츠가 포함된다. <head>의 형제 요소로 위치 하며, 문서 내에서 단 한 번만 사용할 수 있다.

```
<body> ... </body>
```

## 6. HTML 기본 요소

### ❖ meta.html

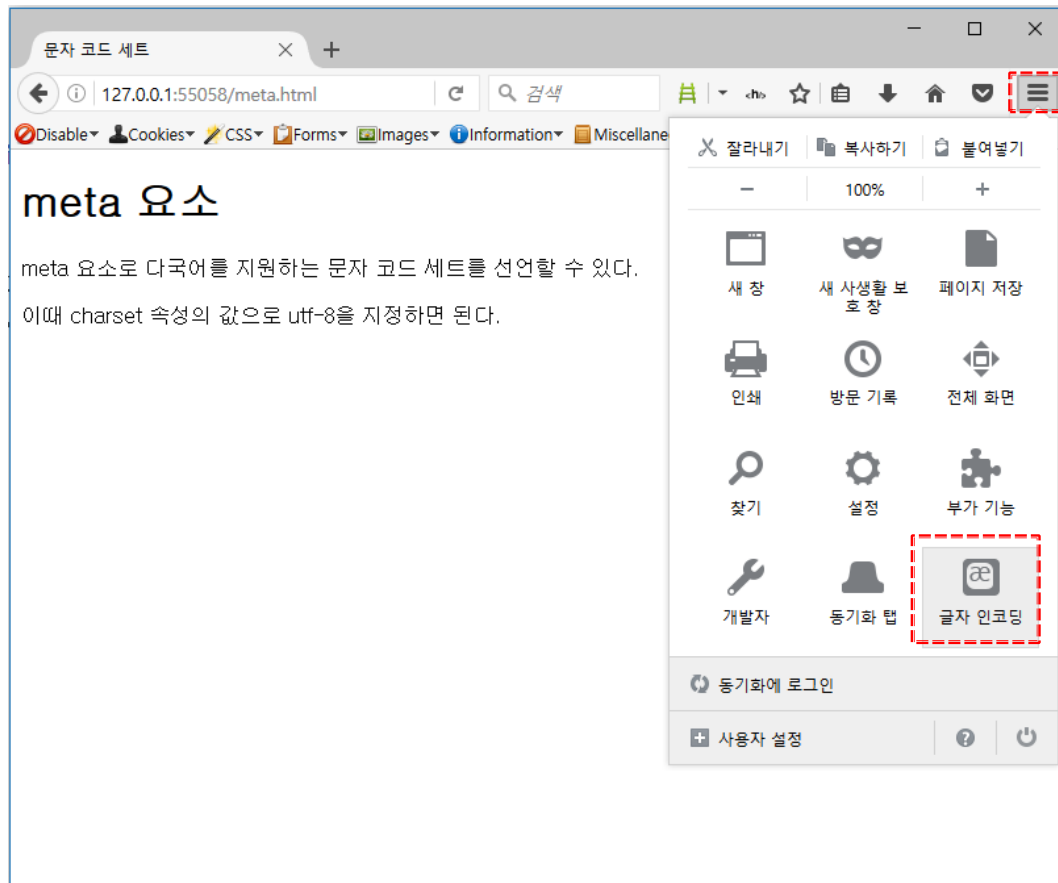
- 실습 폴더에 meta.html 파일을 생성하고 다음과 같이 작성한다.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>meta 요소</title>
  </head>
  <body>
    <h1>문자 코드 세트</h1>
    <p>meta 요소로 다국어를 지원하는 문자 코드 세트를 선언할 수 있다.</p>
    <p>이때 charset 속성의 값으로 utf-8을 지정하면 된다.</p>
  </body>
</html>
```

## 6. HTML 기본 요소

### ❖ meta.html

- 작성 한 meta.html 파일을 웹 브라우저로 실행해 보자.  
만약 인코딩을 바꾸고 싶다면 웹 브라우저에 글자 인코딩을 변경할 수 있다.

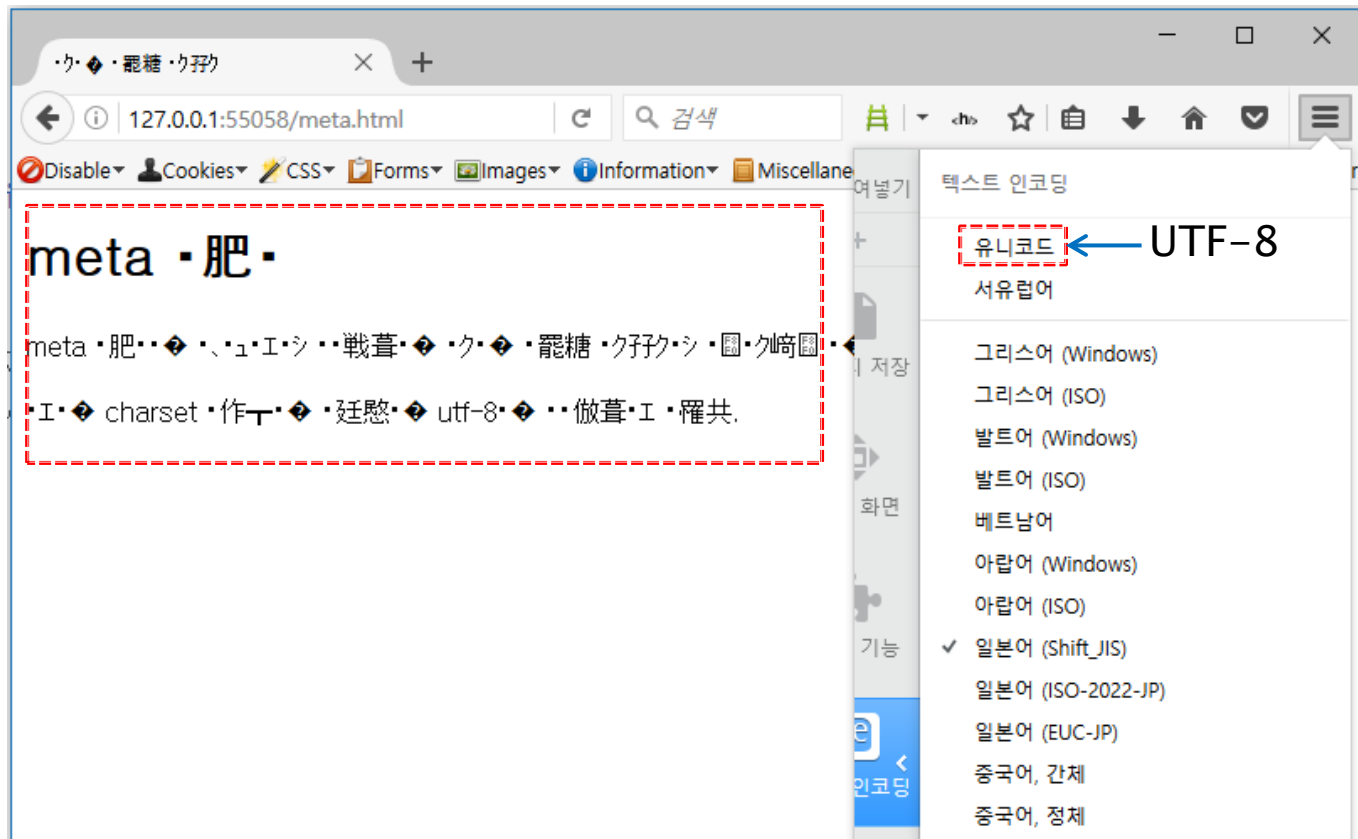


파이어폭스  
메뉴 -> 글자 인코딩

## 6. HTML 기본 요소

### ❖ meta.html

- 인코딩 미 선언시 웹 브라우저에서 글자가 깨질 경우 글자 인코딩의 값을 유니코드로 변경하여 해결 할 수 있다. 그러나 HTML 문서 작성 시 명시적으로 인코딩 선언을 하는 것이 가장 바람직한 해결책이다.





## 6. HTML 기본 요소

### ❖ title.html

- 실습 폴더에 title.html 파일을 생성하고 다음과 같이 작성한다.

```
<!DOCTYPE html>
```

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

```
<head>
```

```
<meta charset="utf-8">
```

```
<title>웹 문서의 제목</title>
```

```
</head>
```

```
<body>
```

```
<h1>웹 문서의 제목</h1>
```

```
<p>웹 문서의 제목을 선언하려고 할 때 사용하는 요소로, 문서마다 title의  
내용은 유니크(Unique)하게 제공해야 한다.</p>
```

```
<p>따라서 title 요소는 문서의 내용을 이해하고 구별할 수 있도록 중요한  
핵심 정보를 적어주는 것이 필요하다.</p>
```

```
<p>title 내용에 장식을 위한 특수문자 등을 삽입하는 것은 웹접근성 관점에서  
바람직하지 않으므로 사용을 지양하는 것이 필요하다. </p>
```

```
</body>
```

```
</html>
```

## 6. HTML 기본 요소

### ❖ title.html

- 작성 한 title.html 파일을 웹 브라우저로 실행해 보자.

title 요소는 브라우저 탭에 표시된다.



## 6. HTML 기본 요소

### ❖ link-style.html

- 실습 폴더에 link-style.html 파일을 생성하고 다음과 같이 작성한다.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>CSS 삽입하기</title>
    <link rel="stylesheet" href="css/style.css">
    <style>
      body { color : #fff ; }
    </style>
  </head>
  <body>
    <h1>CSS 스타일 적용</h1>
    <p>&lt;link&gt; 요소로 CSS 파일을 HTML 문서에 삽입할 수 있다.</p>
    <p>&lt;style&gt; 요소로 HTML 문서 내에 직접 CSS 코드를 작성할 수 있다.
  </p>
</body>
</html>
```

## 6. HTML 기본 요소

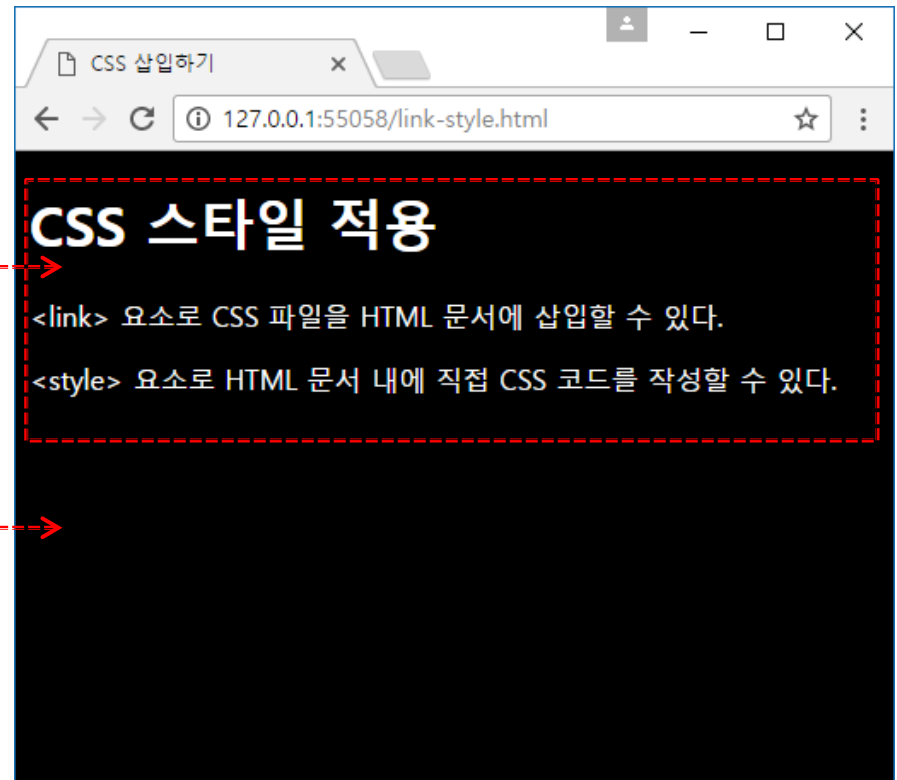
### ❖ style.css

- 실습 폴더 하위에 있는 css 폴더에 style.css 파일을 생성하고 다음과 같이 작성한다.

```
@charset "utf-8";  
body {  
    background-color : #000 ;  
}
```

<style> 요소에 작성한  
CSS 코드에 의해 글자 색상이  
흰색으로 변경

<link> 요소로 삽입한 style.css에  
의해 본문의 배경 색상이  
검정색으로 변경



## 6. HTML 기본 요소

### ❖ script.html

- 실습 폴더에 script.html 파일을 생성하고 다음과 같이 작성한다.

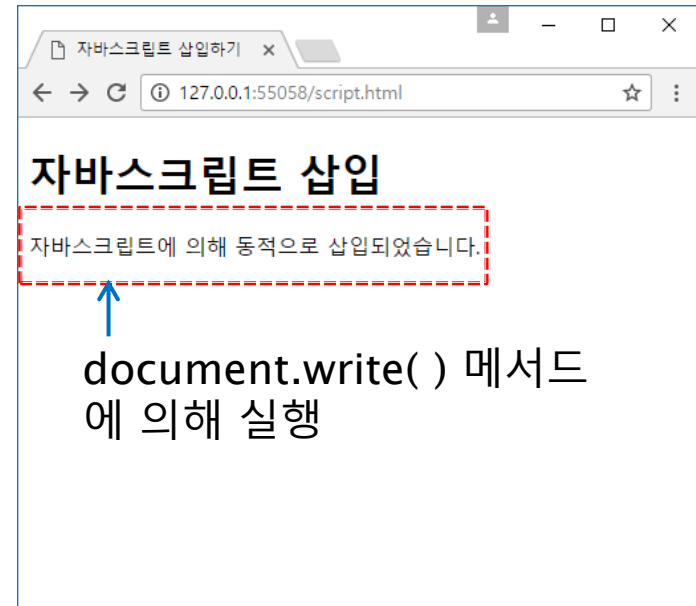
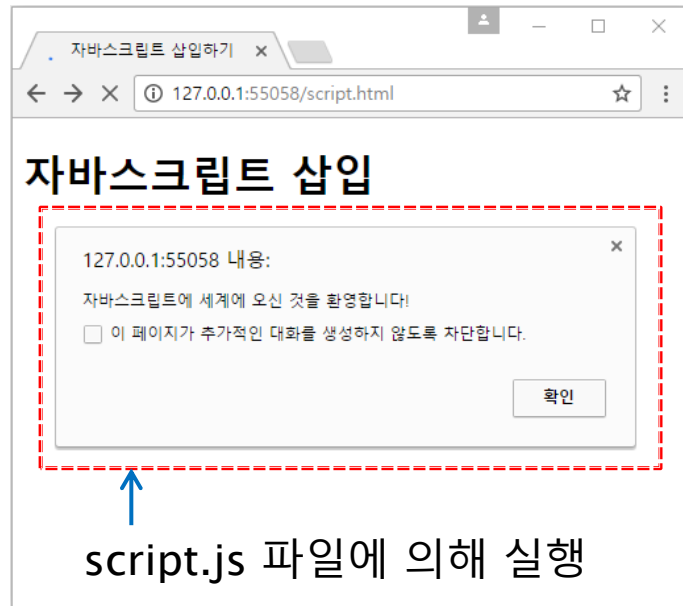
```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>자바스크립트 삽입하기</title>
  </head>
  <body>
    <h1>자바스크립트 삽입</h1>
    <script src="js/script.js"></script>
    <script>
      document.write('<p>자바스크립트에 의해 동적으로 삽입되었습니다.</p>');
    </script>
  </body>
</html>
```

## 6. HTML 기본 요소

### ❖ script.js

- 실습 폴더 하위에 있는 js 폴더에 script.css 파일을 생성하고 다음과 같이 작성한다.

```
function msg(){  
var str = "자바스크립트에 세계에 오신 것을 환영합니다!" ;  
alert(str) ;  
}  
window.onload = msg ;
```



## 7. 파일 경로(Path)

### ❖ 파일의 경로

- HTML 문서에 CSS나 자바스크립트 또는 이미지 등의 파일을 삽입하고자 할 때 경로에 대한 이해가 필요하다.
- 경로는 절대 경로와 상대 경로로 구분할 수 있다.
- 절대 경로는 우리가 웹 브라우저를 이용하여 특정 사이트로 접속 시 입력하는 URL을 떠올리면 된다. 이대 URL의 전체 경로를 절대 경로라고 한다.  
다만 웹 사이트 접속 시 보여주는 기본 HTML 문서의 경우 파일명을 생략할 수 있다.  
대체적으로 이 파일은 index.html을 사용하지만 경우에 따라 서버에서 최상위 계정으로 접속 시 보여줄 파일을 따로 지정할 수도 있다.

<http://www.webcafe2010.com/>

<http://www.webcafe2010.com/index.html>

절대 경로

↑  
생략 가능

## 7. 파일 경로(Path)

### ❖ 파일의 경로

- 상대 경로는 같은 사이트 안에 있는 문서나 이미지 등을 현재 작업 중인 파일을 기준으로 사용하는 방법이다.
- 가령 현재 작업 중인 HTML 문서에서 삽입하고자 하는 CSS 파일이나 자바스크립트 파일이 같은 폴더 내에 있는지 하위 폴더에 있는지 또는 상위 폴더에 있는지에 따라 경로를 지정하는 것이다.

### ❖ HTML 문서에 삽입하려는 파일이 같은 폴더에 있는 경우

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



파일명만 명시

### ❖ HTML 문서에 삽입하려는 파일이 상위 폴더에 있는 경우

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

../파일명 명시

### ❖ HTML 문서에 삽입하려는 파일이 하위 폴더에 있는 경우

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

하위 폴더명/파일명 명시



# HTML5 요소

# 1. Global 속성

## ❖ Global 속성(Attribute)

- Global 속성(attributes)은 모든 HTML 요소에 사용할 수 있는 공통 속성이다. 몇몇 요소에는 적용되지 않을 수 있지만, 원칙적으로 Global 속성은 모든 요소에 허용된다.
- 다음은 대표적인 Global 속성을 정리 한 표이다.

속성 명	의미
id	요소에 유일한 식별자(고유의 이름)를 부여할 때 사용
class	요소에 이름(중복사용 가능)을 부여할 때 사용
style	요소에 인라인 스타일을 지정할 때 사용
title	요소에 툴팁으로 대체 설명을 제공할 때 사용
tabindex	요소에 키보드 포커스를 받을 수 있도록 하거나 키보드 포커스를 받지 못하도록 지정할 때 사용
lang	요소에 언어를 지정할 때 사용

# 1. Global 속성

## ❖ id

- id 속성에 값은 문서 내에서 유일한 이름을 가져야 한다. 개인을 식별하기 위한 고유의 번호인 주민등록 번호 같은 개념이라 볼 수 있다.

```
<div id="gnb"> ... </div>
```

## ❖ class

- 사람마다 이름이 있듯 특정 요소에 이름을 지정하여 의미를 명확하게 정의할 수 있다. 사람은 다르지만 동명이인이 있는 것처럼 class 속성 값은 문서 내에서 여러 번 사용하는 것이 가능하다.
- 사람의 경우 이름 이외에 닉네임이나 회사에서의 직위 등 여러 호칭을 가질 수 있는 것처럼 하나의 요소에 class 속성도 여러 개 지정하는 것이 가능하다.
- 하나의 요소에 여러 개의 class 속성 값을 지정하고자 할 경우 class 속성 값과 값 사이에 공백으로 구분하면 된다.

```
<div class="gnb"> ...  
</div>
```

↑ class 속성 값을 한번 만 지정할 경우

```
<div class="gnb memu"> ... </div>
```

↑ 여러 개의 class 속성 값을 지정할 경우

# 1. Global 속성

## ❖ style

- HTML 요소에 직접 CSS 코드를 적용할 때 style 속성을 사용할 수 있다.  
그러나 구조와 표현의 분리라는 관점에서는 권장하고 싶지 않은 방법이다.  
다만 자바스크립트로 CSS 스타일을 지정할 경우 HTML 요소에 style 속성 형식으로 삽입되기 때문에 이를 알아둘 필요가 있다.

```
<div style="color : #fff ; background-color : #000 ;"> ...  
</div>
```




↑ CSS 코드

## ❖ title

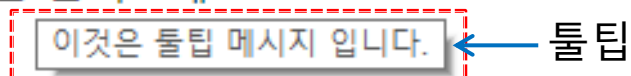
- HTML 요소에 마우스를 올렸을 때 툴팁(Tooltip)으로 설명을 제공할 수 있다.  
모든 요소에 사용이 가능하다. 다만 해당 툴팁의 기본 스타일을 변경할 수는 없다.

```
<div title="툴팁 메시지..."> ... </div>
```



↑ 해당 요소에 마우스 오버 시 보여질 내용

마우스를 올려보세요!



이것은 툴팁 메시지입니다. ← 툴팁

# 1. Global 속성

## ❖ tabindex

- HTML 요소 중 키보드 포커스를 받을 수 있는 요소는 <a> 요소나 <form> 관련 요소가 있다. 해당 요소 이외에는 기본적으로 키보드 포커스를 받을 수 없다.
- 만약 기본적으로 키보드 포커스를 받을 수 없는 요소가 키보드 포커스를 받도록 하려면 tabindex="0"을 지정하면 된다.
- 반대로 키보드 포커스를 받지 못하도록 하려면 tabindex="-1"을 지정하면 된다.

`<div tabindex="0"> ...`  
`</div>` ↑ 키보드 포커스를 받도록 지정할 경우

`<a href="#" tabindex="-1"> ...`  
`</a>` ↑ 키보드 포커스를 받지 못하도록 지정할 경우

## ❖ lang

- HTML 요소의 콘텐츠가 특정 언어일 때 이를 명시적으로 알려줄 수 있는 lang 속성이 있다. lang 속성은 앞서 <html> 요소에 사용했던 방식과 동일하게 사용하면 된다.

`<p lang="js"> こんにちは。</a>`

## 2. 제목 및 단락

### ❖ <h1> ~ <h6> 요소

- <h1>~<h6> 요소는 제목을 의미한다. <h1> 요소는 대 제목, <h2> 요소는 중 제목, <h3> 요소는 소 제목 등을 의미하며, 'h' 뒤의 숫자가 커질수록 제목의 레벨은 낮아진다. 가장 낮은 제목 레벨은 <h6> 요소이다.

<h1> ... </h1>

<h2> ... </h2>

<h3> ... </h3>

<h4> ... </h4>

<h5> ... </h5>

<h6> ... </h6>

### ❖ <p> 요소

- <p> 요소는 단락을 정의할 때 사용할 수 있다. 해당 요소 안에는 텍스트를 비롯하여, <a>, <em>, <img> 요소 등을 포함할 수 있다.

<p> ... </p>

## 2. 제목 및 단락

### ❖ heading.html

- 실습 폴더에 heading.html 파일을 생성하고 다음과 같이 작성한다.

```
<!DOCTYPE html>
```

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

```
<head>
```

```
<meta charset="utf-8">
```

```
<title>제목 요소</title>
```

```
</head>
```

```
<body>
```

```
<h1>제목 1 단계</h1>
```

```
<h2>제목 2 단계</h2>
```

```
<h3>제목 3 단계</h3>
```

```
<h4>제목 4 단계</h4>
```

```
<h5>제목 5 단계</h5>
```

```
<h6>제목 6 단계</h6>
```

```
<p>제목 요소는 1~6까지의 숫자로 구분할 수 있으며 숫자가 커질 수록 제목의  
단계가 낮아진다.</p>
```

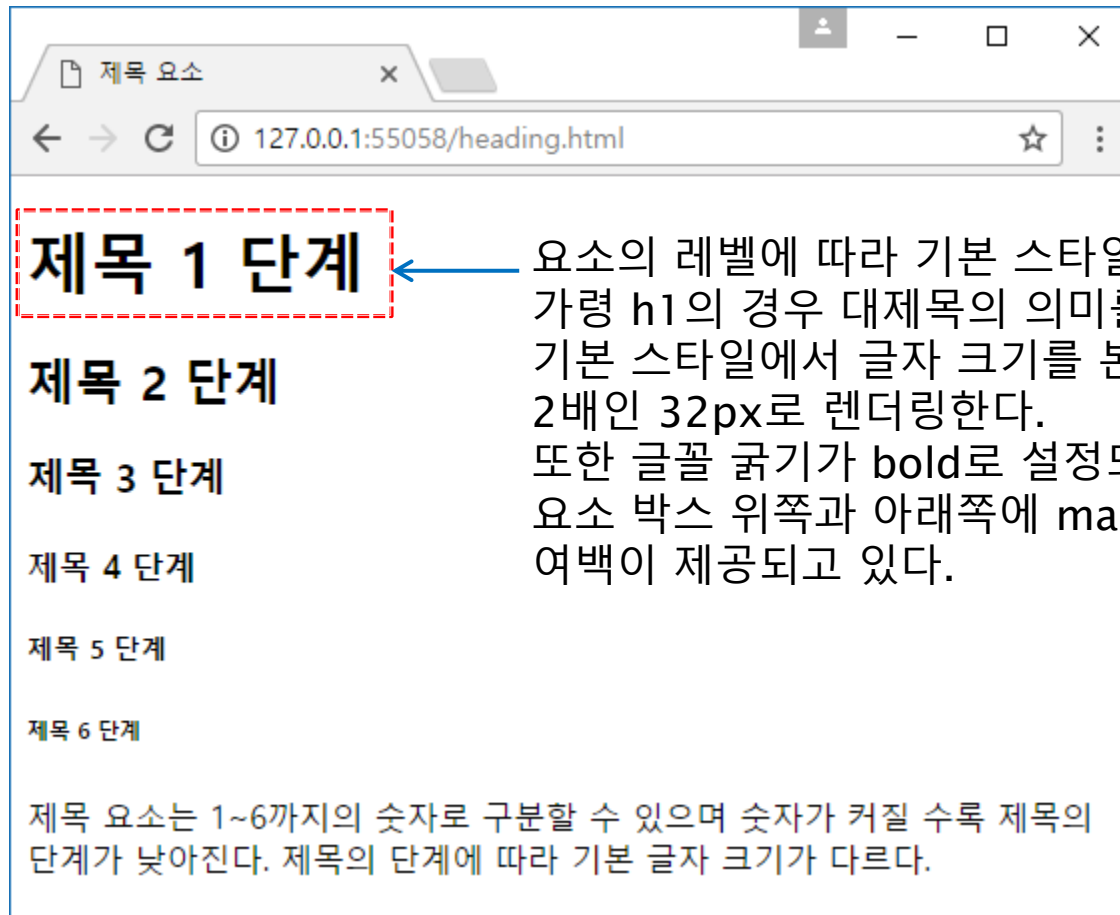
```
</body>
```

```
</html>
```

## 2. 제목 및 단락

### ❖ heading.html

- 작성 한 heading.html 파일을 웹 브라우저로 실행해 보자.

