

HTML5와 CSS3 그리고 API

2018.8.20 ~ 8.24

백명숙



과정개요

■ HTML5는 데이터 구조(HTML)를 다루고 공유하는 방식으로 웹 표준을 채택하고 이를 기반으로 한 표현과 동작(CSS/DOM/Javascript), 상호작용(Javascript,Ajax) 및 공유(XML,JSON) 방식을 효과적으로 다룰 수 있는 기술입니다. 그리고, 웹 표준의 미래 지향적인 가치를 판단하고자 하며, 웹 브라우저가 기존 HTML보다 HTML5, CSS3, Javascript API를 지원하게 된다면 지금 웹사이트를 구현할 때 상호 호환성을 확보하는 것이 중요합니다. 이러한 시대의 웹 표준에 대한 연결 고리를 찾아보고 실무에 바로 적용할 수 있는 능력을 본 강좌에서 다루고자 합니다.

과정목표

- HTML5, CSS3, HTML5 API를 사용하여 웹 개발 업무에 바로 적용할 수 있는 능력을 키우는 것이 이 과정의 목표입니다.

러닝 맵



강사 프로필

성명	백명숙
소속 및 직함	휴클라우드 이사
주요 경력	일은시스템(제일은행 IT자회사), Sun MicroSystems 교육서비스, 인터넷커머스코리아
강의/관심 분야	Java, Framework, Python, Data Science, Machine Learning, Deep Learning
자격/저서/대외활동	Java기반 오픈소스 프레임워크/NCS 개발위원

Learning Object(학습모듈) 및 커리큘럼



LO	커리큘럼
1. HTML5	<ul style="list-style-type: none">- HTML5 문서의 기본- HTML5 문서의 구조화 태그들- Hypertext 와 Link- Audio와 Video 다루기- HTML5의 다양한 입력 Form
2. CSS3	<ul style="list-style-type: none">- CSS 선택자- 문자와 색상 지정하기- Box Model 설정하기- 레이아웃 설정하기
3. HTML5 API	<ul style="list-style-type: none">- Web Storage API- Drage & Drop API- File API- SVG(Scalable Vector Graphics)와 Canvas- Web Socket API

1장.HTML의 발전

1.1 마크업 언어 및 HTML 언어의 역사

1.2 HTML5 언어의 특징

마크업 언어 및 HTML 언어의 역사

■ 마크업 언어(Markup Language)

- 인쇄 교정지의 '마크-업(Mark-up)'에서 유래
- 문서의 속성을 설정하기 위한 마크업을 태그의 형태로 표시
- 대표적인 마크업 언어 : SGML, HTML, XML 등
 - ▶ HTML 언어는 SGML 표준에 따라 정의
 - ▶ 일반 텍스트 형식의 파일로 저장되며 확장자는 *.html 또는 *.htm

• 마크업(Mark-Up)의 유래 :
활자의 식자를 위한 수기형태의
주석

• 마크업 언어는 :
문서의 구조와 내용에 추가적인
의미를 부여하는 마크업 규칙을
규정하는 언어

```
<li><font size="12pt">마크업(Mark-Up)의 유래 :</font>
<br><font size="10pt">활자의 식자를 위한 수기 형태의
주석</font></li>
<li><font size="12pt">마크업 언어 :</font>
<br><font size="10pt">문서의 구조와 내용에 추가적인
의미를 부여하는 마크업 규칙을
규정하는 언어</font></li>
```

마크업 언어 및 HTML 언어의 역사

■ SGML(Standard Generalized Markup Language)

- 1986년 국제표준기구인 ISO에서 개발
 - ▶ 다양한 형식의 전자문서들의 구조와 내용을 기술하는 국제표준
- 시스템 및 응용에 독립적으로 문서를 호환하기 위한 목적
 - ▶ 전자도서, 전자상거래 문서 등 다양한 문서 형식을 정의하는데 사용
 - ▶ HTML은 SGML로 정의된 문서 형식으로 주로 웹문서 작성에 사용

■ HTML(HyperText Markup Language)

- 1994년 HTML 버전 2.0, 1997년 HTML 4.0 버전
- 배우기 쉽고 사용하기 편리하여 인터넷의 대중화에 매우 큰 기여
 - ▶ 반면에 태그가 제한적이고 정교한 페이지를 표현하기에는 부족
- HTML 4.0에서는 동적 HTML (Dynamic HTML) 문서 표현
 - ▶ 스타일시트를 설정하는 CSS 기능과 상호작용을 코드로 표현하기 위한 자바스크립트(Javascript) 언어가 포함.



마크업 언어 및 HTML 언어의 역사

■ XML(eXtensible Markup Language) 및 XHTML

- 문서나 자료의 교환이 필요한 경우 새로운 언어가 필요
- SGML을 간소화한 XML이 1998년 제정
 - ▶ XML 언어로 원하는 문서 형식을 정의하여 다양한 정보를 표현/교환
- HTML 언어도 XML에 기반한 XHTML로 발전
 - ▶ XHTML 1.0은 XML로 문서형식만 정의, 태그는 그 전의 HTML과 동일

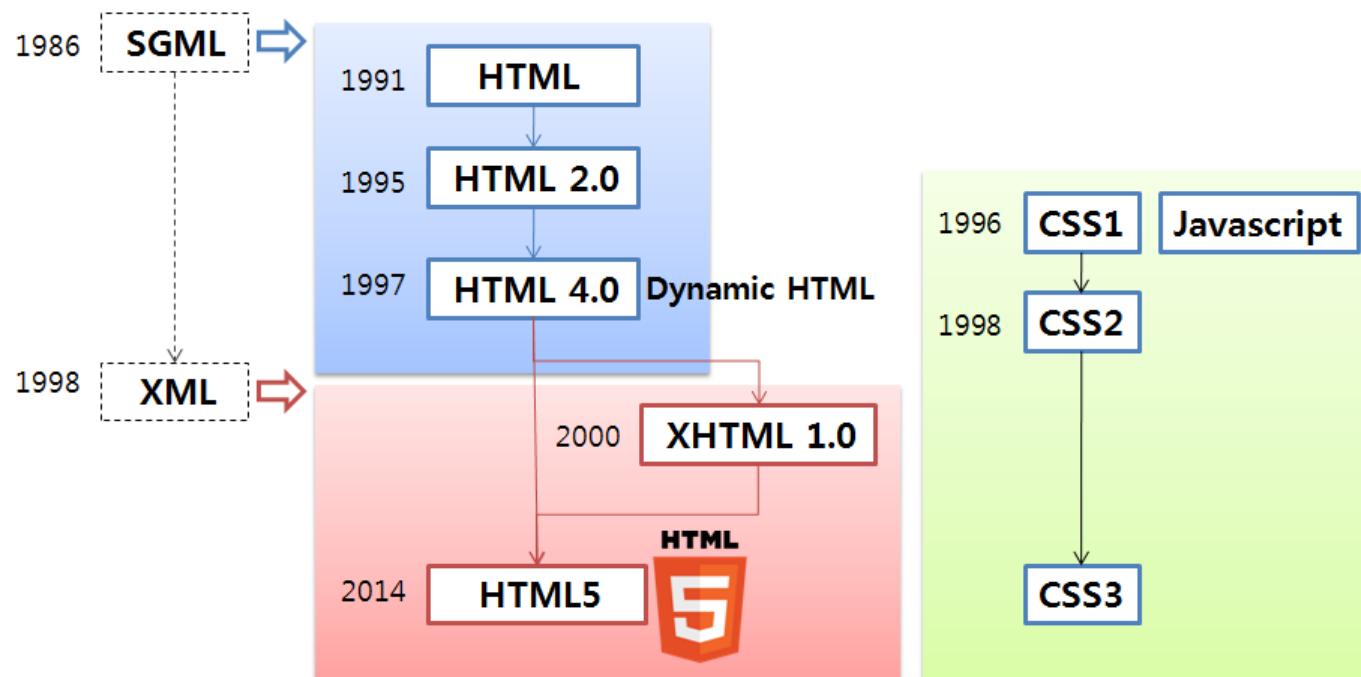
■ HTML5

- 웹 하이퍼텍스트 워킹그룹(WWHATWG)
 - ▶ 모질라, 애플, 오페라 등 웹브라우저 개발사 주도로 2006년 결성
- W3C도 WWHATWG과 협력하여 2007년 HTML5 워킹그룹 신설
- 현재 최종 후보안이 개발된 상황, 2014년 최종 표준안 완성



마크업 언어 및 HTML 언어의 역사

■ HTML의 발전 역사



HTML5 언어의 특징

■ HTML 4.0 이후 요구사항

- 다양한 인터페이스, 다양한 형식의 미디어 파일, 비동기 처리
- 웹 표준 기술을 사용하자는 시도
 - ▶ 웹 애플리케이션 개발에 XML, CSS, 자바스크립트 등 이용

■ HTML5의 방향

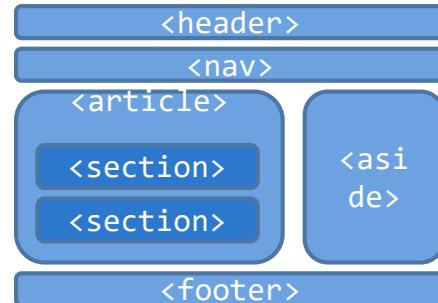
- 마크업에 보다 의미를 부여, 스타일은 분리하도록 CSS3 활용
- 플러그인 대신 웹 표준을 적용하도록 SVG, MathML 지원
- 인터랙션 개발을 위해 자바스크립트를 지원
- 웹 애플리케이션의 개발을 위하여 다양한 API를 제공
 - ▶ 특히 위치관련 및 오프라인 등 모바일 환경까지 고려한 API 제공



(1) 강화된 마크업 요소

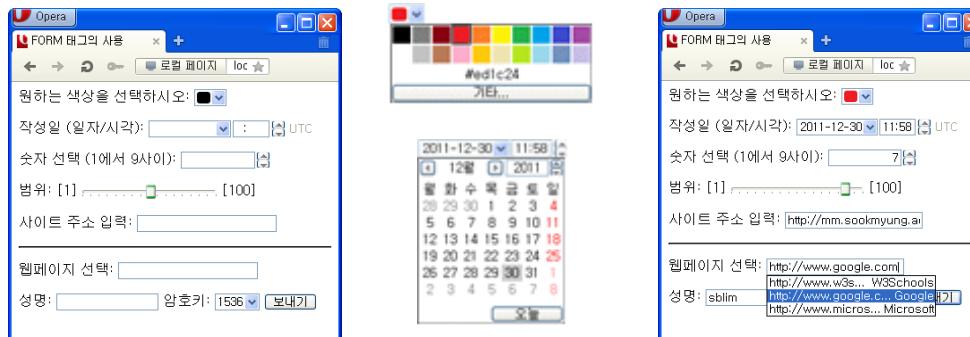
■ 의미를 부여할 수 있는 구조적 마크업 요소

- 페이지 단위의 문서 구조를 표현
- <header>와 <footer> 요소,
- <article> 요소, <section> 요소
- <nav> 요소, <aside> 요소



■ 다양하고 편리한 웹 폼(WebForm) 입력 기능

- 편리한 사용자 인터페이스 개발을 위해 Form 기능 대폭 개선
- <input> 요소에 date, number, color, file 등 각종 type 속성 추가



(1) 강화된 마크업 요소

■ 웹 미디어 기능의 강화

- 멀티미디어 및 그래픽스 관련 기능의 추가
- <video> 요소와 <audio> 요소: 동영상이나 비디오 스트리밍 처리
- <canvas> 요소: 래스터 그래픽스 그리기 기능
- SVG(Scalable Vector Graphics): 벡터 그래픽스 처리
- MathML 언어: 수식의 의미와 모양까지 표현



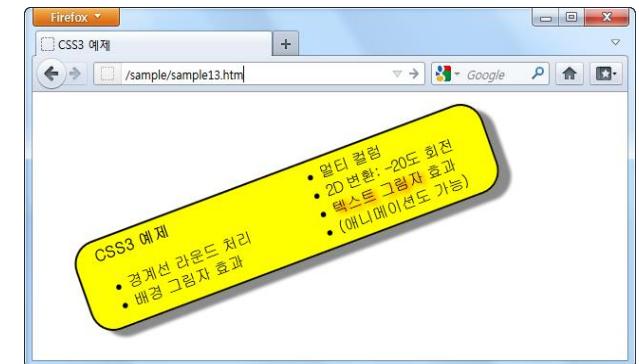
(2) CSS3의 완전 지원

■ 스타일시트(Stylesheet)

- 웹 문서의 외형 스타일을 지정하는 언어
- HTML 문서에는 일반적으로 CSS(Cascading Style Sheet)가 사용
- 1996 CSS1, 1998 CSS2, 2005년 이후 CSS3는 모듈별로 개발 중

■ HTML5에서는 CSS1, CSS2, CSS3까지 완전히 지원

- 기존 CSS는 주로 텍스트, 배경 및 색상, 목록, 박스모델 등 속성
- CSS3에는 더욱 다양한 스타일 지정 기능을 포함
- 예제 화면
 - ▶ 박스의 모서리 처리, 그림자 효과, 다단(multi-column) 지정, 2D/3D 기하변환, 텍스트의 그림자 등 다양한 효과
 - ▶ 이 외에도 장면 변환(transition), 애니메이션, 사용자 인터페이스에 관련된 속성



(3) 다양한 API 지원

- 웹 애플리케이션 개발에 많은 도움을 줄 수 있는 다양한 API(Application Programming Interface) 제공
 - 자바스크립트로 구현 가능
 - <video>와 <audio> 요소를 제어하는 API
 - <canvas> 요소에 그림을 그리는 API
- 별도의 사양으로 분리된 API
 - 웹 소켓, 웹 워커, 웹 스토리지, 로컬 데이터베이스, 웹 메세징, 위치정보 등의 API도 지원
 - 넓은 범위에서 보면 이들 모두가 HTML5의 기능
 - 자바스크립트로 매우 다양하고 강력한 기능의 웹애플리케이션 구현 가능해짐



(3) 다양한 API 지원

- 드래그앤 드롭(Drag & Drop) API
 - ▶ 아이콘, 텍스트, 이미지, 파일 등 요소를 드래깅 할 때 동작을 제어
- 오프라인 웹 어플리케이션(Offline Web Applications) API
 - ▶ 인터넷에 접속 못할 때에 웹 애플리케이션을 사용하도록 하는 기능
- 웹 스토리지(Web Storage) API
 - ▶ 클라이언트 쪽 로컬 스토리지에 저장
- 인덱스드 데이터베이스(Indexed Database) API
 - ▶ 클라이언트에 데이터베이스를 저장하고 사용
- 파일(File) API
 - ▶ 클라이언트 내의 로컬 파일을 읽고 쓸 수 있는 기능
- 웹 소켓(Web Sockets) API
 - ▶ 서버와 브라우저 사이에 양방향 통신 채널을 제공
- 웹 메세징(Web Messaging) API
 - ▶ 애플리케이션 간에 메시지를 주고받을 수 있는 기능
- 위치정보(Geo-Location) API
 - ▶ 모바일 단말기에서 현재 위치를 파악할 수 있게 해주는 기능



(4) 모바일 웹 환경 고려

■ 모바일 환경

- 일찍부터 HTML5를 지원하는 웹브라우저를 탑재
 - ▶ 웹브라우저마다 구현 상황이 서로 다른 데스크탑 환경보다는 모바일 환경이 HTML5 어플리케이션을 개발하기에 더 나은 입장

■ 모바일 환경을 위한 HTML5의 특별한 기능

- 위치정보 API
 - ▶ 위치정보는 모바일 애플리케이션 개발에 최적 활용
- 오프라인 어플리케이션 API
 - ▶ 모바일 환경에서 접속이 끊기거나 트래픽 최적화에 긴요하게 활용
 - ▶ 오프라인이 될 경우에 로컬 스토리지, 웹 데이터베이스, 어플리케이션 캐시 등이 유용하게 활용
- 새로운 유형의 다양한 입력 품 지원
 - ▶ 모바일 애플리케이션의 사용자 인터페이스 개발 및 사용이 편리



2장. HTML5 문서의 기본

- 2.1 기본 문서 만들기
- 2.2 문서 꾸미기
- 2.3 목록 나열하기
- 2.4 표 그리기
- 2.5 문서 특정 부분 구별하기
- 2.6 문서 구조화하기

HTML5 문서 기본

- HTML 문서
 - 태그(tag)로 표기하는 요소(element)와 요소의 속성(attribute)으로 이루어짐.
- HTML 문서가 나타난 초기에는 이와 같은 요소만을 사용하여 표현을 가지고 있는 문서를 작성
- 웹 기술이 발전함에 따라 동적으로 변경 가능한 문서, 응용프로그램 성격을 가지는 문서로 개념 발전
- 학습내용
 - HTML 문서를 만들기 위해 필요한 문서 구성요소와 개념
 - 문서 구조화와 단락을 위한 요소
 - 목록과 표를 만들기 위한 요소



2.1 기본 문서 만들기

2.1.1 문서 구조

2.1.2 요소와 속성

2.1.3 기타 문서 구성

웹 문서의 기본 구조

■ <!DOCTYPE>

- 문서의 종류는 알려주는 <!DOCTYPE>로 시작
 - ▶ 웹 문서에 DOCTYPE 정보가 없으면 웹 브라우저는 웹 문서에서 사용된 규약을 정확히 알 수 없어 웹 문서를 잘못 처리할 수 있음
- <!DOCTYPE html>: HTML5 문서규약

■ <html> 요소

- 문서 내용을 나타냄
- 문서에 대한 정보를 나타내는 <head> 요소
- 문서 내용을 나타내는 <body> 요소
 - ▶ 문서를 구성하는 여러 크기의 단락 제목, 단락, 목록, 표를 표현하기 위한 요소 모음이 나타날 수 있음



<head> 요소

- <head> 요소
 - 문서 제목을 나타내는 하나의 <title> 요소
 - 문서 제목 외에 다른 정보를 나타내기 위한 <meta> 요소
 - 기타 요소: <link> 요소, <script> 요소, <style> 요소
- <title> 요소
 - 문서 제목을 나타냄
- <meta> 요소
 - 웹 문서에서 사용하고 있는 문자 인코딩 정보

```
<meta charset="UTF-8" >
```
 - 검색엔진을 위한 문서의 검색 키워드 정보
- 기타 요소
 - <script> 요소: 자바스크립트로 기술된 웹 문서의 사용자와의 상호 작용을 위한 스크립트
 - <style> 요소: 웹 문서를 표현에 사용되는 CSS(Cascade Style Sheet)
 - <link> 요소: 웹 문서 외부에 기술된 자바스크립트와 CSS 파일 링크 정보

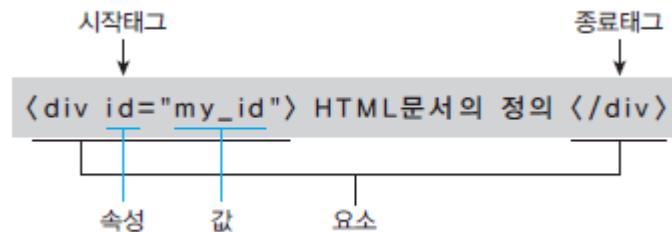


요소와 속성

■ 웹 문서: 태그로 표현되는 요소(element)로 구성

■ 요소

- 시작을 나타내는 시작태그
- 요소의 내용
- 종료 태그
- 속성(attribute) 모음



■ 속성

- 요소의 정보를 나타냄
- 이름과 값의 쌍의 모음으로 구성



태그

- 문서에서 요소를 표시하기 위해 문서 내용에 붙여주는 일종의 꼬리표
- 태그
 - 요소의 시작을 나타내는 시작태그: '<' 문자와 태그 이름, '>' 문자로 구성
 - 요소의 끝을 나타내는 종료태그: '</' 문자와 태그 이름, '>' 문자로 구성
 - 종료태그를 없는 경우 있는 것으로 간주하는 요소도 있음
 - ▶ 가능하면 종료태그를 써주는 것이 좋음
 - 시작과 종료를 한 번에 나타내는 태그: '<' 문자와 태그 이름, '/>' 문자로 구성
 - ▶ 다른 요소나 내용을 가지지 않는 요소를 위한 태그

복합태그와 단독태그

- 복합태그: 요소가 시작태그와 종료태그로 구성되는 경우에 사용하는 태그
- 단독태그: 시작과 종료를 같이 나타내는 태그로 구성

```
<tag_name> ... </tag_name> // 복합태그  
<tag_name/> // 단독태그
```

- <title> 요소: 시작태그인 <title>와 종료태그인 </title>로 구성된 복합태그
- 요소: 시작과 종료를 같이 의미하는 단독태그인 태그



속성

■ 요소에 추가정보를 주기 위해서 사용

■ 추가정보

- 요소의 모양을 나타내는 스타일 등이 있음
- 요소에 공통적으로 적용되는 것과 요소마다 다르게 적용되는 것이 있음.

■ 속성

- 요소의 시작태그 내에 나타남
- 이름과 값으로 구성
- 이름과 값 사이에는 '=' 문자가 나타나고, 값은 "" 문자 또는 " 문자에 감싸서 나타냄

```
<p style=" font-size=0" > ... </p>
```

- 요소는 여러 속성을 가질 수 있으며, 이 경우 속성은 빈 칸으로 구분

```
<p style=" font-size=0" script=" alert( 'error' );" > ... </p>
```



기타 문서 구성

■ 주석

- 웹 브라우저에서는 보이지 않으나 문서 작성자가 문서에 대한 정보를 남기기 위해서 사용

```
<!-- 내용-->
```

■ 특수문자

- '<' 문자나 '>' 문자와 같은 문자는 웹 문서에서 태그를 표현하기 위해 사용

& 특수 문자 이름 :

특수문자 표현	설명
&nbsp	공백 문자
<	<
>	>
"	“
&	&

- 문자코드

& #문자코드 ;



2.2 문서 꾸미기

2.2.1 단락제목

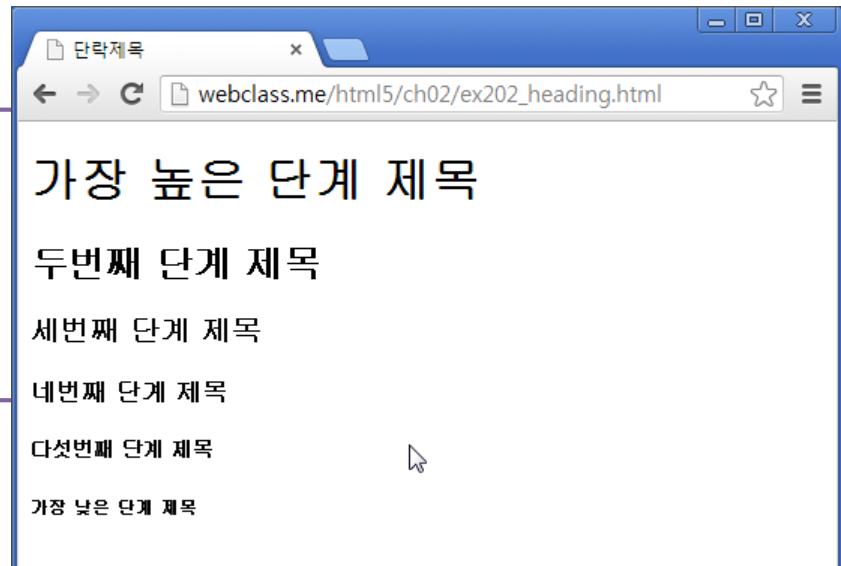
2.2.2 단락과 줄

2.2.3 다양한 텍스트 표현

단락제목

- 웹 문서에서 단락제목을 지정할 때 `<h1>~<h6>` 요소를 이용
 - 숫자가 클수록 글자가 작아짐.
 - ▶ `<h1>` 요소는 첫 번째 단계 수준의 단락제목
 - ▶ `<h2>` 요소는 두 번째 단계 수준의 단락제목
 - ▶ `<h6>` 요소는 가장 낮은 단계 수준의 단락제목을 표현
 - 웹 문서의 내용 중 다른 텍스트보다 굵게 나타남

```
<h1>가장 높은 단계 제목</h1>
<h2>두번째 단계 제목</h2>
<h3>세번째 단계 제목</h3>
<h4>네번째 단계 제목</h4>
<h5>다섯번째 단계 제목</h5>
<h6>가장 낮은 단계 제목</h6>
```



단락과 줄

- 단락: <p> 요소
 - 웹 문서에서 단락을 만들기 위해 사용
 - 웹 브라우저는 단락과 단락 사이에 약간의 공백을 추가하여 시각적으로 두 단락이 구분되어 있을 표시
- 줄 바꿈:
요소
 - 줄 바꿈은 단락을 구분하지 않지만, 앞 문장과 다른 문장간에 줄을 바꾸고자 할때 사용
 - 단일 태그 형태로 사용
- 가로줄: <hr> 요소
 - 웹 문서에서 가로줄을 표시하여 앞뒤로 글의 주제가 바뀌었다는 것을 나타낼 때 사용
- 형식 유지: <pre> 요소
 - 요소 내용을 웹 브라우저 화면에 그대로 보이고자 할 때 사용
 - 웹 문서에서는 빈칸 문자나 탭 문자, 줄바꿈 문자와 같은 공백문자(white space character)를 하나의 공백 문자로 취급
 - 소스코드와 같이 공백문자를 원문 그대로 보이는 것이 중요한 경우에 사용
- 인용: <blockquote> 요소
 - 다른 글의 내용을 인용할 때 사용
 - 웹 브라우저에서는 인용 내용을 들여쓰기를 하여 화면에 표시



예제: 단락과 줄

```
<h1>단락과 줄바꾸기</h1>
<br> 줄이 바뀝니다.
<br> 줄이 바뀝니다.
<p> 문단이 바뀝니다
<hr>
<h1>형식 그대로</h1>
<pre> 입 력 한 대 로
    보 입 니 다
</pre>
<h1>인용</h1>
<blockquote>
멀티미디어 배움터 2.0은 미디어의 특성과 미디어 처리
미디어의 활용 환경 (인터넷, 모바일, 사이버스페이스)
있도록, 멀티미디어가 활용되는 방식과 적용사례를 소개
</blockquote>
```



다양한 텍스트 표현

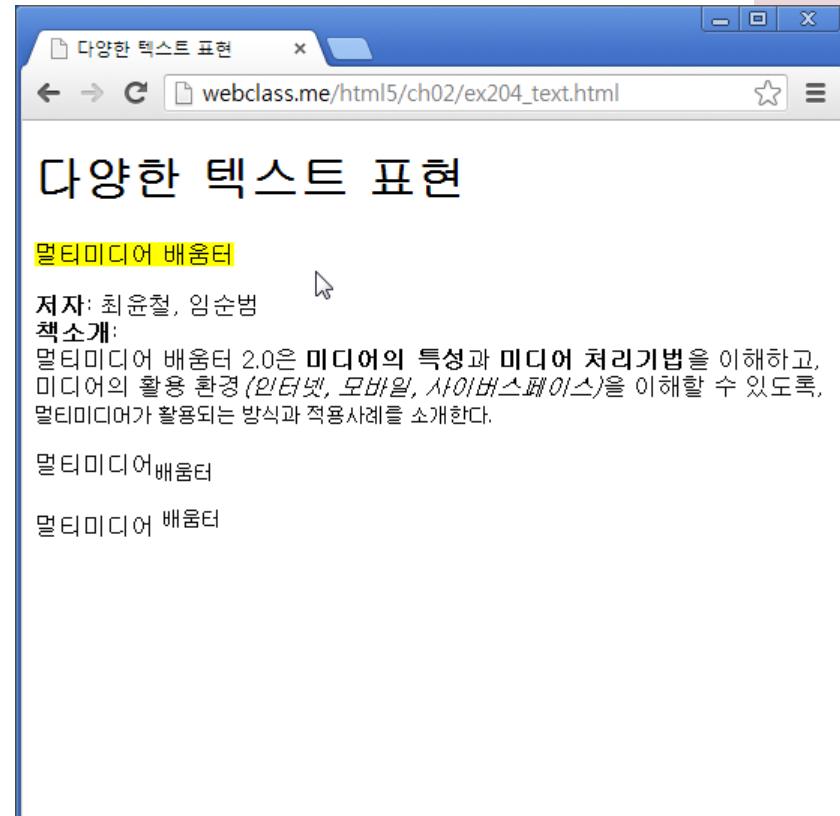
- 텍스트 강조: 요소
 - 문장 내의 특정 텍스트를 강조하고자 할 때 사용하는 요소이며, 웹 브라우저에서 진하게 표시
- 텍스트 구분: 요소
 - 외국어나 분류학적인 용어, 대본에서 무대 지문, 생각 등 다른 일반 텍스트와 구분을 해야 하는 텍스트를 표시하는데 사용하며, 웹 브라우저에서 기울여진 모습으로 나타남
- 작은 글씨 표현: <small> 요소
 - 웹 문서의 꼬릿말에 있는 저작권 문장이나 라이센스 정보처럼 주변의 다른 텍스트보다 포커스를 받지 못하는 텍스트를 표시할 때 사용하며, 웹 브라우저에서는 약간 작은 글씨로 표시
- 하이라이트 효과: <mark> 요소
 - 문장 내의 텍스트를 마킹 하고자 할 때 사용하며, 웹 브라우저에서는 형광펜으로 표시한 것과 같이 나타남
- 첨자 넣기: <sub> 요소와 <sup> 요소
 - <sub> 요소는 아래 첨자(subscript)를 나타내며, <sup> 요소는 위 첨자(superscript)를 만듦



예제: 다양한 텍스트 표현

```
<h1>다양한 텍스트 표현</h1>
<p><mark>멀티미디어 배움터</mark></p>
<strong>저자</strong>: 최윤철, 임순범
<br>
<strong>책소개</strong>:<br>
멀티미디어 배움터 2.0은 <strong>미디어
의 특성</strong>과 <strong>미디어 처리
기법</strong>을 이해하고, 미디어의 활용
환경 <em>(인터넷, 모바일, 사이버스페이
스)</em>을 이해할 수 있도록, <small>멀
티미디어가 활용되는 방식과 적용사례를 소개
한다.</small>

<p>멀티미디어 <sub>배움터</sub></p>
<p>멀티미디어 <sup>배움터</sup></p>
```



2.3 목록 나열하기

- 2.3.1 순서 없는 목록
- 2.3.2 순서 있는 목록
- 2.3.3 정의 목록

순서 없는 목록

■ 순서 없는 목록 : 요소

- 규칙과 순서가 없는 목록(unordered list)를 만들기 위해 사용
- 목록을 만들 때에는 목록 내에 포함된 항목을 나타내기 위한 하위요소인 요소를 사용
- 요소는 요소를 한번에 감싸주는 역할
- 요소 안에 있는 요소는 목록의 한 항목

```
<ul>
  <li>첫 번째 항목</li>
  <li>두 번째 항목</li>
  <li>세 번째 항목</li>
</ul>
```

순서 있는 목록

■ 순서 있는 목록 : 요소

- 규칙과 순서를 숫자로 나타내는 목록(ordered list)에서 사용
- 규칙과 순서에 관계가 있기 때문에 목록 앞에는 숫자가 옴
- 요소는 이 요소를 한번에 감싸주는 역할
- 요소 안에 있는 요소는 목록의 한 항목

```
<ol>
  <li>첫 번째 항목</li>
  <li>두 번째 항목</li>
  <li>세 번째 항목</li>
</ol>
```

정의 목록

■ 정의 목록 : <dl> 요소

- 사전과 같은 정의 목록(definite list)을 만들기 위해서 사용
 - 사전은 찾으려고 하는 단어와 단어를 설명하는 문장으로 구성
-
- <dl> 요소: <dt> 요소와 <dd> 요소를 사용
 - <dt> 요소: 정의할 용어의 제목
 - <dd> 요소: 용어의 설명(정의)

```
<dl>
  <dt>이름</dt>
  <dd>설명</dd>
</dl>
```

예제: 목록

```
<h1>순서 없는 목록</h1>
<ul>
  <li>스티브잡스</li>
  <li>해를 품은 달</li>
</ul>
```

```
<h1>순서 있는 목록</h1>
<ol>
  <li>스티브잡스</li>
  <li>해를 품은 달</li>
</ol>
```

```
<h1>사전 목록</h1>
<dl>
  <dt> html5 </dt>
  <dd> HTML5는 HTML의 차기 주요 제안 버전
  으로 월드 와이드 웹의 핵심 마크업 언어이다.
  2004년 6월 Web Hypertext Application
  Technology Working Group(WWHATWG)에서 웹
  애플리케이션 1.0이라는 이름으로 세부 명세 작
  업을 시작하였다. </dd>
</dl>
```

The screenshot shows a web browser window with the URL webclass.me/html5/ch02/ex205_list.html. The page content is as follows:

- 순서 없는 목록**
 - 스티브잡스
 - 해를 품은 달
- 순서 있는 목록**
 1. 스티브잡스
 2. 해를 품은 달
- 사전 목록**

html5

HTML5는 HTML의 차기 주요 제안 버전으로 월드 와이드 웹의 핵심 마크업 언어이다. 2004년 6월 Web Hypertext Application Technology Working Group(WWHATWG)에서 웹 애플리케이션 1.0이라는 이름으로 세부 명세 작업을 시작하였다.



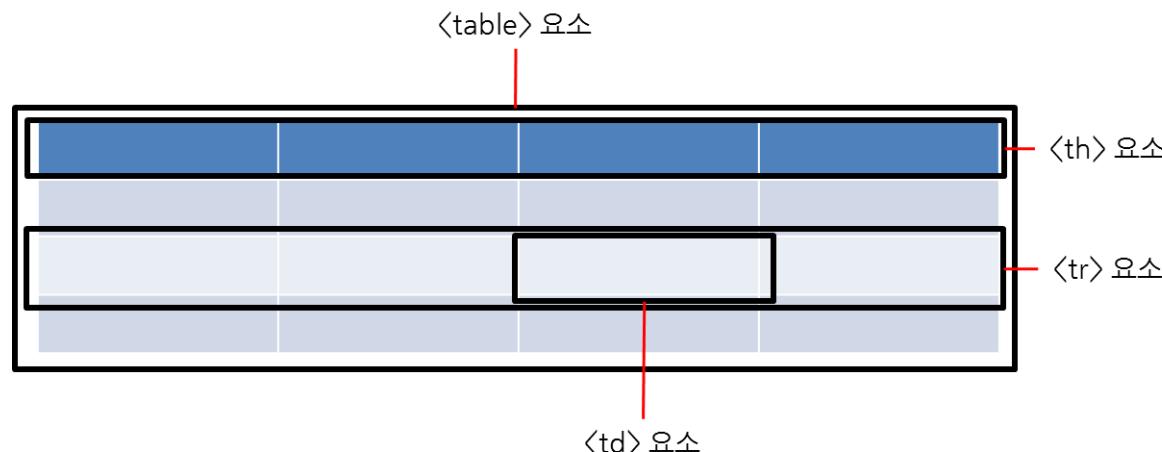
2.4 표 그리기

- 2.4.1 표의 기본 구조
- 2.4.2 표의 장식

표의 기본 구조

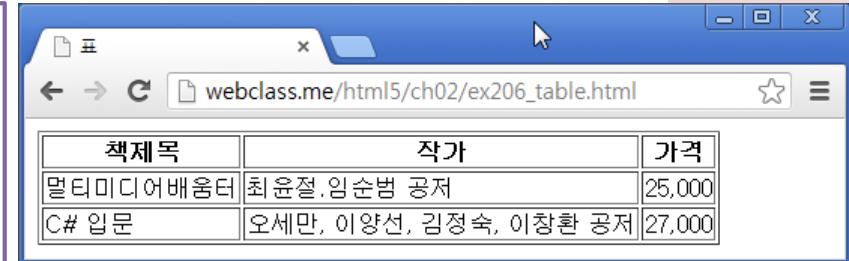
■ <table> 요소

- 웹 문서에 표를 표현하자고 할 때 사용
- <tr> 요소(table row): 하나의 행
- <td> 요소(table data): 표의 각 셀을 만들며, 각 셀의 데이터를 나타냄
- <th> 요소(table header)
 - ▶ <td> 요소와 유사하지만 <th> 요소는 표의 제목에 해당하는 셀을 나타냄
 - ▶ 웹 브라우저에서 <th> 요소 내용은 일반적으로 굵은 글씨로 중앙에 표시



예제: 표

```
<table border="1">
  <tr>
    <th>책 제목</th>
    <th>작가</th>
    <th>가격</th>
  </tr>
  <tr>
    <td>멀티미디어배움터</td>
    <td>최윤절.임순범 공저</td>
    <td>25,000</td>
  </tr>
  <tr>
    <td>C# 입문</td>
    <td>오세만, 이양선, 김정숙, 이창환 공저</td>
    <td>27,000</td>
  </tr>
</table>
```



A screenshot of a web browser window displaying a table. The browser's title bar says "webclass.me/html5/ch02/ex206_table.html". The table has three columns: 책 제목, 작가, and 가격. It contains two rows of data.

책 제목	작가	가격
멀티미디어배움터	최윤절.임순범 공저	25,000
C# 입문	오세만, 이양선, 김정숙, 이창환 공저	27,000

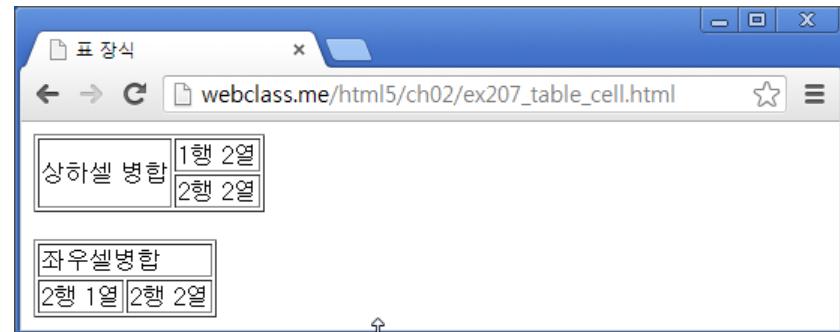
표의 장식

■ 셀 합치기

- <td> 요소의 rowspan 속성과 colspan 속성을 사용
- rowspan 속성: 상하셀 병합
- colspan 속성: 좌우셀 병합

```
<table border="1">
  <tr>
    <!-- 첫번째 테이블 상하 셀을 병합 --
    <td rowspan="2">상하셀 병합</td>>
    <td>1행 2열</td>
  </tr>
  <tr>
    <td>2행 2열</td>
  </tr>
</table><br>
```

```
<table border="1">
  <tr>
    <!-- 두번째 테이블 좌우 셀을 병합한다. -->
    <td colspan="2">좌우셀병합</td>
  </tr>
  <tr>
    <td>2행 1열</td>
    <td>2행 2열</td>
  </tr>
</table>
```



표의 장식: 기타 요소

■ 표 제목이 행 표현: <thead> 요소

- <table> 요소를 머리말, 본문, 꼬리말로 나누었을 때 머리말에 해당하는 요소
- <caption> 요소의 뒤에 <tbody>, <tfoot>, <tr> 요소의 앞에 나타남
- 열 제목으로 구성된 행의 집합을 나타내며 table 요소에 한번만 사용

■ 표 마지막 행 표현: <tfoot> 요소

- 테이블의 꼬리말에 해당
- <tbody> 요소 앞에 사용하며, 표 내에서 한 번만 사용할 수 있음

■ 표 본문 행 표현: <tbody> 요소

- 테이블을 머리말, 본문, 꼬리말로 나누었을 때 본문에 해당하는 내용을 나타냄
- 여러 번 사용할 수 있음



예제: 목록

```
<h1>추천도서 Table</h1>
<table border="1">
  <caption>추천 도서 테이블</caption>
  <thead>
    <tr>
      <th scope="col">책제목</th>
      <th scope="col">작가</th>
      <th scope="col">가격</th>
    </tr>
  </thead><!-- 표의 헤더 영역부분 -->

  <tfoot>
    <tr>
      <th scope="row">총</th>
      <td>2권</td>
      <td>30,000원</td>
    </tr>
  </tfoot><!-- 표의 푸터 영역부분 -->
```

```
<tbody>
  <tr>
    <th scope="row">스티브잡스</th>
    <td>월터아이작슨 저,안진환 역</td>
    <td>25,000원</td>
  </tr>
  <tr>
    <th scope="row">멀티미디어 배움터</th>
    <td>최윤철.임순범 공저</td>
    <td>25,000원</td>
  </tr>
</tbody><!-- 표의 내용 영역부분 -->
</table>
```

The screenshot shows a web browser window with the title "표 고급". The address bar contains the URL "webclass.me/html5/ch02/ex208_table_advanced.html". The main content area displays a table titled "추천 도서 표". The table has three columns: "책제목", "작가", and "가격". It contains four rows: a header row, two data rows, and a footer row. The data rows correspond to the entries in the code above. The footer row shows a total of 2권 and 30,000원.

책제목	작가	가격
스티브잡스	월터아이작슨 저,안진환 역	25,000원
멀티미디어 배움터	최윤철.임순범 공저	25,000원
총	2권	30,000원

2.5 문서 특정 부분 구분하기

2.5.1 <div> 요소와 요소

2.5.2 요소의 id 속성과 class 속성

<div> 요소와 요소

■ 웹 문서에서 특별한 의미를 갖지 않으며 콘텐츠를 그룹화하는데 사용하는 요소

■ <div> 요소

- 한 줄의 공간을 준비하며, 블록(block) 단위로 영역을 묶음
- 테두리, 정렬, 문단 모양 등 필요한 기능들은 CSS의 스타일(style)을 이용하여 지정

■ 요소

- 인라인.inline 단위로 영역을 묶고 입력하는 내용만큼 공간을 준비
- 줄바꿈이 생기지 않기 때문에 앞뒤 내용이 이어짐
- 인라인(inline) 요소 안에 블록(block)요소는 들어갈 수 없음



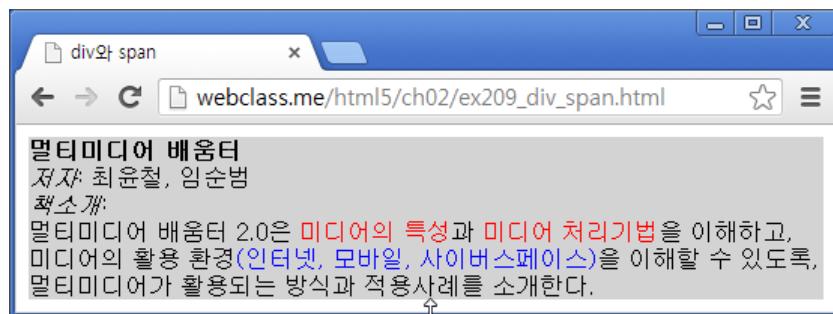
<div> 요소와 요소

인라인(Inline)요소	블록(block)요소
<ul style="list-style-type: none">- 텍스트 요소, 기존 글에 이어서 작성- 요소, 요소, 요소- 줄 바꿈 태그를 사용하지 않거나 블록레벨 태그 안에 단독을 포함되지 않는 한 가로로 쭉 나열됨- 주변에는 공간이 생기지 않음- 인라인 요소	<ul style="list-style-type: none">- 문서 구조 요소, 새로운 행에 작성- <div> 요소, <p> 요소, <h1> 요소 등- 줄 바꿈 태그를 사용하지 않아도 스스로 줄 바꿈이 되며, 요소들은 가로로 흐르지 않고 세로로 흐름- 주변에 일정량의 공간을 만듦. 너비를 정해주지 않으면 가로로 가득 참- <h1> 블록레벨 요소 </h1>



예제:div와 span

```
<div style="background-color:lightgray">
<span style="font-weight:bold">멀티미디어 배움터</span><br>
<span style="font-style:italic">저자</span>: 최윤철, 임순범<br>
<span style="font-style:italic">책소개</span>:<br>
멀티미디어 배움터 2.0은 <span style="color:red">미디어의 특성</span>과 <span
style="color:red">미디어 처리기법</span>을 이해하고, 미디어의 활용 환경<span
style="color:blue">(인터넷, 모바일, 사이버스페이스)</span>을 이해할 수 있도록,
멀티미디어가 활용되는 방식과 적용사례를 소개한다.
</div>
```



요소의 id 속성과 class 속성

■ 요소의 id 속성과 class 속성

- 스타일시트와 자바스크립트에서 문서에 있는 요소를 참조하기 위한 속성

■ id 속성과 class 속성의 차이점

- id 속성은 문서에 하나의 요소를 참조하기 위해 사용
- class 속성은 여러 개의 요소를 참조하기 위해서 사용

```
<div class="book" id="book_multimedia">멀티미디어 배움터</div>
<div class="book" id="book_csharp">C# 입문</div>
```

■ class 속성

- 공백으로 구분하여 여러 개의 값이 올 수 있음
- 스타일시트나 자바스크립트에서 각각의 이름으로 요소를 참조할 수 있음

```
<div class="author book">
```



2.6 문서 구조화하기

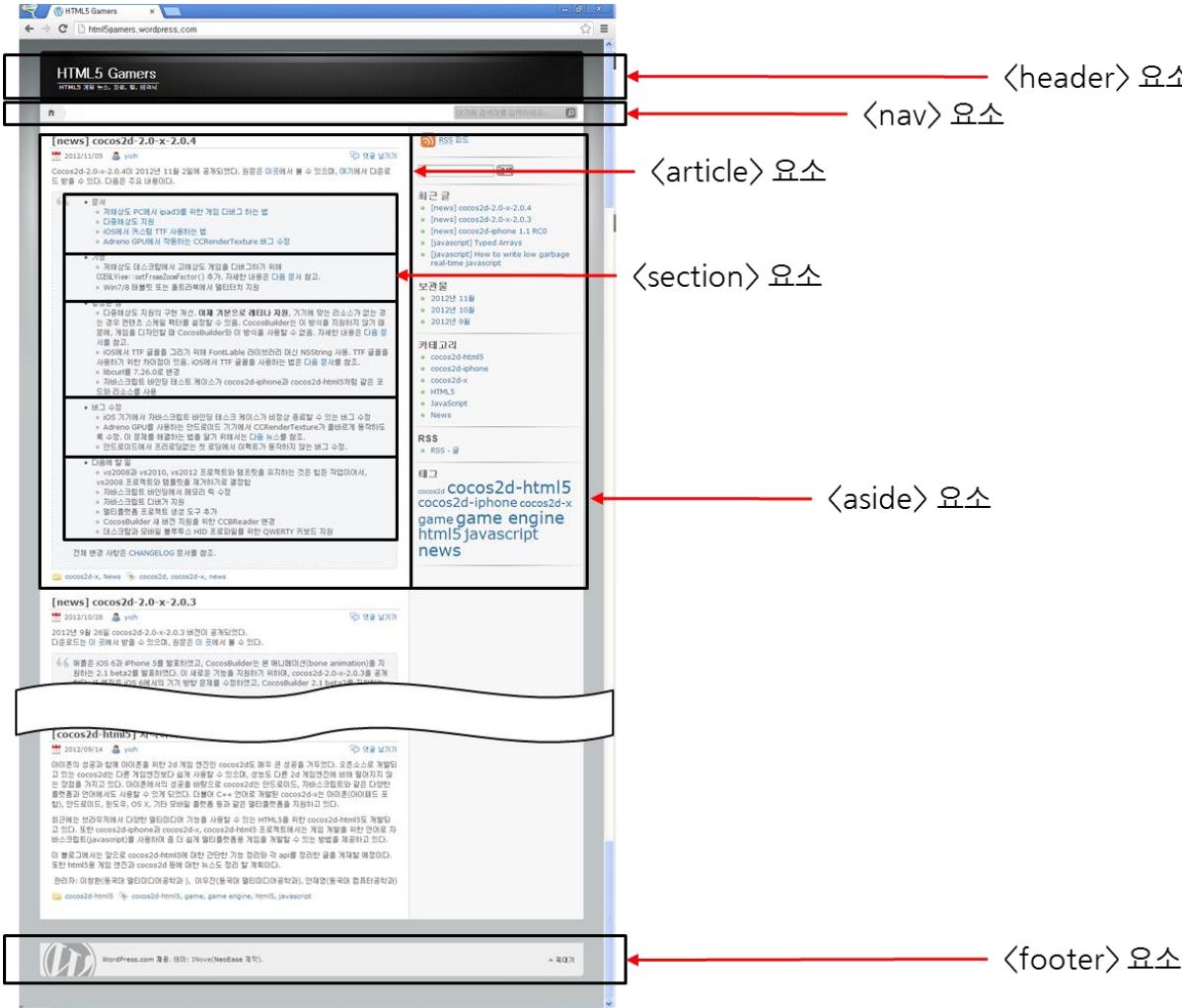
- 2.6.1 문서 구조화의 필요성
- 2.6.2 문서 구조화 요소

문서 구조화의 필요성

- 웹 문서는 이전에는 웹 브라우저를 통해 사람에게 정보를 보여주고 전달하는 용도로 주로 사용
- 최근에는 검색엔진이나 장애인을 위한 스크린 리더와 같이 사람이 아닌 컴퓨터가 문서를 이해해야 하는 경우가 늘어나고 있음
 - 사람은 문장의 위치나 모양, 이미지 등을 통해 문서 내용을 이해할 수 있으나, 컴퓨터와 같은 기계는 단순 문장이나 문장 위치나 모양만으로 내용을 이해하기 힘듬
- 문서 구조화
 - 웹문서 안에 있는 문장의 의미를 특정 의미를 가진 요소를 사용하여 명확하게 나타내는 것