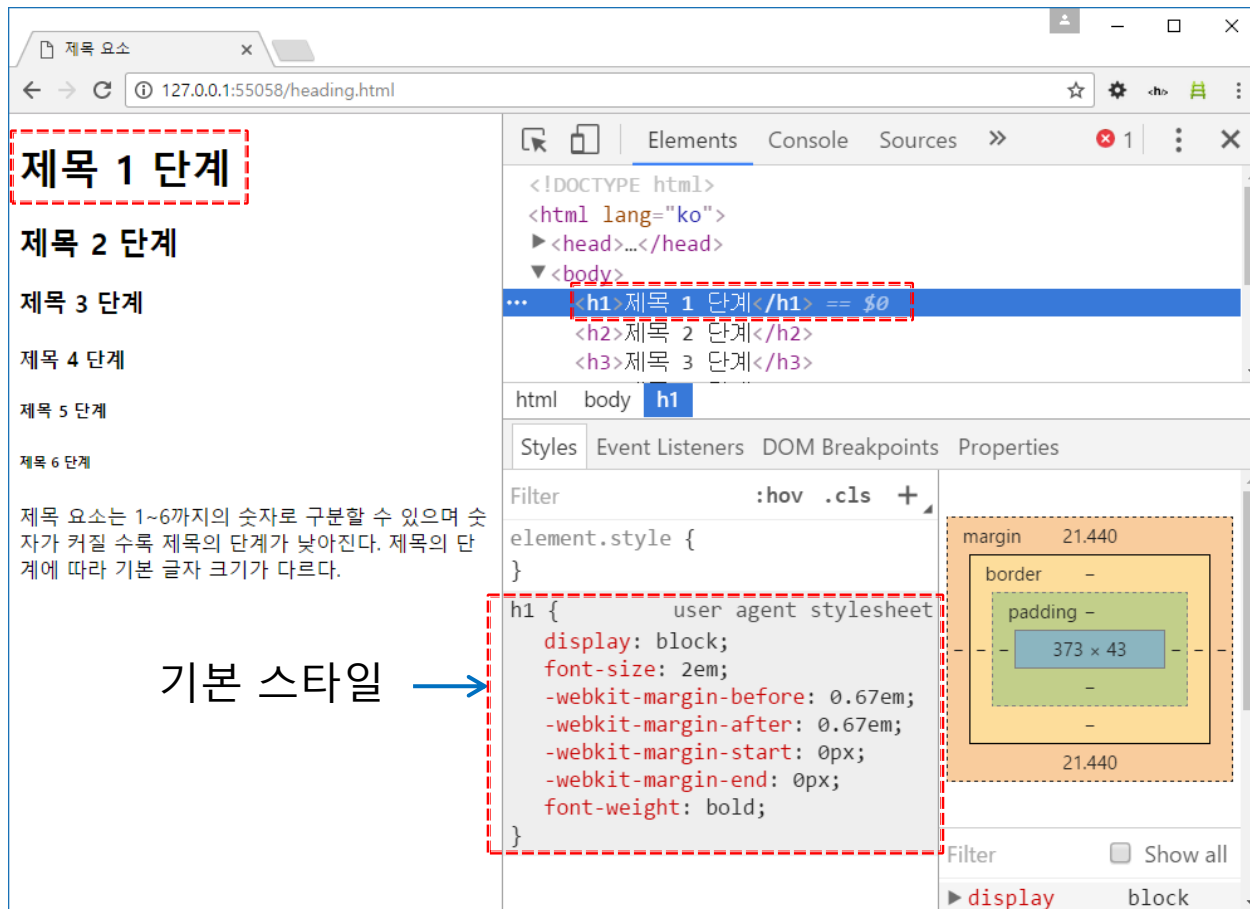




## 2. 제목 및 단락

### ❖ 개발자 도구의 활용

- 웹 브라우저에서 제공하는 ‘개발자 도구’를 통해 요소에 지정된 기본 스타일을 확인할 수 있다.



## 2. 제목 및 단락

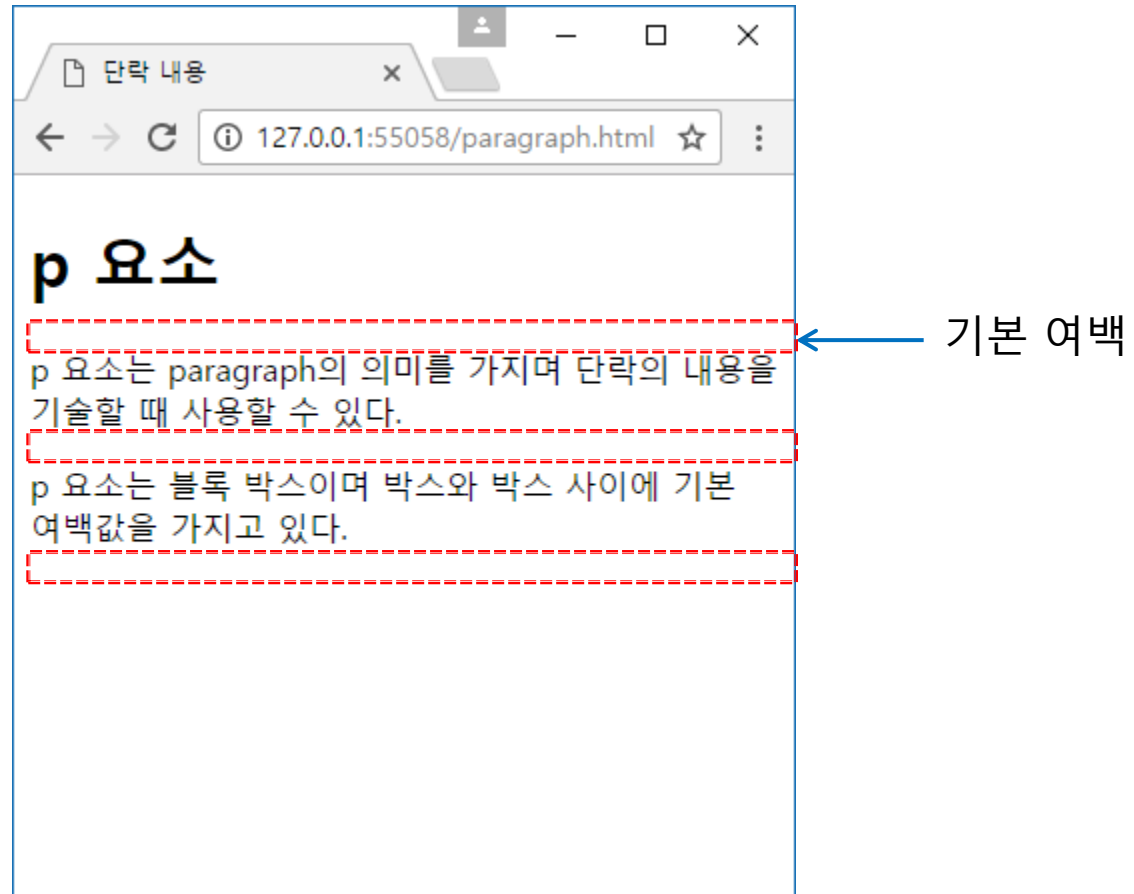
❖ paragraph.html

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>본문 단락 요소</title>
  </head>
  <body>
    <h1>단락 제목</h1>
    <p>
      p 요소는 paragraph의 의미를 가지며 단락의 내용을 기술할 때 사용할 수 있다.
    </p>
    <p>
      p 요소는 블록 박스이며 박스와 박스 사이에 기본 여백값을 가지고 있다.
    </p>
  </body>
</html>
```

## 2. 제목 및 단락

### ❖ paragraph.html

- 작성 한 paragraph.html 파일을 웹 브라우저로 실행해 보자.



### 3. 구조를 나타내는 요소

#### ❖ <section> 요소

- <section> 요소는 일반적으로 콘텐츠를 장 이나 절 등으로 구분하고자 할 경우에 사용한다. 그리고 장 이나 절 등으로 구분되기 때문에 <section> 요소 안에는 제목을 제공해야 한다.
- 제목은 콘텐츠 블록의 의미를 부여하기 위한 중요한 역할을 하기때문에 꼭 <section> 요소가 아닌 새로운 콘텐츠 블록이 등장할 때 적절한 제목을 부여하는 것을 잊지 말자.

<section> ... </section>

#### ❖ <article> 요소

- <article> 요소는 독립된 하나의 콘텐츠를 의미한다. 그 예로는 블로그의 포스트나 뉴스 본문 등을 들 수 있다. <section> 요소와 성격이 비슷하여 경우에 따라 <section> 요소와 비슷한 용도로 사용하기도 한다.

<article> ... </article>

### 3. 구조를 나타내는 요소

#### ❖ <nav> 요소

- <nav> 요소는 주요 내비게이션을 마크업할 때 사용한다.  
내비게이션은 메인 메뉴나 서브 메뉴 등을 의미하며, 서로 다른 페이지로 이동할 수 있는 링크 항목을 포함한다.
- 그러나 모든 목록 형태의 하이퍼링크 콘텐츠에 <nav> 요소를 사용하는 것은 바람직하지 않다. 또한 한 문서 내에서 너무 많은 수의 <nav> 요소를 사용하는 것은 권장하지 않는다. 최대 3개를 넘지 않도록 주의하자.

<nav> ... </nav>

#### ❖ <main> 요소

- <main> 본문 영역의 주요 콘텐츠 블록을 그룹화할 때 사용한다. 주요 콘텐츠가 아닌 사이드 바나 검색 등의 콘텐츠 영역을 <main> 요소로 그룹화해서는 안 된다. 또한 <main> 요소는 <body> 요소 내에서 한 번만 사용할 수 있다.

<main> ... </main>

### 3. 구조를 나타내는 요소

#### ❖ <header>, <footer> 요소

- <header> 요소와 <footer> 요소는 페이지 헤더 또는 푸터나 섹션 헤더 또는 푸터로 사용할 수 있으며, 페이지 헤더의 경우 사이트 제목, 로고, 검색 영역 등을 콘텐츠로 포함할 수 있다. 페이지 푸터는 사이트의 연락처 정보 및 저작권 등을 포함할 수 있다.

<header> ... </header> <footer> ... </footer>

#### ❖ <aside> 요소

- <aside> 요소는 메인 콘텐츠와 관련이 적은 콘텐츠를 마크업할 때 사용할 수 있다. 예를 들어 사이드 바, 광고 등의 콘텐츠 등을 들 수 있다.

<aside> ... </aside>

#### ❖ <address> 요소

- <address> 요소는 주소 등 연락처 정보를 마크업할 때 사용한다. HTML5에서는 <address> 요소의 사용을 명확하게 제시하고 있는데, 해당 요소는 페이지 또는 포스트의 관리자 및 저자의 연락처 정보를 의미합니다. 주의할 점은 모든 주소 콘텐츠를 <address> 요소로 마크업해서는 안 된다.

<address> ... </address>

### 3. 구조를 나타내는 요소

#### ❖ structure.html

- 실습 폴더에 structure.html 파일을 생성하고 다음과 같이 작성한다.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>HTML5 구조 관련 요소</title>
</head>
<body>
  <header class="heaer">
    <h1>브랜드</h1>
    <p>...</p>
  </header>
  <nav class="navigation">
    <h2>메인 메뉴</h2>
    ...
  </nav>
  .....
```

### 3. 구조를 나타내는 요소

❖ structure.html

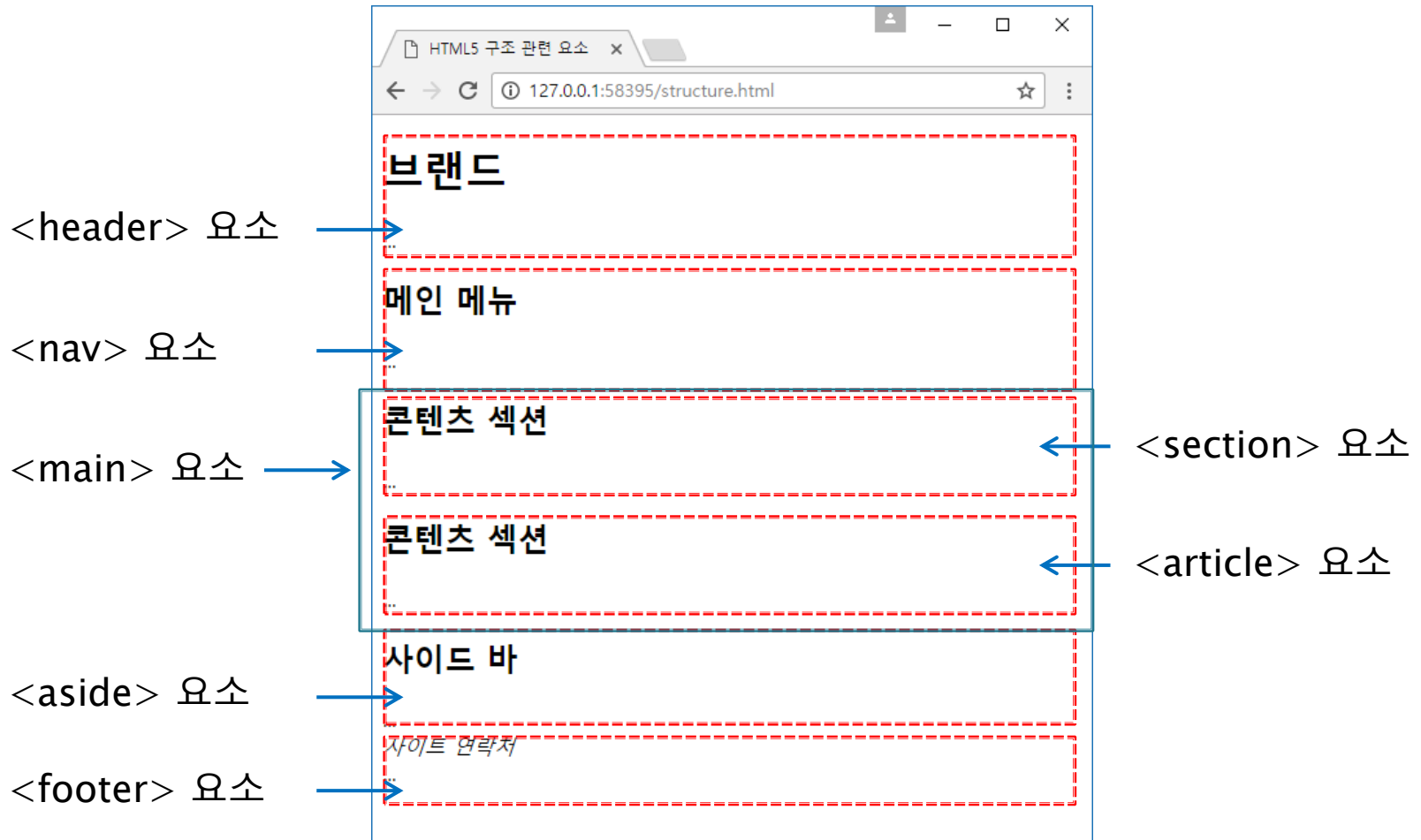
```
<main class="main-content">
  <section class="content-section">
    <h2>콘텐츠 섹션</h2>
    ...
  </section>
  <article class="content-article">
    <h2>콘텐츠 섹션</h2>
    ...
  </article>
</main>
<aside class="sidebar">
  <h2>사이드 바</h2>
  ...
</aside>
<footer class="footer">
  <address>사이트 연락처</address>
  ...
</footer>
</body>
</html>
```



### 3. 구조를 나타내는 요소

#### ❖ structure.html

- 작성 한 structure.html 파일을 웹 브라우저로 실행해 보자.



## 4. 독립적인 의미를 가지는 요소

### ❖ <div> 요소

- 콘텐츠 블록의 시맨틱 한 의미를 가지고 있지는 않지만, 디자인이나 개발 이슈로 인해 콘텐츠 블록을 그룹화 하고자 할 때 사용한다.

<div> ... </div>

### ❖ <span> 요소

- <div> 요소처럼 의미를 가지지 않지만 특정 이슈로 인해 그룹화 시 사용한다.  
단 <div> 요소가 콘텐츠 블록을 그룹화 한다면 <span> 요소는 인라인 콘텐츠를 그룹화할 때 사용한다.

<span> ... </span>

## 5. 목록 관련 요소

### ❖ <ul> 요소

- <ul> 요소는 비순서형 목록을 의미한다. <ul> 요소는 <li> 요소만을 자식 요소로 사용할 수 있다. <ul> 요소는 목록 항목의 순서를 변경하더라도 의미가 변하지 않는 콘텐츠를 마크업할 때 사용해야 한다.

```
<ul>  
  <li> ... </li>  
</ul>
```

### ❖ <ol> 요소

- <ol> 요소는 같은 문법을 가지며 순서형 목록을 마크업할 때 사용한다. 주로 순위를 나타내고자 할 때와 같이 순서가 중요한 의미를 가지는 콘텐츠에 적합하다.

```
<ol>  
  <li> ... </li>  
</ol>
```

### ❖ <li> 요소

- <ul>과 <ol> 요소의 자식 요소로 목록 항목(아이템)을 의미한다.

## 5. 목록 관련 요소

### ❖ ul-ol.html

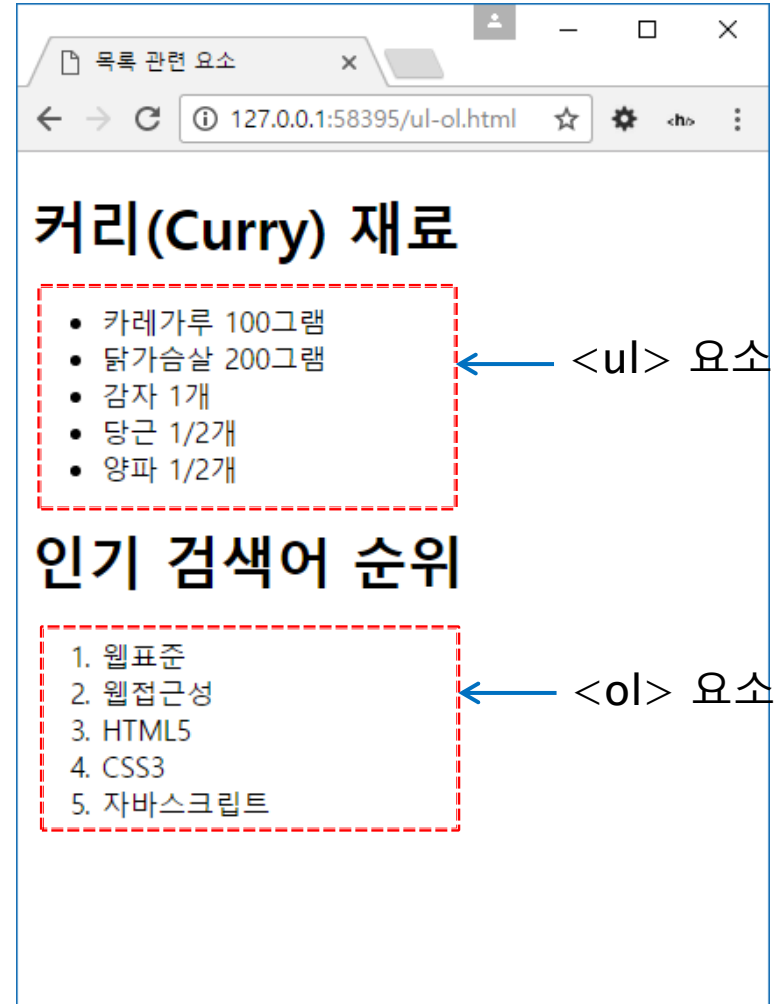
- 실습 폴더에 ul-ol.html 파일을 생성하고 다음과 같이 작성한다.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>목록 관련 요소</title>
</head>
<body>
  <div class="unordered-list">
    <h1>커리(Curry) 재료</h1>
    <ul>
      <li>카레가루 100그램</li>
      <li>닭가슴살 200그램</li>
      <li>감자 1개</li>
      <li>당근 1/2개</li>
      <li>양파 1/2개</li>
    </ul>
  </div>
  .....
```

## 5. 목록 관련 요소

### ❖ ul-ol.html

```
<div class="ordered-list">
  <h1>인기 검색어 순위</h1>
  <ol>
    <li>웹표준</li>
    <li>웹접근성</li>
    <li>HTML5</li>
    <li>CSS3</li>
    <li>자바스크립트</li>
  </ol>
</div>
</body>
</html>
```



## 5. 목록 관련 요소

### ❖ <dl> 요소

- 정의형 목록을 마크업 하고자 할 때 사용한다. <dl> 요소의 자식 요소로 <dt>와 <dd> 요소만 올 수 있다.
- HTML5에서는 기존보다 정의형 목록의 의미가 확장되어 이름과 값 형식을 가지는 유형의 콘텐츠를 마크업 할 때도 사용할 수 있다.

```
<dl>  
  <dt> ... </dt>  
  <dd> ... </dd>  
</dl>
```

### ❖ <dt> 요소

- 정의형 목록에서 용어 제목을 마크업 할 때 사용한다.

### ❖ <dd> 요소

- 정의형 목록에서 용어 설명 내용을 마크업 할 때 사용한다.

## 5. 목록 관련 요소

### ❖ dl.html

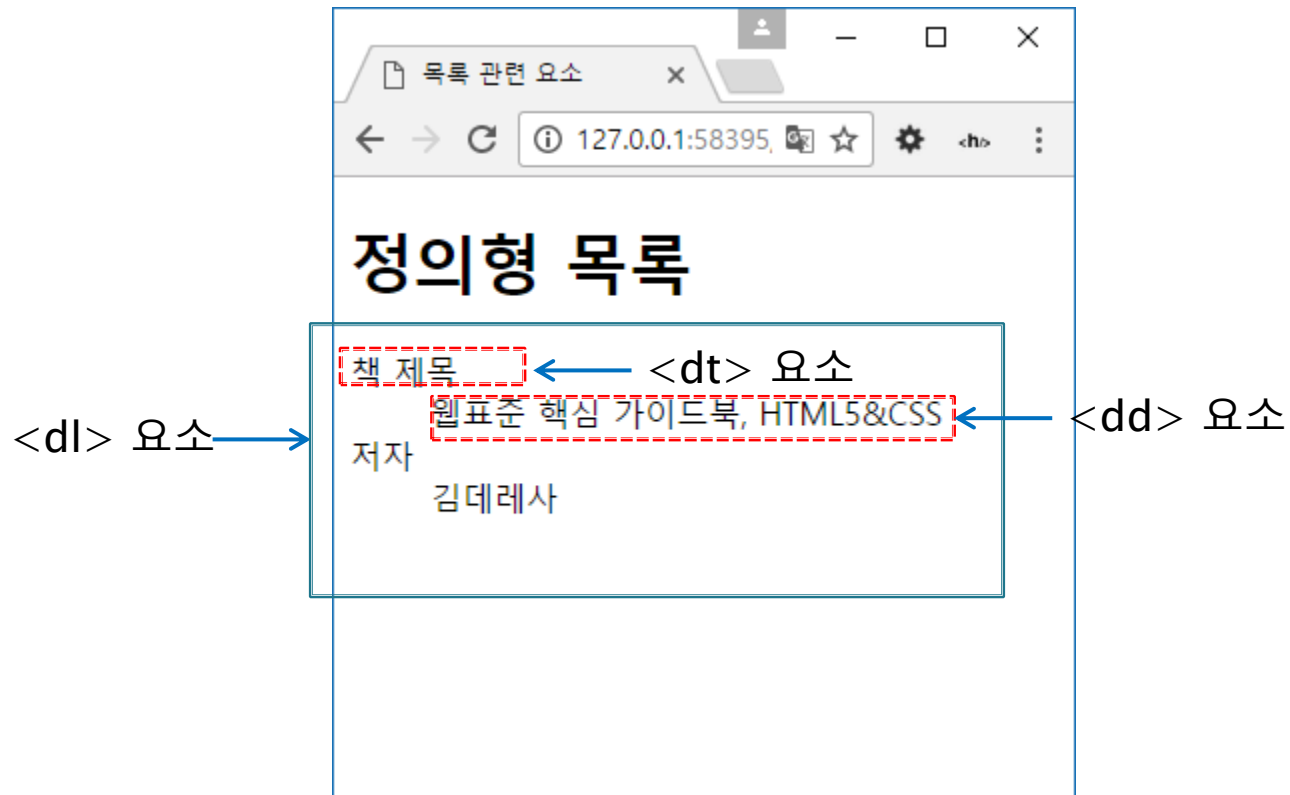
- 실습 폴더에 dl.html 파일을 생성하고 다음과 같이 작성한다.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>목록 관련 요소</title>
</head>
<body>
  <div class="description-list">
    <h1>정의형 목록</h1>
    <dl>
      <dt>용어 제목</dt>
      <dd>
        용어에 대한 설명내용을 지정합니다. dd 요소 안에는 div 요소나 p 요소
        같은 블록 요소도 포함이 가능합니다. 또한 이미지도 삽입할 수 있습니다.
      </dd>
    </dl>
  </div>
</body>
</html>
```

## 5. 목록 관련 요소

### ❖ dl.html

- 작성 한 dl.html 파일을 웹 브라우저로 실행해 보자.