문서 구조화 요소



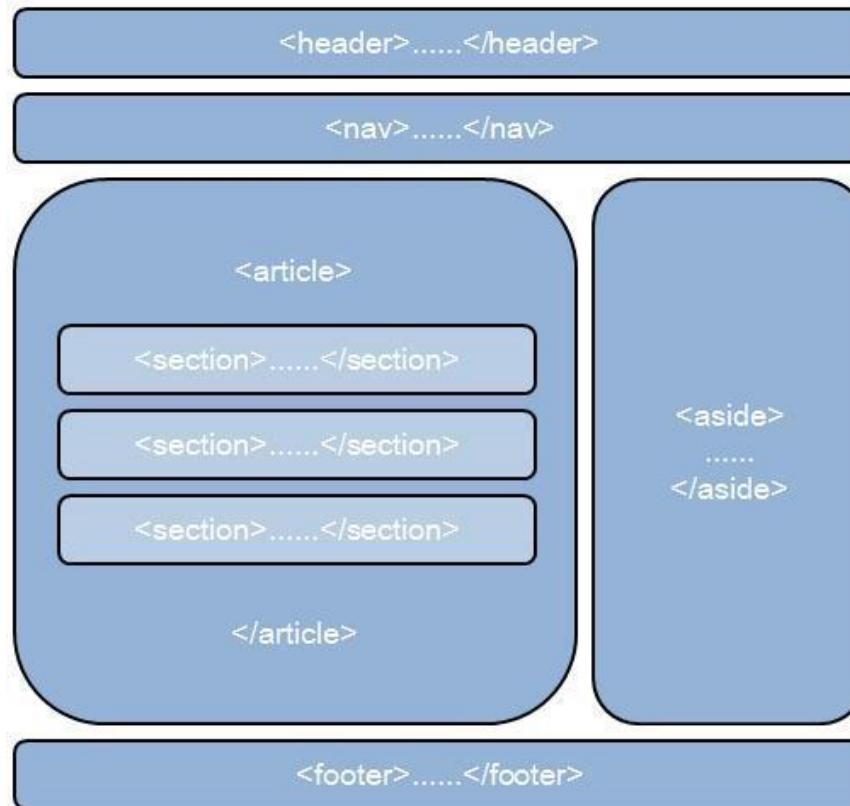
문서 구조화 요소

■ 웹 문서 구조화 요소

- <header>와 <article>, <section>, <aside>, <footer>, <nav> 와 같은 요소가 있음
- 구조화 요소는 화면 상으로는 표시되지는 않지만, 구조화 요소에 포함된 다른 요소들의 용도와 성격을 나타내는 용도로 사용됨
- HTML5에 새로 추가된 부분
 - ▶ 이전 HTML에서는 <div> 요소와 요소를 사용하여 문서의 영역을 구분하였음



문서 구조화 요소



문서 구조화 요소

■ <header> 요소

- 웹 문서에서 머리말 영역을 나타낼 때 사용
- 머리말 영역에는 제목, 제목 설명 문장 등을 나타내는 다른 요소를 포함함

■ <nav> 요소

- 웹 문서에서 다른 웹 문서나 문서내의 다른 부분으로 이동하는 부분을 나타내고자 할 때 사용

■ <article> 요소

- 웹 문서에서 주요 내용을 가진 본문을 나타낼 때 사용
- 웹 문서에서 실제로 나타내고자 하는 내용을 가지고 있으므로, 기계에 의해 중요한 부분으로 인식되어 처리됨
- 웹 문서에서는 여러 개의 <article> 요소가 나타날 수 있으며, 한 <article> 요소는 하나의 독립된 문서 내용을 나타냄



문서 구조화 요소

■ <section> 요소

- 비슷한 그룹이나 절 또는 문서를 묶기 위해서 사용

■ <aside> 요소

- 문서 내용과 관련된 여러 부수 정보를 나타내는데 사용

■ <footer> 요소

- 웹 문서의 저자 정보, 저작권 정보, 이용조건 등을 나타내기 위해서 사용



예제: 문서구조화

```
<header>웹</header>
<nav>
  <ul>
    <li>html5</li>
    <li>css3</li>
    <li>javascript</li>
  </ul>
</nav>
<aside>
  <ul>
    <li>W3C</li>
    <li>ECMA</li>
    <li>IETF</li>
  </ul>
</aside>
<article>
  <h1>html5 문서 기본</h1>
  <p>기본 문서 만들기</p>
  <p>문서 꾸미기</p>
  <p>목록 나열하기</p>
  <p>표 그리기</p>
  <p>문서 특정부분 구분하기</p>
  <p>문서 구조화하기</p>
</article>
<footer>작성자: 임순범, 박희민, 이창환</footer>
```



3장. 링크와 멀티미디어

- 3.1 링크 달기
- 3.2 이미지 사용하기
- 3.3 오디오와 비디오 다루기

3.1 링크 달기

- 3.1.1 하이퍼텍스트와 링크
- 3.1.2 문서간 이동
- 3.1.3 문서 내 특정 위치로 이동
- 3.1.4 <iframe>으로 다른 문서의 내용 표시하기

하이퍼텍스트/하이퍼미디어

■ HTML(HyperText Markup Language)

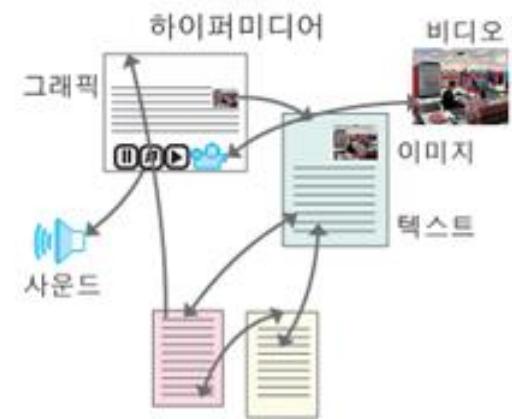
- 하이퍼텍스트의 마크업 언어

■ HTML의 기본 개념

- 하이퍼텍스트 : 서로 연관된 문서나 텍스트 조각들을 연결
- 하이퍼미디어 : 텍스트 뿐 아니라 이미지, 그래픽, 오디오, 비디오 등의 멀티미디어 정보가 서로 연결

■ 하이퍼텍스트/하이퍼미디어의 구조

- 각 정보의 조각은 링크에 의해 서로 연결
- 모든 정보의 접근은 연결 링크를 선택하여 내비게이션



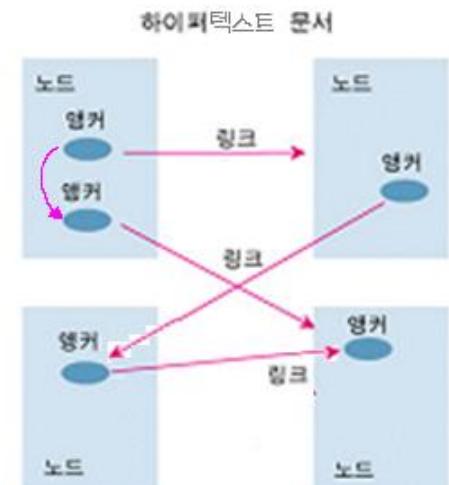
노드, 링크, 앵커

■ 노드, 링크, 앵커의 개념

- 노드 : HTML 문서나 멀티미디어 정보를 표현하는 기본단위
- 링크 : 노드를 연결하여 내비게이션이 가능토록 하는 구성요소
- 앵커 : HTML 문서 내에서 링크의 출발점이나 도착점을 의미
 - ▶ 앵커 영역 : 앵커가 설정되어 있는 영역

■ HTML 문서에서 사용되는 링크의 종류

- 특정 단어나 문장, 혹은 이미지에서 다른 문서로 이동하는 링크
- 외부 URL로 연결하는 링크
- 문서 내의 다른 지점으로의 링크



문서간 이동 <a>

■ <a> 요소 : 링크의 시작점 앵커를 표현

- href 속성 : 이동하고자 하는 목적지 문서의 파일 주소(URL)를 지정
- title 속성에는 말 풍선 창에 나올 설명을 기입

```
<a href="파일이름 혹은 URL 주소" title="설명"> 링크 텍스트</a>
```



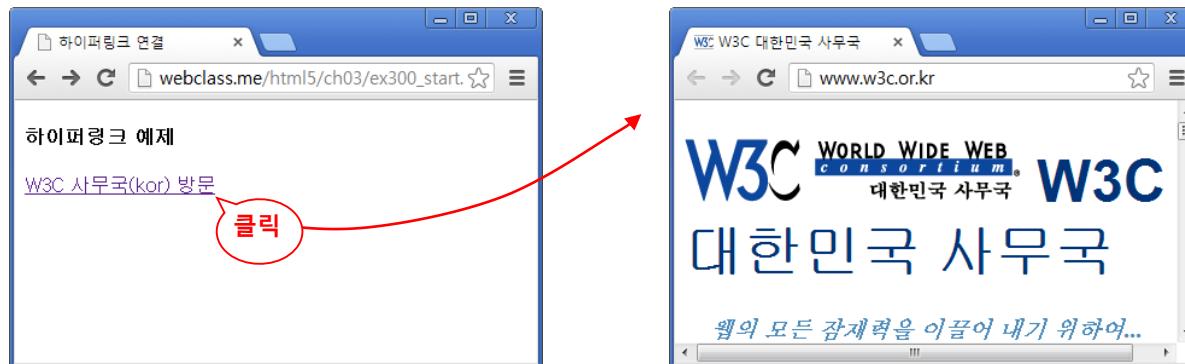
문서 간 이동 <a>

- href 속성 : 이동하고자 하는 문서의 위치 지정

- 절대 주소 : 다른 웹 사이트의 문서로 이동

- ▶ href 속성에 http://로 시작하는 URL 형식의 인터넷 주소 기입

W3C 사무국(kor) 방문



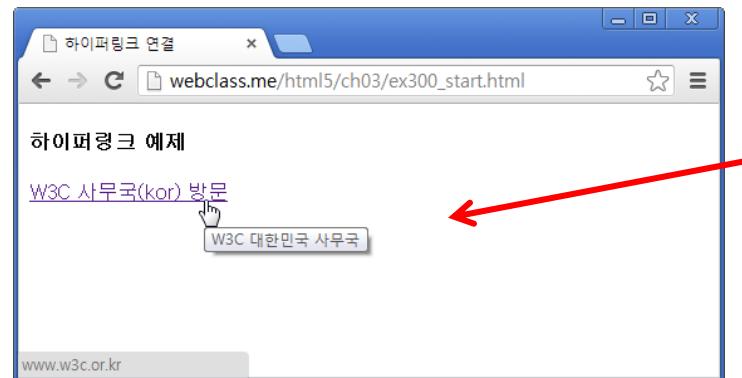
- 상대 주소
 - ▶ 현재의 문서와 같은 폴더의 위치에서부터 상대주소로 링크
- 책 목록

문서 간 이동 <a>

■ title 속성

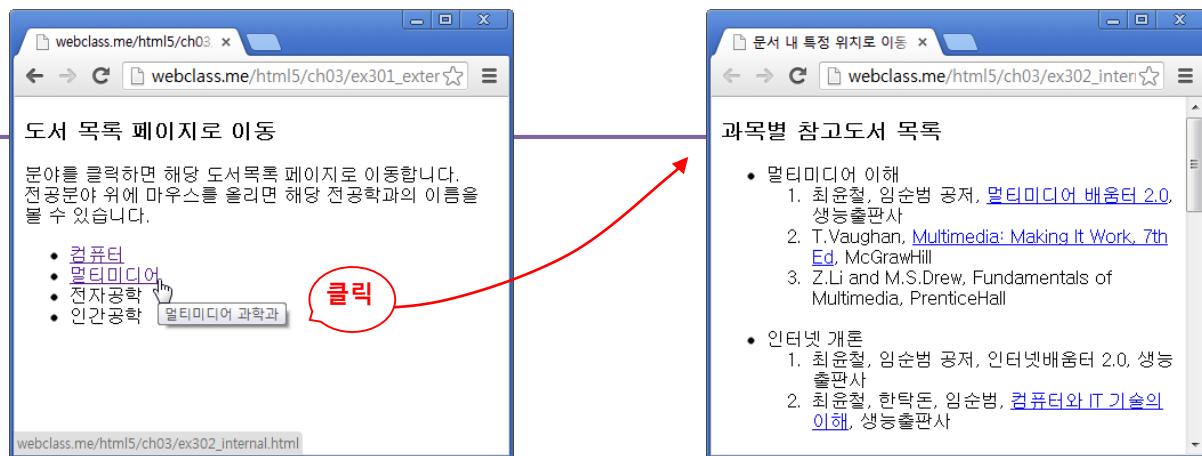
- 하이퍼링크에 대한 설명을 하고 싶을 때 사용
- 하이퍼링크 위에 마우스를 가져가면 말풍선에 설명 등장

```
<a href="http://www.w3c.or.kr" title ="W3C 대한민국 사무국">  
W3C 사무국(kor) 방문</a>
```



링크 예제 (1) : 문서간 이동하기

```
<!DOCTYPE html>
<html>
<body>
  <h3> 도서 목록 페이지로 이동</h3>
  <p> 분야를 클릭하면 해당 도서목록 페이지로 이동합니다.<br>
    전공 분야 위에 마우스를 올리면 해당 전공학과의 이름을 볼 수 있습니다.</p>
  <ul>
    <li> <a href="" title="컴퓨터 공학과"> 컴퓨터 </a> </li>
    <li> <a href="ex302_internal.html" title="멀티미디어 과학과"> 멀티미디어 </a> </li>
    <li>전자공학</li>
    <li>인간공학</li>
  </ul>
</body>
</html>
```



문서 내 특정 위치로 이동 <a>

■ 동일한 문서 내에서 특정 지점으로 연결

- 목적지인 특정 지점은 일종의 책갈피 링크
 - ▶ 예를 들어, 목차를 선택하면 원하는 책갈피 위치로 바로 간다
 - ▶ 문서가 길 경우 사용하면 효과적
- 목적지 앵커의 설정
` 문서 내 이동할 목적지 `
- 시작점 앵커의 설정
` 링크 설정된 ‘고유아이디’ 위치로 이동`

[노트: id 속성 사용]

이전 버전의 HTML에서는 name 속성을 사용

HTML5에서는 name 속성 대신 id 속성을 사용 권장



문서 내 특정 위치로 이동 <a>

■ 링크의 목적지 앵커 지정:

- id 속성 : 문서 내의 원하는 위치에 목적지 앵커를 설정
 - ▶ 이름을 지정하는 것이므로 <a> 와 사이에 텍스트 필요 없음

예) 멀티미디어 배움터 2.0

■ 시작점 앵커에서 링크 연결:

- href 속성에 목적지 앵커의 아이디를 지정
 - ▶ 목적지 앵커에서 아이디 설정할 때는 이름만 기입
 - ▶ 아이디를 이용할 때는 #으로 시작

예) 최윤철, 임순범 공저, 멀티미디어 배움터 2.0



링크 예제 (2) : 문서 내 위치로 이동

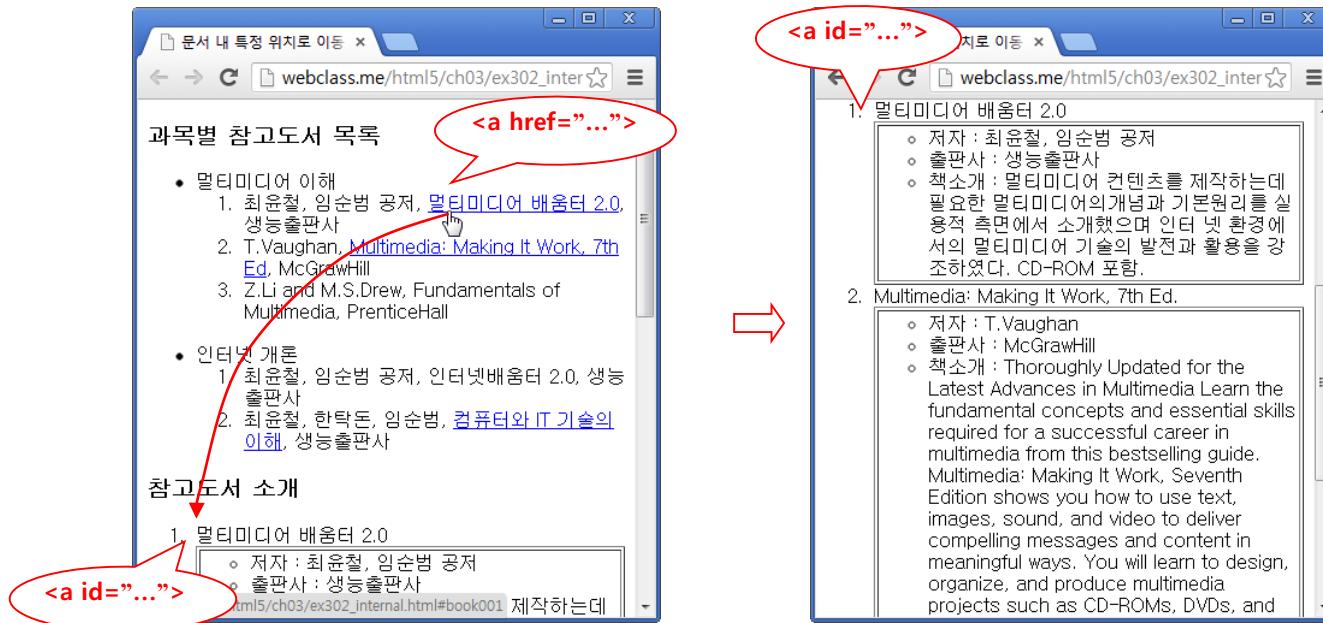
```
<ol>
  <li>최윤철, 임순범 공저, <a href="#book001">멀티미디어 배움터 2.0</a>, 생능출판사</li>
  <li>T.Vaughan, <a href="#book002">Multimedia: Making It Work, 7th Ed</a>,
McGrawHill</li>
  <li>Z.Li and M.S.Drew, Fundamentals of Multimedia, PrenticeHall</li>
</ol><br>
...
<h3>참고도서 소개</h3>
<ol>
  <a id="book001"></a><li>멀티미디어 배움터 2.0</li>
  <table border="1"> <tr><td>
    <ul> <li> 저자 : 최윤철, 임순범 공저</li>
      <li> 출판사 : 생능출판사</li>
      <li> 책소개 : 멀티미디어 컨텐츠를 제작하는데 필요한 멀티미디어의 개념과 ... </li>
    </ul>
  </td></tr> </table>

  <a id="book002"></a><li>Multimedia: Making It Work, 7th Ed.</li>
  <table border="1"> <tr><td>
    <ul> <li> 저자 : T.Vaughan</li>
      <li> 출판사 : McGrawHill</li>
      <li> 책소개 : Thoroughly Updated for the Latest Advances in Multimedia ... 생략 ...
    </ul>
  </td></tr> </table>
```



링크 예제 (2) : 문서 내 위치로 이동

■ 실행결과



<iframe>으로 다른 문서의 내용 표시

- <iframe> 요소 : 한 화면에서 링크로 연결된 내용 보기
 - 브라우저 페이지 내에 또 다른 브라우저 페이지 프레임을 삽입
 - 여기에서 링크된 문서의 내용을 확인

```
<iframe src="내부 프레임에 출력할 파일 경로"></iframe>
```

- <iframe> 요소의 속성 : src, width, height, name

```
<iframe src ="파일주소" width="폭" height="높이" name="이름">  
</iframe>
```

- src 속성 : 내부 브라우저 내부 프레임에 출력할 파일의 url 지정
- width와 height 속성 : 브라우저 프레임의 가로, 높이 크기
- name 속성 : 브라우저 프레임의 이름
 - ▶ <a> 요소의 target 속성에 <iframe>의 이름을 지정

<iframe>으로 다른 문서의 내용 표시

```
...
<ul>
    <li><a href="http://book.naver.com/bookdb/book_detail.nhn?bid=6232247"
target="intro">
        최윤철, 임순범 공저, 멀티미디어배움터 2.0</a> </li>
    <li>T.Vaughan, Multimedia: Making It Work, 7th Ed.</li>
    <li>Z.Li and M.S.Drew, Fundamentals of Multimedia</li>
</ul>
<li>인터넷 개론</li>
<ul>
    <li>최윤철, 임순범 공저, 인터넷배움터 2.0</li>
    <li><a href="http://book.naver.com/bookdb/book_detail.nhn?bid=5339292"
target="intro">
        최윤철, 한탁돈, 임순범, 컴퓨터와 IT 기술의 이해</a> </li>
    </ul>
</ol>
<dl>
    <dt>지정도서</dt>
    <dd>도서관에 여러권 비치되어 있으며 항상 열람 가능</dd>
</dl>
    <iframe src ="" name="intro" width="420" height="400"> </iframe>
</body>
```



<iframe>으로 다른 문서의 내용 표시

■ 실행결과

The image shows two browser windows illustrating the use of the `<iframe>` element.

Left Browser Window: A list of references for multimedia understanding. A red oval highlights the anchor tag ``. Another red oval highlights the `<iframe name="intro">` element in the page's source code. A red arrow points from the "target" attribute to the `<iframe>` element, indicating they refer to the same frame.

Right Browser Window: A screenshot of the Naver Book search results for "IT융합 시대의 멀티미디어 배움터 2.0". The book cover and details are visible, including the price of 25,000 won.

Page Navigation: The left side of the slide has a blue navigation bar with icons for back, forward, and search. The right side has a standard browser-style header with tabs, address bar, and menu.

3.2 이미지 사용하기

- 3.2.1 이미지 파일 종류
- 3.2.2 이미지 삽입

이미지 파일 종류

■ GIF(Graphic Interchange Format)

- JPEG, PNG에 비하여 파일 크기가 작다
- 표현 가능 색상이 256개이므로 자연스러운 장면에는 부적합
- 사진이 아닌 클립아트나 드로잉 같은 종류의 이미지에 적합

■ JPEG(Joint Photographic Experts Group)

- 24비트의 칼라를 사용하므로 수백만 개의 색상을 표현
- 복잡한 그림이나 사진 등 색상을 많이 사용하는 이미지에 적합

■ PNG(Portable Network Graphic)

- GIF와 JPEG 형식의 장점, 비압축 형식인 BMP 형식의 장점도 고려
- 24비트(또는 32비트) 칼라를 사용

이미지 파일 종류

■ 압축률 비교

원본 이미지 (BMP 포맷)	GIF 포맷	JPEG 포맷	PNG 포맷
 419KB (317x451픽셀)	 8.01KB (압축률 52.3)	 17.9KB (압축률 23.4)	 22.8KB (압축률 18.4)
 414KB (352x402픽셀)	 49.7KB (압축률 8.33)	 27.2KB (압축률 15.2)	 206KB (압축률 2.07)

이미지 삽입

■ 요소의 src, width, height 속성

- src 속성 : 이미지 파일의 이름을 지정
- width와 height 속성 : 크기를 조정하고 싶을 때 사용
- src 속성 : 파일의 경로 + 파일 이름

▶ 예

```
<img src ="images/multimedia.jpg">멀티미디어 배움터
```

```
<img src ="images/multimedia.jpg" width="70">멀티미디어 배움터
```

```
<img src ="images/multimedia.jpg" width="70" height="50">멀티미디어  
배움터
```



이미지 삽입

■ 요소의 alt 속성

- “alternate text(대체 텍스트)”의 약어
- 이미지에 대한 설명 텍스트를 지정
 - ▶ 화면에 이미지를 로드 못할 경우 그 위치에 텍스트가 대신 출력
 - ▶ 이미지 파일의 주소가 잘못되거나, 인터넷 연결이 너무 느려서 미처 이미지를 표시하지 못하는 경우 등
 - ▶ 예, 멀티미디어 배움터



[노트: 의 align 속성]

이전 버전의 HTML에서는 요소의 align 속성으로 이미지 위치를 정렬
HTML5에서는 요소에서 CSS 스타일지정으로 위치를 설정하도록 권장

<figure>, <figcaption>

■ <figure> 요소

- 그림, 사진, 다이어그램과 텍스트 등의 콘텐츠를 함께 묶어서 하나의 독립된 단위로 취급하고 싶을 때 사용

■ <figcaption> 요소

- <figure> 요소를 위한 제목을 표현, <figure> 요소 내에 위치

```
<figure>
  <p>멀티미디어 배움터 2.0</p>
  <br>
  <figcation>[그림 1] 책 소개</figcaption>
</figure>
```



■ 이미지에 하이퍼링크 연결

```
<a href =”링크할 곳의 파일이름”>
<img src=”이미지 파일이름”></a>
```

이미지 삽입 예제

```
...
<table border="1"> ...
<tr>
  <td>
    <figure>
      <a href="http://book.naver.com/bookdb/book_detail.nhn?bid=6232247">
        <br>
        <figcation> 멀티미디어 배움터</figcation></a>
      </figure>
    </td>
    <td> 최윤철, 임순범 공저 </td>
    <td> 생능출판사 </td>
  </tr>
  <tr>
    <td>
      <figure>
        <a href="http://book.naver.com/bookdb/book_detail.nhn?bid=6746965">
          <br>
          <figcation> 스티브 잡스</figcation></a>
        </figure>
      </td>
    ...
  ...
</table>
```



이미지 삽입 예제

■ 실행결과



3.3 오디오와 비디오 다루기

- 3.3.1 지원하는 오디오/비디오 파일 형식
- 3.3.2 오디오 삽입하기
- 3.3.3 비디오 삽입하기

지원하는 오디오/비디오 파일 형식

- MP3 (*.mp3) : MPEG Audio Layer-3
 - MPEG-1의 오디오 규격으로 개발된 형식, 대중적으로 널리 사용
- Wave (*.wav, *.wave)
 - 마이크로소프트와 IBM이 개발, 비압축 방식의 오디오 형식
- MPEG4 (*.mp4, *.m4v)
 - MPEG-4의 part14에서 규정된 비디오 파일 형식, H.264 코덱 사용
- Ogg (*.ogg, *.ogv)
 - 스트리밍 방식의 멀티미디어 표현을 위한 공개소스 기반 형식
 - Vorbis, FLAC 등의 오디오 코덱, Ogg Theora 등의 비디오 코덱 사용
- WebM (*.webm)
 - 구글이 HTML5의 동영상에 사용하기 위해 최근에 개발