## 6. 이미지 및 하이퍼 링크

### ❖ <img> 요소

- <img> 요소는 빈 요소(Empty Element)로 이미지 콘텐츠를 삽입할 때 사용한다. 주의할 점은 검색 엔진이나 시각 장애인들이 사용하는 스크린리더 등 이미지를 볼 수 없는 환경에서도 이미지의 내용을 이해할 수 있도록 동등한 대체 텍스트를 제공해야 한다는 것이다.
- 대체 텍스트 제공을 위해 <img> 요소 내에서 alt 속성을 사용할 수 있다.

``

↑  
src 속성과 더불어 필수 속성임.

### ❖ <a> 요소

- <a> 요소는 하이퍼링크를 지정할 때 사용하는 요소로, HTML5에서는 문법의 변화가 생겼다 기존 HTML4.01와 XHTML1.0에서는 <a> 요소가 블록 요소를 포함할 수 없었지만 HTML5에서는 블록 요소를 포함하는 게 가능해 졌다.

```
<a href="url">
  <h1>    ...    </h1>
  <p>    ...    </p>
</a>
```

## 6. 이미지 및 하이퍼 링크

### ❖ a-img.html

- 실습 폴더에 a-img.html 파일을 생성하고 다음과 같이 작성한다.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>이미지 및 하이퍼 링크</title>
</head>
<body>
  <h1>이미지 및 하이퍼 링크 삽입</h1>

  <a href=https://tacademy.sktechx.com/ target="_blank">
    <p>
      
    </p>
    <p>
      HTML5에서 블록 전체에 하이퍼 링크를 지정하는 것도 가능합니다.
    </p>
    .....
  </a>
```

## 6. 이미지 및 하이퍼 링크

### ❖ a-img.html

<p>

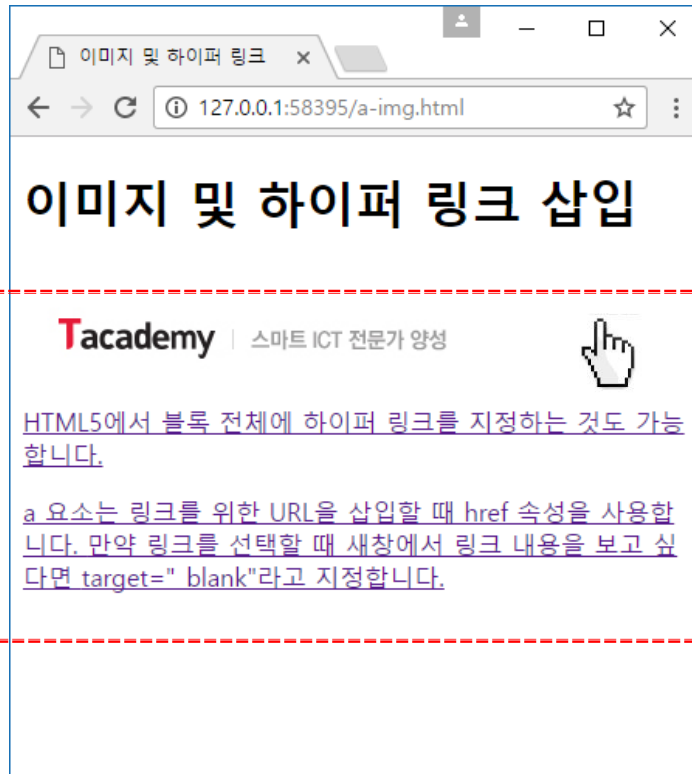
a 요소는 링크를 위한 URL을 삽입할 때 href 속성을 사용합니다.  
만약 링크를 선택할 때 새창에서 링크 내용을 보고 싶다면  
target="\_blank"라고 지정합니다.

</p>

</a>

</body>

</html>

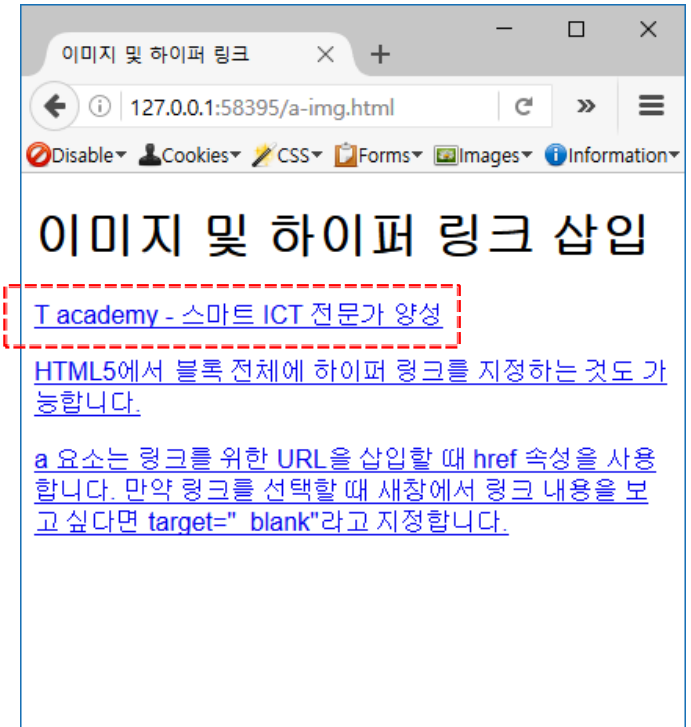
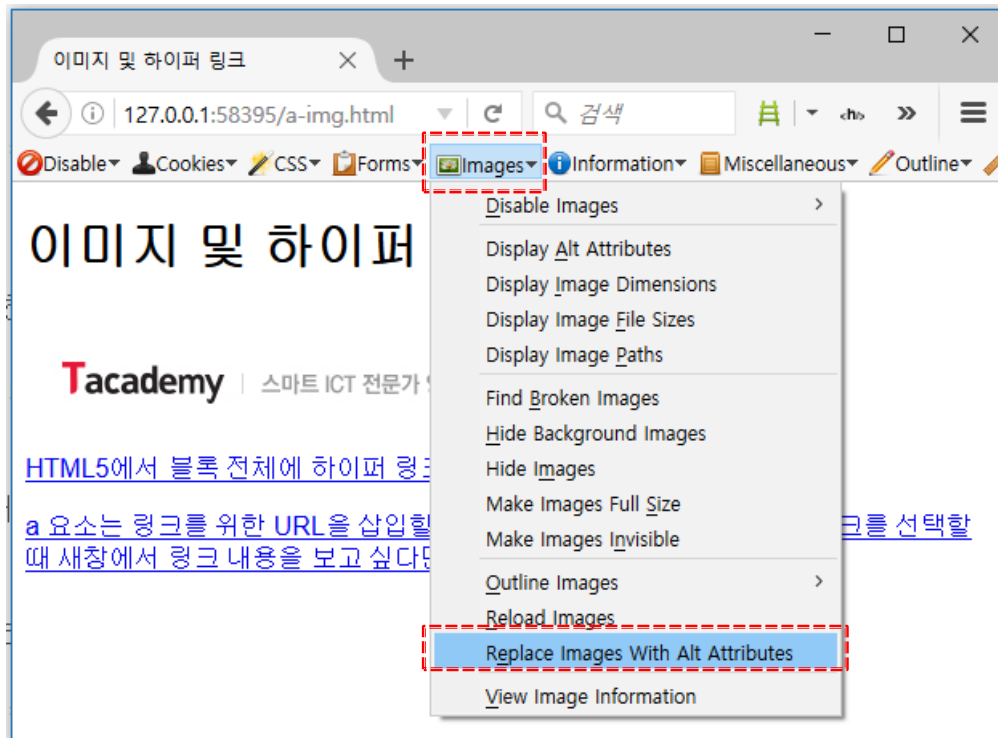


<p> 요소 전체가  
하이퍼링크로 지정되어 있음

## 6. 이미지 및 하이퍼 링크

### ❖ 대체 텍스트

- 웹접근성 측면에서 중요한 정보인 대체 텍스트 제공이 적절한지 확인하기 위해 앞서 설치한 Web Developer를 통해 검사해 보자.
- 다음은 파이어폭스 브라우저에서 이미지를 대체 텍스트로 표시한 모습이다.



## 7. 콘텐츠 캡션을 위한 요소

### ❖ <figure>, <figcaption> 요소

- <figure> 요소는 HTML5에 새롭게 추가된 요소로, 이미지, 비디오 오디오 등을 포함할 수 있으며 <figure> 요소에 포함된 콘텐츠에 캡션을 제공하고자 할 때 사용한다. 이때 <figcaption> 요소로 <figure> 요소에 포함된 정보에 대한 캡션을 지정할 수 있으나 <figcaption> 요소는 생략이 가능하다.

<figure>



<figcaption>이미지 캡션</figcaption>

</figure>

## 8. 텍스트 레벨의 시맨틱 요소

### ❖ <time> 요소

- <time> 요소는 HTML5에 새롭게 추가된 요소로, 컴퓨터가 이해할 수 있는 날짜 및 시간 정보를 마크업 할 수 있다.
- <time> 요소의 필수 속성으로는 datetime 속성이 있으며 datetime 속성의 값으로 날짜 및 시간 정보를 지정할 수 있다. 이때 시간은 24시간 방식, 날짜는 GMT 표준시 형식으로 표현해야 하며 날짜와 시간 정보를 구분할 때는 시간 정보 앞에 T를 붙인다.

```
<time datetime="2017-10-04T13:30:00">2017년 10월 4일</time>
```

### ❖ <blockquote>, <q> 요소

- <blockquote> 요소와 <q> 요소는 인용문을 나타낼 때 사용한다. 단, <blockquote> 요소는 블록 콘텐츠 전체를 인용할 때 사용하며, <q> 요소는 단락의 텍스트 일부를 인용할 때 사용한다. 특히 <q> 요소를 사용하여 마크업하면 대부분의 웹 브라우저에서 렌더링 시 <q> 요소로 마크업 한 문장의 앞뒤에 인용 부호가 자동으로 삽입된다.
- <blockquote> 요소와 <q> 요소는 속성으로 cite를 제공하며 해당 속성은 인용문의 출처를 표기한다.

```
<blockquote cite="http://www.w3.org"> ... </blockquote>
```

```
<q cite="http://www.w3.org"> ... </q>
```

## 9. 비디오 및 오디오

### ❖ <video>, <audio>, <source>, <track> 요소

- 비디오 및 오디오 관련 요소는 HTML5에 새롭게 도입된 요소로, 동영상 및 오디오 콘텐츠를 삽입하고자 할 때 사용한다. 동영상 및 오디오 콘텐츠의 경우 기존 HTML에서는 플러그 인에 의존적이었지만, <video> 요소와 <audio> 요소의 등장으로 더 이상 플러그 인에 의존하지 않고 동영상 및 오디오 콘텐츠를 제공할 수 있게 되었다.
- 또한 플러그 인에 의존적이지 않기 때문에 CSS를 활용하여 디자인을 적용할 수 있고, 자바스크립트로 동영상 및 오디오 등을 제어할 수 있게 되었다.
- <video> 및 <audio> 요소를 사용하여 동영상 및 오디오 콘텐츠를 삽입하는 경우, 웹 브라우저가 지원하는 동영상 및 오디오 형식이 통일되어 있지 않다. 아직까지는 이러한 문제가 해결되지 않은 상황이다. 현재 HTML5에서는 영상 및 음성 코덱 그리고 해당 콘텐츠의 컨테이너 조합 형식으로 H.264, AAC, MP4와 Theora, Vorbis, OGG 등을 후보로 생각하고 있지만, 아직까지 웹 브라우저 제조사 간 이견으로 규정되어 있지 않다.

## 9. 비디오 및 오디오

❖ <video>, <audio>, <source>, <track> 요소

```
<video controls poster="media/google_stories.jpg"
      preload="none" width="640" height="360">
  <source src="media/google-developer-stories.webm"
        type='video/webm; codecs="vp8, vorbis"'>
  <track src="media/video-subtitles-en.vtt" kind="captions" srclang="en"
        label="English captions" default>
</video>
```

자막 파일

```
<audio controls="controls" preload="auto">
  <source src="media/interview.mp3" type="audio/mp3">
</audio>
```



## 9. 비디오 및 오디오

### ❖ 브라우저 별 지원 동영상 및 오디오 형식

#### ◎ 동영상 형식

Codecs/Container	IE9+	Chrome29+	Firefox23+	Sarari6+	Opera16+
Theora+Vorbis+Ogg	×	○	○	×	○
H.264+AAC+MP4	○	○	○	○	×
WebM	×	○	○	×	○

#### ◎ 오디오 형식

형식	IE9+	Chrome29+	Firefox23+	Sarari6+	Opera16+
MP3(.mp3)	○	○	○	○	×
Ogg Vorbis(.ogg)	×	○	○	×	○
Windows Audio(.wav)	×	○	○	○	○
MPEG4 AAC(.m4a)	○	○	○	○	×

## 10. 표 관련 요소

### ❖ <table>, <tr>, <th>, <td> 요소

- 표 관련 요소는 열과 행으로 구분되는 2차원 형태의 데이터 테이블을 마크업 할 때 사용한다. 주요 요소로는 <table>, <caption>, <colgroup>, <col>, <thead>, <tbody>, <tfoot>, <tr>, <th>, <td> 요소가 있다.
- 기존에 웹 문서를 제작할 때는 <table> 관련 요소를 레이아웃을 위해 주로 사용하였는데, 이는 구조적인 마크업을 해치며 웹 접근성에도 나쁜 영향을 끼치는 사례라고 볼 수 있다. 레이아웃을 위해서라면 테이블이 아닌 CSS를 사용할 것을 권장한다.
- <table>에는 접근성을 위해 <caption> 요소를 반드시 제공해야 한다. 그리고 함께 요약정보를 제공해야 하는데 요약 정보는 표에 제공되는 정보가 어떤 것이 있는지 열과 행의 내용을 명시적으로 알려 주어야 한다. 이때 <p> 요소로 요약 정보를 마크업 한 후 WAI-ARIA의 aria-describedby 속성을 사용하여 관련 테이블과 연결하여 제공하면 된다.
- 제목 셀인 <th> 요소의 경우 행 제목인지 열 제목인지에 대한 정보를 scop 속성을 통해 제공할 수 있다. 참고로 행 제목은 scope="row" 열 제목은 scope="col"로 지정한다.

## 10. 표 관련 요소

### ❖ table.html

- 실습 폴더에 table.html 파일을 생성하고 다음과 같이 작성한다.

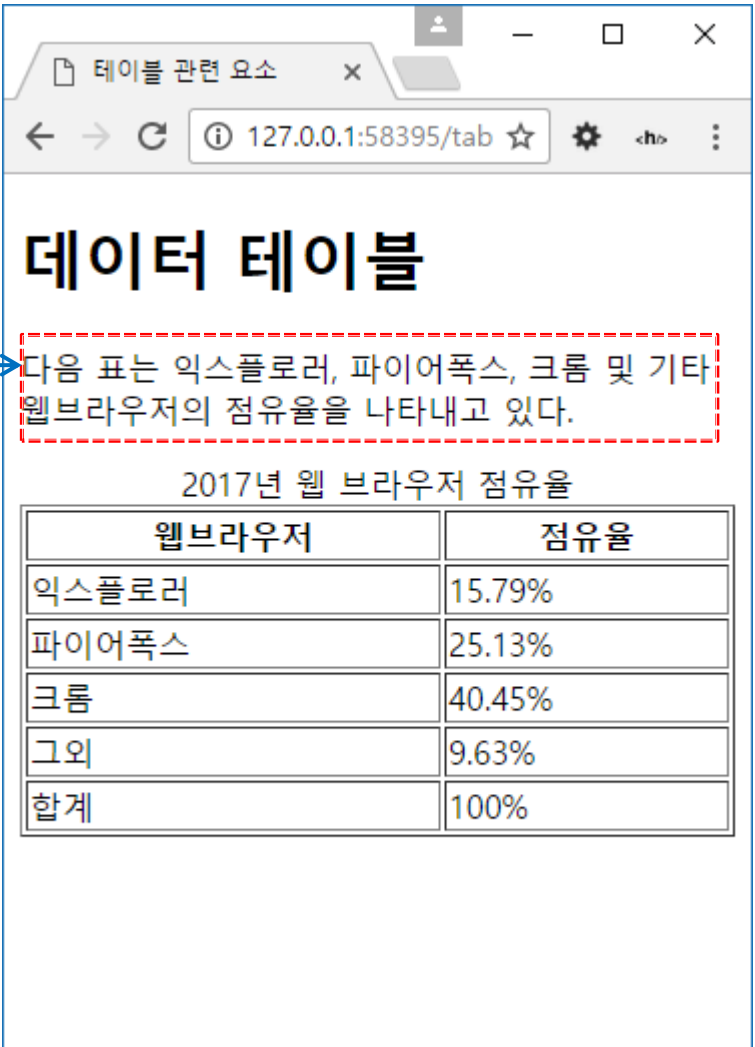
```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="utf-8">
  <title>테이블 관련 요소</title>
</head>
<body>
<h1>데이터 테이블</h1>
<p id="description">다음 표는 익스플로러, 파이어폭스, 크롬 및 기타 웹 브라우저의
  점유율을 나타내고 있다.</p>
<table border="1" width="100%" aria-describedby="description">
  <caption>2017년 웹 브라우저 점유율</caption>
  <tr>
    <th scope="col">웹브라우저</th>
    <th scope="col">점유율</th>
  </tr>
  <tr>
    <td>익스플로러</td>
    <td>15.79%</td>
  </tr>
  .....
```

## 10. 표 관련 요소

### ❖ table.html

```
<tr>
  <td>파이어폭스</td>
  <td>25.13%</td>
</tr>
<tr>
  <td>크롬</td>
  <td>40.45%</td>
</tr>
<tr>
  <td>그외</td>
  <td>9.63%</td>
</tr>
<tr>
  <td>합계</td>
  <td>100%</td>
</tr>
</table>
</body>
</html>
```

표 요약정보



데이터 테이블

다음 표는 익스플로러, 파이어폭스, 크롬 및 기타 웹브라우저의 점유율을 나타내고 있다.

2017년 웹 브라우저 점유율

웹브라우저	점유율
익스플로러	15.79%
파이어폭스	25.13%
크롬	40.45%
그외	9.63%
합계	100%

## 11. 폼 관련 요소

### ❖ <form>, <fieldset>, <legend> 요소

- HTML5에서는 다수의 폼 관련 요소가 추가되었다.
- <form> 요소는 서버에 사용자 입력 값을 전송하는 역할을 담당하는 요소이다. action 속성이 필수이며 해당 속성에 값을 전송할 서버의 URL을 지정하면 된다.
- <fieldset> 요소는 폼 관련 서식을 그룹화하고 <legend> 요소를 사용하여 그룹화 한 폼 서식의 목적이나 이름을 지정할 수 있다.

```
<form action="login.php" method="post">  
  <fieldset>  
    <legend> ..... </legend>  
  </fieldset>  
</form>
```

## 11. 폼 관련 요소

### ❖ <label> 요소

- <label> 요소는 웹접근성 측면에서도 중요한 요소로, 다양한 폼 서식의 설명을 의미한다. 가령 아이디 입력 상자나 비밀번호 입력 상자의 경우, 기계가 해당 입력 상자가 어떤 데이터를 입력하는 영역인지 알 수 없기 때문에 <label> 요소로 연관 관계를 알려주어야 한다. 이렇게 폼 서식에 대한 설명을 명확하게 지정하면 보조 기기 에서 폼 서식의 접근이나 이해가 쉬워진다.
- <label> 요소로 설명을 제공할 수 있는 폼 서식으로는 <button>, <input>, <keygen>, <meter>, <output>, <progress>, <select>, <textarea> 요소를 들 수 있다.
- <label> 요소를 사용하여 폼 서식과의 관계를 지정할 때는 for 속성을 사용하여 연결하고자 하는 폼 서식의 id 속성 값을 지정하면 된다.

<fieldset>

<legend>로그인 정보</legend>

<label for="user-id">아이디</label> <input type="text" id="user-id">

<label for="user-pw">비밀번호</label> <input type="password" id="user-pw">

</fieldset>

## 11. 폼 관련 요소

### ❖ <input> 요소

- <input> 요소는 여러 가지 폼 서식을 마크업할 때 사용한다. 특히 HTML5에서는 기존에 제공하던 10가지 type 이외에 search, tel, url 등 다수의 type이 새롭게 추가되었다. 또한 type 속성 이외에 required, placeholder 속성 등 다수의 신규 속성도 추가되어 성능이 한층 더 강화되었다. 그러나 많은 신규 기능이 아직 모든 브라우저에서 지원되고 있지 않으므로 사용 시 해당 브라우저 지원여부를 확인하는 것이 필요하다.

### ❖ <input> 요소가 지원하는 type 속성 값

속성 값	설명	속성 값	설명
type="text"	한 줄 글 상자	type="email"	이메일 입력 상자
type="password"	비밀번호 입력 상자	type="date"	날짜 입력 서식
type="search"	검색어 입력 상자	type="month"	월 입력 서식
type="tel"	전화번호 입력 상자	type="week"	주 단위 입력 서식
type="url"	URL 입력 상자	type="time"	시간 입력 서식

## 11. 폼 관련 요소

### ❖ <input> 요소

속성 값	설명	속성 값	설명
type="datetime"	날짜 및 시간 입력 서식	type="file"	첨부 파일 서식
type="datetime-local"	타임 존이 없는 날짜와 시간 입력 서식	type="radio"	라디오 버튼 서식
type="number"	숫자 입력 서식	type="submit"	전송 버튼
type="range"	숫자 입력 서식이지만 슬라이더 형태의 UI로 렌더링	type="reset"	리셋 버튼
<div> <div>&lt;label for="username"&gt;이름&lt;/label&gt;</div> <div>&lt;input type="text" id="username"&gt;</div> </div>	색상 입력 서식	type="button"	버튼(스크립트 실행)
type="checkbox"	체크 박스 서식	type="hidden"	숨김 서식

폼 관련 서식은 1대1로 매칭되는 레이블을 반드시 제공하여야 함.



## 11. 폼 관련 요소

### ❖ <input> 요소에 추가 된 새로운 속성

- required 속성은 해당 폼 서식이 필수 속성인지 아닌지를 나타내는 논리 속성이다. required 속성을 지정한 폼 서식에 값을 입력하지 않은 상태로 서버로 데이터를 전송할 경우, 에러 메시지가 웹 브라우저 화면에 출력된다.

```
<label for="userId">아이디</label>  
<input type="text" required id="userId" name="userId">  
<input type="submit" value="전송">
```

- placeholder 속성은 사용자에게 해당 폼 서식에 어떤 값을 입력해야 하는지를 알리기 위한 힌트로 사용 된다. 이때 placeholder 속성에 입력한 값은 해당 폼 서식에 포커스가 있을 때 새로운 값을 입력하면 사라진다.

```
<label for="userId">아이디</label>  
<input type="text" placeholder="예)guest" id="userId" name="userId">
```

## 11. 폼 관련 요소

### ❖ <button> 요소

- <button> 요소는 버튼 서식을 삽입할 때 사용한다. <button> 요소에 type 속성을 지정하지 않으면 전송 버튼이 된다. <button> 요소의 type 속성에 지정할 수 있는 값은 "submit", "reset", "button" 등이며, <input> 요소의 "submit", "reset", "button"과 기능이 동일하다.

<button type="submit">전송</button>

<button type="reset">취소</button>

<button type="button">버튼</button>

## 11. 폼 관련 요소

### ❖ form.html

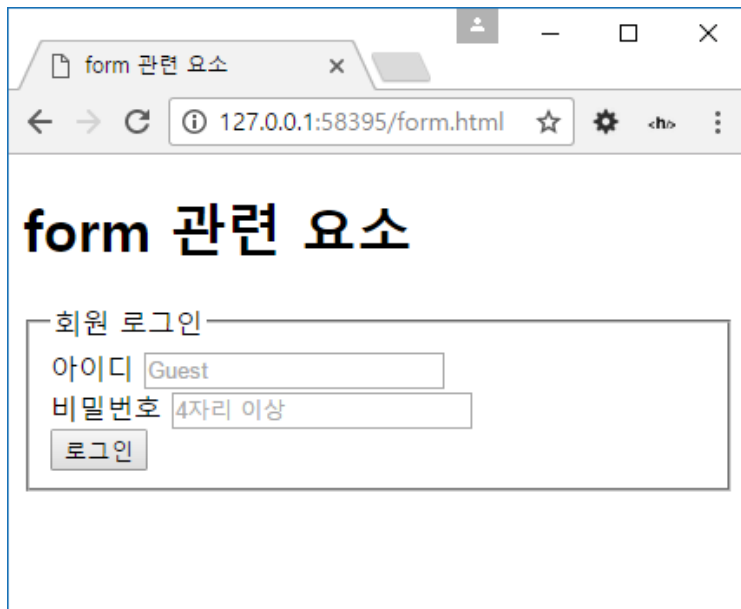
- 실습 폴더에 form.html 파일을 생성하고 다음과 같이 작성한다.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>form 관련 요소</title>
  </head>
  <body>
    <h1> form 관련 요소</h1>
    <form action="login.php" method="post">
      <fieldset>
        <legend>회원 로그인</legend>
        <div>
          <label for="user-id">아이디</label>
          <input type="text" id="user-id" required placeholder="Guest">
        </div>
        <div>
          <label for="user-pw">비밀번호</label>
          <input type="password" id="user-pw" required placeholder="4자리 이상">
        </div>
        .....
      </fieldset>
    </form>
  </body>
</html>
```

## 11. 폼 관련 요소

### ❖ form.html

```
<button type="submit">로그인</button>
</fieldset>
</form>
</body>
</html>
```



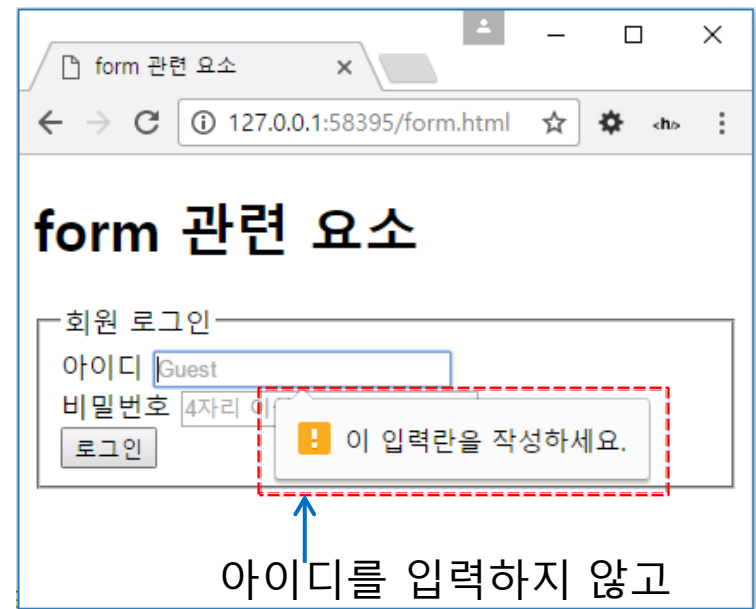
form 관련 요소

회원 로그인

아이디 Guest

비밀번호 4자리 이상

로그인



form 관련 요소

회원 로그인

아이디 Guest

비밀번호 4자리 이상

로그인

이 입력란을 작성하세요.

아이디를 입력하지 않고  
로그인 버튼을 클릭했을 때

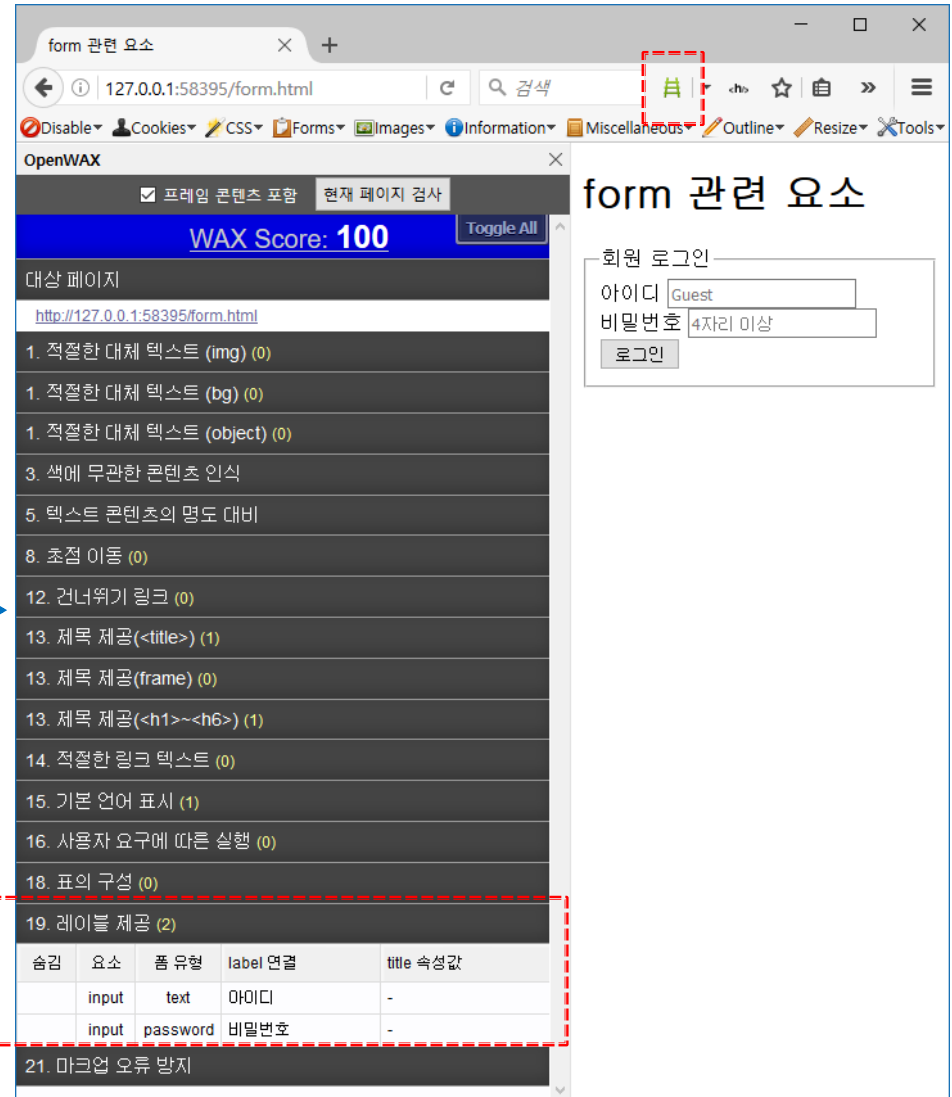
## 11. 폼 관련 요소

### ❖ 레이블 제공여부 검사

- 앞서 작성한 form.html 파일을 웹 브라우저에서 실행한 후 OpenWax 버튼을 클릭하여 레이블 제공에 해당 하는 항목에서 레이블이 적절하게 제공되었는지 살펴보자.

파이어폭스 →

폼 서식에 매칭 된  
레이블 정보 확인. →



form 관련 요소

WAX Score: 100

대상 페이지  
<http://127.0.0.1:58395/form.html>

1. 적절한 대체 텍스트 (img) (0)
1. 적절한 대체 텍스트 (bg) (0)
1. 적절한 대체 텍스트 (object) (0)
3. 색에 무관한 콘텐츠 인식
5. 텍스트 콘텐츠의 명도 대비
8. 초점 이동 (0)
12. 건너뛰기 링크 (0)
13. 제목 제공(<title>) (1)
13. 제목 제공(frame) (0)
13. 제목 제공(<h1>~<h6>) (1)
14. 적절한 링크 텍스트 (0)
15. 기본 언어 표시 (1)
16. 사용자 요구에 따른 실행 (0)
18. 표의 구성 (0)
19. 레이블 제공 (2)
21. 마크업 오류 방지

숨김	요소	폼 유형	label 연결	title 속성값
	input	text	아이디	-
	input	password	비밀번호	-

form 관련 요소

회원 로그인

아이디

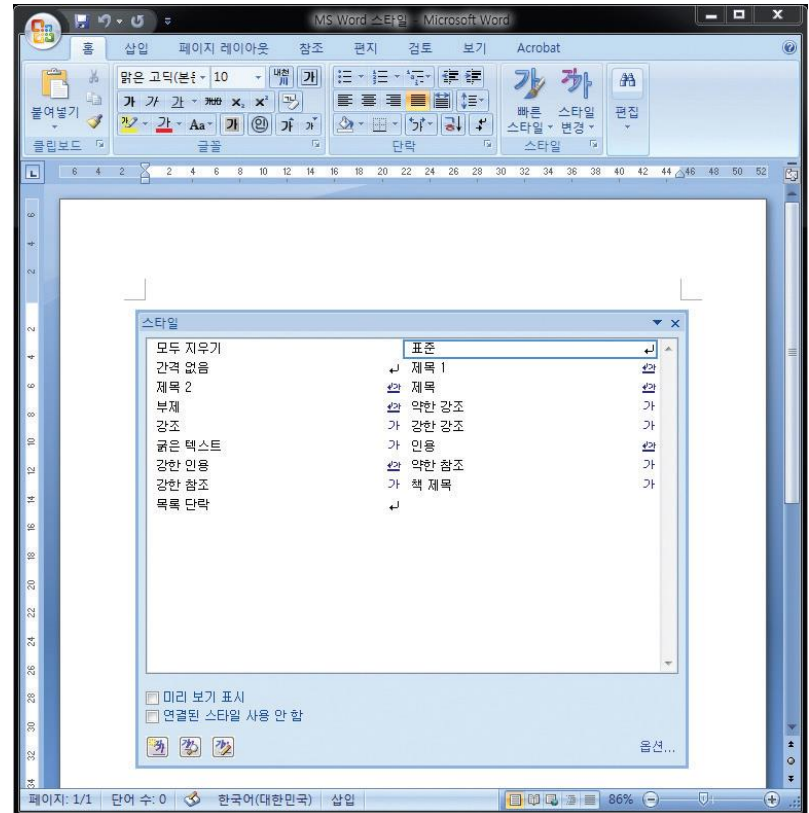
비밀번호

# CSS 기초

# 1. CSS 개요

## ❖CSS(Cascading Style Sheet)

- CSS 또는 캐스케이딩 스타일 시트 (Cascading Style Sheet)라고 하며 주로 마크업 언어가 실제로 표시되는 방법을 기술하는 언어이다.
- CSS는 HTML과 XHTML에 주로 쓰이며, XML에서도 사용할 수 있다.
- CSS는 W3C의 표준이며, 레이아웃과 스타일을 정의할 때의 자유도가 높다.
- CSS는 웹 사이트에서 사용되는 디자인을 위한 언어로 웹의 독창적인 개념이 아니라 워드 프로세스 등에 다양한 응용 프로그램에서 사용했던 개념이다.



## 2. CSS2 VS CSS3

### ❖CSS의 과거와 현재

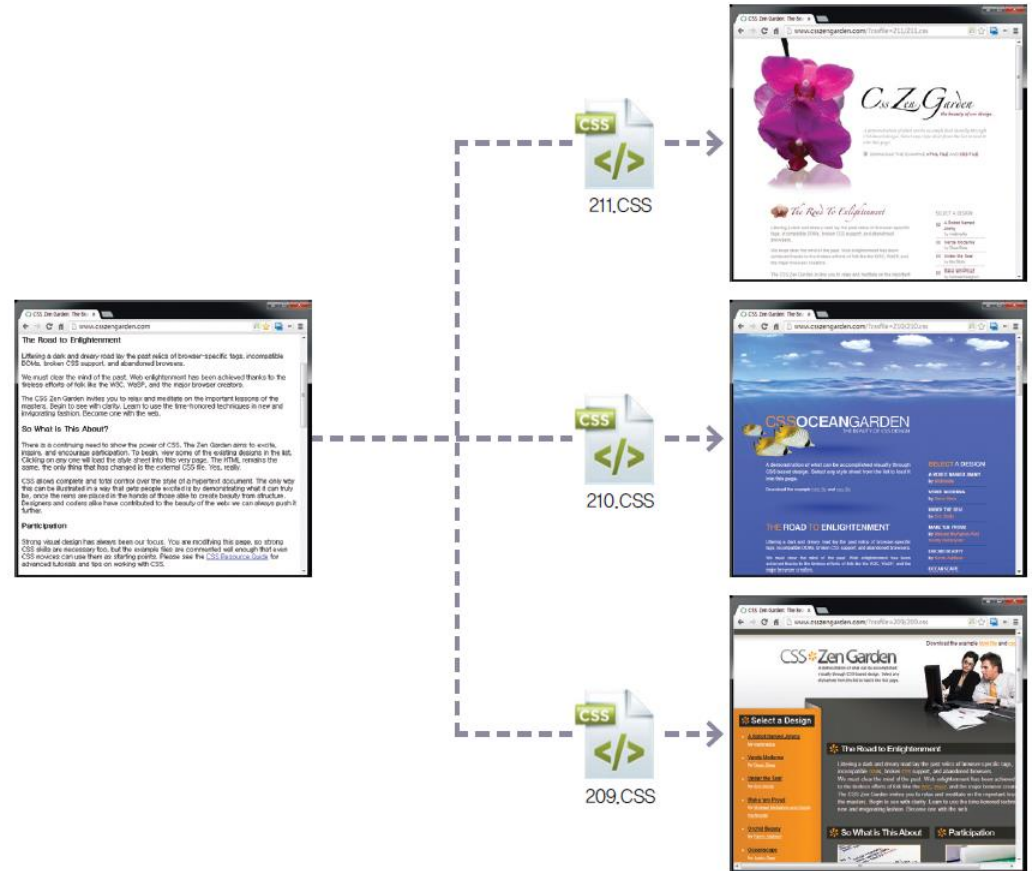
- 1996년 W3C의 주도하에 첫 번째 CSS 버전인 CSS Level 1이 발표 되었다.
- 1998년에 CSS Level 2가 등장하면서 대부분의 웹 브라우저들이 CSS Level 2를 지원하기 시작했다.
- 그 후 CSS Level 2의 버그를 수정한 CSS Level 2.1이 2006년에 발표되면서 현재까지 표준으로 사용되고 있다.
- CSS Level 3는 CSS Level 2.1과 달리 모든 명세가 포함된 버전이 아닌 모듈 단위로 개발되고 있으며, 표준화 단계 역시 모듈 단위로 진행되고 있다.
- 이 중 몇몇 모듈은 현재 Recommendation(권고안) 단계에 있으며, Working Draft (초안) 단계에 머물러 있는 모듈도 있다.



### 3. CSS 사용의 의의

#### ❖ 구조와 표현의 분리

- 스타일 사용의 중요한 의의는 문서의 구조와 표현을 분리할 수 있다는 점이다.
- 이는 구조와 표현을 분리함으로써 문서 구조의 수정 없이 스타일의 변경만으로 다양한 표현을 할 수 있다는 것을 의미한다.



## 4. CSS 사용 규칙

### ❖ 외부 스타일시트(External Style Sheet)

- CSS 파일을 외부에 생성하여 HTML 문서에 연결하는 방식으로 <link> 요소를 사용하는 방법과 @import 명령을 사용하는 두 가지 방식이 있다.

```
<link rel="stylesheet" type="text/css" href="css/style.css">  
@import url(css/external.css) ;
```

### ❖ 내부 스타일시트(Embedded Style Sheet)

- HTML 파일 내에 CSS 코드를 직접 포함하여 스타일이 적용되도록 하는 방법으로, CSS 코드는 <style> 요소 내에 선언한다.

```
<style>  
  p { color:#00f ; }  
</style>
```

### ❖ 인라인 스타일시트(Inline Style Sheet)

- 특정 HTML 요소에 style 속성을 사용하여 CSS 코드를 선언하는 방법이다.
- 이 방법은 구조와 표현의 분리라는 관점에서 바람직하지 않다. 특히 inline 방식의 스타일은 선택자의 우선순위가 가장 높기 때문에 스타일의 재정의가 어렵거나 불가능한 경우가 발생할 수 있으므로 사용에 주의가 필요하다.

```
<p style="color : #6a099d ;">인라인 스타일시트</p>
```

## 4. CSS 사용 규칙

### ❖ CSS 주석

- CSS 주석은 슬래시(/)와 별표(\*)로 묶어서 지정한다.  
아래는 CSS에서 사용하는 주석의 예시를 나타낸 것이다.

```
/* CSS에서 사용할 수 있는 주석입니다. */
```

### ❖ CSS 단위

- 문자열 타입 : "inherit"와 같이 CSS에서 미리 정의된 키워드나 제작자가 정의한 저작자 키워드 등 텍스트로 입력하는 속성 값이 이에 속한다.
- 숫자 타입 : 개수 비율들을 나타낼 수 있는 정수, 실수 값, 퍼센트 값이 이에 속한다.
- 길이 단위 타입 : 길이 단위 타입은 상대 길이 단위 값과 절대 길이 단위 값이 있으며 상대 길이 단위 값은 특정 길이에 비율로 정해지는 값이다.

## 4. CSS 사용 규칙

### ❖ CSS 단위

#### 상대 단위

단위	의미
em	폰트의 기본 크기 값에 비례한 단위
ex	폰트의 기본 X 높이에 비례한 단위
ch	"0"의 기본 폭에 비례한 단위
rem	최상위 요소의 폰트 크기에 비례한 단위
vw	뷰포트 너비에 비례한 단위
vh	뷰포트 높이에 비례한 단위
vmin	뷰포트의 최소 너비, 높이에 비례한 단위

#### 절대 단위

단위	의미
cm	센티미터 단위
mm	밀리미터 단위
in	인치(inch) 단위로, 2.54cm와 같음.
px	픽셀(pixels) 단위로, 1/96inch와 같음.
pt	포인트(point) 단위로, 1/72inch와 같음.
pc	피카(picas) 단위로, 12pt와 같음.

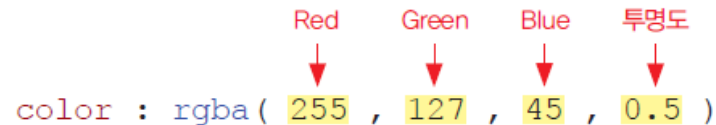
- 기타 단위들로는 각도를 지정하는 "deg", "grad", "rad", "turn", 시간을 나타내는 "s", "ms", 빈도를 지정하는 "Hz", "kHz" 그리고 해상도를 지정하는 "dpi", "dpcm", "dppx"를 들 수 있다.
- 이 단위들은 기존 CSS2에서 음성 브라우저 등의 특수 경우를 상정하여 정의되었던 것이지만, CSS3에서는 표현 속성이 다양해짐에 따라 각도나 시간은 일반적으로 사용되는 값의 형식이 될 것이다.

## 4. CSS 사용 규칙

### ❖ CSS 색상

- CSS3에서의 색상 단위는 좀 더 확대되었다. 특히, 투명 값(alpha)이나 색상, 채도, 명도 등을 지정 할 수 있는 HSL 방식 그리고 currentcolor 키워드 등이 추가되었다. (색상 이름, transparent, 16진수 RGB 색상 값, 256단계 RGB 색상 값, 백분율(%) RGB 색상 값, RGBA 색상 값, HSL 색상 값, HSLA 색상 값, currentcolor)

#### ※ RGBA 형식



color : rgba( 255 , 127 , 45 , 0.5 )

The diagram shows the CSS code `color : rgba( 255 , 127 , 45 , 0.5 )` with four labels above it: "Red" above 255, "Green" above 127, "Blue" above 45, and "투명도" (Transparency) above 0.5. Red arrows point from each label to its corresponding value in the code.

#### ※ HSLA 형식



color : hsla( 0 , 0% , 100% , 0.5 )

The diagram shows the CSS code `color : hsla( 0 , 0% , 100% , 0.5 )` with four labels above it: "색상" (Hue) above 0, "채도" (Saturation) above 0%, "명도" (Lightness) above 100%, and "투명도" (Transparency) above 0.5. Red arrows point from each label to its corresponding value in the code.

## 4. CSS 사용 규칙

### ❖ external.css

- 실습 폴더 하위에 있는 css 폴더에 external.css 파일을 생성하고 다음과 같이 작성한다.

```
@charset "utf-8";  
/* 제목과 본문에 대한 글자 색상 지정 */  
h1 { color:#f00 ; }  
p { color:#00f ; }
```

### ❖ link.html

- 실습 폴더에 link.html 파일을 생성하고 다음과 같이 작성한다.

```
<!DOCTYPE html>  
<html lang="ko">  
<head>  
  <meta charset="utf-8">  
  <title>외부 스타일시트</title>  
  <link rel="stylesheet" type="text/css" href="css/external.css">  
</head>  
<body>  
  <h1>link 요소 사용</h1>  
  <p>link 요소를 사용하여 외부에 생성한 CSS 파일을 삽입합니다.</p>  
</body>  
</html>
```

## 4. CSS 사용 규칙

### ❖ import.html

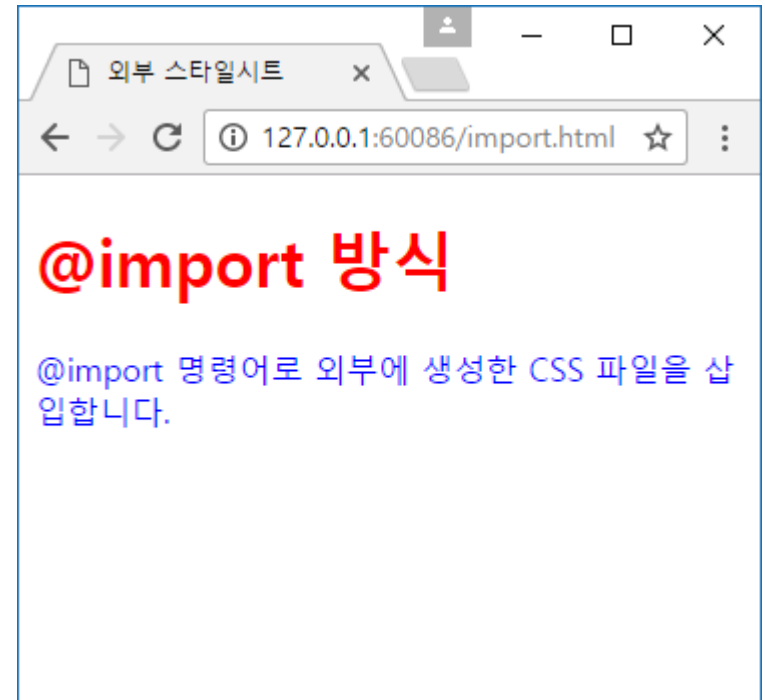
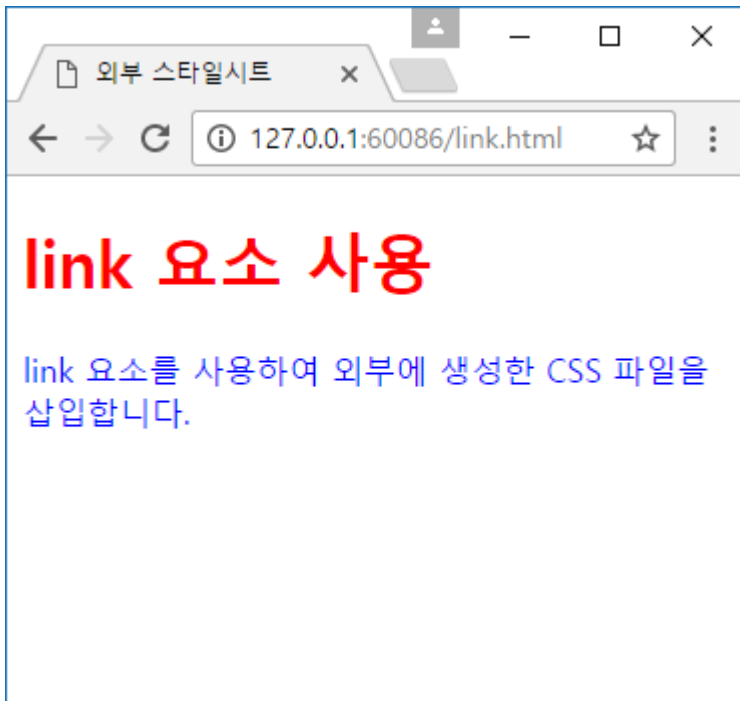
- 실습 폴더에 import.html 파일을 생성하고 다음과 같이 작성한다.