지원하는 오디오/비디오 파일 형식

■ 웹브라우저에서 오디오/비디오 코덱의 지원 현황

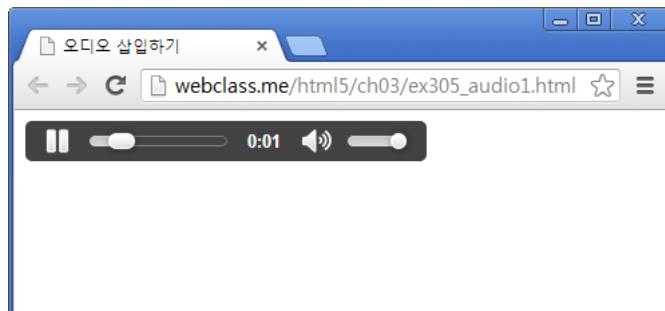
코덱	웹브라우저				
	 크롬 (버전6 이상)	 파이어폭스 (버전3.6 이상)	 익스플로러 (버전9 이상)	 사파리 (버전5 이상)	 오페라 (버전10.6 이상)
mp3	지원	미지원	지원	지원	미지원
Wav	지원	지원	미지원	지원	지원
Ogg/Theora	지원	지원	미지원	미지원	지원
mp4 (H.264)	지원	미지원	지원	지원	미지원
WebM	지원	미지원	미지원	미지원	지원

오디오 삽입하기 <audio>

```
<audio controls autoplay src="재생할 사운드 파일 이름">
```

- src 속성 : 사운드 파일 이름을 지정
- controls 속성 : 기본적인 미디어 제어기를 표시할지 여부를 지정
- autoplay 속성 : 파일이 로드되자마자 자동으로 재생시킨다는 의미
- loop 속성 : 사운드를 반복 재생시킬 횟수를 지정

```
<audio controls src="song.mp3" loop="3" autoplay>  
브라우저에서 <audio>를 지원하지 않습니다. <br>  
(song.mp3파일이 3회 자동재생 됩니다.)  
</audio>
```



오디오 삽입하기 <audio>

■ <source> 요소와 같이 사용하기

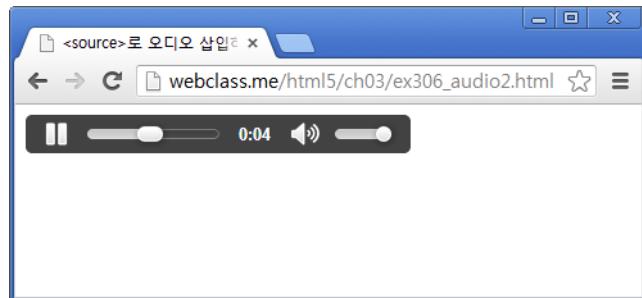
- 브라우저에서 오디오 파일이 지원되지 않는 경우를 대비
- <source> 요소에서 같은 내용을 여러 형식으로 작성한 파일 지정
- src 속성 : 오디오 파일의 이름 지정
- type 속성 : 오디오 파일의 MIME 형식을 지정
 - ▶ 예, “audio/mp3, audio/ogg, audio/wav” 와 같이 지정

```
<audio controls autoplay>
  <source src="song.mp3" type="audio/mp3">
  <source src="song.ogg" type="audio/ogg">
</audio>
```

- ▶ 웹브라우저에서 오디오 파일 로드 전에 재생가능 여부를 확인
- ▶ 가장 앞에 있는 파일의 형식부터 재생이 가능한지 확인

오디오 예제 : <audio>와 <source>

```
<body>
  <audio controls autoplay>
    <source src="song.mp3" type="audio/mp3">
    <source src="song.ogg" type="audio/ogg">
    <source src="song.wav" type="audio/wav">
    브라우저에서 &lt;audio&gt; 요소, 혹은 mp3/ogg/wav 를 지원하지 않습니다.
  </audio>
</body>
```



오디오 삽입하기 <audio>

■ <audio> 요소의 preload 속성

- 미리 로드 되어야 하는지의 여부를 지정
- preload 속성값
 - ▶ auto(기본값) : 페이지를 로드하고 바로 오디오 파일을 다운로드
 - ▶ metadata : 사용자가 재생 시키기 전까지는 오디오의 크기, 관련 정보 등과 같은 메타데이터만 다운로드
 - ▶ none : 재생을 시작 하기 전까지 오디오 파일을 다운로드 안함

비디오 삽입하기 <video>

```
<video controls src="비디오 파일 이름" width="폭" height="높이" >
```

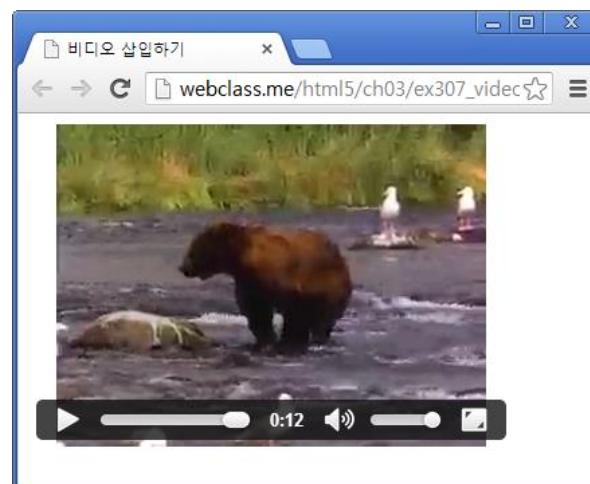
■ <video> 요소의 속성

- src, controls, loop, autoplay 속성: <audio> 요소의 속성과 동일
- width, height 속성: 화면에서 비디오가 표시될 영역의 크기를 설정
- videoWidth, videoHeight 속성: 비디오 자체의 너비와 높이를 반환
- Poster 속성: 동영상이 로딩되고 있을 때 보여줄 이미지를 지정
- preload 속성: 브라우저가 미리 동영상을 로딩 할지 지정



비디오 예제 1: 비디오 삽입

```
<html>
  <head> <title> 비디오 삽입하기 </title> </head>
  <body>
    <video controls autoplay width="360" height="240" src="bear.mp4">
      비디오를 재생할 수 없습니다.
    </video>
  </body>
</html>
```



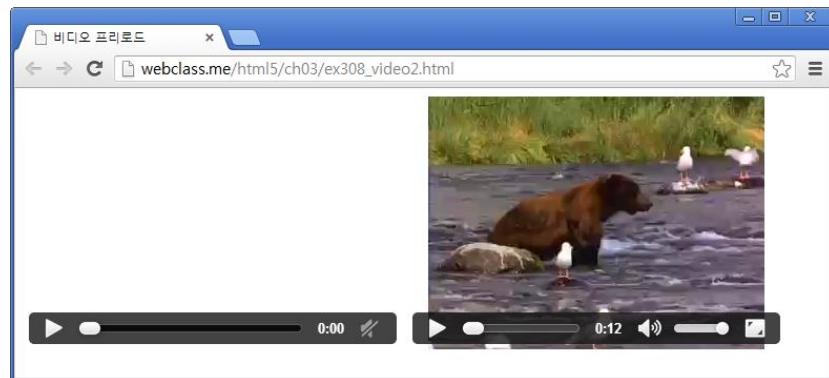
비디오 미리 로딩하기

■ preload 속성

- auto(기본값) : 웹브라우저가 페이지를 로드하고 바로 비디오 파일을 다운로드
- metadata : 사용자가 재생 시키기 전까지는 비디오의 크기, 첫 프레임, 비디오 관련 정보 등과 같은 메타데이터만 다운로드
- none : 재생을 시작 하기 전까지 비디오 파일을 다운로드 안함

비디오 예제 2: 비디오 미리 로딩

```
<body>
  <video width="360" height="240" src="bear.mp4" controls preload="none">
    비디오를 재생할 수 없습니다.
  </video>
  <video width="360" height="240" src="bear.mp4" controls preload="metadata">
    비디오를 재생할 수 없습니다.
  </video>
</body>
```



4장. 다양한 입력 폼

4.1 폼 이해하기

4.2 기본 형식으로 입력하기

4.3 고급 형식으로 입력하기

4.1 폼 이해하기

<form> 요소의 사용

■ 폼 요소의 사용

- 회원가입, 상품구매, 키워드 검색 등 사용자로부터 정보를 받을 때
- 사용자와 애플리케이션이 상호작용
 - ▶ 사용자가 원하는 내용을 입력 => 전송버튼을 통해 데이터를 전송 => 애플리케이션으로 전달 => 작업 처리 후에 실행 결과를 반환

■ <form> 요소의 역할

- 사용자가 입력하는 정보를 하나로 묶어서 애플리케이션에 전달할 수 있도록 다양한 입력 양식을 그룹핑 하고 전송 방법을 설정
- <form> 요소 내 사용자의 정보 입력 양식
 - ▶ <input>, <textarea>, <select>, <button> 등의 입력 요소를 이용



다양한 입력 폼 예제

```
<h3>다양한 입력 폼</h3>
<form method="get" action="form_app.js">
    성명: <input type="text" name="person"/> <br>
    성별: <input type="radio" name="sex" value="male"/> 남성
           <input type="radio" name="sex" value="female"/> 여성 <br>
    직업: <select name="job" size="1">
        <option>학생</option> <option>회사원</option>
        <option>공무원</option> <option>기타</option>
    </select>
    <p> 구입희망분야(복수선택 가능)<br>
    - 분야: <input type="checkbox" name="books" value="computer"/> 컴퓨터
           <input type="checkbox" name="books" value="economy"/> 경제
           <input type="checkbox" name="books" value="common"/> 상식 <br>
    비고: <br>
        <textarea name="comments" rows="4" cols="40"/> </textarea>
    </p> <hr>
    <input type="submit" value="신청"/>
    <input type="reset" value="취소"/>
</form>
```

다양한 입력 폼

성명:

성별: 남성 여성

직업:

구입희망분야(복수선택 가능)

- 분야: 컴퓨터 경제 상식

비고:

다양한 입력 폼

성명: 홍길동

성별: 남성 여성

직업:

구입희망분야(복수선택 가능)

- 분야: 컴퓨터 경제 상식

비고:

손쉬운 경제 살살이 필요합니다.

<form> 요소의 주요 속성

```
<form name="폼 이름" method="get/post" action="애플리케이션 주소">
```

- method 속성 : 데이터들이 전송되는 방식을 지정한다.
 - ▶ 전송 방식의 종류는 get방식과 post방식
- action 속성 : 데이터를 처리할 애플리케이션 프로그램의 주소
 - ▶ 웹서버 프로그램인 경우에는 URL 주소
- name속성 : 폼 요소에 대한 이름을 지정

■ HTML 문서에서 <form> 의 역할

- 다양한 <input> 요소
 - => 입력된 데이터를
 - => 애플리케이션에 전달
 - => 그 실행 결과를 받는 것



get방식과 post방식

▶ Get방식

- 전송할 데이터를 URL 주소에 포함하여 문자열 형태로 전달
- URL 뒤에 ?에 이어서 "변수명=값", 변수가 여러 개일 경우는 &로 구분
- 한글은 16비트 유니코드로 표현

...

ch04/application.js?person=%C8%AB%B1%E6%B5%BF&sex=male&job=

...

- get 방식은 간단한 데이터를 전달할 때에는 편리
- 그러나 주소 창에 전달하는 값이 노출되어 보안에 취약

▶ Post방식

- 프로그램의 입출력 방식을 사용하여 데이터의 양에 제한이 없다
- 전송하는 데이터가 드러나지 않으므로 보안이 필요한 경우에 많이 사용



4.2 기본 형식으로 입력하기

4.2.1 텍스트 입력

4.2.2 선택항목의 입력

4.2.3 버튼 입력

4.2.4 기타 입력 필드

4.2.5 입력 필드의 그룹핑

입력 폼의 형태

■ 기본적인 입력 폼의 형태

- <form> 요소 안에 <input>, <textarea>, <select>, <button> 등 요소

■ <input> 요소의 속성

<input type="입력 형식" name="변수명", value="입력 값"/>

- name 속성은 입력 요소의 이름
- value 속성은 입력 요소의 값
- type 속성은 입력 폼의 유형을 결정

텍스트 관련	text (문자열 입력 필드), password (암호 문자열)
선택 관련	radio (단일 선택), checkbox (복수 선택)
버튼	submit (전송 버튼), reset (초기화 버튼), button (임의 기능 버튼), image (이미지를 전송 버튼으로 정의)
기타	file (파일 선택), hidden (숨김 필드)



텍스트 입력

■ 문자열 입력 필드

```
<input type="text" name="변수명" value="초기값"/>
```

- 한 줄의 문자열을 입력 받는 가장 기본적인 입력 형식
- name 속성 : 애플리케이션에 전달될 변수명을 지정
- value 속성 : 사용자로부터 입력 받은 값을 전달
 - ▶ 문자열 입력창에 초기 문자를 보여주려면 value 속성에 값을 지정

■ 암호 입력 필드

```
<input type="password" name="변수명"/>
```

- 비밀번호나 주민등록번호 등 보안이 필요한 문자 입력
- 사용자 입력 데이터가 남들이 볼 수 없도록 ‘•’으로 표시
 - ▶ 화면에 표시되는 형식만 다를 뿐 더 이상의 보호 기능은 없다.



텍스트 입력

■ 텍스트 영역 필드 : <textarea> 요소

- 여러 줄에 걸치는 텍스트를 입력
- name 속성 : <textarea> 요소의 이름을 지정
- cols 속성 : 텍스트 영역의 한 줄에 해당하는 문자수, 즉, 열의 개수
- rows 속성 : 텍스트 영역의 행의 개수
 - ▶ 표시 영역보다 많은 텍스트를 입력하면 스크롤바
 - ▶ 텍스트 영역에 초기 문장은 <textarea>와 </textarea> 사이에 기입

```
<textarea name="이름" cols="열의 수" rows="행의 수">  
    텍스트 영역에 표시되는 초기 문장  
</textarea>
```



아이디와 비밀번호, 요청사항 입력

```
<h3> 문자열, 암호 입력 및 텍스트 영역</h3>
<form method="post" action="form_app.js">
    아이디 : &nbsp; <input type="text" name ="id" value="...ID 입력..."> <br>
    비밀번호: <input type="password" name="pwd">
    <p>요청사항: <br>
        <textarea name="comment" cols="40" rows="5"> 전달하실 내용을 적으세요:<br>
        </textarea>
    </p>
</form>
```

선택 항목의 입력

■ 라디오 버튼

```
<input type="radio" name="변수명" value="선택값"/>
```

- 여러 항목 중에서 하나만 선택
- name속성과 value속성을 반드시 지정
 - ▶ 동일한 그룹의 라디오 버튼은 name속성을 동일하게
 - ▶ 즉, name 속성값이 같은 그룹에서 하나만 선택
 - ▶ 선택된 라디오 버튼의 value 속성값이 애플리케이션에 전달
- checked 속성 : 초기화면에서 미리 선택해 놓도록 설정
 - ▶ 회원과 남자를 선택한 경우 “member=yes&sex=male”의 형태로 전달

회원여부:

```
<input type="radio" name="member" value="yes" checked/>회원
```

```
<input type="radio" name="member" value="no"/>비회원<br>
```

성별 : <input type="radio" name="sex" value="male"/>남성

```
<input type="radio" name="sex" value="female"/>여성
```

회원여부: 회원 비회원
성별 : 남성 여성



선택 항목의 입력

■ 체크박스 선택

```
<input type="checkbox" name="변수명" value="선택값"/>
```

- 여러 항목에 대하여 개별적으로 선택, 여러 개 선택 가능
- name 속성값은 모두 같은 값, value 값은 고유한 값
- checked 속성 : 초기 화면에 기본적으로 체크

▶ 체크박스에 표시된 항목의 value 속성값들이 애플리케이션으로 전송

취미(중복선택) :

```
<input type="checkbox" name="hobby" value="read"/>독서  
<input type="checkbox" name="hobby" value="movie" checked/>영화  
<input type="checkbox" name="hobby" value="music"/>음악  
<input type="checkbox" name="hobby" value="sports"/>스포츠
```

취미(중복선택) : 독서 영화 음악 스포츠

선택 항목의 입력

■ 선택목록에서 선택 : <select> 요소 내에 <option> 항목

- 여러 개의 목록 중에서 하나를 선택
 - ▶ 드롭다운 형태 혹은 스크롤 박스 형태의 선택목록에서 항목을 선택
 - ▶ name 속성 : 선택목록의 변수명
 - ▶ size 속성 : 사용자들에게 보여줄 항목의 개수
 - size 속성이 1이면 드롭다운 목록, 2 이상이면 스크롤 박스의 크기
 - ▶ multiple 속성 : 하나 이상의 항목을 선택 가능하도록 설정
- 각 항목은 <select> 요소 내에서 <option> 요소로 정의
 - ▶ value 속성 : 각 항목을 구별하기 위한 값
 - ▶ selected 속성 : 초기 화면에서 특정 항목을 기본값으로 지정

```
<select name="job" size="1">
  <option value="student" selected>학생</option>
  <option value="company">회사원</option>
  <option value="teacher">교사</option>
  <option value="sales">자영업</option>
  <option value="others">기타</option>
</select>
```

작업:

작업:
학생
회사원
교사
자영업
기타

작업:
학생
회사원
교사
자영업

버튼 입력

■ 전송 버튼

```
<input type="submit" value="버튼라벨"/>
```

- 사용자가 입력한 값을 애플리케이션으로 전달
 - ▶ <form> 요소의 영역 안에 있는 모든 입력 데이터가 <form>의 action 속성에서 지정한 애플리케이션 프로그램으로 전송
- value 속성 : 버튼에 표시하고 싶은 라벨 지정

■ 초기화 버튼

```
<input type="reset" value="버튼라벨"/>
```

- 사용자가 폼에 입력한 데이터 값들은 모두 초기화
- value 속성 : 버튼에 표시할 라벨 지정



버튼 입력

■ 일반 버튼

```
<input type="button" value="버튼라벨"/>
```

- 다양한 용도로 사용할 수 있는 버튼
- value 속성값 : 버튼에 표시할 라벨 지정

■ 이미지 버튼

```
<input type="image" src="이미지 파일" alt="문자열"/>
```

- 버튼 모양이 맘에 들지 않는 경우 원하는 이미지로 대체
- src 속성 : 이미지의 경로
- alt 속성 : 이미지에 문제가 있을 때 보여 줄 대체 텍스트

버튼 입력

```
<form method="get" action="form_app.js">
<p>취미(중복선택) :
    <input type="checkbox" name="hobby" value="read"/>독서
    <input type="checkbox" name="hobby" value="movie" checked/>영화
    <input type="checkbox" name="hobby" value="music"/>음악
    <input type="checkbox" name="hobby" value="sports"/>스포츠
</p>
직업:
<select name="job" size="4" multiple>
    <option value="student" selected>학생</option>
    <option value="company">회사원</option>
    <option value="teacher">교사</option>
    <option value="sales">자영업</option>
    <option value="others">기타</option>
</select>
<hr>
<input type="submit" value="전송하기"/> &nbsp;
<input type="reset" value="초기화"/> &nbsp;
<input type="button" value="확인하기" onClick="alert('입력값 확인')"/> &nbsp;
<input type="image" src="help.gif" alt="전송 버튼"/>
</form>
```

HTML



버튼 예제

취미(중복선택) : 독서 영화 음악 스포츠

직업: 회사원
교사
자영업
기타

전송하기 초기화 확인하기 Submit

버튼 예제

취미(중복선택) : webclass.me의 페이지 내용:

취미
입력값 확인

직업: 자영업

전송하기 초기화 확인하기 Submit

기타 입력 필드

■ 파일 선택하기 : <input type="file"/>

- 사용자가 폼 입력에서 파일을 선택

- ▶ 초기 화면에는 파일 경로를 보여줄 문자열 필드와 “파일선택” 버튼
- ▶ 버튼을 누르면 파일 선택창 등장, 문자열 필드에 선택된 파일 이름

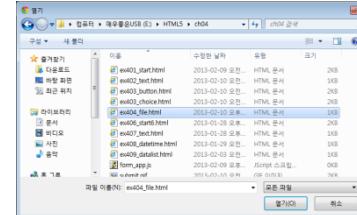
파일 업로드하기


```
<input type="file" name="myfile"/>
```

크롬 브라우저

파일 업로드하기

[파일 선택] 선택된 파일 없음



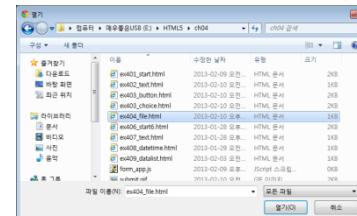
파일 업로드하기

[파일 선택] ex04_file.html

파이어폭스 브라우저

파일 업로드하기

[찾아보기 ...]



파일 업로드하기

[찾아보기 ...]

기타 입력 필드

■ 데이터 숨김

```
<input type="hidden" name="변수명" value="값" />
```

- 사용자가 입력하거나 선택하는 데이터가 아니라 시스템에서 특정 데이터 항목을 입력 받아 처리하고 싶은 경우가
- 사용자의 아이피 주소나 이미 넘겨받은 데이터를 다시 보여줄 필요가 없을 때
- 화면에 아무것도 보여주지 않지만, name과 value 속성값을 전달

기타 입력 필드

■ 텍스트 라벨

```
<label for="입력아이디">
```

- 텍스트 라벨과 특정 입력 필드를 연결
 - ▶ 해당 글자를 보여주는 것 이외에 데이터 전달에는 영향이 없다
- for 속성과 이와 연결되는 입력 양식의 id 속성에 동일한 아이디값

```
성별: <label for="male">남성</label>
```

```
  <input type="radio" name="sex" id="male" value="male">
```

```
  <label for="female">여성</label>
```

```
  <input type="radio" name="sex" id="female" value="female">
```

텍스트 라벨을 클릭해도 됩니다.

성별: 남성 여성

텍스트 라벨을 클릭해도 됩니다.

성별: 남성 여성

입력 필드의 그룹핑

■ 그룹핑 : <fieldset> 요소

- 폼 양식을 그룹핑하는 범위를 지정
 - ▶ 사용자의 시각적 편의를 위해서 제공
 - ▶ 입력 필드의 그룹 주위에 기본 스타일로 테두리 선
- name 속성 : 그룹의 이름을 지정하는
- form 속성 : 폼과 연결
- disabled 속성 : 그룹 내의 모든 하위 입력 요소들을 비활성화

■ 그룹의 라벨 : <legend> 요소

- 그룹을 구분하기 위한 제목 라벨
 - ▶ <fieldset> 요소에 포함되는 첫 번째 자식 요소로서 한번만 사용
- 그룹 라벨의 스타일은 그룹을 구분하는 선의 중간에 걸쳐서 표시



도서 검색 예제

```
<form method="post" action="form_app.js">
  <fieldset>
    <legend>로그인</legend>
    <label for="user_id">아이디 : </label>
    <input type="text" name="id" size="20" id="user_id"> <br>
    <label for="user_pw">비밀번호 :</label>
    <input type="password" name="pw" size="20" id="user_pw">
  </fieldset> <br>
  <fieldset>
    <legend>통합 검색</legend>
    <label for="book_name">도서명 : </label>
    <input type="text" name="book_search" size="50" id="book_name">
    <br>검색범위 :
    <input type="radio" name="s_type" value="keyword" id="keyword">
    <label for="keyword">키워드</label>
    <input type="radio" name="s_type" value="content" id="content">
    <label for="content">본문내용</label>
    <br>자료유형 :
    <input type="checkbox" name="d_type" value="all">전체
    <input type="checkbox" name="d_type" value="book">단행본
    <input type="checkbox" name="d_type" value="paper">학술지
    <input type="checkbox" name="d_type" value="non_book">비도서<br>
  </fieldset> <br> ...
```



도서 검색 예제

```
<button type="submit">검색</button>
<button type="reset">지우기</button>
</form>
```

The screenshot shows a web browser window with the title '그룹핑 예제'. The address bar displays the URL 'webclass.me/html5/ch04/ex404_group.html'. The main content area contains a form titled '도서 검색'. The form is divided into two sections: '로그인' and '통합 검색'. The '로그인' section contains two input fields labeled '아이디:' and '비밀번호:'. The '통합 검색' section contains an input field labeled '도서명:', a radio button group for '검색범위' (with '키워드' selected), and a checkbox group for '자료유형' (with '전체' selected). At the bottom of the form are two buttons: '검색' and '지우기'.