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="utf-8">
  <title>External 방식</title>
  <style>
    /* @import 방식으로 CSS 파일을 삽입할 수 있다.
       @import 방식은 CSS 파일 내에서 다른 CSS 파일을 삽입할 때도
       사용할 수 있다. */
    @import url(css/external.css) ;
  </style>
</head>
<body>
  <h1>@import 방식</h1>
  <p>@import 명령어로 외부에 생성한 CSS 파일을 삽입합니다.</p>
</body>
</html>
```

## 4. CSS 사용 규칙

### ❖ link.html, import.html

- 작성 한 link.html 파일과 import.html 파일을 웹 브라우저로 실행해 보자.



link.html 파일과 import.html 파일 모두 external.css 파일에 작성한 CSS 코드가 적용됨



## 4. CSS 사용 규칙

### ❖ style.html

- 실습 폴더에 style.html 파일을 생성하고 다음과 같이 작성한다.

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="utf-8">
  <title>내부 스타일시트</title>
  <style>
    h1 { color:#f00 ; }
    p { color:#00f ; }
  </style>
</head>
<body>
  <h1>style 요소 사용</h1>
  <p>style 요소 내에 직접 CSS 코드를 삽입합니다.</p>
</body>
</html>
```

## 4. CSS 사용 규칙

### ❖ inline.html

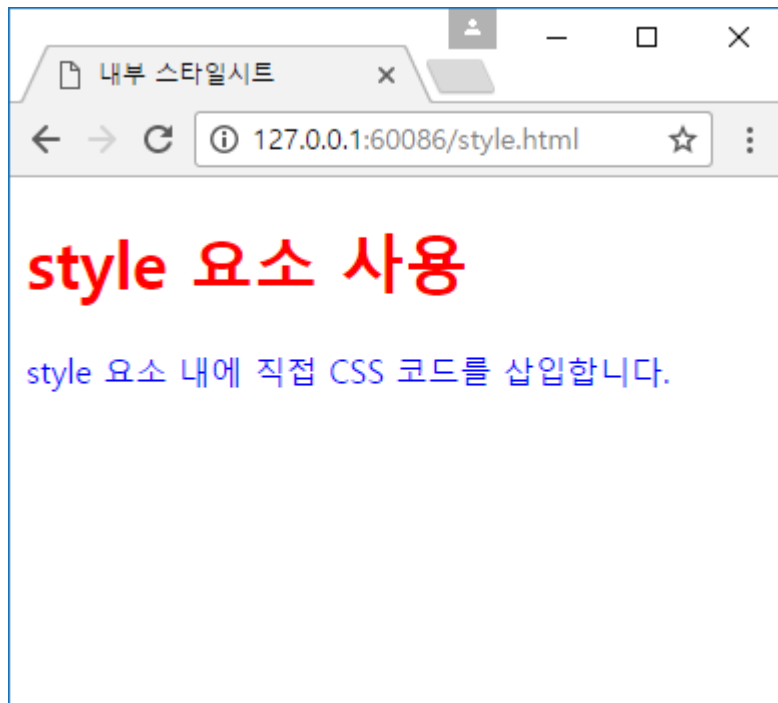
- 실습 폴더에 inline.html 파일을 생성하고 다음과 같이 작성한다.

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="utf-8">
  <title>인라인 스타일시트</title>
  <style>
    h1 { color:#f00 ; }
    p { color:#00f ; }
  </style>
</head>
<body>
  <h1 style="color : #00f ;">style 속성 사용</h1>
  <p style="color : #f00 ;">
    style 속성으로 요소에 직접 CSS 코드를 삽입합니다.
  </p>
</body>
</html>
```

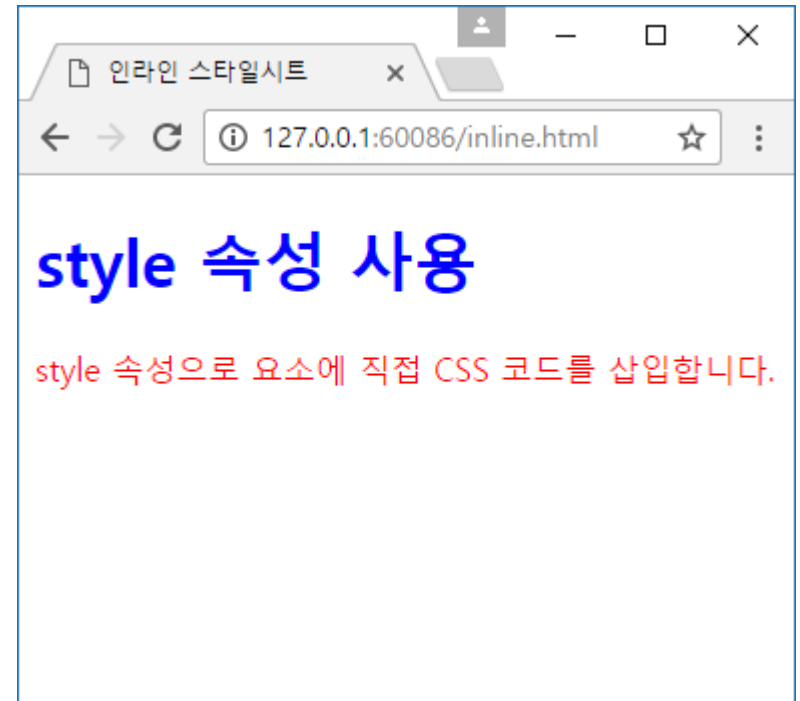
## 4. CSS 사용 규칙

❖ style.html, inline.html

- 작성 한 style.html 파일과 inline.html 파일을 웹 브라우저로 실행해 보자.



↑  
<style> 요소 내에 선언한 CSS가 적용



↑  
style 속성으로 직접 선언한 CSS 코드가 적용

## 5. 선택자(Selector)

### ❖ 전체 선택자(Universal Selector)

- 전체 선택자(Universal Selector)는 모든 요소를 선택하는 방법으로, "\*"를 선택자로 선언한다.

```
사용 예    * {  
  
        margin : 0 ;  
        padding : 0 ;  
    }
```

### ❖ 요소 선택자(Type Selector)

- 요소 선택자(Type Selector)는 HTML 요소를 선택하는 방법으로, "h1", "p", "div" 등의 요소를 선택자로 지정할 수 있다.

```
사용 예    div {  
  
        color : rgba( 255 , 0 , 0 , 0.5 ) ;  
    }
```

## 5. 선택자(Selector)

### ❖ class 선택자(Class Selector)

- class 선택자는 HTML 요소의 class 속성 값을 참조하여 선택하는 방법이다. 이때 class 속성 값은 하나의 HTML 요소에 여러 개를 지정할 수 있기 때문에 다중 class를 선택자로 지정할 수도 있다.

```
사용 예   .note {  
           font-size : 1.2em ;  
        }
```

### ❖ id 선택자(Identification Selector)

- id 선택자는 HTML 요소의 id 속성 값을 참조하여 선택하는 방법이다. 이때 id 속성 값은 하나의 HTML 문서에 한 번만 사용할 수 있기 때문에 id 선택자를 사용하면 유일한 요소를 선택할 수 있다.

```
사용 예   #main {  
           font-weight : bold ;  
        }
```

## 5. 선택자(Selector)

### ❖ 속성 선택자(Attribute Selector)

- 속성 선택자는 HTML 요소의 속성을 참조하여 선택하는 방법을 의미하며, 이때 속성의 지정 여부나 속성 값의 일치 여부로 선택할 수 있다.

```
사용 예   a[title] {  
           text-decoration : underline ;  
        }
```

### ❖ 가상 클래스 선택자(Pseudo-classes Selector)

- 가상 클래스 선택자는 요소의 상태나 상황에 따라 선택하는 방법으로, 링크의 경우 방문하기 전, 방문한 후, 링크 위에 마우스를 올려놓거나 포커스 시 등의 상황을 선택하여 스타일을 지정할 수 있다. 또한 언어에 따른 구분이나 마크업 구조에 따라 특정 요소를 선택할 수도 있다.

```
사용 예   a:link {  
           color : rgba( 255 , 100 , 100 , 0.8 ) ;  
        }
```

## 5. 선택자(Selector)

### ❖ 가상 요소 선택자(Pseudo-element Selector)

- 가상 요소 선택자는 요소의 첫 글자나 첫 줄 또는 요소 앞이나 뒤 등 가상의 영역을 선택하고자 할 때 사용한다.

```
사용 예  p:first-letter {  
           color : #ff0000 ;  
           font-size : 5em ;  
        }
```

### ❖ 구조 선택자(Pseudo-classes Selector)

- 구조 선택자는 요소의 위치나 자식 요소가 유일한지 등 DOM의 구조를 기준으로 요소를 선택하는 방법이다.

```
사용 예  ul li:nth-child(odd) {  
           color : #ff0000 ;  
           font-size : 5em ;  
        }
```

## 5. 선택자(Selector)

### ❖ 다양한 가상 선택자 및 구조 선택자

| 선택자(Selector)  | 의미  |
|----------------|---|
| E:link         | 아직 방문하지 않는 하이퍼링크를 가진 "E" 요소를 선택함            |
| E:visited      | 이미 방문한 하이퍼링크를 가진 "E" 요소를 선택함                |
| E:active       | 현재 사용자 액션을 받고 있는 "E" 요소를 선택함                |
| E:hover        | 마우스 포인터가 올라간 "E" 요소를 선택함                    |
| E:focus        | 키보드의 포커스를 받은 "E" 요소를 선택함                    |
| E:target       | "E" 요소가 하이퍼링크 타겟이 되는 경우에 선택함                |
| E:lang(fr)     | Lang 속성이 "fr(프랑스어를 나타내는 속성 값)"인 "E" 요소를 선택함 |
| E:enabled      | 사용 가능 상태의 "E" 요소를 선택함                       |
| E:disabled     | 사용 불가 상태의 "E" 요소를 선택함                       |
| E:checked      | 체크된 "E" 요소를 선택함                             |
| E:root         | 문서 최상위의 요소를 선택함                             |
| E:nth-child(n) | 상위 요소의 n 번째 자식 요소가 "E"이면 선택함                |



## 5. 선택자(Selector)

### ❖ 다양한 가상 선택자 및 구조 선택자

| 선택자(Selector)         | 의미  |
|-----------------------|---|
| E:nth-last-child(n)   | 상위 요소의 역순으로 n 번째 자식 요소가 "E"이면 선택함               |
| E:nth-of-type(n)      | 동일한 "E" 타입의 형제 요소중 n 번째 "E" 요소를 선택함             |
| E:nth-last-of-type(n) | 동일한 "E" 타입의 형제 요소 중 역순으로 n 번째 "E" 요소를 선택함       |
| E:frist-child         | 첫 번째 자식 요소의 타입이 "E"이면 선택함                       |
| E:last-child          | 마지막 자식 요소가 타입이 "E"이면 선택함                        |
| E:first-of-type()     | 상위 요소에 대하여 첫 번째 자식 요소의 타입이 "E"이면 선택함            |
| E:last-of-type()      | 상위 요소에 대하여 마지막 자식 요소의 타입이 "E"이면 선택함             |
| E:only-child          | 상위 요소에 대하여 유일한 자식 요소가 타입이 "E"이면 선택함             |
| E:only-of-type        | 상위 요소에 대하여 자식 요소중 다른 "E" 요소가 없이 유일한 "E"일 경우 선택함 |
| E:empty               | 텍스트 노드를 포함하여 아무런 자식 요소를 갖고 있지 않는 "E" 요소를 선택함    |
| E:not(S)              | "S"로 지정된 선택자와 일치하지 않는 "E" 요소를 선택함               |

## 5. 선택자(Selector)

### ❖ 하위 선택자(Descendant Combinator)

- 하위 선택자 방식은 선택자와 선택자를 공란으로 선언하며, 선행 선택자의 하위 요소 중 후행 선택자에 해당하는 요소를 선택하는 방법이다.

```
사용 예   #main div {  
           border : 3px solid currentcolor ;  
        }
```

### ❖ 자식 선택자(Child Combinators)

- 자식 선택자 방식은 선행 선택자인 부모 요소 하위에 포함된 후행 선택자인 자식 요소를 선택하는 방법이다. 이때 부모 선택자와 자식 선택자는 ">"로 구분하여 선언한다.

```
사용 예   #main > div {  
           border : 3px solid red ;  
        }
```

## 5. 선택자(Selector)

### ❖ 형제 선택자(Sibling Combinator)

- 형제 선택자는 기본(General) 형제 선택자와 인접(Adjacent) 형제 선택자로 구분할 수 있다. 이때 기본 형제 선택자는 선행 선택자와 후행 선택자를 "+"로 구분하여 선언하고, 인접 형제 선택자는 "~"로 구분하여 선언한다.

```
사용 예   h1 + p {  
           color : red ;  
        }  
        .note ~ p {  
           color : blue ;  
        }
```

### ❖ 선택자의 그룹화

- 앞에서 살펴본 모든 선택자는 콤마(,)를 사용하여 그룹으로 한 번에 선언할 수 있다. 선택자를 그룹으로 선언할 경우, 선언된 모든 선택자에는 동일한 선언이 적용된다.

```
사용 예   h1 , h1 + p , .note, #main, p[title] {  
           font-size : 1.2em ;  
        }
```

## 5. 선택자(Selector)

### ❖ universal.html

- 실습 폴더에 universal.html 파일을 생성한 후 다음과 같이 작성하고 브라우저에서 확인해 보자.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>전체 선택자 - Universal Selector</title>
    <style>
      * {
        margin : 0 ;
        padding: 0 ;
      }
    </style>
  </head>
  <body>
    <h1>전체 선택자</h1>
    <p>
      전체 선택자를 사용하여 모든 요소에 동일한 스타일 선언을 지정할 수 있습니다.
    </p>
  </body>
</html>
```

## 5. 선택자(Selector)

### ❖ type.html

- 실습 폴더에 type.html 파일을 생성한 후 다음과 같이 작성하고 브라우저에서 확인해 보자.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>요소 선택자 - Type Selector</title>
    <style>
      h1 { color : rgba( 255 , 0 , 0 , 0.5 ) ; }
    </style>
  </head>
  <body>
    <h1>요소 선택자</h1>
    <p>
      요소 선택자를 사용하여 특정 요소에 스타일을 지정할 수 있습니다.
    </p>
  </body>
</html>
```

## 5. 선택자(Selector)

### ❖ class.html

- 실습 폴더에 class.html 파일을 생성한 후 다음과 같이 작성하고 브라우저에서 확인해 보자.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>class 선택자 - Class Selector</title>
    <style>
      .note { font-size : 1.2em ; font-weight : bold ; color : #054D4A ; }
    </style>
  </head>
  <body>
    <h1 class="note">class 선택자</h1>
    <p>class 선택자를 사용하여 특정 class가 지정된 요소에 스타일을
      지정할 수 있습니다.</p>
    <p class="note">class 선택자를 사용하여 특정 class가 지정된 요소에 스타일
을
      지정할 수 있습니다.</p>
    <p>class 선택자를 사용하여 특정 class가 지정된 요소에 스타일을
      지정할 수 있습니다.</p>
  </body>
</html>
```

## 5. 선택자(Selector)

### ❖ id.html

- 실습 폴더에 id.html 파일을 생성한 후 다음과 같이 작성하고 브라우저에서 확인해 보자.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>id 선택자 - Identification Selector</title>
    <style>
      #main { font-weight : bold ; color : #054D4A ; text-decoration :
underline ; }
    </style>
  </head>
  <body>
    <h1>id 선택자</h1>
    <p>id 선택자를 사용하여 특정 아이디가 지정된 요소에 스타일을 지정할 수
      있습니다. 이 때 하나의 문서에는 중복된 아이디가 존재해서는 안됩니다.</p>
    <p id="main">id 선택자를 사용하여 특정 아이디가 지정된 요소에 스타일을
      지정할 수 있습니다. 이 때 하나의 문서에는 중복된 아이디가 존재해서는
      안됩니다.</p>
  </body>
</html>
```

## 5. 선택자(Selector)

### ❖ attribute.html

- 실습 폴더에 attribute.html 파일을 생성한 후 다음과 같이 작성하고 브라우저에서 확인해 보자.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>속성 선택자 - Attribute Selector</title>
    <style>
      [title] { text-decoration : underline ; background : #FC9C04 ; }
    </style>
  </head>
  <body>
    <h1>속성 선택자</h1>
    <p title="속성 선택자에 대한 상세 설명 내용입니다.">
      속성 선택자는 요소에 정의 된 속성명이나 속성 값으로 요소를 선택하여
      스타일을 지정할 수 있습니다.</p>
    <p>속성 선택자는 요소에 정의 된 속성명이나 속성 값으로 요소를 선택하여
      스타일을 지정할 수 있습니다.</p>
  </body>
</html>
```



## 5. 선택자(Selector)

### ❖ pseudo-class.html

- 실습 폴더에 pseudo-class.html 파일을 생성한 후 다음과 같이 작성하고 브라우저에서 확인해 보자.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>가상 클래스 선택자 – Pseudo Classes Selector</title>
    <style>
      [title] { text-decoration : underline ; background : #FC9C04 ; }
    </style>
  </head>
  <body>
    <h1>가상 클래스 선택자</h1>
    <a href="https://tacademy.sktechx.com/">T 아카데미</a>
    <p>가상 클래스 선택자는 상황에 따라 요소를 선택하여 스타일을 지정할 수
      있습니다.</p>
  </body>
</html>
```

## 5. 선택자(Selector)

### ❖ pseudo-element.html

- 실습 폴더에 pseudo-element.html 파일을 생성한 후 다음과 같이 작성하고 브라우저에서 확인해 보자.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>가상 요소 선택자 - Pseudo Element Selector</title>
    <style>
      h1::before{ content : " * " ; }
      h1::after{ content : " * " ; }
      p:first-letter{ font-size : 5em ; color : #0000FF ; background : #FFFF00 ; }
    </style>
  </head>
  <body>
    <h1>가상 요소 선택자</h1>
    <p>가상 요소 선택자는 첫 줄, 첫 글자, 특정 요소 앞이나 뒤 등 요소 가상 영역을
      선택하여 스타일을 지정할 수 있습니다.</p>
  </body>
</html>
```

## 5. 선택자(Selector)

### ❖ descendant.html

- 실습 폴더에 descendant.html 파일을 생성한 후 다음과 같이 작성하고 브라우저에서 확인해 보자.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>하위 선택자 - Descendant Combinator</title>
    <style>
      #main div{
        color : #393 ;
        border : 3px solid currentcolor ;
        background : #ff0;
      }
    </style>
  </head>
  .....
```

## 5. 선택자(Selector)

❖ descendant.html

```
<body>
  <h1>하위 선택자</h1>
  <div id="main">
    <p>하위 선택자는 선행 선택자의 하위 요소 중 후행 선택자에 해당하는 요소를
      선택하는 방법입니다.</p>
  </div>
  <p>하위 선택자는 선행 선택자의 하위 요소 중 후행 선택자에 해당하는 요소를
    선택하는 방법입니다.</p>
</body>
</html>
```

## 5. 선택자(Selector)

### ❖ child.html

- 실습 폴더에 child.html 파일을 생성한 후 다음과 같이 작성하고 브라우저에서 확인해 보자.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>자식 선택자 - Child Combinator</title>
    <style>
      ol > li{
        list-style-type : square ;
        font-weight : bold ;
        color : #ff0 ;
      }
    </style>
  </head>
  .....
```

## 5. 선택자(Selector)

❖ child.html

```
<body>
  <h1>자식 선택자</h1>
  <p>선행 선택자인 부모 요소 하위에 포함된 후행 선택자인 자식 요소를 선택하는
    방법입니다.</p>
  <ol>
    <li>첫 번째 목록
      <ul>
        <li>첫 번째 하위 목록</li>
      </ul>
    </li>
    <li>두 번째 목록
      <ul>
        <li>두 번째 하위 목록</li>
      </ul>
    </li>
  </ol>
</body>
</html>
```

## 5. 선택자(Selector)

### ❖ sibling.html

- 실습 폴더에 sibling.html 파일을 생성한 후 다음과 같이 작성하고 브라우저에서 확인해 보자.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>형제 선택자 - Sibling Combinator</title>
    <style>
      h1 + p { color : #f00 ; }
      h2 ~ p { color : #00f ; }
    </style>
  </head>
  <body>
    <h1>기본 형제 선택자</h1>
    <p>기본 형제 선택자는 선행 선택자와 후행 선택자를 "+"로 구분하여
      선언합니다. </p>
    <p>기본 형제 선택자는 선행 선택자와 후행 선택자를 "+"로 구분하여
      선언합니다. </p>
    .....
  </body>
</html>
```

## 5. 선택자(Selector)

❖ sibling.html

```
<h1>인접 형제 선택자</h1>
```

```
<h2>인접 형제 선택자</h2>
```

```
<p>인접 형제 선택자는 선행 선택자와 후행 선택자를 "~"로 구분하여  
선언합니다.</p>
```

```
<p>인접 형제 선택자는 선행 선택자와 후행 선택자를 "~"로 구분하여  
선언합니다.</p>
```

```
</body>
```

```
</html>
```



## 5. 선택자(Selector)

### ❖ group.html

- 실습 폴더에 group.html 파일을 생성한 후 다음과 같이 작성하고 브라우저에서 확인해 보자.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>선택자의 그룹 - Selector Grouping</title>
    <style>
      #heading, .example, p { color : #00f ; }
    </style>
  </head>
  <body>
    <h1 id="heading">선택자 그룹</h1>
    <h2 class="example">선택자 그룹</h2>
    <p>모든 선택자는 콤마(,)를 사용하여 그룹으로 한 번에 선언할 수 있습니다.</p>
  </body>
</html>
```

## 6. 구체성, 겹침

### ❖ 선택자의 구체성(specificity) - 우선순위

- CSS에서는 하나의 요소에 여러 스타일이 겹칠 경우, 선택자의 우선순위에 따라 구체성(specificity)이 높은 선택자의 스타일이 적용된다.

```
<ul id="list" class="note">  
  <li>HTML5</li>  
  <li>CSS3</li>  
  <li>javascript</li>  
  <li>jQuery</li>  
</ul>
```

선택자	A	B	C	구체성	우선순위
li	0	0	1	1	5
ul li	0	0	2	2	4
.note li	0	1	1	11	3
#list li	1	0	1	101	2
ul#list li	1	0	2	102	1

## 6. 구체성, 겹침

### ❖ cascade.html

- 실습 폴더에 cascade.html 파일을 생성한 후 다음과 같이 작성하고 브라우저에서 확인해 보자.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>스타일 겹침에 따른 우선 순위</title>
    <style>
      p { color : #00f ; }
      #main { color : #0f0 ; }
      .note { color : #f00 ; }
    </style>
  </head>
  <body>
    <h1>선택자의 우선 순위</h1>
    <p id="main">하나의 요소에 여러 스타일이 중복 지정될 경우, 선택자의
    우선순위에 따라 구체성(specificity)이 높은 선택자의 스타일이 적용됩니다.</p>
    <p class="note">하나의 요소에 여러 스타일이 중복 지정될 경우, 선택자의
    우선순위에 따라 구체성(specificity)이 높은 선택자의 스타일이 적용됩니다.</p>
  </body>
</html>
```

# CSS3 속성

# 1. 폰트와 텍스트

## ❖ @font-face

- @font-face는 일반적인 속성과 달리 CSS에서 사용할 font-family의 이름과 자원을 정의할 수 있는 규칙으로, 사용자의 환경에 설치된 폰트 또는 제작자가 서버에서 제공한 폰트를 다운로드하여 사용할 수 있도록 해준다.
- @font-face 규칙은 과거 인터넷 익스플로러가 지원했던 기능으로, 지금은 CSS3의 표준으로 자리 잡았다. 그러나 브라우저별로 지원하는 폰트 형식이 다름으로 인해 여러 개의 폰트를 선언해야 하는 단점이 있다.
- 인터넷 익스플로러 6~8의 경우 eot 형식의 글꼴만 지원한다. 그리고 인터넷 익스플로러 9의 경우에는 eot 형식과 함께 woff 형식을 지원한다. 그 밖에 파이어폭스, 크롬, 사파리 그리고 오페라의 경우 woff 형식을 지원한다.