4.3 고급 형식으로 입력하기

4.3.1 서식이 있는 텍스트 입력

4.3.2 날짜와 시간 입력

4.3.3 색상 및 숫자 입력

4.3.4 고급 입력 요소

<input> 요소에 추가된 입력 형식

■ text, password, radio, checkbox, button 외에 새로운 양식

텍스트 관련	email (이메일 주소), URL (URL 주소), tel (전화번호), search (검색창 입력)
날짜와 시간	date (날짜), month (연도/월), week (연도/주), time (시간), datetime (UTC time zone의 날짜/시간), datetime-local (Time zone이 없는 날짜/시간)
색상 및 숫자	number (숫자), range (일정 범위의 수), color (색상 선택)

■ <input> 요소에 새로운 속성

- ▶ autocomplete 속성 : 자동으로 사용자의 입력을 완성
- ▶ placeholder 속성 : 입력 값에 대한 설명을 희미하게 표시
- ▶ required 속성 : 사용자가 필수적으로 입력을 하도록 검증
- ▶ autofocus 속성 : 문서가 로드 될 때 입력 영역에 마우스 커서 표시
- ▶ step 속성 : 입력 필드의 숫자나 범위를 조절하는 단계를 지정

추가 입력요소 및 유효성 검사

■ <input> 요소 이외에 추가된 입력 요소

- <output> 요소 : 폼의 처리 결과
- <datalist> 요소 : 입력 양식에 대한 내용을 옵션 리스트로 제공
- <keygen> 요소 : 암호화 키를 생성

■ 유효성 검사

- 입력값이 형식에 맞게 입력되었는지 검사하는 기능
 - ▶ 예, 이메일 입력 필드, 필수적으로 입력해야 하는 필드
- 유효성 검사 대상
 - ▶ <input> 요소, <select> 요소, <textarea> 요소
 - ▶ <button> 요소의 경우 submit 형식일 때
- required 속성 : 오류 발생시 메시지를 표시하고 폼 전송 중단
- novalidate 속성 : 유효성 검사 대상에서 제외



서식이 있는 텍스트 입력

■ 이메일 주소 입력 : <input type="email"/>

- 이메일을 입력 받기 위한 입력창
 - ▶ 이메일 주소의 형식이 ****@**.*에 맞게 작성되었는지 확인
 - ▶ 이메일 주소가 존재하는지 까지는 미검사
- multiple 속성 : 여러 개의 이메일 주소를 입력, 콤마로 구분

Email :

전송

Email :

전송

! 이메일 주소를 입력하세요.

■ URL 주소 입력 : <input type="url"/>

- 웹 주소의 URL을 직접 입력하는 경우를 지원
 - ▶ 인터넷 주소 표기에 맞는 'http://' 형식으로 입력되었는지 확인
- value 속성에 "http://"를 지정하여 입력창에 미리 표시 가능

URL :

전송

URL :

전송

! URL을 입력하세요.

URL :

전송

서식이 있는 텍스트 입력

■ 전화번호 입력 : <input type="tel" />

- 전화번호를 입력할 수 있는 입력 창
- pattern 속성 : 원하는 전화번호와 자리수에 유효한 패턴을 지정
 - ▶ 정규 표현식으로 정의, 자바스크립트에서 사용하는 패턴과 동일
- placeholder 속성 : 입력할 자리수를 표시 (단순히 표시만)

```
Tel : <input type="tel" placeholder="00*-000*-0000"  
pattern="[0-9]{2,3}-[0-9]{3,4}-[0-9]{4}" />
```

Tel :

[전송]

Tel :

[전송]

Tel :

[전송] ❌ 요청한 형식과 일치시키세요.

[노트: pattern 속성의 예]

pattern="[A-Za-z]{no}" : no개 만큼의 영문자를 입력

pattern="[0-9]{no}" : no개 만큼의 숫자를 입력

pattern="[A-Za-z0-9]{min, max}" : 영문자와 숫자를 min~max 만큼의 글자 수 입력

pattern="[0-9]{" : 숫자를 1개 이상 입력



서식이 있는 텍스트 입력

■ 검색창 입력 : <input type="search"/>

- 검색을 위한 문자열을 입력할 수 있는 입력 필드
- 외형적으로 사용자 인터페이스에 차이
- 입력한 검색어 취소

Search :

Search : ×

날짜와 시간 입력

■ 날짜 입력(일, 월, 주)

- <input type="date"/> 연-월-일의 날짜를 입력
- <input type="month"/>연-월만 입력
- <input type="week"/> 연-주를 입력 (번호순서 1월 첫 주가 'W01')
- min 속성 : 최소값, max 속성 : 최대값, value 속성 :초기값
 - ▶ 속성값은 '연-월-일' 형식으로 날짜 입력

A screenshot of a web browser window titled '날짜 입력 예제'. The URL is 'webclass.me/html5/ch04/ex405_date.html'. The page contains three input fields: 'date (연-월-일)', 'month (연도-월)', and 'week (연도-주)'. The 'date' field has a dropdown menu set to '연도-월-일' with a yellow border. Below it is a calendar for April 2013. The date '28' is highlighted in blue. At the bottom are '오늘' and '삭제' buttons.

A screenshot of a web browser window titled '날짜 입력 예제'. The URL is 'webclass.me/html5/ch04/ex405_date.html'. The page contains three input fields: 'date (연-월-일)', 'month (연도-월)', and 'week (연도-주)'. The 'date' field has a dropdown menu set to '연도-월-일' with a yellow border. Below it is a calendar for April 2013. The date '28' is highlighted in blue. At the bottom are '오늘' and '삭제' buttons.

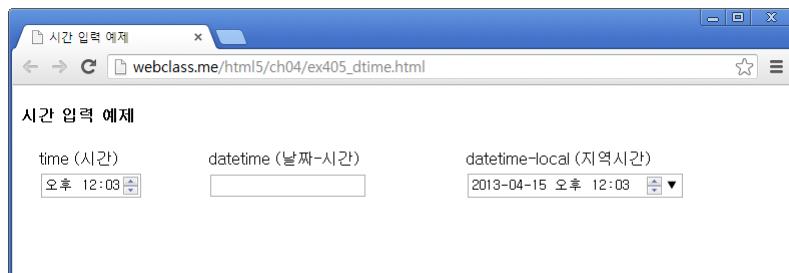
A screenshot of a web browser window titled '날짜 입력 예제'. The URL is 'webclass.me/html5/ch04/ex405_date.html'. The page contains three input fields: 'date (연-월-일)', 'month (연도-월)', and 'week (연도-주)'. The 'week' field has a dropdown menu set to '연도-주' with a yellow border. Below it is a calendar for April 2013. The week starting from April 1st is highlighted in blue. At the bottom are '오늘' and '삭제' buttons.

날짜와 시간 입력

■ 시간 입력

- <input type="time"/>
 - ▶ 시간을 입력, “위/아래()” 버튼을 통해 시간과 분을 별도 조정
- <input type="datetime"/>
 - ▶ 날짜와 시간 선택, UTC(Universal Time Coordinated) 국제표준 시간대
- <input type="datetime-local"/>
 - ▶ 원하는 지역의 현지 시간 입력

time, datetime, datetime-local 형식의 예



날짜와 시간 입력 예제

```
<form method="get" action="form_app.js">
    성명 : <input type="text" name="p_name"/> <br>
    전화 : <input type="tel" name="p_tel" placeholder="00*-000*-0000"
            pattern="[0-9]{2,3}-[0-9]{3,4}-[0-9]{4}" /> <br>
    이메일 : <input type="email" name="p_mail" placeholder="***@***.***"/>
    <p>
        도서명 : <input type="text" size="25" name="book_title"/><br>
        예약 희망일 : <input type="date" name="last_date" min="2013-01-30"><br>
        수령 시간 : <input type="time" name="time_from" min="09:00" max="18:00">
        에서 <input type="time" name="time_until" min="09:00" max="18:00">사이<br>
    <hr>
    <input type="submit" value="예약하기"/>
</form>
```

초기상태

도서 예약 대출

성명 :

전화 :

이메일 :

도서명 :

예약 희망일 :

수령 시간 : 에서 사이

입력오류

도서 예약 대출

성명 :

전화 :

이메일 :

도서명 :

예약 희망일 :

수령 시간 : 에서 사이

값은 09:00 이상이어야 합니다.



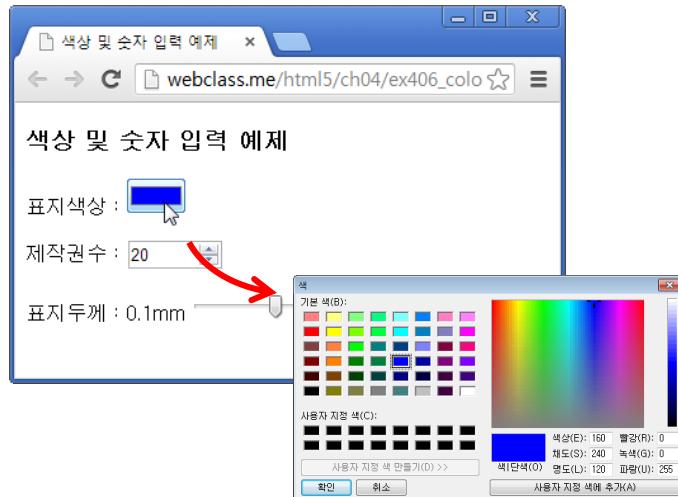
색상 및 숫자 입력

■ 색상 입력 : <input type="color"/>

- 사용자가 직접 색상을 선택하는 색상 선택메뉴 제공

- ▶ 크롬 브라우저

- 색상 입력 양식을 클릭하면 전체 색상을 보여주는 시스템 색상메뉴 등장



색상 및 숫자 입력

■ 숫자 입력 : <input type="number"/>

- 화살표 버튼으로 숫자를 조정할 수 있는 입력 형식
- 최소값 min 속성, 최대값 max 속성, 초기값 value 속성
- step 속성은 증가/감소 간격을 지정

```
<input type="number" min="0" max="100" step="10" value="20"/>
```

제작권수 : [닫기]

■ 범위 입력 : <input type="range"/>

- 스크롤바를 움직여서 일정한 범위의 숫자를 입력
- 최소값 min 속성, 최대값 max 속성
- 막대가 움직이는 칸의 간격 값은 step 속성

0.1mm <input type="range" min="1" max="5" value="3" /> 0.5mm

표지 두께 : 0.1mm  0.5mm

고급 입력 요소

■ 데이터 목록 기능 : <datalist> 요소

- 검색어 자동완성 혹은 제시어 기능 구현 가능
 - ▶ <input> 요소의 list 속성과 <datalist> 요소의 id 속성을 동일하게 지정
- 데이터목록의 옵션 항목들이 제시어 목록으로 사용
 - ▶ 입력창에 포커스가 들어오면 옵션 목록이 등장

```
참가국 : <input type="text" size="12" list="country" />
<datalist id="country">
    <option value="스페인"/>
    <option value="영국"/>
    <option value="독일"/>
</datalist>
```

참가국 :

참가국 :

The diagram illustrates the interaction between an input field and a dropdown menu. On the left, a simple input field labeled "참가국 :" contains a placeholder. An arrow points from this state to the middle state. In the middle state, the input field is empty, but a small blue rectangular box is positioned above it, representing the dropdown menu. This menu contains three options: "스페인", "영국", and "독일", each in a separate row. The option "영국" is highlighted with a blue background, indicating it is selected. Another arrow points from the middle state to the right state. In the right state, the input field now contains the text "영국", which corresponds to the selected item in the dropdown menu.

참가국 :

자동 계산 출력

■ 자동 계산 출력 : <output> 요소

- 입력된 데이터로부터 계산된 결과 표현, 폼의 출력을 위한 요소
- name 속성 : 요소의 이름을 지정
- for 속성 : 변수의 이름을 지정
- 폼 전송 시 onForminput 이벤트를 이용하여 계산을 수행
 - ▶ 예, 가격 및 권수가 새로 입력되면 onForminput 이벤트 발생
=> <output> 요소의 value 값이 “price x num”으로 계산

금액 :

```
<input type="number" name="price" min="0" step="100" value="0" /> 원 X  
<input type="number" name="num" min="0" step="1" value="0" /> 권 =  
<output onForminput="value=price.valueAsNumber*num.valueAsNumber+'원'" />
```

금액 : 원 × 권 =



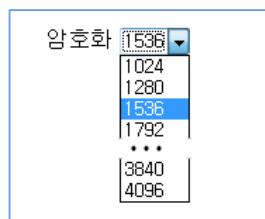
금액 : 원 × 권 = 125000원

암호화 키 생성

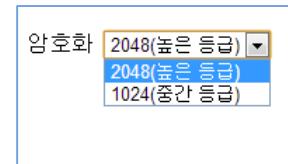
■ 암호화 키 생성 : <keygen> 요소

- 서버와 통신할 때의 보안을 위해 암호화 키를 생성
- 웹 보안을 위해 공개키 기반 암호화 방식에서 사용
 - ▶ 폼이 제출되면 공개키와 개인키가 쌍을 생
=> 공개키는 서버로 전송, 개인키는 브라우저에서 사용자 키를 보관
- 브라우저 별로 지원되는 암호화 방식에는 차이

(a) 오페라 브라우저



(b) 크롬 브라우저



도서 구입 요청 예제

```
<form>
  <h3>도서 구입 요청</h3>
  도서명 : <input type="text" list="book" />
  <datalist id="book">
    <option value="멀티미디어배움터" label="30,000"/>
    <option value="인터넷배움터" label="34,000"/>
    <option value="컴퓨터와 IT 기술의 이해" label="28,000"/>
  </datalist>
  <p>선호도 : 1 <input type="range" min="1" max="5" value="3" /> 5
  <p>가격 : <input type="number" name="price" min="0" step="100" value="10000"/> 원
  <p>권수 : <input type="number" name="num" min="0" step="1" value="0" /> 권
  <p>합계 : <output onForminput="value=price.valueAsNumber*num.valueAsNumber+'원'" />
  <p>암호화 <keygen name="key"/>
  <p><input type="submit" value="구입"/>
</form>
```

The figure consists of three side-by-side screenshots of an Opera web browser window. Each screenshot shows a form titled "도서 구입 요청" (Book Purchase Request) with the following fields:

- 도서명 :** An input field.
- 선호도 :** A range input field with a slider from 1 to 5, currently set to 3.
- 가격 :** An input field containing "10000 원".
- 권수 :** An input field containing "0 권".
- 합계 :** An output field showing the calculated total.
- 암호화 :** A keygen input field.
- 구입 :** A submit button.

In the first screenshot, the "도서명 :" field is empty. In the second screenshot, the "도서명 :" field contains "인터넷배움터", the "권수 :" field contains "5 권", and the "합계 :" output field shows "170000원". In the third screenshot, the "도서명 :" field contains "인터넷배움터", the "권수 :" field contains "5 권", and the "합계 :" output field shows "170000원".

5장. CSS3 스타일시트 기초

5.1 CSS3 시작하기

5.2 CSS 기본 사용법

5.3 문자와 색상 지정하기

5.4 박스모델 설정하기

5.1 CSS3 시작하기

- 5.1.1 스타일시트와 CSS3 기본 개념
- 5.1.2 CSS 속성선언

스타일시트와 CSS3 기본 개념

■ 스타일시트란?

- 웹 문서의 출력될 외형 스타일을 좀더 다양하면서 손쉽게 적용
 - ▶ HTML 태그로는 세세한 부분까지 모두 다 지정하기에는 부족
 - ▶ 스타일시트를 이용하면 크기, 색상 등의 스타일을 일괄 적용
 - ▶ 글자간격, 문단간격, 위치 등 자세한 부분까지 제어가 가능
- 콘텐츠의 내용과 디자인의 분리가 가능한 점이 큰 장점
 - ▶ 웹문서에서 마크업 요소는 보다 내용의 구조에 치중
 - ▶ 디자인 요소는 별도로 작성
 - ▶ 다수의 개발자가 동시에 웹사이트 개발 가능, 또한 유지 보수 용이
- 결과적으로 파일의 용량 축소, 사이트의 성능도 향상



스타일시트와 CSS3 기본 개념

■ CSS(Cascading Style Sheet)의 특징

- 웹컨소시엄에서 웹 문서용으로 개발한 스타일시트 언어
- 기능의 복잡도에 따라 Level1, Level2, Level3로 구분
 - ▶ 1996년 CSS Level1 (CSS1), 1998년 CSS Level2 (CSS2),
 - ▶ Level3인 CSS3는 모듈 별로 구분하여 2005년 이후 현재까지 개발 중
- CSS3의 가장 큰 차이점은 모듈 기반으로 개발된다는 점
 - ▶ 각종 디바이스에 따라 CSS3에서 원하는 모듈만을 탑재
 - ▶ 필요한 모듈만을 빠르게 업데이트 하는 것이 가능
- CSS3는 화려하고 동적인 스타일 작성 가능
 - ▶ 기존의 플래시나 그래픽 디자인 도구에 의존하던 부분을 CSS3 스타일시트만을 이용하여 상당부분 가능하게 됨

역호환(backward-compatibility) : CSS3를 지원하면 CSS2와 CSS1은 당연히 지원
- 이 책에서는 경우에 따라 CSS1, CSS2, CSS3를 구분하지 않고 간단히 CSS로 표기



CSS 속성선언

■ CSS 기본 문법

- CSS 속성(Property) 설정 :
특정 엘리먼트 혹은 그 일부분에 대해 외형의 스타일을 추가

■ CSS 스타일시트 구성요소

- 선택자(Selector)
 - ▶ HTML 문서 내에서 스타일을 설정할 대상이 되는 태그를 선택
 - ▶ 여러 개 나열될 경우 콤마(,)로 구분
- 속성 선언(Property Declaration)
 - ▶ 속성(Property)과 속성값(value)으로 구성
 - ▶ 속성과 속성값은 콜론(:)으로 구분하고 선언은 세미콜론(;)으로 종료

선택자(Selector) 속성선언(Declaration)

선택자 { 속성:값; 속성:값; . . . }

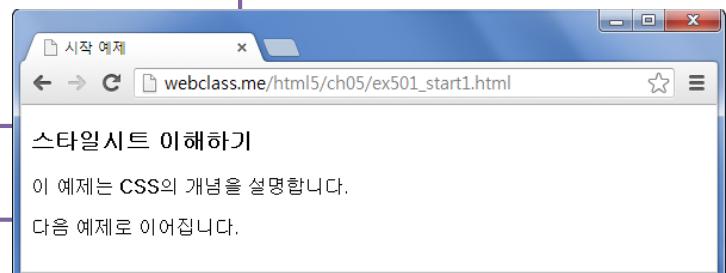
예, h3 {color:red; font-style:italic; }



CSS 시작 예제

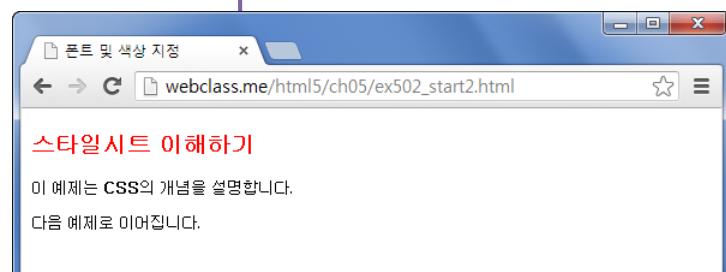
■ 스타일 지정이 없는 문서

```
<html>
  <body>
    <h3>스타일시트 이해하기</h3>
    <p>이 예제는 <strong>CSS</strong>의 개념을 설명합니다. </p>
    <p>다음 예제로 이어집니다.</p>
  </body>
</html>
```



■ 폰트 및 색상 지정

```
<html>
  <head>
    <style type="text/css">
      h3 { color:red }
      p { font-size:10pt }
    </style>
  </head>
  <body>
    <h3>스타일시트 이해하기</h3>
    <p>이 예제는 <strong>CSS</strong>의 개념을 설명합니다.</p>
    <p>다음 예제로 이어집니다.</p>
  </body>
</html>
```



CSS 시작 예제

■ 속성 누적하기

```
<html>
  <head>
    <style type="text/css">
      h3 {color:red; font-style:italic; }
      p {font-size:10pt}
      strong { color:red; font-style:italic; }
    </style>
  </head>
  <body>
    <h3>스타일시트 이해하기</h3>
    <p>이 예제는 <strong>CSS</strong>의
       개념을 설명합니다.</p>
    <p>다음 예제로 이어집니다.</p>
  </body>
</html>
```



5.2 CSS 기본 사용법

- 5.2.1 HTML 문서에서 스타일시트 선언 방법
- 5.2.2 CSS 선택자의 종류

스타일시트 선언 방법

■ 내부 스타일시트 선언

- HTML 문서의 <head>에서 <style> 태그를 이용하여 선언
- 주석은 /* 와 */ 사이에 기입

```
<head>
    <style type="text/css"> /*CSS 스타일 선언*/ </style>
</head>
```

```
<html>
    <head>
        <style type="text/css">
            h3 {color:red}          /* h3의 색상을 빨간색으로 */
            p {font-size:10pt}      /* p의 글자를 한 크기 작게 */
        </style>
    </head>
    <body>
        <h3>스타일시트 이해하기</h3>
        <p>이 예제는 <strong>CSS</strong>의 개념을
설명합니다.</p>
    </body>
</html>
```



스타일시트 선언 방법

■ 외부 스타일시트 연결

- HTML 문서의 <head>에서 <link> 태그를 이용하여 연결

```
<head>
```

```
  <link rel="stylesheet" type="text/css" href="CSS 파일 이름" />  
</head>
```

```
/* 외부 스타일시트 */  
h3 { color:red}  
p { font-size:10pt }
```

```
<html>
```

```
  <head>
```

```
    <link rel="stylesheet" type="text/css" href="extern.css"/>
```

```
  </head>
```

```
  <body>
```

```
    <h3>스타일시트 이해하기</h3>
```

```
    <p>이 예제는 <strong>CSS</strong>의 개념을  
설명합니다.</p>
```

```
  </body>
```

```
</html>
```



스타일시트 선언 방법

■ 인라인 스타일시트 삽입

- 모든 요소에서 사용할 수 있는 style 속성을 이용
 - 해당 태그에만 특정한 스타일을 적용하고 싶을 때 사용
- <태그 style="CSS 속성선언">

```
<html>
  <head>
    <style type="text/css">
      h3 {color:red; font-style:italic; }
      p {font-size:10pt}
    </style>
  </head>
  <body>
    <h3>스타일시트 이해하기</h3>
    <p>이 예제는
      <strong style="font-style:italic; color:red;"> CSS</strong>
      의 개념을 설명합니다.</p>
    <p>다음 예제로 이어집니다.</p>
  </body>
</html>
```



CSS 선택자의 종류

■ 태그 선택자

- 엘리먼트의 태그를 나열
- 다중 태그는 콤마(,)로 구분
- 다중 속성의 경우는 세미콜론(;)으로 구분
예) h3, strong { color: red; font-style: italic }
- 다중 속성값은 콤마로 나열 : 순서대로 가능한 속성값을 적용
예) p { font: Palatino, Garamond, "Times New Roman", serif; font-size: small }



CSS 선택자의 종류

■ 클래스 선택자

- 같은 태그에 다른 스타일을 적용, 혹은 여러 태그에 특정 스타일을 공통으로 적용하고자 할 때
 - ▶ 해당 태그에 같은 이름의 클래스를 설정 : class 속성 이용
 - ▶ <태그이름 class="클래스이름"> ... </태그>
- 클래스 선택자는 클래스이름 앞에 점(.)을 삽입
 - ▶ “.클래스이름”의 형태 : 해당 클래스에 모두 적용
 - ▶ “선택자.클래스이름”의 형태 : 특정 태그에서 해당 클래스만 지정
- 예,

```
.red1 {color: red; font-style: italic; }  
strong.red1 {font-size: 12pt }
```



CSS 선택자의 종류

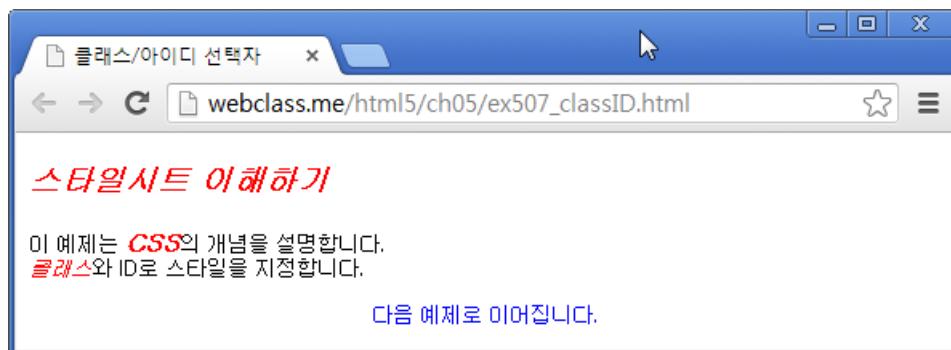
■ 아이디 선택자

- 아이디는 html 문서 내에서 한 군데에서만 지정 가능
`<태그이름 id="아이디이름"> ... </태그>`
- 해당 아이디로 설정된 태그에만 특정 스타일을 적용
 - ▶ 아이디 선택자는 아이디이름 앞에 샵(#)을 붙인다
 - ▶ `#next { color:blue; text-align:center }`



클래스 선택자 및 아이디 선택자

```
<head>
  <style type="text/css">
    p {font-size: 10pt}
    .red1 {color: red; font-style: italic;} /* red1 클래스는 빨간색 이탤릭으로 */
    strong.red1 {font-size: 12pt}          /* strong 요소중 red1 클래스는 12pt 크기 */
    #next { color: blue; text-align: center} /* next 아이디는 파란색 가운데 정렬 */
  </style>
</head>
<body>
  <h3 class="red1">스타일시트 이해하기</h3>
  <p>이 예제는 <strong class="red1">CSS</strong>의 개념을 설명합니다.
  <br><span class="red1">클래스</span>와 ID로 스타일을 지정합니다.</p>
  <p id="next">다음 예제로 이어집니다.</p>
</body>
```



가상클래스 선택자

■ 가상클래스(pseudo class) 선택자

- 요소 이름 다음 콜론(:) 뒤에 예약어
 - ▶ 요소를 선택할 수 있는 특별한 상태를 표현
 - ▶ a:link는 링크를 의미, 방문한 링크는 a:visited로 표현
 - ▶ :before 와 :after 는 content 속성으로 원하는 콘텐츠 추가