```
@font-face {  
font-family : 'Nanum' ;  
src : url('webfont/NanumGothic.eot') ;  
src : url('webfont/NanumGothic.eot?#iefix') format('embedded-opentype'),  
      url('webfont/NanumGothic.woff') format('woff'),  
      url('webfont/NanumGothic.ttf') format('truetype') ;  
}
```

```
div { font-family : 'Nanum' ; }
```

# 1. 폰트와 텍스트

## ❖ font-family

- font-family는 텍스트의 폰트를 지정하는 속성으로, 속성 값은 복수로 지정할 수 있으며, 지정된 순서대로 표시할 수 있는 폰트를 찾게 된다. 웹 브라우저가 표시할 수 있는 폰트를 찾으면 이후에 지정된 속성 값들은 무시된다.

```
div { font-family : "맑은 고딕", "돋움", sans-serif; }
```

1순위      2순위      Generic-Family

## ❖ font-weight

- font-weight는 폰트가 표시되는 굵기를 지정하는 속성으로, 100~900 사이의 굵기 단계를 표시하는 수치 값이나 normal, bold 등의 키워드로 선언할 수 있다. 이때 굵기 단계의 수치가 높을수록 더 굵게 표시된다. 키워드의 경우 수치 값과 대응되어 "normal"은 수치 값 "400"과 같이 중간 굵기로, "bold"는 수치 값 "700"과 같이 두꺼운 굵기로 표시된다.

```
div { font-weight : 400 ; }  
div { font-weight : bold ; }
```

# 1. 폰트와 텍스트

## ❖ font-weight

- font-weight는 폰트가 표시되는 굵기를 지정하는 속성으로, 100~900 사이의 굵기 단계를 표시하는 수치 값이나 normal, bold 등의 키워드로 선언할 수 있다. 이때 굵기 단계의 수치가 높을수록 더 굵게 표시된다. 키워드의 경우 수치 값과 대응되어 "normal"은 수치 값 "400"과 같이 중간 굵기로, "bold"는 수치 값 "700"과 같이 두꺼운 굵기로 표시된다.

```
div { font-weight : 400 ; }  
div { font-weight : bold ; }
```

## ❖ font-style

- font-style은 폰트의 표시 형태를 지정하는 속성으로, 폰트의 기본 형태로 표시하는 normal 값과 이탤릭 체 또는 기울임꼴로 표시하는 italic과 oblique가 있다. font-style 속성 값이 "italic"으로 지정될 경우 sans-serif는 serif로 font-family가 변경되기도 한다.

```
div { font-weight : normal; }  
div { font-weight : italic; }
```

# 1. 폰트와 텍스트

## ❖ font-size

- font-size는 폰트의 크기를 지정하는 속성이다.

```
div { font-size : 16px ; }  
div { font-size : 1.6em ; }  
div { font-size : 160% ; }  
div { font-size : 1.6rem ; }
```

## ❖ line-height

- line-height 속성은 줄의 높이를 지정하는 속성이다. 그러나 줄의 높이라는 표현보다 줄간격으로 해석하는게 좋다.

한글의 경우 line-height를 100% 이상으로 지정하여 줄과 줄사이의 간격을여유있게 제공하는 편이 가독성 측면에서 도움이 된다.

```
div { line-height : 15px ; }  
div { line-height : 1.5 ; }
```



# 1. 폰트와 텍스트

## ❖ text-transform

- text-transform은 텍스트의 대소 문자를 변환하기 위한 속성으로, 모두 대문자로 변환하는 uppercase, 모두 소문자로 변환하는 lowercase, 그리고 단어의 첫 글자를 대문자로 변환하는 capitalize 값과 함께 CSS3에서 full-width 값이 추가되었다.

```
div { text-transform : capitalize ; }  
div p { text-transform : full-width ; }
```

## ❖ white-space

- white-space는 텍스트의 공백과 줄바꿈의 처리 방법을 지정하는 속성으로 nowrap 값을 지정할 경우 텍스트의 줄바꿈이 무시되며 요소의 너비에 다른 자동 줄바꿈도 적용되지 않는다.

```
div { white-space : nowrap ; }
```

# 1. 폰트와 텍스트

## ❖ text-align

- text-align은 단락 내 텍스트의 가로 방향 정렬 방법을 지정하는 속성이다. 속성 값 중 left, right, center, justify, [string]은 CSS2.1에서도 정의된 속성 값으로 CSS3에서는 좀 더 많은 값이 추가되었다.

```
div { text-align : "." ; } ← 특정 문자를 기준으로 정렬
div p { text-align : center ; }
```

## ❖ text-indent

- text-indent는 단락의 첫 줄 들여쓰기를 지정하기 위한 속성이다. 이때 음수 값을 지정하였을 경우에는 내어 쓰기 효과로 보여진다. 그리고 백분율 값으로 속성 값을 지정하였을 경우에는 부모 요소의 너비 값을 참조하여 들여쓰기 길이를 계산한다.

```
div { text-indent : 2em ; }
div p { text-indent : -50px ; }
```

# 1. 폰트와 텍스트

## ❖ text-decoration

- text-decoration은 텍스트를 장식하는 속성으로, CSS3에서 세분화된 하위 속성이 추가되었다. CSS3에서 추가된 하위 속성을 이용하면 검은색 문장에 빨간색 점선으로 취소선을 작성할 수 있다.

```
div {  
  text-decoration-line : underline ;  
  text-decoration-style : wavy ;  
  text-decoration-color : #f00 ;  
}  
div p{ text-decoration : line-trough dotted #00f; }
```

← CSS3에서 추가 된 방식

## ❖ text-shadow

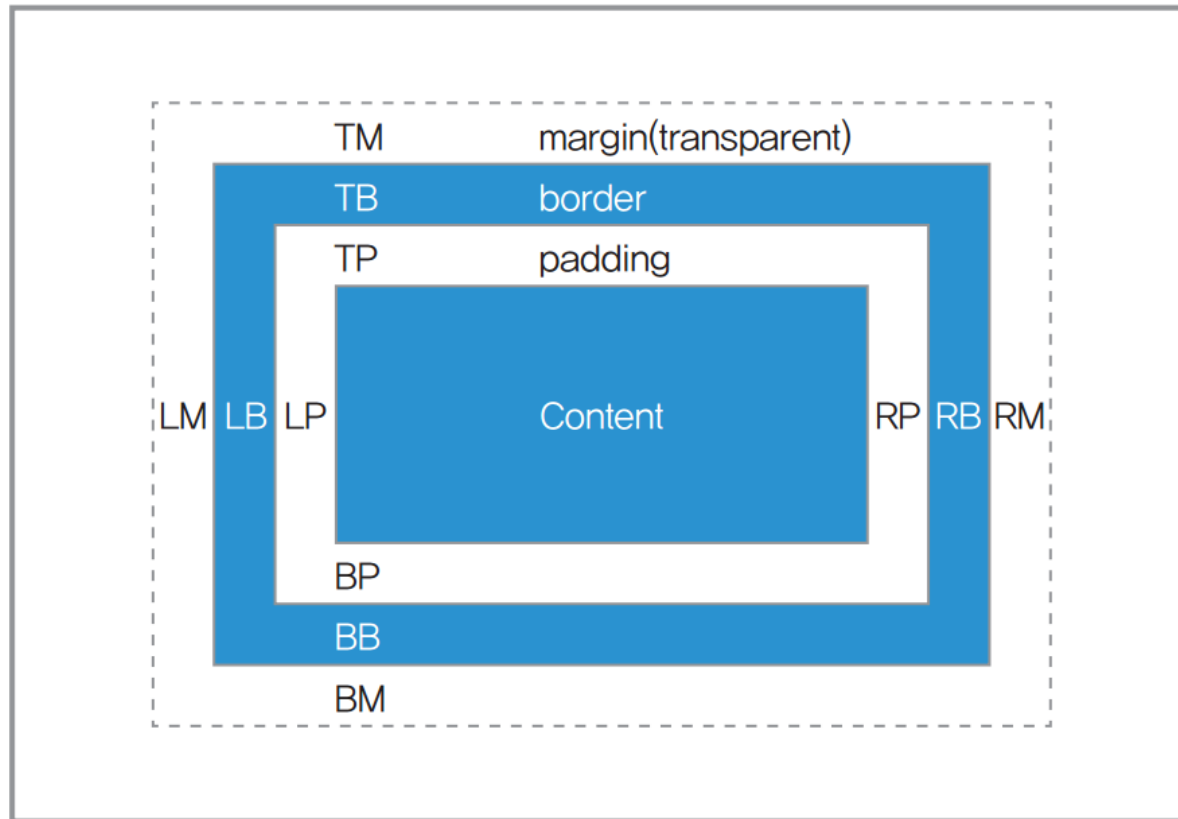
- text-shadow는 텍스트에 그림자를 지정하는 속성이다. 텍스트의 offset-x와 offset-y, blur-radius와 그림자 색상을 지정하여 표현할 수 있으며 콤마로 구분하여 여러 개의 그림자를 겹쳐서 지정할 수도 있다.

```
div { text-shadow : 1px 5px 7px rgb(231,231,231),  
                  0 0 0 rgba(5,77,74,0.8),  
                  1px 5px 7px rgb(231,231,231) ;  
}
```

## 2. 박스 모델

❖ 박스가 실제 화면에서 차지하는 크기를 결정하는 요인

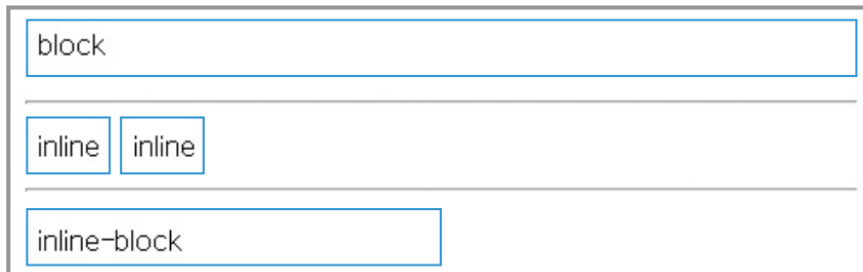
- CSS는 요소를 표시하기 위해 요소별로 사각형의 박스를 생성한다. HTML 요소 박스의 크기는 콘텐츠 크기에 안쪽 여백인 padding과 테두리인 border 그리고 바깥쪽 여백인 margin에 의해 결정된다.



## 2. 박스 모델

### ❖ display

- display는 HTML 요소의 표현 방식을 지정하는 속성으로, 속성 값이 "inline"으로 지정된 요소는 자신의 전후로 줄바꿈을 만들지 않으며, 포함하는 요소가 너비에 의해 자동 줄바꿈 된다. 요소의 폭은 콘텐츠에 따라 결정되며, width 속성으로 요소의 너비를 직접 지정할 수는 없다. "block"으로 지정된 요소는 기본적으로 부모 요소의 너비 전체를 채우며, 요소 박스 전 후로 줄바꿈이 생긴다. 그리고 "inline-block"으로 지정된 요소는 "inline"과 같이 요소 전 후로 줄바꿈이 생기지는 않지만, block 요소처럼 width 속성으로 요소의 너비를 직접 지정할 수 있다. 그리고 none 값을 지정할 경우 존재하지 않는 요소가 된다.



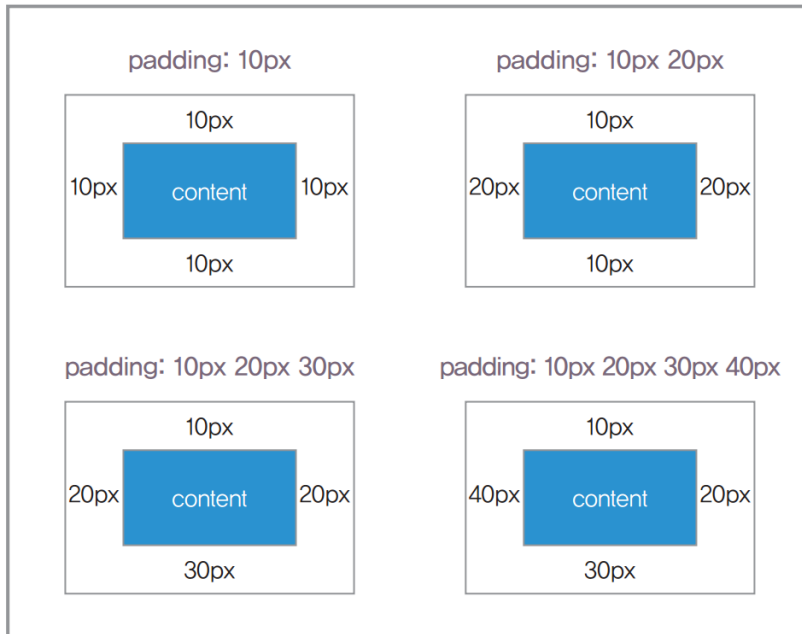
display 속성에 따른 요소 박스의 렌더링 결과

```
span { display : block ; }  
div { display : inline ; }  
span { display : inline-block ; }
```

## 2. 박스 모델

### ❖ padding

- padding은 content 영역과 border 사이의 안쪽 여백을 지정하는 속성으로, 속성 값은 1에서 4개까지 입력할 수 있으며, 입력 값의 개수에 따라 적용되는 방향이 다르다.

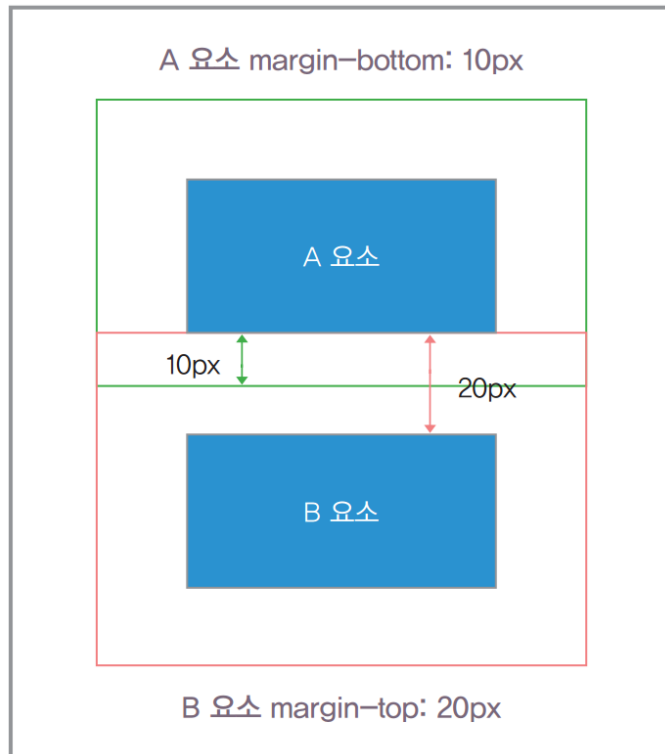


```
div { padding : 10px ; }  
div { padding : 10px 20px ; }  
div { padding : 10px 20px 30px ; }  
div { padding : 10px 20px 30px 40px ; }  
div {  
    padding-top : 10px ;  
    padding-right : 20px ;  
    padding-bottom : 30px ;  
    padding-left : 40px ;  
}
```

## 2. 박스 모델

### ❖ margin

- margin 속성은 border를 기준으로 다른 요소와의 바깥쪽 여백을 지정하는 속성으로 padding 속성과 사용 방법이 동일하다. padding 속성과 다른 점은 음수 값을 사용할 수 있다는 것과 요소 박스가 상하로 인접한 박스의 display 속성 값이 "block"인 경우, 마진 겹침(Margin Collapsing) 현상이 발생한다.



```
div { margin : 10px ; }  
div { margin : 10px 20px ; }  
div { margin : 10px 20px 30px ; }  
div { margin : 10px 20px 30px 40px ; }  
div {  
    margin-top : 10px ;  
    margin-right : 20px ;  
    margin-bottom : 30px ;  
    margin-left : 40px ;  
}
```

## 2. 박스 모델

### ❖ border

- border는 요소 박스의 테두리를 지정하는 속성이다. border 속성은 border-[방향] 등으로 박스의 특정 위치만을 지정할 수 있다. 또한 border-style, border-width, border-color 등의 세부 속성을 사용하여 테두리 선의 모양과 굵기 및 색상을 지정할 수도 있다.

```
div { border : 1px solid #ccc ; }  
div { border-width : 1px ; }  
div { border-style : solid ; }  
div { border-color : #ccc ; }
```

### ❖ width, height, min-width, max-width, min-height, max-height

- width와 height는 요소의 너비와 높이를 지정하는 속성이다. display 속성 값이 "inline"인 요소에는 적용되지 않는다. min과 max로 시작하는 속성은 최소와 최대 크기를 의미한다.

```
div { width : 50% ; }  
div { height : 100px ; }  
div { max-width : 100% ; }  
div { min-height : 100px ; }
```



## 2. 박스 모델

### ❖ overflow

- `overflow`는 콘텐츠가 요소의 콘텐츠 영역을 벗어나는 경우의 처리 방법을 지정하는 속성으로, "`visible`"은 오버 플로우 된 콘텐츠를 그대로 표시한다. 이때 오버 플로우 된 콘텐츠의 영역은 요소의 크기에 반영되지 않는다. "`hidden`"의 경우 오버 플로우 된 콘텐츠를 숨기고 "`scroll`"은 요소 박스에 스크롤 바를 생성한다.

```
div { overflow : auto; }  
div { overflow : hidden; }  
div { overflow : scroll; }
```

## 2. 박스 모델

### ❖ box-model.html

- 실습 폴더에 box-model.html 파일을 생성하고 다음과 같이 작성해보자.

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <title>박스 모델</title>
  <link rel="stylesheet" href="css/box-model.css">
</head>
<body>
  <h1>박스 모델</h1>
  <div class="box01">요소 박스의 크기를 결정하는 것은 width, height,
border,
margin, padding 등의 박스 모델 관련 속성입니다.</div>
  <div class="box02">요소 박스의 크기를 결정하는 것은 width, height,
border,
margin, padding 등의 박스 모델 관련 속성입니다.</div>
</body>
</html>
```

## 2. 박스 모델

### ❖ box-model.css

- 실습 폴더에 하위에 있는 css 폴더에 box-model.css 파일을 생성하고 다음과 같이 작성한 후 웹 브라우저로 확인해보자.

```
/* 모든 p 요소의 배경 색상 지정 */
div{
    width: 200px;
    height: 50px;
    border: 1px solid #000;
    padding: 10px 20px;
    margin: 10px 20px 30px 40px;
    background: #ffc;
}
/* 요소 박스 내 콘텐츠가 실제 크기를 넘칠 경우 넘치는 콘텐츠 숨김 */
.box01{
    overflow: hidden;
}
/* 요소 박스 내 콘텐츠가 실제 크기를 넘칠 경우 넘치는 스크롤 바 생성 */
.box02{
    overflow: auto;
}
```

### 3. 색상 및 배경, 테두리

#### ❖ color

- color는 요소의 전경색을 지정하는 속성이다. 앞에서 살펴본 바와 같이 색상과 관련된 값은 CSS3에서 기존 색상 값인 색의 3원색, 즉 빨강(Red), 초록(Green), 파랑(Blue) 값으로 표현하는 RGB 형식뿐만 아니라 색상(hue), 채도(saturation), 명도(lightness)를 의미하는 HSL 형식과 함께 투명도(alpha) 등도 제공되기 때문에 CSS가 표현할 수 있는 색상의 폭이 더욱 넓어졌다.

```
div { color : rgb(255,0,0) ; }  
div { color : rgba(255,0,0,0.5) ; }  
div { color : hsl(27,88%,58%) ; }  
div { color : hsla(27,88%,58%,0.7) ; }
```

#### ❖ opacity

- opacity는 CSS3에 새롭게 추가된 속성으로 요소 박스의 투명도를 지정하는 속성이다. opacity 속성 값은 "0"에서 "1" 사이의 값을 가지며, "0"은 투명, "1"은 불투명을 의미한다. opacity 속성은 요소 전체에 적용되며, 속성이 상속되기 때문에 포함하는 자식 요소들에게도 영향을 끼친다.

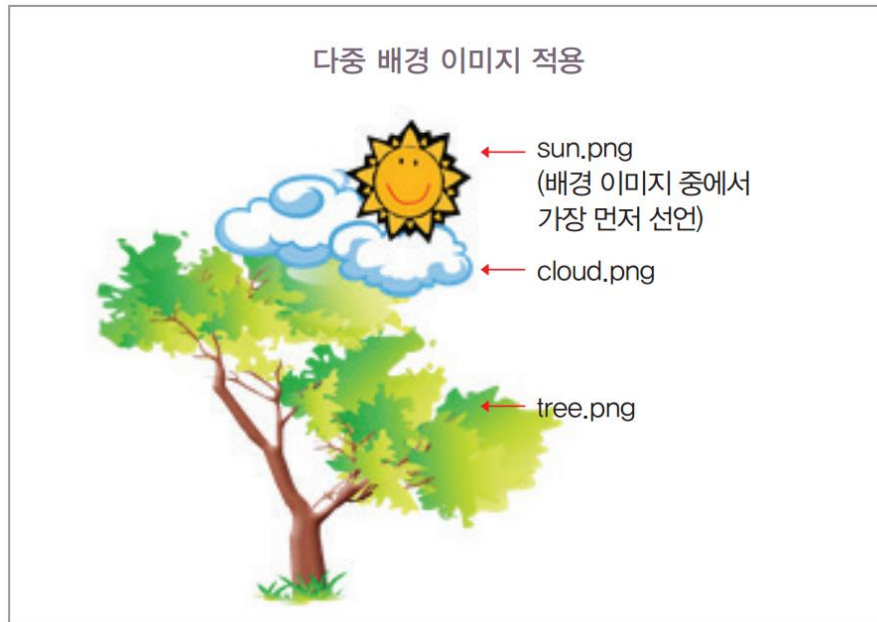
```
div { opacity : 0.5 ; }
```

### 3. 색상 및 배경, 테두리

#### ❖ background-image

- `background-image`는 요소 박스에 배경 이미지를 지정하는 속성이다. CSS3에서는 배경 이미지의 속성 값에 콤마(,)로 구분하여 여러 개의 이미지를 지정함으로써 하나의 요소에 다중 배경 이미지를 지정할 수 있게 되었다. 또한 여러 개의 배경 이미지를 지정할 때 배경 이미지의 겹치는 순서는 가장 먼저 지정한 이미지가 가장 위쪽에 위치한다.

```
div { background-image : sun.png, cloud.png, tree.png ; }
```

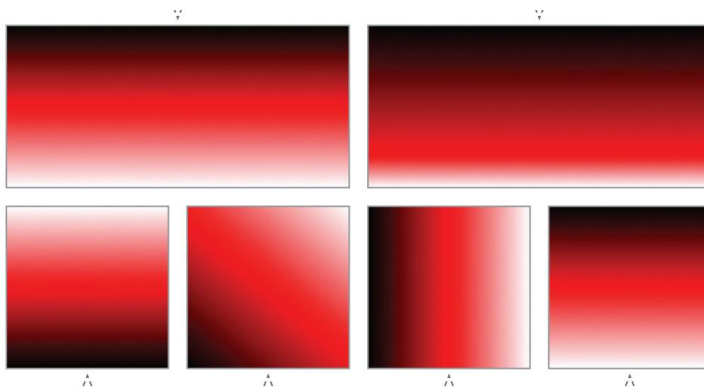


### 3. 색상 및 배경, 테두리

#### ❖ linear-gradient( )

- CSS3에서 background-image 속성의 가장 큰 변화를 꼽자면 요소 박스에 직접 그라데이션 효과를 지정할 수 있게 되었다는 점이다. 이로 인해 이미지 없이 점진적인 색상 표현이 가능하고 확대, 축소 시에도 해상도의 손실 없이 표현할 수 있게 되었다.

```
div {  
  background-image :  
    linear-gradient(rgba(0,0,0,0.5), rgba(255,0, 0,0.5),  
  rgba(255,255,255,0.5) ;  
}  
div { background-image : linear-gradient(45deg, #fff 0%, #000 100%) ; }
```



angle color-stop color-stop

### 3. 색상 및 배경, 테두리

#### ❖ background-repeat

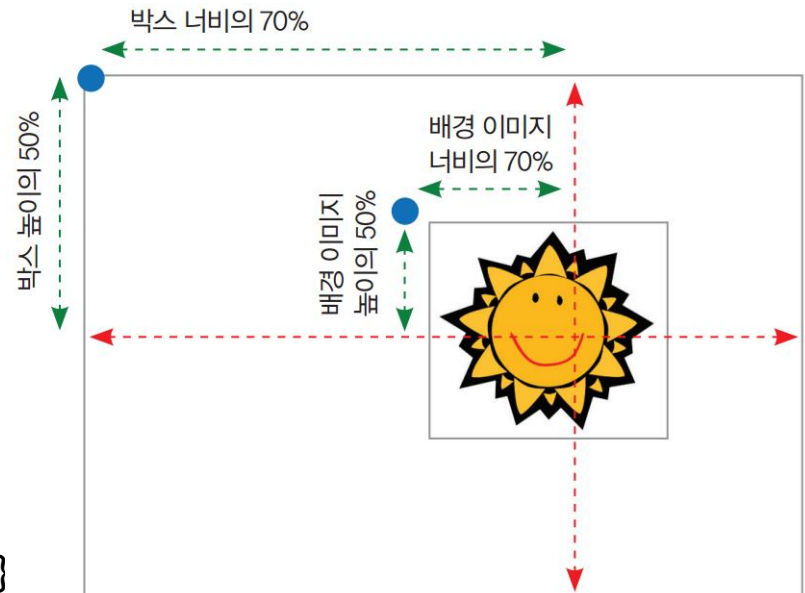
- background-repeat는 배경 이미지의 반복 여부를 지정하는 속성이다. 배경 이미지가 여러 개 지정되었을 경우, background-repeat 속성 값도 콤마(,)로 구분하여 여러 번 지정할 수 있으며, 배경 이미지의 순서대로 반복 스타일을 지정하면 된다.

```
div { background-repeat : no-repeat ; }  
div { background-repeat : space, no-repeat, round ; }
```