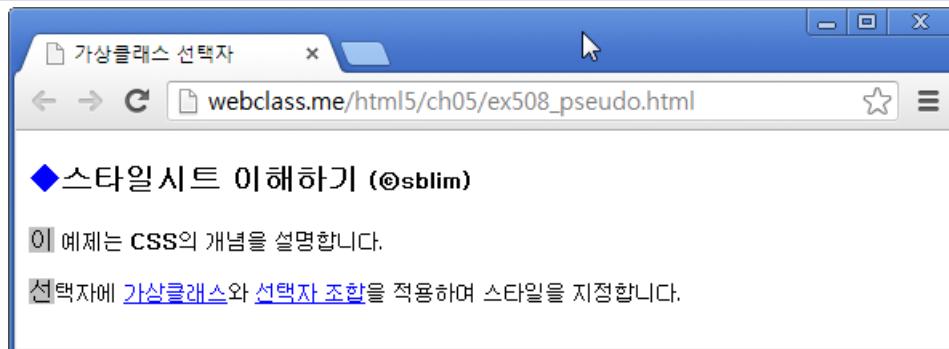
■ 대표적인 가상클래스 선택자

- 하이퍼링크 관련 :link :visited
- 마우스 관련 :active :hover :focus
- 콘텐츠 삽입 :before :after
- 문자 블록 관련 :first-letter :first-line



가상클래스 선택자

```
<head> <style type="text/css">
    p {font-size:10pt}
    a:link { color: blue; }          /* a태그의 하이퍼링크 */
    a:visited { color: green; }     /* 방문한 a태그의 링크 */
    p:last-child { color: blue; text-align:center; }           /* 마지막 p요소 */
    p:first-letter { font-size: 12pt; background-color:silver } /* p요소의 첫글자 */
    h3:before { content: "◆"; color: blue }           /* h3요소의 앞에 파란색 ◆문자 삽입 */
    h3:after { content: " (@sblim)"; font-size:10pt } /* h3요소의 뒤에 콘텐츠 삽입 */
</style> </head>
<body>
    <h3><strong>스타일시트</strong> 이해하기</h3>
    <p>이 예제는 <strong>CSS</strong>의 개념을 설명합니다.</p>
    <p>선택자에 <a href="http://www.w3c.org">가상클래스</a>와
        <a href="http://mm.sm.ac.kr">선택자 조합</a>을 적용하여 스타일을 지정합니다.</p>
</body>
```



선택자 조합

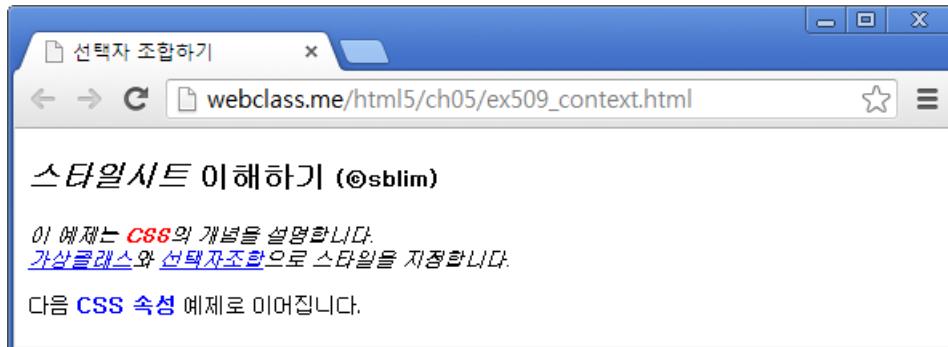
■ 선택자 조합

- 보다 구체적인 요소의 선택이 가능
 - ▶ 특정 요소에 포함되어있는 하위 요소 선택,
뒤에 이어서 나오는 형제 요소 선택
- 대표적인 선택자 조합
 - A E (후손 요소)
 - A > E (자식 요소)
 - B + E (형제 요소)



선택자 조합

```
<head> <style type="text/css">
  p { font-size: 10pt }
  h3:after { content: " (@sblim)"; font-size: 10pt } /* h3요소의 뒤에 콘텐츠 삽입 */
  h3 strong { font-style: italic } /* h3에 속하는 strong 요소 */
  p strong { color: red } /* p에 속하는 strong 요소 */
  p > strong { color: blue } /* p요소 바로 아래에 속하는 strong 요소 */
  h3 + p { font-style: italic } /* h3요소 바로 다음에 있는 p요소 */
</style> </head>
<body>
  <h3> <strong>스타일시트</strong> 이해하기</h3>
  <p>이 예제는 <span id="test"><strong>CSS</strong></span>의 개념</p>을 설명합니다.
  <br><a href="http://www.w3c.org">가상클래스</a>와 <a href="http://mm.sm.ac.kr">
  선택자조합</a>으로 스타일을 지정합니다.</p>
  <p>다음 <strong>CSS 속성</strong> 예제로 이어집니다.</p>
</body>
```



5.3 문자와 색상 지정하기

5.3.1 폰트(Font)의 지정

5.3.2 문자(Text)의 조정

5.3.3 색상(Color) 및 배경
(Background)의 지정

폰트(Font)의 지정

■ 스타일 관련 태그의 사용

- HTML5에서는 내용과 스타일을 분리
 - ▶ , , <i>, <u> 등 출력스타일 지정 태그의 사용을 비권장
 - ▶ 본문에서는 구조나 의미 위주의 태그 사용, 출력 스타일은 CSS 사용
- CSS 속성을 이용
 - ▶ 선택된 요소에 대해 글꼴 지정, 글자 크기, 폰트 굵기, 기울임 등 다양한 모양의 출력스타일을 지정
- 예,

```
p {font-family: "맑은고딕", "돋움", sans-serif; font-size: 10pt; }  
strong { font-weight: bold; font-style: italic; }
```

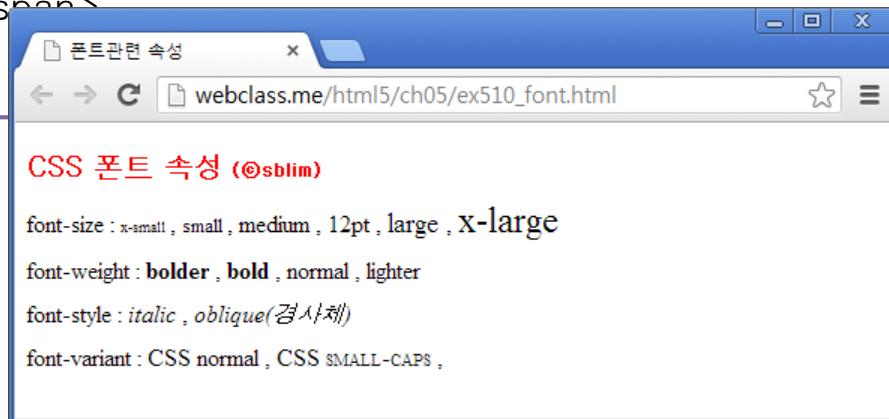
폰트(Font)의 지정

■ 폰트관련 CSS 속성

- font-family : 글꼴 지정
- font-size : 글자크기
- font-weight : 글자굵기
- font-style : 기울임 모양
- font-variant : 영문 소문자의 표시방법

폰트 지정 예제

```
<head> <style type="text/css">
    h3 { font-family:"맑은고딕" "돋움" sans-serif; color: red }
    h3:after { content: " (@sblim)"; font-size: 10pt }
    p { font-family:"Times New Roman" "돋움" serif; line-height: 10pt}
    #x-small { font-size:x-small }
    /* 중간 생략 */
    #v-normal { font-variant: v-normal }
    #small-caps { font-variant: small-caps }
</style> </head>
<body>
    <h3>CSS 폰트 속성</h3>
    <p>font-size : <span id="x-small">x-small</span>, <span id="small">small</span>,
        <span id="medium"> medium </span>, <span id="pt12"> 12pt </span>,   <!--중
    간 생략 -->
        <p>font-variant : <span id="v-normal"> CSS normal , </span> <span id="small-
    caps"> CSS small-caps , </span>
</body>
```



문자(Text)의 조정

■ 문자 관련 CSS 속성

- 단락 줄 맞추기, 문자/줄 간격, 들여쓰기, 밑줄 등 다양한 문자 장식
 - ▶ text-align : 문장의 정렬
 - ▶ letter-spacing : 문자 간격
 - ▶ word-spacing : 단어 간격
 - ▶ vertical-align : 세로방향의 정렬,
 - ▶ line-height : 줄간격
 - ▶ text-indent : 들여쓰기/내어쓰기 지정
 - ▶ text-decoration : 장식 효과 [밑줄, 윗줄, 가운데줄, 깜빡임]
 - ▶ text-transform : 영어 대소문자의 변경
 - ▶ text-shadow : 문자에 그림자를 추가

[노트: 길이값의 단위]

cm, mm, in 미터법/인치의 단위, 1 pc(파이카) = 12 pt(포인트), 1 pt = 1/72 인치

em과 ex는 영문자 대문자 M과 소문자 x의 글자높이 (한 글자/소문자 높이)

px(픽셀)는 출력화면에서 점의 갯수, %는 해당 공간에서의 비율

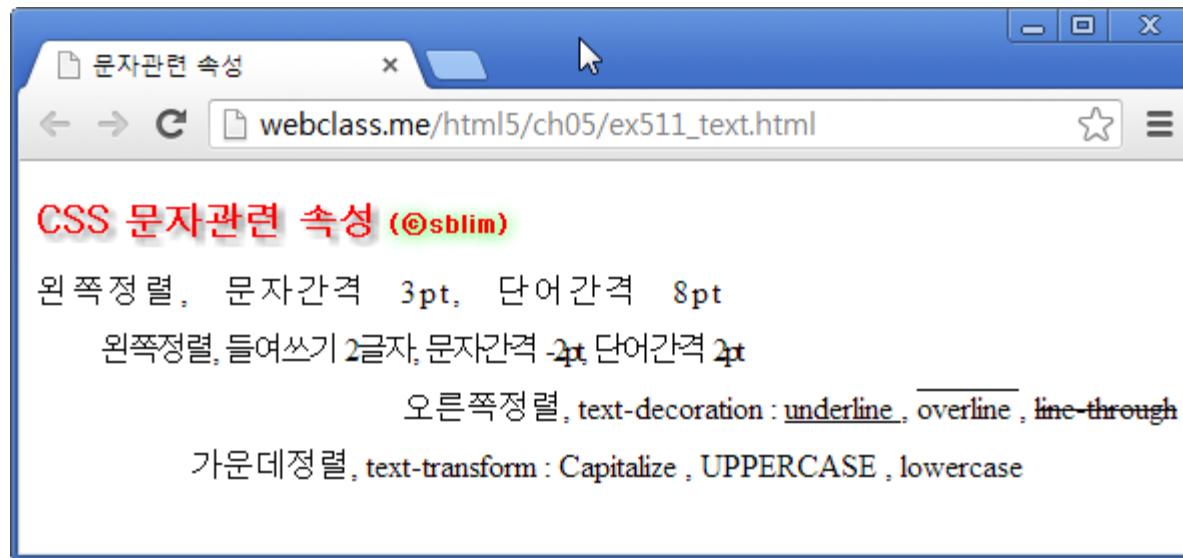


문자 조정 예제

```
<head> <style type="text/css">
    h3 {font-family:"맑은고딕" sans-serif; color:red; text-shadow:3px 3px 4px grey }
    h3:after { content: " (@sblim)"; font-size:10pt; text-shadow: 1px 1px 12px green }
    p { font-family:"Times New Roman" "돋움" serif; line-height: 10pt }
    #first { text-align:left; letter-spacing: 2pt; word-spacing: 8pt }
    #second { text-align:left; text-indent:2em; letter-spacing:-2pt; word-spacing:2pt}
    #third { text-align:right; }
    #fourth { text-align:center; }
    /* 중간 생략 */
    #cap { text-transform: capitalize }
    #upper { text-transform: uppercase }
    #lower { text-transform: lowercase }
</style> </head>
<body>
    <h3><strong>CSS</strong> 문자관련 속성</h3>
    <p id="first">왼쪽정렬, 문자간격 3pt, 단어간격 8pt </p>
    <p id="second">왼쪽정렬, 들여쓰기 2글자, 문자간격 -2pt, 단어간격 2pt </p>
    <p id="third">오른쪽정렬, text-decoration : <span id="under"> underline </span>,
        <span id="over"> overline </span>, <span id="thru"> line-through </span> </p>
    <p id="fourth">가운데정렬, text-transform : <span id="cap"> capitalize </span>,
        <span id="upper"> uppercase </span> , <span id="lower"> Lowercase
    </span></p>
</body>
```



문자 조정 예제



색상(Color) 및 배경(Background)

■ 색상의 표현 : RGB 혹은 RGBA 모델

- 화면의 각 점(픽셀)은 3바이트 혹은 4바이트의 메모리에 표현
 - 각각 1 바이트씩 RGB(Red, Green, Blue) 색상값 : 0~255까지 표현
 - RGBA 모델의 경우 4번째 바이트는 투명색 표현 등 특수용도 사용

색상 값의 표현

- RGB 색상 값을 각각 16진수, 10진수, 혹은 비율(%)로 표현
 - ▶ 16진수 표현: #RRGGBB 예) #ff0000, #080800
 - ▶ 10진수 표현 함수: rgb(R, G, B) 예) rgb(255, 0, 0), rgb(128, 128, 0)
 - ▶ 백분율 표현 함수: rgb(R%, B%, G%) 예) rgb(100%, 0%, 0%), rgb(50%, 50%, 0%)
 - ▶ 키워드 표현: 예) red, olive
 - 투명색은 transparent라는 키워드로 표현

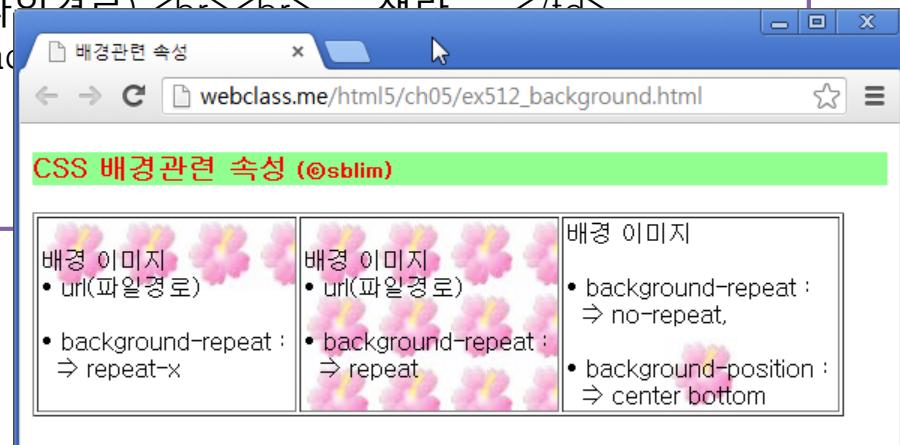
배경관련 속성

■ 배경 관련 속성

- 각 요소의 영역에 배경색이나 이미지를 배경으로 지정
- 관련 속성
 - ▶ background-color : 배경색 지정
 - ▶ background-image : 배경 이미지 파일 지정, url(파일경로)
 - ▶ background-repeat : 배경 이미지의 반복사용 (양쪽, x방향/y방향 반복)
 - ▶ background-attachment : 스크롤과 함께 이동 여부
 - ▶ background-position : 배경의 시작 위치
 - ▶ background : 배경 속성을 한번에 모두 지정 (shorthand)

배경 지정 예제

```
<head> <style type="text/css">
  h3 { color: red; background-color: #90ff90 }
  h3:after { content: " (@sblim)"; font-size:10pt; background-color: yellow }
  #first { background-image: url(flower.jpg); background-repeat:repeat-x; }
  #second { background-image: url(flower.jpg); }
  #third { background-image: url(flower.jpg); background-repeat: no-repeat;
            background-position: center bottom }
</style> </head>
<body>
  <h3> CSS 배경관련 속성</h3>
  <table border="1">
    <tr>
      <td id="first"> 배경 이미지<br>• url(파일경로) <br><br> ... 생략 ...
      <td id="second"> 배경 이미지<br>• url(파일경로) <br><br> ... 생략 ...
      <td id="third"> 배경 이미지 <br><br>• bac
    </tr>
  </table>
</body>
```



5.4 박스모델 설정하기

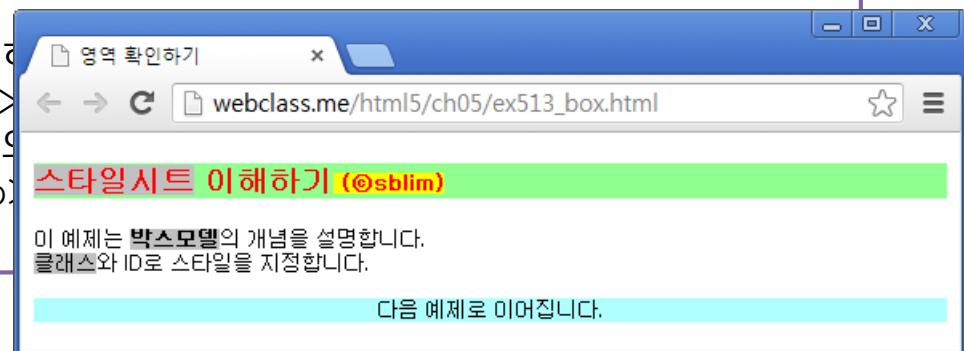
- 5.4.1 영역설정을 위한 박스모델
- 5.4.2 박스모델 유형의 지정

영역 설정을 위한 박스 모델

■ 배경 영역

- <h3>, <p>, <div>와 같은 요소는 해당하는 줄만큼 배경
- 이나 과 같은 요소는 해당하는 글자들만 배경
- <table>이나 와 같은 요소는 자신의 영역이 미리 결정

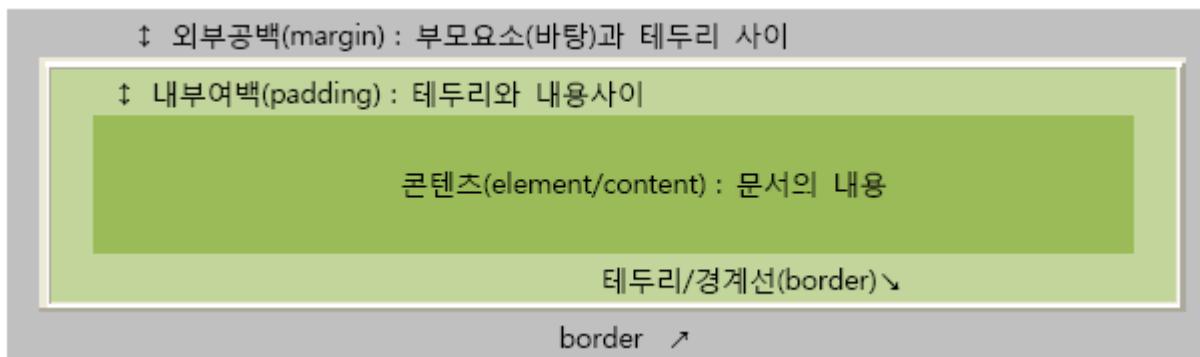
```
<head> <style type="text/css">
  p {font-size: 10pt}
  h3 { color: red; background-color: #90ff90 }
  h3:after { content: " (@sblim)"; font-size:10pt; background-color: yellow }
  strong, .red1 { background-color: silver }
  #next { text-align: center; background-color: #B0ffff }
</style> </head>
<body>
  <h3> <strong>스타일시트</strong> 이해하기</h3>
  <p>이 예제는 <strong>박스모델</strong>입니다.
  <br><span class="red1">클래스</span>와
  <p id="next">다음 예제로 이어집니다.</p>
</body>
```



영역 설정을 위한 박스 모델

■ 박스공간의 구성

- HTML의 모든 요소들은 네모 박스 모양의 공간을 차지
- 이러한 요소가 차지하는 공간의 개념이 박스모델(box model)
- 요소가 차지하는 공간 이외에 내부여백(padding), 테두리(border), 외부공백(margin) 지정

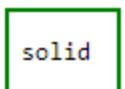


박스 공간을 위한 속성

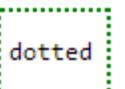
■ 박스 공간을 위한 속성

- 외부공백 : margin,
margin-top, margin-right, margin-left, margin-bottom
- 내부여백 : padding,
padding-top, padding-right, padding-left, padding-bottom
- 테두리/경계선의 두께 : border-width, border-top-width,
border-bottom-width, border-left-width, border-right-width
- 테두리의 모양 : border-style (실선, 점선, 이중선 등)
- 테두리의 색상 : border-color
- 테두리 지정 줄여쓰기(shorthand) :
`border: <width> <style> <color>`

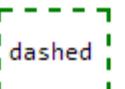
border-style



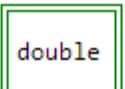
solid



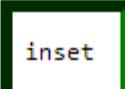
dotted



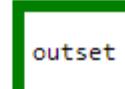
dashed



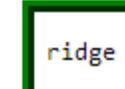
double



inset



outset



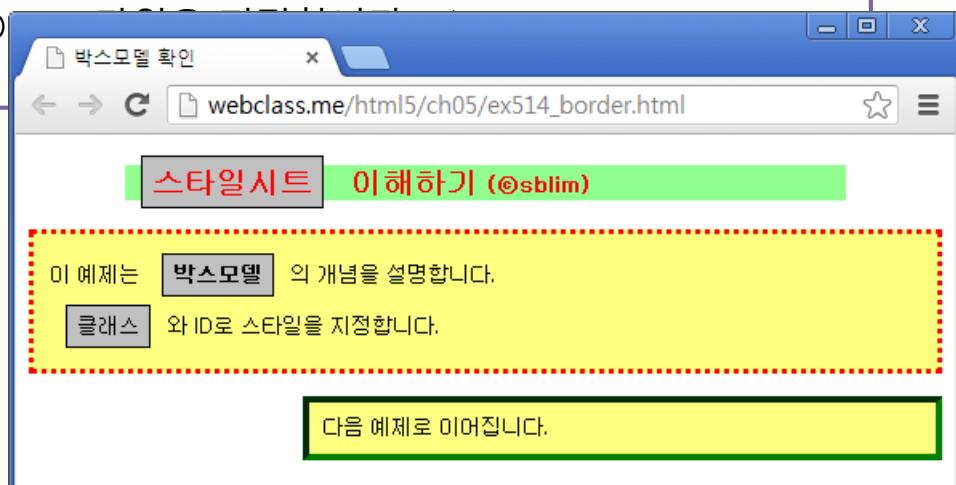
ridge



groove

박스 모델 확인 예제

```
<head> <style type="text/css">
  p {font-size: 10pt; line-height: 24pt}
  h3 { color: red; background-color:#90ff90; margin-left: 60px; margin-right: 60px }
  h3:after { content: " (@sblim)"; font-size:10pt;}
  p { background-color: #ffff80; padding: 10px; border: medium dotted red }
  #next { line-height: 2pt; margin-left:30%; padding:8px; border: 4px double blue }
  strong, .red1 { background-color: silver; margin: 10px; padding: 6px; border: 1px solid black }
</style> </head>
<body>
  <h3> <strong>스타일시트</strong> 이해하기</h3>
  <p>이 예제는 <strong>박스모델</strong>의 개념을 설명합니다.
  <br><span class="red1">클래스</span>와 ID
  <p id="next">다음 예제로 이어집니다.</p>
</body>
```



박스 모델 유형의 지정

■ 박스모델의 유형

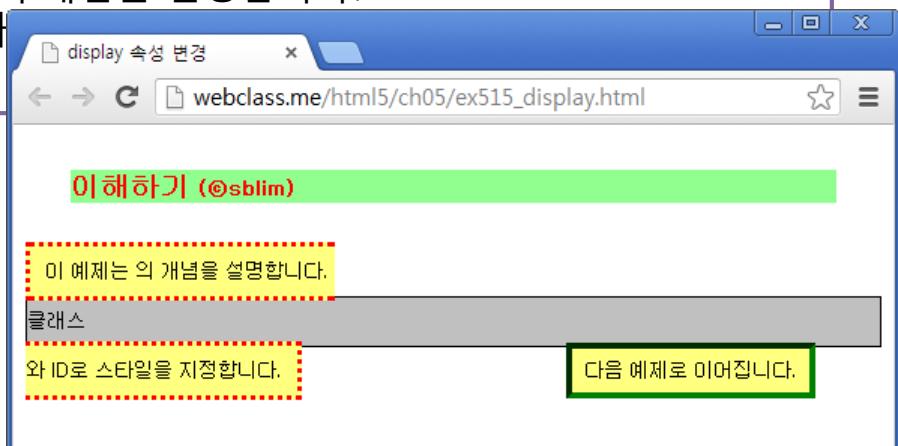
- 블록(block) 유형, 인라인(inline) 유형, 표(table) 유형, 목록(list-item) 유형, 모양이 없는(none) 유형

■ display 속성 값

- block : 글줄(문단) 단위의 박스 모델로 지정
- inline : 글줄 내의 글자 단위의 박스 모델로 지정
- table : 표 단위로 박스 모델을 지정
- list-item : 목록의 열거 항목 단위로 박스 모델을 지정
- none : 공간을 차지하지 않는다.

display 속성 변경 예제

```
head> <style type="text/css">
/* 생략 */
p { display: inline; background-color: #ffff80; padding: 10px; border: thin dotted red }
#next { display: inline; line-height: 12pt; margin-left: 30%; padding: 8px;
border: 4px inset green }
strong { display: none; background-color: silver; margin: 10px; padding: 6px;
border: 1px solid black }
.red1 { display: block; background-color: silver; padding: 6px; border: 1px solid
black }
</style> </head>
<body>
<h3> <strong>스타일시트</strong> 이해하기</h3>
<p>이 예제는 <strong>박스모델</strong>의 개념을 설명합니다.
<br><span class="red1">클래스</span>와
<p id="next">다음 예제로 이어집니다.</p>
</body>
```



6장. 고급 표현을 위한 CSS3

활용

- 6.1 목록과 표 장식하기
- 6.2 레이아웃 설정하기
- 6.3 다양한 효과 설정하기
- 6.4 움직임 설정하기

6.1 목록과 표 장식하기

- 6.1.1 목록의 스타일 설정
- 6.1.2 표의 스타일 설정

목록의 스타일 설정

■ 목록의 글머리 기호 설정

list-style-type 속성

구분	속성값	설명
순서없는 목록 (unordered list)	disc circle square	● 속이 찬 둥근 기호 ○ 속이 빈 둥근 기호 ■ 속이 찬 네모 기호
순서있는 목록 (ordered list)	decimal lower-roman upper-roman lower-alpha upper-alpha	정수 글머리 기호(1,2,3, ...) 소문자 로마자 글머리 기호(i, ii, iii, ...) 대문자 로마자 글머리 기호(I, II, III, ...) 소문자 알파벳 글머리 기호(a, b, c, ...) 대문자 알파벳 글머리 기호(A, B, C, ...)

목록에 스타일 설정하기

```
<head> <style type="text/css">
    ol li { list-style-type: upper-alpha }
    li.usa { list-style-type: disc }
    li.kor { list-style-type: circle }
    li.renew { list-style-type: square }
</style> </head>
<body>
    <h3>과목별 추천도서 목록</h3>
    <ol>
        <li>멀티미디어 이해</li>
        <ul>
            <li class="kor">최윤철, 임순범 공저, 멀티미디어배움터 2.0</li>
            <li class="usa">T.Vaughan, Multimedia: Making It Work, 7th Ed.</li>
            <li class="usa">Z.Li and M.S.Drew, Fundamentals of Multimedia</li>
        </ul>
        <li>인터넷 개론</li>
        <ul>
            <li class="kor">최윤철, 임순범 공저, 인터넷배움터 2.0</li>
            <li class="kor">임순범 외 공저, HTML5 웹 프로그래밍 입문</li>
            <li class="renew">신규 추가 예정 </li>
        </ul>
    </ol>
</body>
```



목록에 스타일 설정하기

■ 목록의 글머리 기호에 이미지 사용

`list-style-image : url("이미지 파일주소")`

- 목록의 글머리 기호에 원하는 이미지를 사용하고 싶은 경우

■ 글머리 기호 위치 지정

`list-style-position 속성`

속성	속성값	설명
<code>list-style-position</code>	<code>inside</code> <code>outside</code> (기본값)	- 글머리 기호가 콘텐츠 박스 안쪽에 위치 - 글머리 기호가 콘텐츠 박스 바깥쪽에 위치

목록의 글머리 위치 지정하기

```
<style type="text/css">
    ul.outside-list { list-style-position: outside }
    ul.inside-list { list-style-position: inside }
    li.usa { list-style-image: url("flag_usa.gif") }
    li.kor { list-style-image: url("flag_kor.gif") }
    p { font-weight: bold }
</style>
...
<h3>과목별 추천도서 목록</h3>
<ul class="outside-list">
    <p>멀티미디어 이해</p>
    <li class="kor">최윤철, 임순범 공저, 멀티미디어배움터 2.0</li>
    <li class="usa">T.Vaughan, Multimedia: Making It Work, 7th Ed.</li>
    <li class="usa">Z.Li and M.S.Drew, Fundamentals of Multimedia</li>
</ul>
<ul class="inside-list">
    <p>인터넷 개론</p>
    <li class="kor">최윤철, 임순범 공저, 인터넷배움터 2.0</li>
    <li class="kor">임순범 외 공저, HTML5 웹 프로그래밍 입문</li>
</ul>
```



표의 스타일 설정

■ 표 또는 셀의 폭 지정 방법 : width, table-layout

- width 속성

- ▶ 표나 각 셀의 가로 길이, 즉, 폭의 크기를 지정
- ▶ %, px, pt, in 등의 절대값으로 지정
- ▶ width 값이 없으면 자동으로 셀의 크기 계산

- table-layout 속성

- ▶ 표의 레이아웃 스타일을 지정
- ▶ 속성값이 없거나 auto의 경우 표의 가로 길이를 자동으로 계산
 - 각 열의 넓이는 원래 크기에 비례하여 조정
- ▶ fixed의 경우 지정한 크기 그대로 각 셀과 표의 크기를 결정
 - 열(혹은 셀)의 크기를 지정하지 않으면 각 열은 같은 크기

표의 레이아웃 방식

```
<style type="text/css">
  #books { table-layout: auto; width: 90% }
  #books caption { font-size: 14pt; font-weight: bold; margin: 0.5em }
</style>
...
<table border="1" id="books">
  <caption>추천 도서 테이블</caption>
  <thead>
    <tr>
      <th>작가</th> <th>책제목</th> <th>출판사</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>월터아이작슨</td> <td>스티브잡스</td> <td>민음사</td>
    </tr>
    <tr>
      <td>최윤철, 임순범</td> <td>멀티미디어 배움터</td>
    </tr>
    <tr>
      <td>임순범 외</td> <td>HTML5 웹 프로그래밍 입문</td> <td>생능출판사</td>
    </tr>
  ... 생략 ...
</tbody>
</table>
```

작가	책제목	출판사
월터아이작슨	스티브잡스	민음사
최윤철, 임순범	멀티미디어 배움터	생능출판사
임순범 외	HTML5 웹 프로그래밍 입문	생능출판사

작가	책제목	출판사
월터아이작슨	스티브잡스	민음사
최윤철, 임순범	멀티미디어 배움터	생능출판사



표의 레이아웃 방식

■ 셀의 테두리(border) 모양 지정

- border-spacing 속성 : 셀들간의 간격 (테두리 굵기)
- border-collapse 속성 : 셀의 테두리 분리 여부
- empty-cells 속성 : 빈 셀의 테두리

속성	속성값	설명
border-spacing	<길이값>	표나 셀의 폭 크기를 지정
border-collapse	collapse separate (기본값)	- 셀의 테두리를 분리하지 않고 한 줄로 그린다. - 각 셀의 테두리를 분리하여 그린다.
empty-cells	show (기본값) hide	- 빈 셀의 테두리를 그린다. - 빈 셀의 테두리를 그리지 않는다.
border		5.4wif 박스모델의 border 속성과 동일

■ 캡션의 위치 지정 : caption-side 속성

- 표의 제목이나 해설을 기입하는 캡션의 위치
- 속성값으로는 top(기본값)과 bottom



표의 테두리 및 캡션 모양 지정하기

■ `#books { border-collapse: collapse; }`

A screenshot of a web browser window titled "표의 테두리 지정". The URL is "webclass.me/html5/ch06/ex614_table_border(a).html". The table has a single row with three columns: "작가" (Writer), "책제목" (Book Title), and "출판사" (Publisher). The data is as follows:

작가	책제목	출판사
월터아이작슨	스티브잡스	
최윤철, 임순범	멀티미디어 배움터	생능출판사
임순범 외	HTML5 웹 프로그래밍 입문	생능출판사

■ `#books { border-spacing: 8px; }`

A screenshot of a web browser window titled "표의 테두리 지정(2)". The URL is "webclass.me/html5/ch06/ex614_table_border(b).html". The table structure is identical to the first one, but the border spacing is set to 8px, creating a larger gap between the cells.

작가	책제목	출판사
월터아이작슨	스티브잡스	
최윤철, 임순범	멀티미디어 배움터	생능출판사
임순범 외	HTML5 웹 프로그래밍 입문	생능출판사

■ `#books { empty-cells: hide; caption-side: bottom; margin: 1em }`

A screenshot of a web browser window titled "표의 캡션위치 지정". The URL is "webclass.me/html5/ch06/ex614_table_caption.html". The table has a caption "추천 도서 테이블" located below it. The "empty-cells" property is set to "hide", so the empty cells in the first row are not visible.

작가	책제목	출판사
월터아이작슨	스티브잡스	
최윤철, 임순범	멀티미디어 배움터	생능출판사
임순범 외	HTML5 웹 프로그래밍 입문	생능출판사

6.2 레이아웃 설정하기

6.2.1 콘텐츠의 위치 지정 방법

6.2.2 플로팅 박스 배치하기

6.2.3 콘텐츠 박스의 크기 조정하기

콘텐츠의 위치 지정 방법

■ 위치 및 크기 지정 : top, right, bottom, left, width, height 속성



■ 위치값의 유형 지정 : position 속성

속성	속성값	설명
position	static (기본값) absolute relative fixed	- 요소가 순서대로 배치된다. 즉, 위치지정이 필요 없다. - 문서 혹은 상위 요소 내에서의 절대위치에 배치한다. - 직전 요소에 이어서 상대위치에 배치한다. - 현재 브라우저 화면 내에서의 절대위치에 배치한다.

위치값 유형에 따른 위치지정

```
<head> <style type="text/css">
    #w3c_static { position: static; }
    #h5_static { position: static; top: 100px; left: 300px; }
    #css_relative { position: relative; left: 80px; }
    #h5_absolute { position: absolute; top: 205px; left: 100px; }
    #css_fixed { position: fixed; top: 20px; right: 30px; }
</style> </head>
<body>
    <h3>5.5 새로운 문서 표준 HTML5
        
    </h3>
    <p>
        
        현재 W3C에서 표준안 개발을 하고 있는 HTML5는 차세대 웹문서 표준안으로 ...(중략)....
    </p>
    <h3> 5.5.1 HTML5의 탄생 배경 및 특징
        
        
    </h3>
    
    <p>HTML 4.0이 1997년 발표된 이후 벌써 10년 이상이 경과되었는데, IT 업계 ...(중략)...
</p>
</body>
```

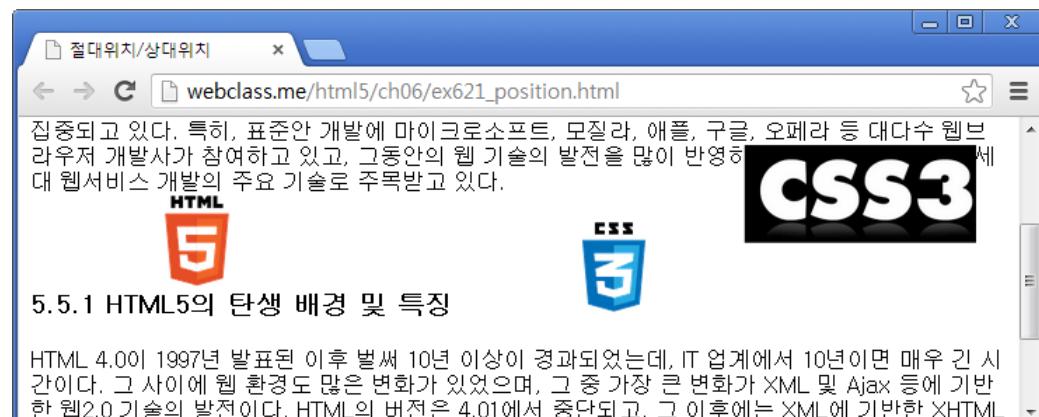


위치값 유형에 따른 위치지정

(a) 실행 결과



(b) 위로 스크롤 하였을 때



순서 지정

■ 앞 뒤 순서 지정 : z-index

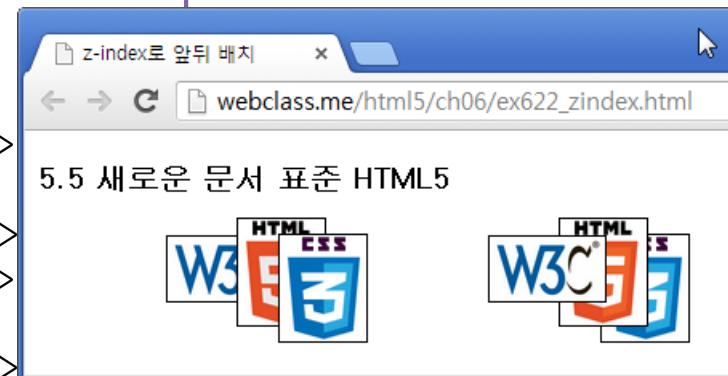
- 여러 개의 콘텐츠를 겹쳐서 배치할 때 앞뒤 순서를 결정
 - z축 상의 좌표는 아니고 순서만을 지정 : 큰 수가 앞 쪽

```
<style type="text/css">
  #w3c_z1 { z-index: 1; position: relative; top: -20px; left: 80px; }
  #h5_z2 { z-index: 2; position: relative; top: -5px; left: 45px; }
  #css_z3 { z-index: 3; position: relative; top: 5px; left: 10px; }
  #w3c_z9 { z-index: 9; position: relative; top: -20px; left: 80px; }
  #h5_z8 { z-index: 8; position: relative; top: -5px; left: 45px; }
  #css_z7 { z-index: 7; position: relative; top: 5px; left: 10px; }
</style>
...
<h3>5.5 새로운 문서 표준 HTML5 </h3>






```



플로팅 박스 배치하기

■ 플로팅 박스의 지정 : float

- 특정 콘텐츠를 주변 콘텐츠와 별도로 분리하여 배치하고 싶을 때
- float 속성은 플로팅 박스와 주변 콘텐츠와 배치 방법을 지정

속성	속성값	설명
float	left	- 플로팅 박스가 왼쪽 경계에 배치, 주변 콘텐츠는 오른쪽에 위치
	right	- 플로팅 박스가 오른쪽 경계에 배치, 주변 콘텐츠는 왼쪽에 위치
	none	- (기본값) 플로팅 박스 적용 없이 순서대로 배치

플로팅 박스 배치하기

```
<style type="text/css">
    #w3c_float { float:left; border: thin solid black; }
    #h5_float { float:left; top: 100px; left: 300px; }
    #css_float { float:right; border: thin solid black; }
</style>
...
<h3>5.5 새로운 문서 표준 HTML5 </h3>



<p>현재 W3C에서 표준안 개발을 하고 있는 HTML5는 차세대 웹문서 ...(생략)....
```



The screenshot shows a web browser window titled "플로팅 박스". The address bar contains the URL "webclass.me/html5/ch06/ex623_float.html". The main content area displays the following text:

5.5 새로운 문서 표준 HTML5

HTML5 현재 W3C에서 표준안 개발을 하고 있는 HTML5는 차세대 웹문서 표준안으로 많은 관심이 집중되고 있다. 특히, 표준안 개발에 마이크로소프트, 모질라, 애플, 구글, 오페라 등 대다수 웹브라우저 개발사가 참여하고 있고, 그동안의 웹 기술의 발전을 많이 반영하여 모바일을 포함한 차세대 웹서비스 개발의 주요 기술로 주목받고 있다.

5.5.1 HTML5의 탄생 배경 및 특징

HTML 4.0이 1997년 발표된 이후 벌써 10년 이상이 경과되었는데, IT 업계에서 10년이면 매우 긴 시



콘텐츠 박스의 크기 조정하기

■ 콘텐츠의 크기 조정 : width, height 속성

- 특정 콘텐츠에서 차지하는 공간 크기를 임의로 조정
- 가로 및 세로 크기를 지정, 최대 크기와 최소 크기도 지정

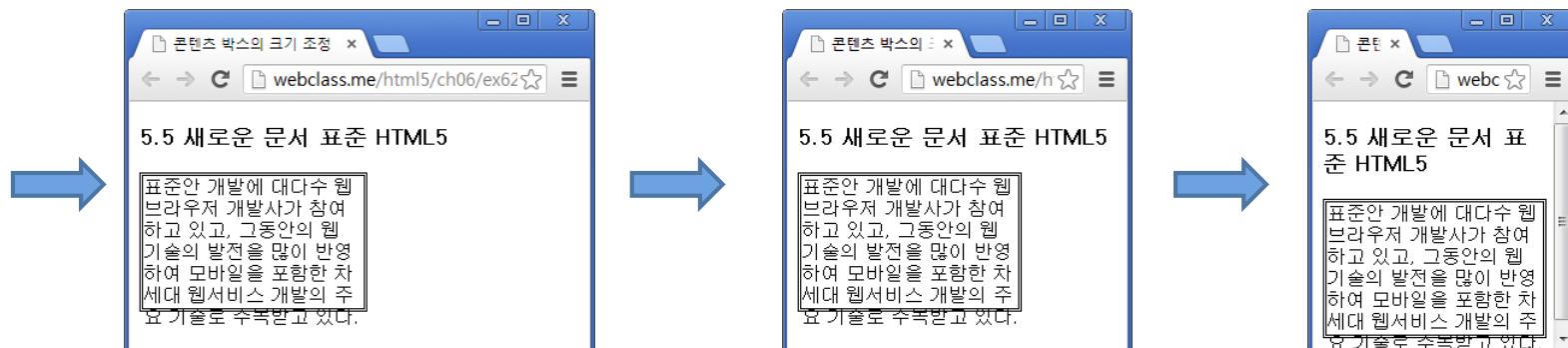
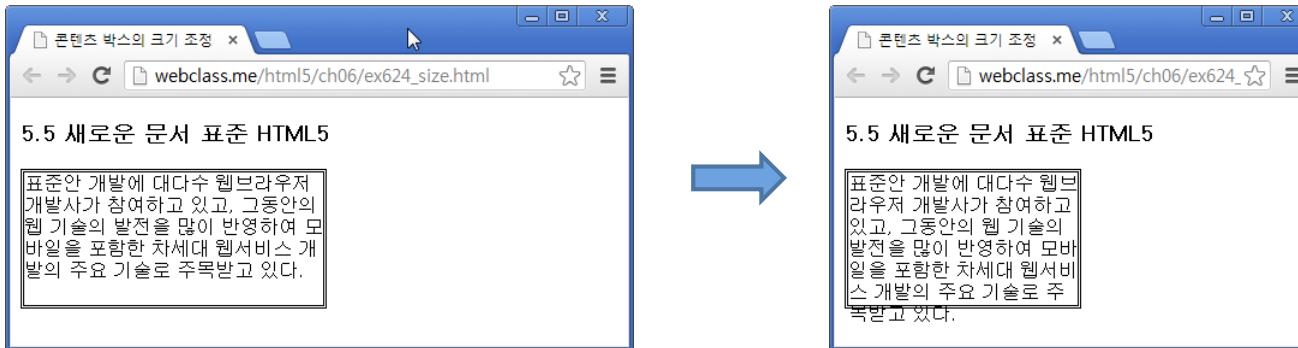
속성	값	설명
width, height	<길이값>, auto	박스 모델의 가로 및 세로 크기를 지정
min-width, min-height	<길이값>, auto	박스의 최소 가로 및 세로 크기를 지정
max-width, max-height	<길이값>, auto	박스의 최대 가로 및 세로 크기를 지정

```
<style type="text/css">
    #intro_text { width: 50%; min-width: 180px; height: 110px;
                  border: medium double black;}
</style>
...
<h3>5.5 새로운 문서 표준 HTML5 </h3>
<p id="intro_text"> 표준안 개발에 대다수 웹브라우저 개발사가 참여하고 있고,
그동안의 웹 기술의 발전을 많이 반영하여 ... (생략)....
```



콘텐츠 박스의 크기 조정하기

■ 실행 결과



overflow 속성

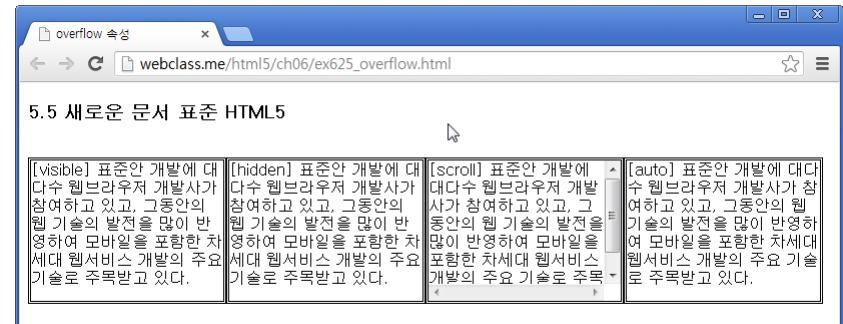
■ 오버플로우 : overflow 속성

- 콘텐츠의 분량이 요소의 박스 크기를 초과할 때의 처리방법

속성	속성값	설명
overflow	visible (기본값)	- 박스 아래쪽에 초과된 콘텐츠가 계속하여 나타난다.
	hidden	- 박스 크기를 초과하는 콘텐츠는 잘려서 보이지 않는다.
	scroll	- 스크롤 바가 있어서 초과하는 콘텐츠를 볼 수 있다.
	auto	- 콘텐츠가 박스를 초과할 경우 스크롤 바가 나타난다.

overflow 속성

```
<style type="text/css">
    #intro1 { overflow: visible; float:left; width: 24%; height: 140px; border: medium double black; }
    #intro2 { overflow: hidden; float:left; width: 24%; height: 140px; border: medium double black; }
    #intro3 { overflow: scroll; float:left; width: 24%; height: 140px; border: medium double black; }
    #intro4 { overflow: auto; float:left; width: 24%; height: 140px; border: medium double black; }
</style>
<h3>5.5 새로운 문서 표준 HTML5 </h3>
<p id="intro1">[visible] 표준안 개발에 대다수 웹브라우저 개발사가 ... (중략) ... </p>
<p id="intro2">[hidden] 표준안 개발에 대다수 웹브라우저 개발사가 ... (중략) ... </p>
<p id="intro3">[scroll] 표준안 개발에 대다수 웹브라우저 개발사가 ... (중략) ... </p>
<p id="intro4">[auto] 표준안 개발에 대다수 웹브라우저 개발사가 ... (중략) ... </p>
```



6.3 다양한 효과 설정하기

6.3.1 박스에 효과 주기

6.3.2 객체의 투명도 및 가시성 설정

박스에 효과 주기

■ 둥근 모서리 : border-radius 속성

- 사각형의 모서리를 둥글게 설정
- 각 모서리의 둥근 정도를 달리 지정 가능

속성	속성값	설명
border-radius	<길이값>	테두리의 모서리 반경을 지정
border-top-left-radius	<길이값>	박스의 각 모서리에서 반경을 지정
border-top-right-radius		
border-bottom-right-radius		
border-bottom-left-radius		

박스에 효과 주기

```
<style type="text/css">
    #intro_text { position: relative; left: 10%; width: 60%; padding: 5px;
                  border: medium double black; border-radius: 10px; }
    #w3c_float { float: right; border: thin solid black; padding: 5px;
                  border-top-left-radius: 8px; border-bottom-right-radius: 8px; }
</style>
...
<h3>5.5 새로운 문서 표준 HTML5 </h3>

<p id="intro_text">
    현재 W3C에서 표준안 개발을 하고 있는 HTML5는 차세대 웹문서 표준안으로 많은 관심이
    집중되고 있다. 특히, 표준안 개발에 ... (중략)...
```



박스에 효과 주기

■ 박스 그림자

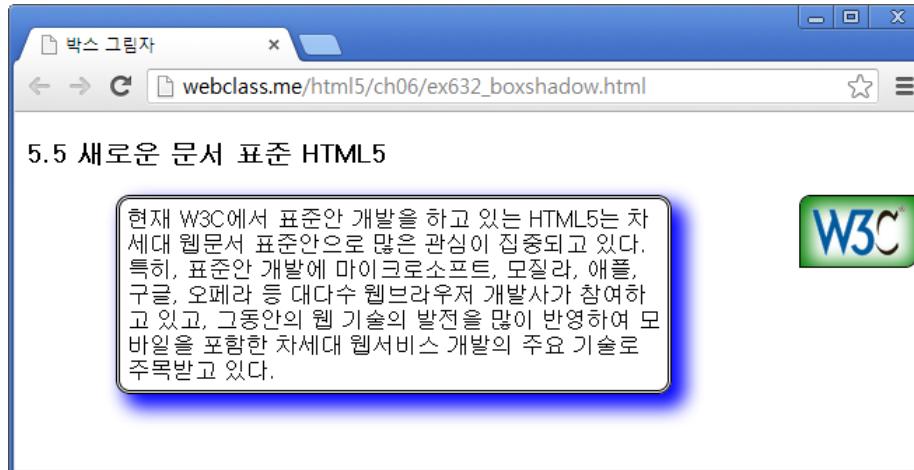
box-shadow: <offset><offset> <blur><spread> <color>
<inset/outset>

- 가로방향 및 세로방향 그림자의 시작 위치, 그림자의 번짐 정도와 크기, 색상, 그림자 진행 방향을 지정

속성값 구분	속성값	설명
<offset> <offset>	(필수) <길이값>	가로방향 및 세로방향의 그림자 시작 위치
<blur> <spread>	(선택) <길이값>	번짐 정도와 그림자의 크기를 지정
<color>	(선택) <색상값>	그림자 색상
<inset/outset>	(선택) inset, outset	그림자 진행 방향: 안쪽/바깥쪽(기본값) 방향

박스에 효과 주기

```
<style type="text/css">
    #intro_text { position: relative; left: 10%; width: 60%; padding: 5px;
                  border: medium double black; border-radius: 10px;
                  box-shadow: 8px 8px 20px 2px blue; }
    #w3c_float { float:right; border: thin solid black; padding: 5px;
                  border-top-left-radius:8px; border-bottom-right-radius:8px;
                  box-shadow: 2px 2px 20px 4px green inset; }
</style>
```



객체의 투명도 및 가시성 설정

■ 투명도 : opacity 속성

- 0.0일 때 투명(fully transparent)하고 1.0일 때 불투명(fully opaque)

■ 가시성 : visibility

- 특정 요소를 보이거나 혹은 보이지 않게 지정
- 표에서 원하는 열이나 행을 보이지 않게도 할 수 있음

속성	속성값	설명
visibility	visible (기본값) hidden collapse	- 콘텐츠를 보이게 한다. - 콘텐츠가 보이지 않게 한다. - 표에서 행이나 열의 내용을 보이지 않게 한다. 표의 레이아웃은 그대로이며, 일반 콘텐츠일 경우 hidden과 같다.