#### ❖ background-attachment

- background-position은 배경 이미지의 위치를 지정하는 속성이다.  
background-position 속성 역시 앞서 살펴본 background 관련 속성과 마찬가지로 콤마(,)로 구분하여 여러 번 지정할 수 있다.

```
div { background-position : center ; }  
div { background-position: 70px  
50px ; }  
div { background-position: 70% 50% ; }
```



### 3. 색상 및 배경, 테두리

#### ❖ background-size

- background-size는 CSS3에 새롭게 추가된 속성으로, 배경 이미지의 크기를 지정하는 속성이다.

```
div { background-size : 50px 50px ; }  
div { background-size : 50% 50% ; }  
div { background-size : cover ; }
```

#### ❖ background

- background 속성은 앞서 살펴본 배경 관련 속성을 한번에 지정하는 단축 표기법 형식의 속성이다.

```
div {  
  background : #ffc url(tree.png) no-repeat 100% 100% / 100px 130px ;  
}
```

배경 색상

배경 이미지

배경 이미지 반복 여부

배경 이미지 위치

배경 이미지 크기



### 3. 색상 및 배경, 테두리

#### ❖ border

- border-width, border-style, border-color 속성을 한 번에 지정할 수 있는 단축 표기법 속성이다. border 관련 세부 속성으로 border-[방향]을 통해 각 방향별로도 지정이 가능하다.

```
div { border : 1px solid #000 ; }
```

#### ❖ border-width

- border-width는 요소 박스 테두리의 선 굵기를 지정하는 속성이다. border-width 속성도 padding이나 margin 속성과 같이 속성 값을 1개에서 4개까지 지정할 수 있으며, border 관련 속성은 사용 방법이 모두 같다.

```
div { border-width : 1em ; }
```

```
div { border-width : 5px 10px 10px 5px ; }
```

### 3. 색상 및 배경, 테두리

#### ❖ border-style

- border-style은 요소 박스 테두리의 선 모양을 지정하는 속성이다.

```
div { border-style : solid ; }
```

#### ❖ border-color

- border-color은 요소 박스 테두리의 선 색상을 지정하는 속성이다.

```
div { border-color : #000 ; }
```

#### ❖ border-radius

- border-radius는 요소 박스의 테두리 선을 둥근 모서리 형태로 지정하는 속성이다. 이때 둥근 모서리의 길이는 가로 방향과 세로 방향의 값을 "슬래시(/)"로 구분하여 다르게 지정할 수도 있다.

```
div { border-radius : 10px ; }
```

```
div { border-radius : 5px 10px 15px 20px ; }
```

```
div { border-radius : 5px 10px 15px 20px / 10px 5px 5px 10px ; }
```

### 3. 색상 및 배경, 테두리

#### ❖ box-shadow

- `box-shadow`는 요소 박스의 그림자를 지정하는 속성이다. `text-shadow` 속성 값과 마찬가지로 처음 등장하는 2개의 길이 값은 그림자의 가로 및 세로 방향의 위치를 의미하고, 세 번째 길이 값은 `blur`의 반지름을 의미한다. 그리고 `text-shadow` 속성에는 없었던 네 번째 길이 값은 `spread` 반경으로, 그림자의 확장 영역을 지정할 수 있다. 또한 `inset` 키워드를 사용하여 박스의 안쪽으로 그림자를 표현할 수 있으며, 콤마(,)로 구분하여 여러 개의 박스 그림자를 지정할 수도 있다.

```
div { box-shadow : inset 5px 5px 5px 5px rgba(0, 0, 0, 0.7) ; }
```



### 3. 색상 및 배경, 테두리

#### ❖ background.html

- 실습 폴더에 background.html 파일을 생성하고 다음과 같이 작성해보자.

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>다중 배경 이미지</title>
    <link rel="stylesheet" href="css/background.css">
  </head>
  <body>
    <h1>다중 배경 이미지</h1>
    <p class="box01">
      다중 배경 이미지 적용
    </p>
    <p class="box02">
      테두리 꾸미기
    </p>
  </body>
</html>
```

### 3. 색상 및 배경, 테두리

#### ❖ background.css

- 실습 폴더에 하위에 있는 css 폴더에 background.css 파일을 생성하고 다음과 같이 작성한 후 웹 브라우저로 확인해보자.

```
p {  
  width:200px;  
  height:200px;  
  padding:20px;  
  font-weight:bold;  
  border:1px solid #000;  
}
```

```
.box01{  
  background-color: #ffc;  
  background-image: url(../images/sun.png), url(../images/cloud.png),  
                   url(../images/tree.png);  
  background-size:50px 50px, 100px 50px, 200px 150px;  
  background-position:150px 50px, 100px 70px, right bottom;  
  background-repeat: no-repeat;  
}
```

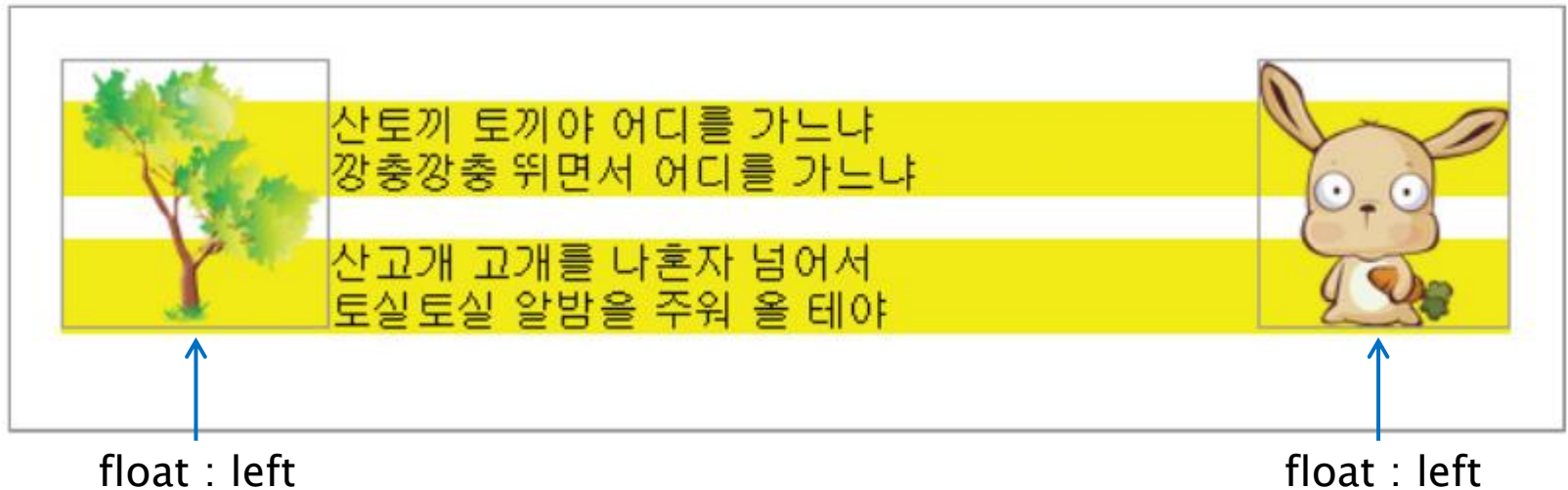
### 3. Float와 Position

#### ❖ float

- float는 일반적인 흐름에서 분리된 요소를 부모 영역을 기준으로 배치하는 속성이다. float 속성 값을 "left"나 "right"로 지정하면, 해당 요소는 일반적인 흐름을 벗어나 부모 요소의 왼쪽이나 오른쪽에 배치된다. 이렇게 float로 지정된 요소는 다른 요소의 배치에 영향을 주게 된다.

```
div { float : left ; }
```

```
div { float : right ; }
```



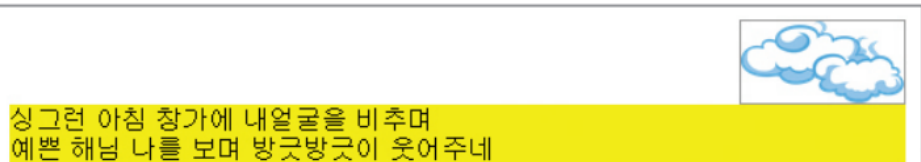
### 3. Float와 Position

#### ❖ clear

- clear는 float 속성의 선언으로 요소의 배치 위치에 영향을 받게 된 경우, 이를 해제하고자 할 때 사용하는 속성이다. 만약 float : left로 지정된 요소로 인해 배치 위치에 영향을 받은 요소의 흐름을 해제할 경우에는 "left" 속성 값을, float : right로 지정된 요소로 인해 배치 위치에 영향을 받은 요소의 흐름을 해제할 경우에는 "right" 속성 값을 사용할 수 있으며, "both"를 지정했을 경우에는 float 방향과 상관없이 해제된다.

```
div { clear: both ; }
```

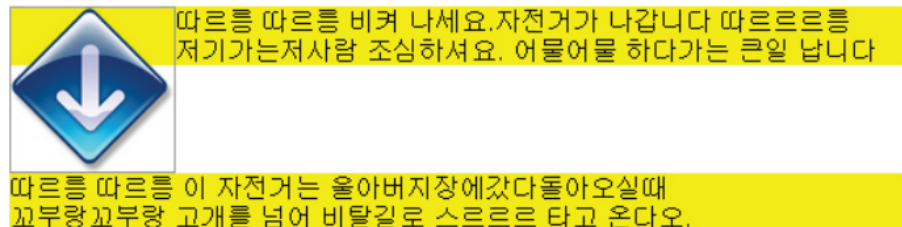
clear 속성이 지정된 요소 →



clear 속성이 지정된 요소 →



clear 속성이 지정된 요소 →



### 3. Float와 Position

#### ❖ position

- `position`은 요소 박스의 배치 방식을 지정할 수 있는 속성이다. 요소 박스는 기본 값인 `static`으로 정의되어 있다. `static`은 일반적인 흐름에 따른 배치 방식을 말하며, 마크업 순서대로 선형화되어 배치된다.
- 이 밖에 `relative`는 일반적인 요소의 흐름 순서에 따라 배치되지만, 본래 위치를 기준으로 `offset` 관련 속성으로 지정한 위치만큼 이동하여 배치된다.
- `absolute`는 일반적인 흐름을 벗어나 레이어처럼 다른 요소 위에 겹쳐 배치되는 방식을 의미한다. 이때 위치 지정은 `relative`와 동일하게 `offset` 관련 속성에서 지정한 만큼 이동한다. 그러나 `absolute`는 본래 위치를 기준으로 이동하는 `relative`와 달리 `absolute`로 지정된 요소를 포함하는 상위 요소 중 `position` 속성 값이 `static`이 아닌 요소(`position`의 속성 값이 `relative`, `absolute`, `fixed` 중 하나인 경우) 중에서 가장 가까운 요소를 기준으로 배치된다는 점이 다르다. 또한 `absolute` 속성 값이 적용된 요소는 블록 요소가 되며, 기본 레이아웃에 영향을 끼치지 않는다.
- `fixed`는 Viewport를 기준으로 `offset` 관련 속성에서 지정한 만큼 이동하여 배치된다.



### 3. Float와 Position

#### ❖ top, right, bottom, left

- top, right, bottom, left 속성은 position 속성 값이 static이 아닌 경우, 해당 요소의 위치를 지정할 때 사용할 수 있는 offset 관련 속성이다.

```
div {  
    position : absolute ;  
    top : 100px ;  
    left : 100px ;  
}
```

#### ❖ z-index

- z-index는 position 속성 값이 static이 아닌 요소의 offset 관련 속성 값과 겹쳤을 경우, 겹치는 순서를 지정하는 속성이다. 기본 값은 auto이며, 마크업 순서에 따라 가장 나중에 선언된 요소 박스가 맨 위에 배치된다. number 속성 값의 경우 단위가 없는 숫자로 작성하며, 높은 숫자가 지정된 요소가 맨 위에 배치되며 z-index 속성 값으로 음수 값도 사용할 수 있다.

```
div { z-index : 3 ; }  
div { z-index : -1 ; }
```

## 4. 애니메이션 관련 속성

### ❖ transform

- transform은 요소 박스를 변형하는 속성으로, 2차원 및 3차원 변형이 가능하며 변형 형태별로 함수 타입의 속성 값을 지정한다. 이때 속성 값을 공란으로 구분하여 복수의 속성 값을 지정할 수 있다.
- transform 속성 값으로 translate( ) 함수는 HTML 요소 박스를 평면상에서 수평 이동하는 기능이다. translation-value 값에는 요소가 변형 기준점으로부터 가로 및 세로 방향으로 이동하는 길이 값을 지정할 수 있다. translateX( )와 translateY( ) 함수로 이동거리를 방향 별로 각각 지정할 수 있다. translate( )로 이동 된 요소는 다른 요소의 배치에 영향을 끼치지 않는다.

```
div { transform : translate(50px, 50px) ; }  
div { transform : translateX(50px) ; }  
div { transform : translateY(50px) ; }
```

- transform 속성 값으로 scale( ) 함수는 HTML 요소 박스의 크기를 변형하는 기능이다.  
translate( ) 함수와 마찬가지로 X, Y 값 을 지정하여 가로 및 세로 크기를 조절할 수 있다.

```
div { transform : scale(50px, 50px) ; }
```

## 4. 애니메이션 관련 속성

### ❖ transform

- transform 속성 값으로 rotate( ) 함수는 HTML 요소 박스를 평면상에서 회전하는 기능으로, 함수의 값으로 회전 각도(deg)를 지정할 수 있다. 이때 회전 각도가 양수 값일 경우에는 시계 방향으로 회전하고, 음수 값일 경우에는 반 시계 방향으로 회전한다.

```
div { transform : rotate(45deg) ; }
```

- transform 속성 값으로 skew( ) 함수는 HTML 요소 박스의 기울임을 지정하는 기능이다. rotate( ) 함수와 마찬가지로 함수 값에 기울 기의 각도(deg)를 지정할 수 있으며, skewX( )와 skewY( )로 각 축 방향 별 요소의 기울기를 지정할 수도 있다.

```
div { transform : skew(45deg) ; }
```

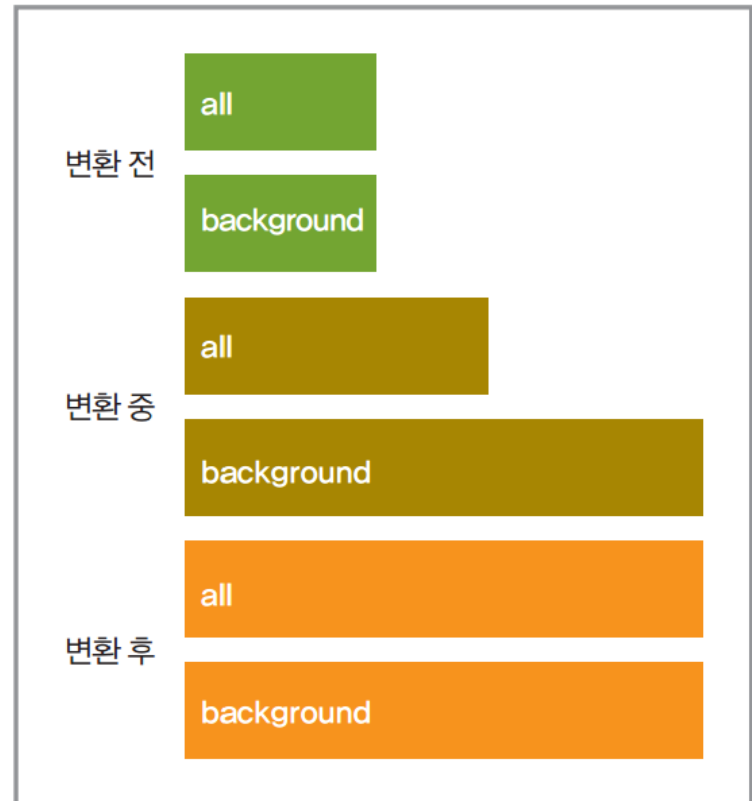
- transform 속성 값으로 matrix( ) 함수는 HTML 요소 박스에 이동 및 회전, 크기 변화, 기울임 등을 복합적으로 적용할 수 있는 기능이다. 함수 값은 3×3 값의 행렬로, 앞서 살펴본 변형 관련 함수 값을 하나 혹은 둘 이상 조합하여 사용할 수 있다.

```
div { transform : matrix(0.5, 0.2, 0.3, 0.5, 50, 100) ; }
```

## 4. 애니메이션 관련 속성

### ❖ transition-property

- transition-property는 요소에 지정된 속성을 변환하고자 할 때 사용하는 속성이다. 속성 값으로 특정 CSS 속성을 지정하면 해당 속성만 변환되고, all을 지정할 경우 요소에 지정된 모든 속성이 변환된다.
- 2개 이상의 속성을 지정할 경우, 콤마(,)로 구분하면 여러 개를 지정할 수 있다.  
이때 속성의 변환이 점진적으로 진행되는 상황을 보여주고자 할 경우에는 뒤에서 살펴볼 transition-duration 속성을 사용하여 변환 시간을 함께 명시해야 한다.



## 4. 애니메이션 관련 속성

### ❖ transition-duration

- transition-duration은 변환이 진행되는 시간을 지정하는 속성이다.  
transition-property 속성 값을 콤마(,)로 구분하여 여러 개로 지정했을 경우 transition-duration 속성 값 역시 콤마(,)로 구분하여 변화되는 속성별로 시간을 지정할 수 있다. 이때 시간은 초 단위(s)로 지정한다.

```
div {  
  margin-bottom : 20px;  
  width : 100px ;  
  height : 50px ;  
  border : #ccc solid 5px ;  
  background : rgba(5,77,74,1) ;  
  transition-property : width, background ;  
  transition-duration : 1s, 2s ;  
}
```

```
div:hover {  
  width : 400px ;  
  background : rgba(255,100,0,0.8) ;  
}
```

← 마우스를 올리면 전이가  
진행되도록 하기 위해서  
:hover 선택자가 아닌  
초기값을 가진 선택자에  
선언하여야 함

## 4. 애니메이션 관련 속성

### ❖ transition-timing-function

- transition-timing-function은 속성이 변환될 때 진행 속도의 형태를 지정하는 속성으로 가속과 감속을 조정하여 좀 더 부드럽게 진행되도록 해주는 기능이다.

```
div{ transition-timing-function : ease-in-out ; }
```

### ❖ transition-delay

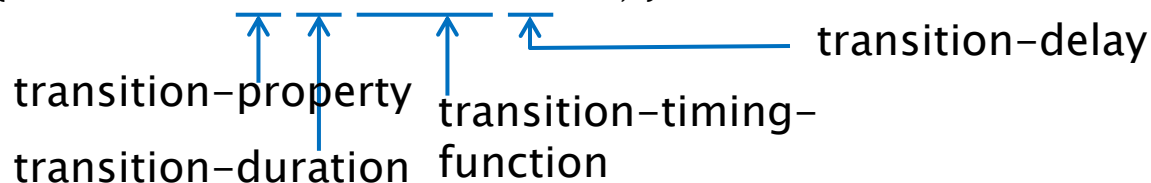
- transition-delay는 전이가 진행되기 전 지연시간을 지정하는 속성이다.

```
div{ transition-delay : 1s ; }
```

### ❖ transition

- transition은 transition 관련 속성 값들을 일괄 적용하는 대표 속성으로, transition 속성 값은 공백으로 구분하여 선언한다. 하나 이상의 그룹을 선언할 경우에는 이를 콤마(,)로 구분하여 지정할 수 있다.

```
div { transition : all 3s ease-in 2s ; }
```



## 4. 애니메이션 관련 속성

### ❖ @keyframes

- @keyframes는 animation 속성에 적용할 키 프레임을 생성하기 위한 규칙이다. 키 프레임이란, 애니메이션을 구현할 때 기준이 되는 특정 시점으로, 각 키 프레임 사이의 애니메이션은 자동으로 구현된다.
- @keyframes 규칙의 구문은 다음과 같다.

```
@keyframes 애니메이션 이름 {  
  from {  
    [ CSS 속성 : 값 ; ]  
  }  
  [percentage]{  
    [ CSS 속성 : 값 ; ]  
  }  
  to {  
    [ CSS 속성 : 값 ; ]  
  }  
}
```

- 애니메이션 이름은 @keyframes 규칙으로 생성된 애니메이션 정보를 대표하는 이름으로, 사용자가 정의하여 선언한다.
- @keyframes 규칙 내부에서 from{ }은 시작 지점을 의미하며, 0%로 선언할 수도 있다. 이때 특정 시점에서 변화하는 선언 값을 지정하고자 할 경우, 중간 단계를 백분율(percentage)로 지정할 수 있다. to{ } 구문은 애니메이션의 종료 지점이다.

## 4. 애니메이션 관련 속성

### ❖ animation-name

- animation-name 속성은 @keyframes 규칙으로 생성한 애니메이션 이름을 지정하여 해당 애니메이션 이 실행되도록 하는 속성이다. 이때 실행하고자 하는 애니메이션 이 여러 개인 경우 애니메이션 이름을 콤마(,)로 구분하여 지정할 수 있다.

```
div{ animation-name: main-ani ; }
```

### ❖ animation-duration

- animation-duration은 animation-name 속성으로, 실행된 애니메이션 진행 시간을 지정하는 속성이다. 지정할 수 있는 속성 값인 time은 초 단위(s)로 1회 진행 시간을 의미한다.

```
div{ animation-duration: 1s ; }
```

### ❖ animation-timing-function

- 앞서 살펴본 transition-timing-function과 동일하다. 가속과 감속을 위한 속성이다.

```
div{ animation-timing-function : ease-in-out ; }
```



## 4. 애니메이션 관련 속성

### ❖ animation-iteration-count

- animation-iteration-count는 애니메이션의 반복 횟수를 지정하는 속성으로, infinite를 지정한 경우 애니메이션이 무한 반복되며, number 값으로 직접 반복 횟수를 지정할 수 있다.

```
div{ animation-iteration-count: infinite ; }
```

- div{ animation-iteration-count: 2 ; }

### ❖ animation-direction

- animation-direction은 애니메이션의 진행 방향을 지정하는 속성으로, 기본 값은 normal이며, normal은 순방향으로 진행된다. 이때 애니메이션의 방향을 역방향으로 지정하고자 할 때는 reverse를 사용하면 된다.

```
div{ animation-direction: alternate ; }
```

### ❖ animation-play-state

- animation-play-state는 애니메이션의 진행 및 정지 상태를 지정하는 속성으로, running은 애니메이션이 진행된 상태대로 표시하는 것을 말하며, paused는 애니메이션이 정지된 상태대로 표시하는 것을 의미한다.

```
div{ animation-timing-function : ease-in-out ; }
```

## 4. 애니메이션 관련 속성

### ❖ animation-delay

- animation-delay는 애니메이션이 실행되기 전 지연 시간을 초 단위(s)로 지정하는 속성으로, 앞서 살펴 본 transition-delay 속성과 동일하다.

```
div{ animation-delay : 2 ; }
```

### ❖ animation-fill-mode

- animation-fill-mode는 애니메이션 실행 이전이나 이후에 애니메이션 효과를 표시할지의 여부를 지정 하는 속성이다. 애니메이션 실행 이전의 지연 시간에는 애니메이션에서 지정한 속성이 적용되지 않는 데, 이 부분을 animation-fill-mode 속성을 통해 정의할 수 있다. animation-fill-mode 속성 값으로 forwards를 지정한 경우에는 애니메이션의 마지막 키 프레임에 선언 된 CSS 속성이 애니메이션 실행이 종료된 후에 표시되고, backwards를 지정한 경우에는 첫 번째 키 프레임에 선언된 CSS 속성이 애니메이션이 시작하기 전이나 지연 시간에 표시되는 것을 의미한다. 이때 both 값을 지정하면 양쪽에 모두 적용된다.

```
div{ animation-fill-mode: backwards ; }
```

## 4. 애니메이션 관련 속성

### ❖ animation

- animation은 애니메이션과 관련된 속성 값들을 일괄 지정하기 위한 대표 속성이다.

```
div { animation : my-animation 5s ease-in-out 2s infinite alternate  
backwards ; }
```

animation-  
name

animation-  
duration

animation-timing-  
function

animation-delay

animation-iteration-  
count

animation-direction

animation-fill-  
mode

## 4. Flex Box

### ❖ Flexible Box Layout

- 웹이 표시되는 기기의 화면 크기가 다양해지면서 요소의 배치도 유연해질 필요성이 생겼다. CSS3에는 기존의 float나 position 속성을 바탕으로 한 배치 방법과는 다른 방식으로 요소를 배치할 수 있는 새로운 방법이 추가되었는데 Flexible Box Layout이 그것이다. 이제부터 Flex와 관련된 다양한 속성들을 살펴보도록 하자.

### ❖ display : flex, display : inline-flex

- CSS3에 추가된 유연한 박스 레이아웃을 사용하려면 가장 먼저 기존에 사용했던 display 속성에 추가된 새로운 값인 flex 또는 inline-flex를 지정해야 한다.
- 두 값의 차이점은 블록 수준의 유연한 박스를 생성할 것인지, 인라인 수준의 유연한 박스를 생성할 것인지를 결정할 수 있다는 것이다.

```
div { display : flex ; }
```

### ❖ flex-direction

- flex-direction은 요소 박스의 배치 방향을 지정하는 속성이다. 해당 속성은 요소 박스가 반드시 display 속성 값으로 flex 또는 inline-flex로 지정되어 있어야 하며, row, column, rowreverse, column-reverse 등의 속성 값을 지정하여 flex 박스에 포함된 자식 요소 박스를 가로 또는 세로 방향이나 역방향으로 배치한다.

```
div { flex-direction : row ; }
```

## 4. Flex Box

### ❖ Flexible Box Layout

- 웹이 표시되는 기기의 화면 크기가 다양해지면서 요소의 배치도 유연해질 필요성이 생겼다. CSS3에는 기존의 float나 position 속성을 바탕으로 한 배치 방법과는 다른 방식으로 요소를 배치할 수 있는 새로운 방법이 추가되었는데 Flexible Box Layout이 그것이다. 이제부터 Flex와 관련된 다양한 속성들을 살펴보도록 하자.

### ❖ display : flex, display : inline-flex

- CSS3에 추가된 유연한 박스 레이아웃을 사용하려면 가장 먼저 기존에 사용했던 display 속성에 추가된 새로운 값인 flex 또는 inline-flex를 지정해야 한다.
- 두 값의 차이점은 블록 수준의 유연한 박스를 생성할 것인지, 인라인 수준의 유연한 박스를 생성할 것인지를 결정할 수 있다는 것이다.

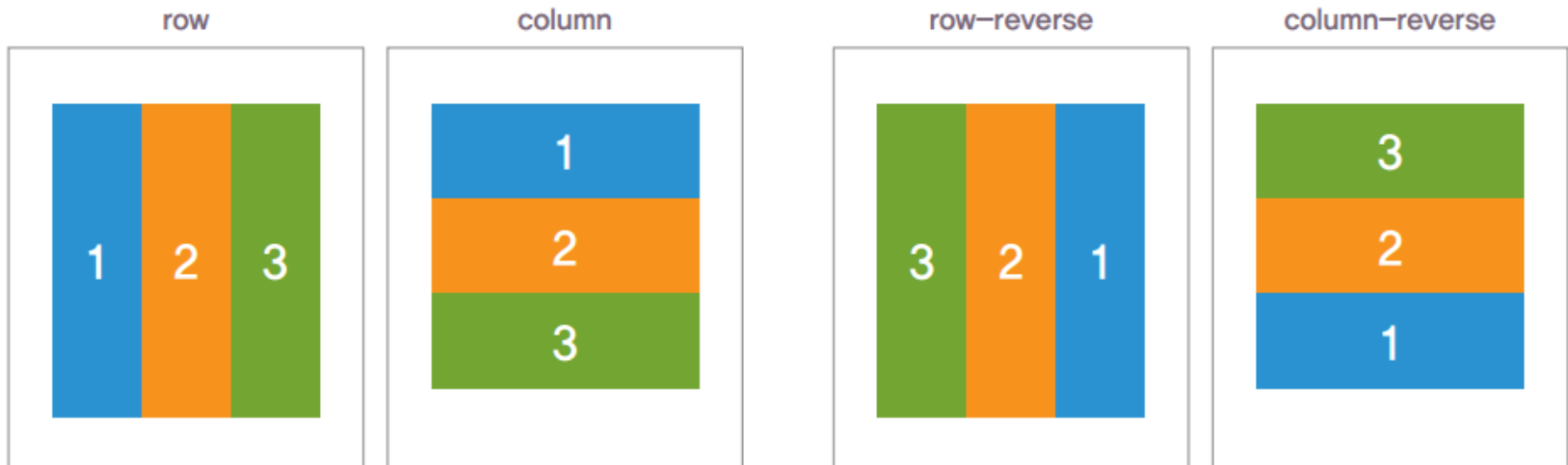
```
div { display : flex ; }
```

## 4. Flex Box

### ❖ flex-direction

- flex-direction은 요소 박스의 배치 방향을 지정하는 속성이다. 해당 속성은 요소 박스가 반드시 display 속성 값으로 flex 또는 inline-flex로 지정되어 있어야 하며, row, column, rowreverse, column-reverse 등의 속성 값을 지정하여 flex 박스에 포함된 자식 요소 박스를 가로 또는 세로 방향이나 역방향으로 배치한다.

```
div { flex-direction : row ; }
```

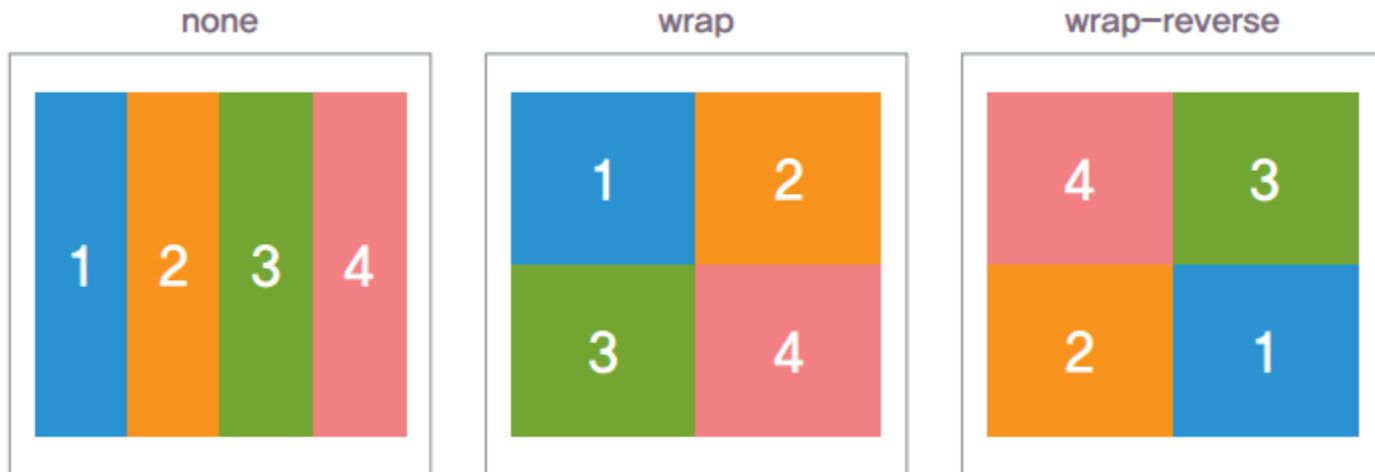


## 4. Flex Box

### ❖ flex-wrap

- flex-wrap은 flex로 지정된 요소에 포함된 자식 요소 박스가 한 줄 또는 여러 줄로 배치될 것인지를 지정하는 속성으로, 초기값 none은 모든 자식 요소를 하나의 단일 행이나 열에 표시한다. 만약 flex-wrap 속성 값을 wrap으로 지정할 경우, 모든 자식 요소는 여러 줄로 배치된다. wrap-reverse는 역방향으로 여러 줄을 배치한다.

```
div { flex-wrap : nowrap; }
```



## 4. Flex Box

### ❖ flex-flow

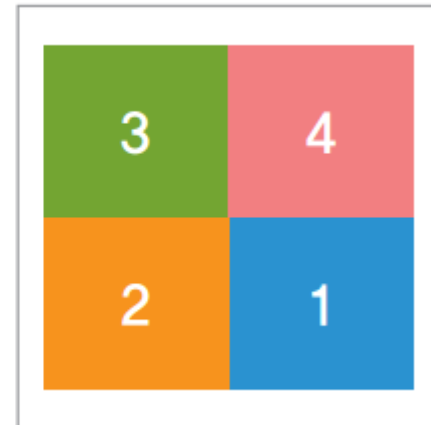
- `flex-flow`은 `flex-direction` 속성과 `flex-flow` 속성을 일괄적으로 지정할 수 있는 단축 표기법 즉 대표 속성이다.

```
div { flex-flow : row wrap ; }
```

### ❖ order

- `order`는 요소 박스의 배치 순서를 지정하기 위한 속성이다. 속성 값에 지정된 숫자 중 가장 낮은 번호의 요소가 가장 먼저 배치된다. 만약 요소 박스의 `order` 값이 동일하게 지정된 경우에는 일반적인 흐름에 따라 먼저 배치된 요소가 우선한다.

```
div { order : 3 ; }
```



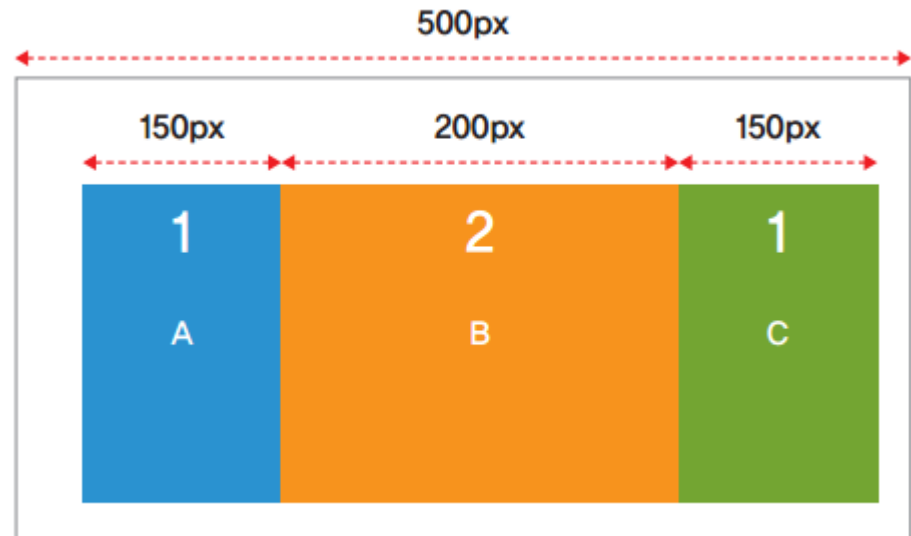


## 4. Flex Box

### ❖ flex-grow

- flex-grow 속성은 flex로 정의된 부모 요소가 자식 요소보다 클 경우, 자식 요소의 크기를 조정하는 방법으로 자식 요소에 확대 비율을 지정하여 부모 요소의 크기에 맞게 자동으로 크기를 늘려서 조정한다. 이때 속성 값에는 음수 값을 사용할 수 없다.

```
div {  
  display : flex ;  
  flex-direction : row ;  
}  
.box1 {  
  width : 100px ;  
  flex-grow : 1 ;  
}  
.box2 {  
  width : 100px ;  
  flex-grow : 2 ;  
}  
.box3 {  
  width : 100px ;  
  flex-grow : 1 ;  
}
```

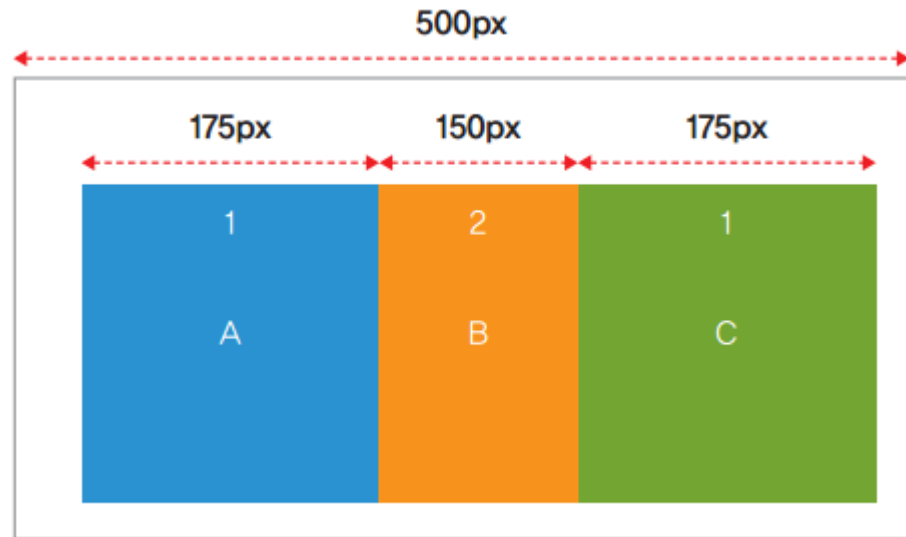


## 4. Flex Box

### ❖ flex-shrink

- flex-grow 속성은 flex로 정의된 부모 요소가 자식 요소보다 작을 경우, 자식 요소의 크기를 조정하는 방법으로 자식 요소에 축소 비율을 지정하여 부모 요소의 크기에 맞게 자동으로 크기를 줄여서 조정한다. 이때 속성 값에는 음수 값을 사용할 수 없다.

```
div {  
  display : flex ;  
  flex-direction : row ;  
}  
.box1 {  
  width : 200px ;  
  flex-shrink : 1 ;  
}  
.box2 {  
  width : 200px ;  
  flex-shrink : 2 ;  
}  
.box3 {  
  width : 200px ;  
  flex-shrink : 1 ;  
}
```

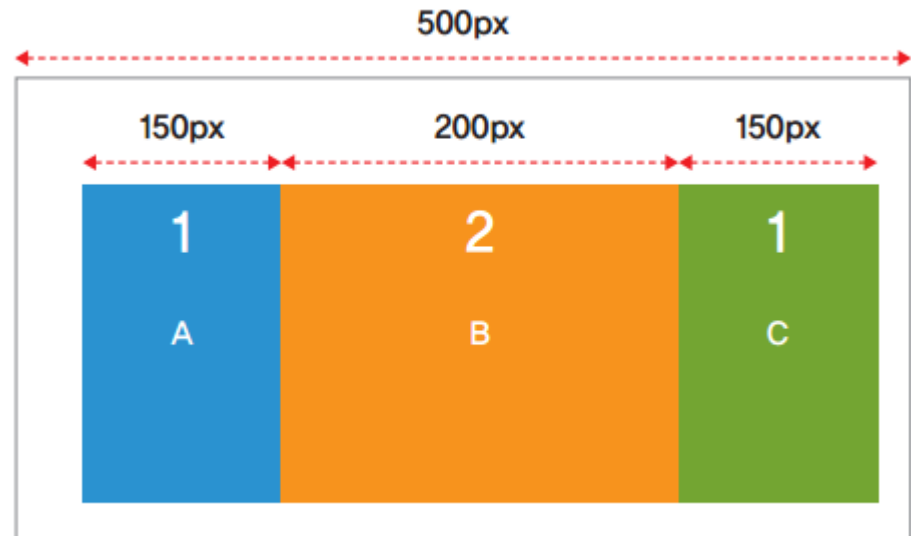


## 4. Flex Box

### ❖ flex-basis

- flex-basis는 유연하게 조절되는 박스의 기준 크기를 지정하는 속성으로, width 속성에서 사용할 수 있는 값과 동일하다. 만일 width 속성과 flex-basis 속성이 함께 선언된 경우, flex로 정의된 요소는 flex-basis속성 값으로 적용된다.

```
div {  
  display : flex ;  
  flex-direction : row ;  
}  
.box1 {  
  width : 200px ;  
  flex-basis : 100px ;  
  flex-grow : 1 ;  
}  
.box2 {  
  width : 100px ;  
  flex-grow : 2 ;  
}  
.box3 {  
  width : 100px ;  
  flex-grow : 1 ;  
}
```

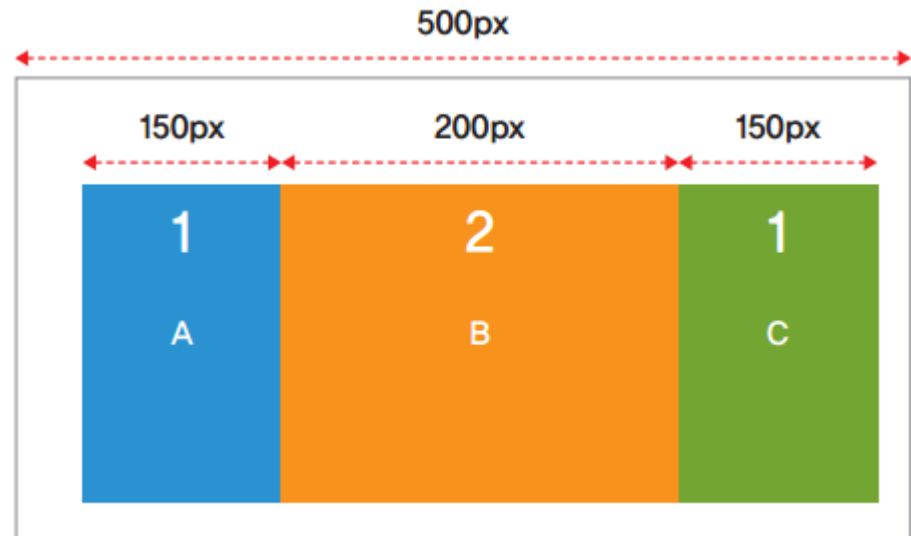


## 4. Flex Box

### ❖ flex

- flex는 flex-grow 속성과 flex-shrink 속성 그리고 flex-basis 속성을 일괄적으로 지정할 수 있는 대표 속성이다.

```
div {  
  display : flex ;  
  flex-direction : row ;  
}  
.box1 {  
  flex : 1 1 100px ;  
}  
.box2 {  
  flex : 2 1 100px ;  
}  
.box3 {  
  flex : 1 1 100px ;  
}
```



## 4. Flex Box

### ❖ justify-content

- justify-content는 flex로 지정된 부모 요소 내에서 자식 요소의 정렬 방식을 지정하기 위한 속성이다. 해당 정렬은 flex-direction 속성 값이 행 방향이나, 열 방향이나에 따라 start와 end 영역이 결정된다. 만약 자식 요소가 행 방향으로 정의되어 있을 경우 flex-start는 부모 요소의 좌측, flex-end는 부모 요소의 우측을 의미한다. 열 방향일 경우에는 상단과 하단이 각각 flex-start와 flex-end로 지정된다.

```
div {  
  display : flex ;  
  justify-content : space-between ;  
}
```

### ❖ align-content

- align-content는 flex로 지정된 부모 요소 내에서 여러 개의 행이나 열로 구성된 요소 박스의 정렬을 지정하기 위한 속성이다.

```
div {  
  display : flex ;  
  flex-wrap : wrap ;  
  align-content : space-between ;  
}
```

## 4. Flex Box

### ❖ align-items, align-self

- align-items, align-self 속성은 라인 박스의 높이에 따라 배치되는 요소 박스에 배치에 영향을 주는 교차축(cross axis)을 지정하는 속성이다. flex-direction 속성 값은 row 또는 column에 따라 요소 박스가 배치되는데, 이때 요소 박스의 너비나 높이가 기준 축으로 정해진다. 전체 요소의 기준 축을 동일하게 지정할 때는 align-items 속성을 flex가 정의된 부모 요소에 지정해야 한다. 만약 자식 요소마다 기준 축을 지정해야 할 경우에는 align-self 속성으로 지정한다.

```
div {  
  display : flex ;  
  flex-direction : row ;  
  align-items : center ;  
}
```

```
div {  
  display : flex ;  
  flex-direction : row ;  
}  
div p{  
  align-self : center ;  
}
```

## 5. 사용자 인터페이스

### ❖ content

- content 속성은 가상 요소 선택자인 :before나 :after를 선택자로 지정한 후, 특정 요소 앞이나 뒤에 콘텐츠를 추가할 수 있도록 하는 기능이다. content 속성에는 텍스트를 포함하여 이미지나 특정 규칙에 따른 값을 지정할 수 있다.

```
body {  
  counter-reset: chapter;  
}  
h1:before {  
  content: "Chapter " counter(chapter) ". ";  
  counter-increment: chapter;  
}  
h1 {  
  counter-reset: section;  
}  
h2:before {  
  content: counter(chapter) "." counter(section) " ";  
  counter-increment: section;  
}
```

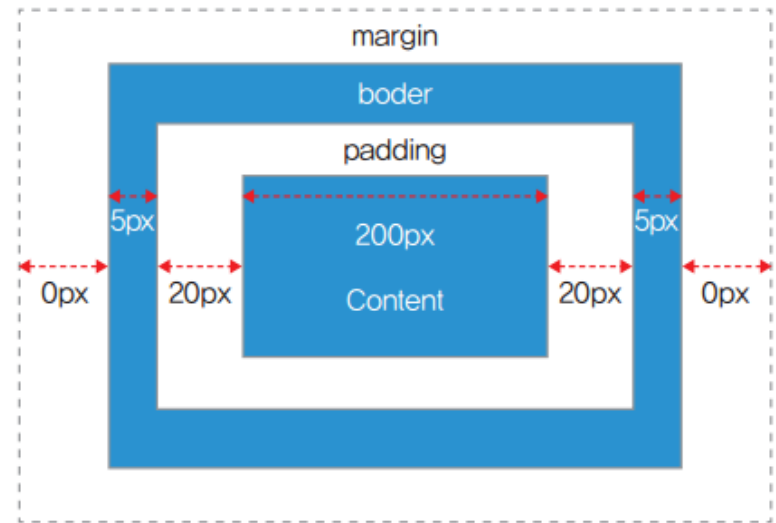
## 5. 사용자 인터페이스

### ❖ box-sizing

- `box-sizing`은 요소 박스의 크기를 결정하는 방식을 지정하는 속성이다. 앞서 살펴본 박스 모델에 따라 박스의 실제 크기는 `margin`과 `border` 그리고 `padding` 값에 콘텐츠의 크기를 더해 결정된다. 그러나 `box-sizing` 속성을 사용하면 박스의 실제 크기를 `width` 값에 `border`와 `padding` 값을 포함하여 렌더링 할 수 있다.

오른쪽 요소 박스의 경우 기본 박스 모델에 따르면 실제 박스의 가로 크기는 250px이다. 그러나 `box-sizing` 속성 값을 `border-box`로 지정하면 200px로 렌더링 된다.

```
div{  
  box-sizing: border-box ;  
}
```



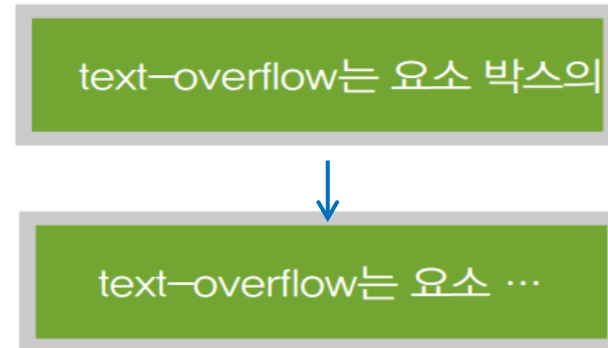


## 5. 사용자 인터페이스

### ❖ text-overflow

- `text-overflow`는 요소 박스의 너비가 고정되어 있고, 요소 박스 내 텍스트가 박스의 크기를 넘는 경우 overflow된 텍스트를 처리하는 방식을 지정하는 속성이다. 속성 값으로 `clip`과 `ellipsis`가 있으며, `clip`은 overflow된 텍스트를 잘라내고, `ellipsis`는 overflow된 텍스트를 숨기고 뒤에 말줄임(...) 표시를 나타낸다. 이때 주의해야 할 점은 `text-overflow` 속성을 지정하려면 요소 박스의 너비가 고정 값으로 지정되어야 하고 줄 바꿈이 되지 않도록 `white-space` 속성 값을 `nowrap`으로 지정해야 한다.

```
div {  
  width : 200px ;  
  white-space : nowrap ;  
  overflow : hidden ;  
  text-overflow : ellipsis ;  
}
```



### ❖ cursor

- `cursor`는 HTML 요소 위에 마우스가 오버될 때 마우스 포인터의 모양을 지정하는 속성이다.

```
div { cursor : pointer; }
```