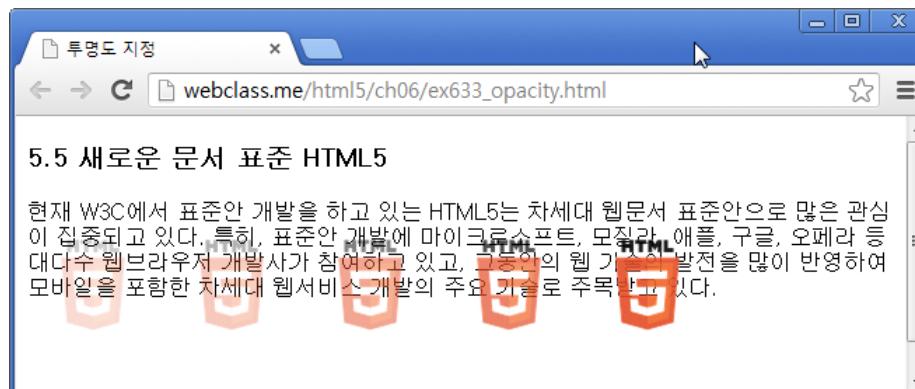
객체의 투명도 및 가시성 설정

```
<style type="text/css">
    #h5_op2 { opacity: 0.2; position: relative; top: -80px; left: 20px; }
    #h5_op3 { opacity: 0.3; position: relative; top: -80px; left: 50px; }
    #h5_op5 { opacity: 0.5; position: relative; top: -80px; left: 80px; }
    #h5_op7 { opacity: 0.7; position: relative; top: -80px; left: 110px; }
    #h5_op9 { opacity: 0.9; position: relative; top: -80px; left: 140px; }
</style>
...(중략)...





```



6.4 움직임 설정하기

6.4.1 전환효과

6.4.2 좌표변환

6.4.3 키프레임 애니메이션

전환효과

■ transition: <property> <duration>

- 객체 모양의 변화는 객체의 CSS 속성(property) 값의 변화
 - ▶ 예, 객체의 크기변화는 width/height 속성값의 변화
- 변화될 속성이름과 전환시간을 지정

속성	속성값 구분	속성값	설명
transition	<property>	none, all (기본값) CSS 속성명	전환효과가 적용될 CSS 속성을 지정
	<duration>	시간값	전환시간을 지정

[노트: 브라우저의 구현 상황] 속성이름

크롬과 사파리에서는 -webkit-으로 시작, 파이어폭스에서는 -moz-로 시작,
오페라에서는 -o-로 시작. 익스플로러에는 아직 구현되어 있지 않다.

이벤트 발생시 속성 변경

```
<head> <style type="text/css">
    #title:hover { border: thick double blue; padding:4px;
                    background-color: yellow; font-size: xx-large; }
    #h5_logo { position: absolute; top: 10px; right: 60px; }
    #h5_logo:hover { border: thin solid red; width: 108px; height: 132px; }
</style> </head>
<body>
    <h3>5.5 새로운 문서 표준 <span id="title">HTML5</span></h3>
    
    <p>현재 W3C에서 표준안 개발을 하고 있는 HTML5는 차세대 웹문서 ... (중략) ... </p>
</body>
```

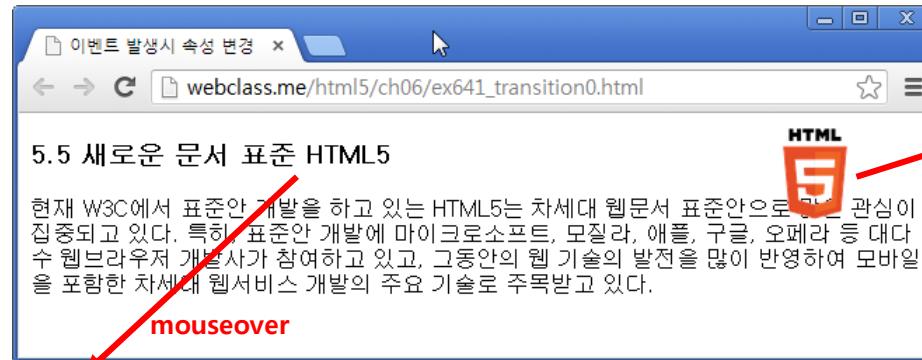


점진적으로 변하는 전환 효과 설정

```
<style type="text/css">
    #title:hover { border: thick double blue; padding:4px;
                    background-color: yellow; font-size: xx-large;
                    -webkit-transition: border 4s, background-color 8s; }

    #h5_logo { position: absolute; top: 10px; right: 60px; }
    #h5_logo:hover { border: thin solid red; width: 108px; height: 132px;
                      -webkit-transition: width 4s; }

</style>
```



문서 표준 **HTML5**

문서 표준 **HTML5**

문서 표준 **HTML5**



mouseover
표준안으로 만든 관심이
풀, 구글, 오페라 등 대다
수 많이 반응해 모바일



표준안으로 만든 관심이
풀, 구글, 오페라 등 대다
수 많이 반응해 모바일



표준안으로 만든 관심이
풀, 구글, 오페라 등 대다
수 많이 반응해 모바일



좌표변환

■ transform: <함수>

- translate(x,y), scale(x,y), rotate(angle), skew(x-angle,y-angle)
- 이동변환, 크기변환, 회전변환, 기울임변환

좌표변환 함수	설명
translate(x,y)	- 이동변환
translateX(x), translateY(y)	- X 방향으로 이동, Y 방향으로 이동
scale(x,y)	- 크기변환
scaleX(x), scaleY(y)	- X 방향으로 크기변환, Y 방향으로 크기변환
rotate(angle)	- angle 각도만큼 회전변환
skew(x-angle,y-angle)	- 기울임변환
skewX(angle), skewY(angle)	- X 방향으로 기울임변환, Y 방향으로 기울임변환

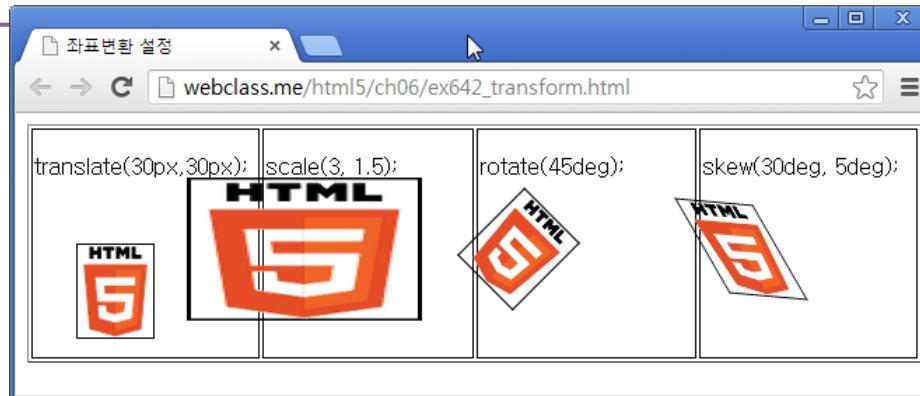
[노트: 브라우저의 구현 상황] 속성이름

크롬과 사파리는 -webkit-으로 시작, 파이어폭스는 -moz-로 시작,
오페라에서는 -o-로 시작, 익스플로러에서는 -ms-로 시작

좌표변환

```
<style type="text/css">
    #h5_trans { -webkit-transform: translate(30px,30px); border: thin solid; }
    #h5_scale { -webkit-transform: scale(3, 1.5); border: thin solid; }
    #h5_rotate { -webkit-transform: rotate(45deg); border: thin solid; }
    #h5_skew { -webkit-transform: skew(30deg, 5deg); border: thin solid; }
    td { width: 160px; height: 160px; vertical-align: top; border: thin solid; } </style>
...
<table border="1">
    <tr>
        <td><p>translate(40px,40px);</p>
             </td>
        <td><p>scale(3, 1.5);</p>
             </td>
    ...

```



키프레임 애니메이션

■ 연속적인 애니메이션을 실행

- 먼저 애니메이션 도중의 주요 장면을 키프레임으로 설정한 후

@keyframe key-name

진행비율 { 속성: 속성값; . . . }

진행비율 { 속성: 속성값; . . . } . . .

}

- 원하는 객체에 이미 작성된 키프레임을 애니메이션으로 지정

animation: <key-name> <duration>

속성	속성값 구분	속성값	설명
animation	<key-name>	(키프레임 이름)	@keyframe에서 설정한 이름
	<duration>	<시간값>	애니메이션 시간을 지정

[노트: 브라우저의 구현 상황] 속성이름

크롬과 사파리에서는 -webkit-으로 시작, 파이어폭스에서는 -moz-로 시작,
오페라에서는 -o-로 시작. 익스플로러에는 아직 미 구현



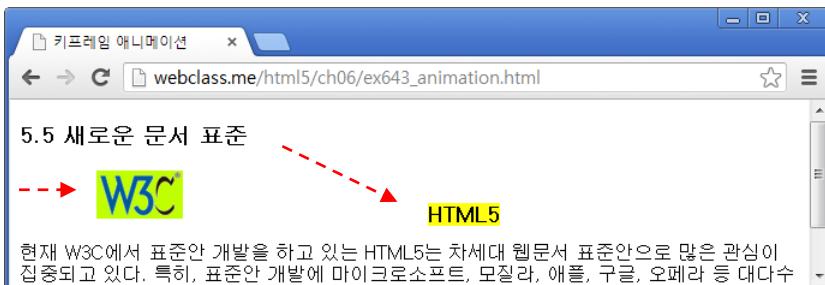
키프레임 애니메이션

```
<style type="text/css">
    @-webkit-keyframes w3cMove {
        from { background: yellow; left: 20px; }
        to { background: lime; left: 200px; }
    }
    @-webkit-keyframes sway {
        0% { background: white; left: 50px; }
        25% { background: yellow; top: 40px; left: 150px; }
        50% { background: lime; top: 40px; left: 250px; }
        75% { background: yellow; left: 150px; }
    }
    #w3c_anim { -webkit-animation: w3cMove 6s; position: relative; }
    #title { -webkit-animation: sway 6s; position: relative; }
</style>
```

...
<h3>5.5 새로운 문서 표준 HTML5 </h3>

<p> 현재 W3C에서 표준안 개발을 하고 있는 HTML5는 차세대 웹문서 표준안으로 많은 관심이 집중되고 있다. 특히, 표준안 개발에 ...(생략)...

키프레이언 애니메이션



7장. HTML5 인터페이스 관련 API

- 7.1 위치정보 사용하기
- 7.2 오디오와 비디오 제어하기
- 7.3 끌어다 놓기

7.1 위치정보 사용하기

- 7.1.1 위치정보 얻어오기
- 7.1.2 위치정보 추적하기

위치정보

■ 위치정보(geolocation)

- 사용 예
 - ▶ 사용자 주변에 있는 주요 관심지점에 대한 정보
 - ▶ 주변 교통 정보 등을 활용한 다양한 웹서비스
 - ▶ 지도와 결합하여 자동차 네비게이션과 같은 서비스
- HTML5에서는 위치정보를 얻을 수 있는 자바스크립트 API를 제공



위치정보 API

- 내장객체인 navigator.geolocation 객체를 이용
- 위치정보 API의 기능
 - 현재 위치정보를 가져오는 기능
 - 위치이동을 추적하는 기능

종류	이름	설명
메소드	getCurrentPosition()	현재 위치 정보를 반환.
	watchPosition()	위치 변경을 추적
	clearWatch()	위치 추적을 취소

위치정보 얻어오기

■ 위치정보 얻어오기

- 웹브라우저를 통해 현재 위치에 관한 정보를 얻는 것

■ 위치정보 얻어오기 과정

1. geolocation 객체를 구하기
2. 위치정보를 알려주는 객체의 `getCurrentPosition()` 메소드를 호출하기
3. 얻은 위치정보를 처리하기
 - ▶ 웹 애플리케이션에 따라 내용이 달라짐

위치정보 얻어오기

■ geolocation 객체

- 웹브라우저가 위치정보 기능을 가지고 있지 않으면, navigator 객체의 geolocation 속성은 null 값을 가짐.
- 이를 이용하여 웹브라우저가 위치정보 기능을 가지고 있는지 알 수 있음

■ getCurrentPosition() 메소드

- 위치정보를 가져오기 위해 사용하는 geolocation 객체의 메소드
- 위치정보를 받아서 처리하는 함수를 인자로 받음
- 오류가 발생했을 때 처리해야 할 부분을 가진 함수와 위치정보 가져오기 기능에 대한 세부 제어를 위한 내용을 추가적으로 줄 수도 있음

■ Position 객체

종류	이름	설명
속성	coords.latitude	위도 정보 (단위: 도).
	coords.longitude	경도 정보 (단위: 도)
	coords.altitude	고도 (단위: m)
	timestamp	위치정보를 가져온 시각



예제: 위치정보 얻어오기

```
<table>
    <tr><td>Timestamp</td><td id="ts"></td></tr>
    <tr><td>Position</td><td id="position"></td></tr>
</table>
<br>
<div>
    <span class="button" onclick="clickGetPosition();">Get position</span>
</div>

<script type="text/javascript">
function clickGetPosition() {
    if (navigator.geolocation) {
        function MySuccess(MyPosition) {
            var d = new Date(MyPosition.timestamp);

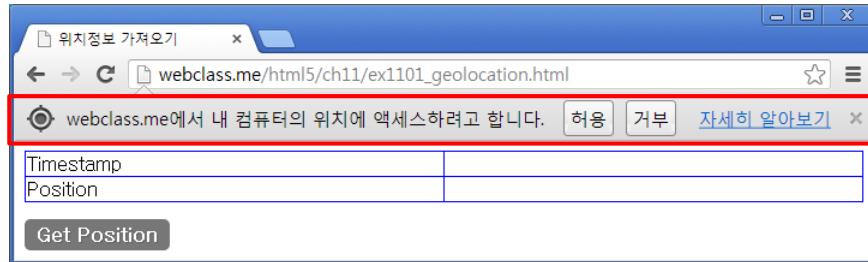
            // Date.toUTCString() 메소드는 시간을 국제표준시 문자열로 변환
            document.getElementById("ts").innerHTML = d.toUTCString() ;
            document.getElementById("position").innerHTML =
                "(" + MyPosition.coords.latitude + ", " +
                MyPosition.coords.longitude + ")";
        }

        navigator.geolocation.getCurrentPosition(MySuccess);
    } else {
        alert("No support");
    }
}
</script>
```

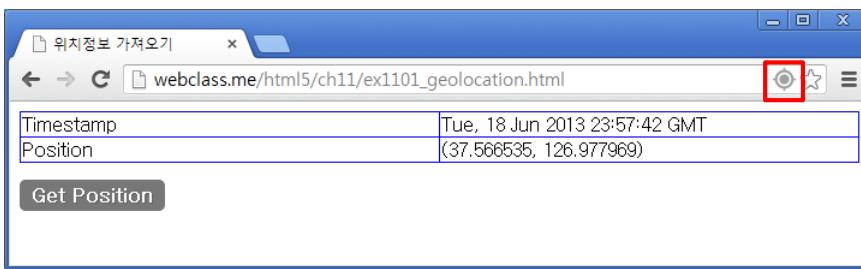


실행결과

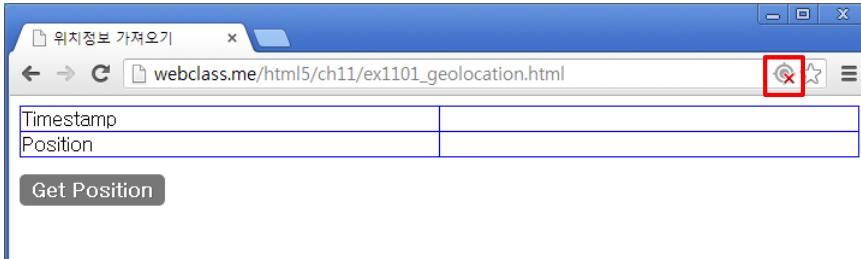
■ 위치정보 사용자의 확인 화면



■ 위치정보 사용 허용 화면



■ 위치정보 사용 거부 화면



위치정보 추적하기

■ 위치정보 추적하기

- 네비게이션과 같이 사용자가 움직였을 때 변경된 위치정보를 계속 가져와야 하는 경우에 사용하는 기능
- 주기적으로 위치정보를 얻어오는 것이 아니라 변경된 경우에만 위치정보를 가져와 위치정보와 관련된 작업을 할 수 있도록 해줌
- geolocation 객체의 watchPosition() 메소드와 clearWatch() 메소드 사용

위치정보 추적하기

■ watchPosition() 메소드

- getCurrentPosition() 메소드의 인자와 의미가 같음
- 위치정보가 바뀔 때마다 watchPosition() 함수의 인자로 전달된 함수가 호출되는 점이 다름

■ clearWatch() 메소드

- 위치정보 추적기능을 멈춤
- watchPosition() 함수를 호출했을 때 반환되는 값을 clearWatch() 함수에 전달

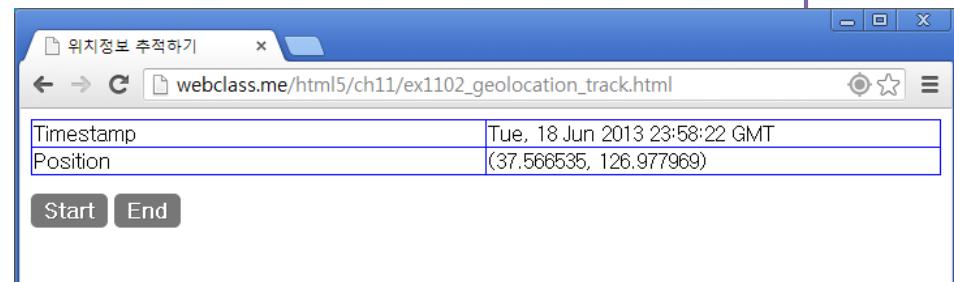
예제: 위치정보 추적하기

```
<script type="text/javascript">
var geoId;

function clickStart() {
    if (navigator.geolocation) {
        function MySuccess(MyPosition) {
            var d = new Date(MyPosition.timestamp);

            document.getElementById("ts").innerHTML = d.toUTCString() ;
            document.getElementById("position").innerHTML = "(" +
                MyPosition.coords.latitude + ", " +
                MyPosition.coords.longitude + ")";
        }
        geoId = navigator.geolocation.watchPosition(MySuccess);
    } else {
        alert("No support");
    }
}

function clickEnd() {
    if (navigator.geolocation) {
        navigator.geolocation.clearWatch(geoId);
    } else {
        alert("No support");
    }
}
</script>
```



7.2 오디오와 비디오 제어하기

오디오와 비디오 제어하기

■ HTML5의 멀티미디어

- <audio> 요소와 <video> 요소
- 자바스크립트 API: 멀티미디어 요소 제어

■ 멀티미디어 관련 요소의 속성(attribute)

- HTML 태그를 통해 값을 지정할 수 있는 속성
 - ▶ src 속성, width 속성, height 속성
- 자바스크립트 코드를 통해서 값을 사용할 수 있는 속성
 - ▶ currentTime 속성, duration 속성 등



멀티미디어 요소 제어 API

종류	이름	설명
메소드	play()	멀티미디어 자료의 재생 시작
	pause()	멀티미디어 자료의 재생 일시정지
속성	currentTime	멀티미디어 자료의 현재 재생 위치
	duration	멀티미디어 자료의 전체 길이
	ended	멀티미디어 자료를 마지막까지 재생했는지 여부
	paused	재생의 일시멈춤 여부
이벤트	canplay	재생이 가능할 때 발생하는 이벤트
	timeupdate	재생 중 주기적으로 발생하는 이벤트

예제: 재생 제어

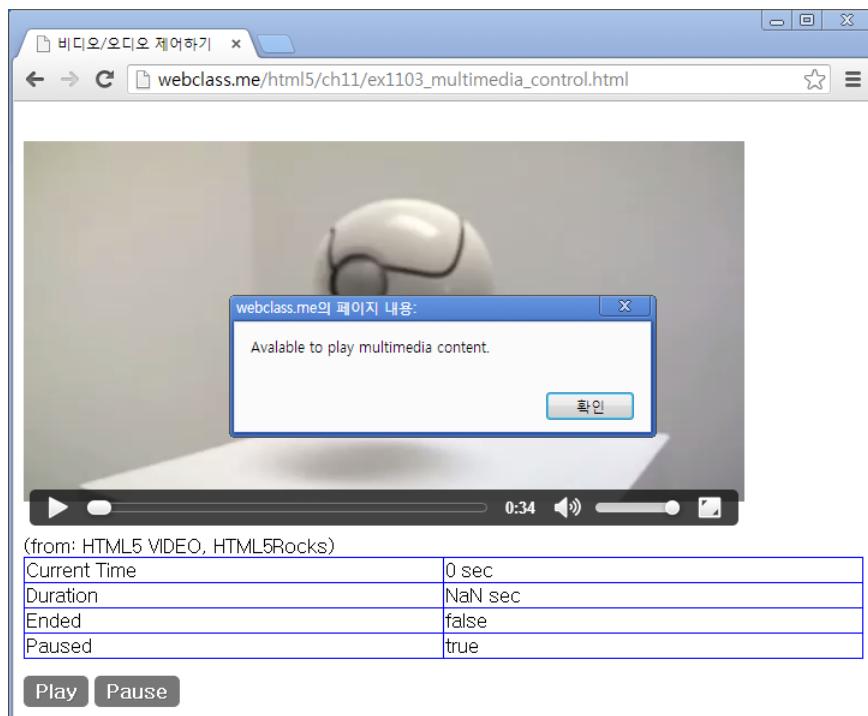
```
<script type="text/javascript">
var videoE = document.getElementById("video");

// videoE.oncanplay = alertCanPlay;
videoE.addEventListener("canplay", function() {
    alert("Available to play multimedia content.");
    updateState();
});
// videoE.ontimeupdate = updateState;
videoE.addEventListener("timeupdate", function() {
    updateState();
});
function clickPlay() {
    videoE.play();
}
function clickPause() {
    videoE.pause();
}
function updateState() {
    document.getElementById("currentTime").innerHTML = videoE.currentTime + " sec";
    document.getElementById("duration").innerHTML = videoE.duration + " sec";
    document.getElementById("ended").innerHTML = videoE.ended;
    document.getElementById("paused").innerHTML = videoE.paused;
}
updateState();
</script>
```

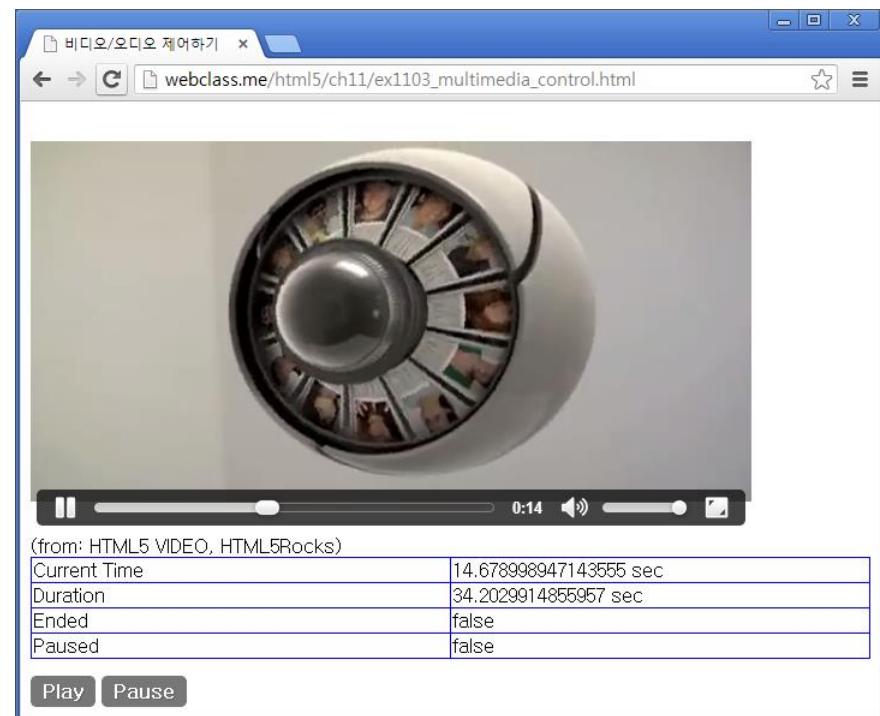


실행결과

- 멀티미디어 요소를 로딩하였을 때



- "Play"를 선택하여 재생상태

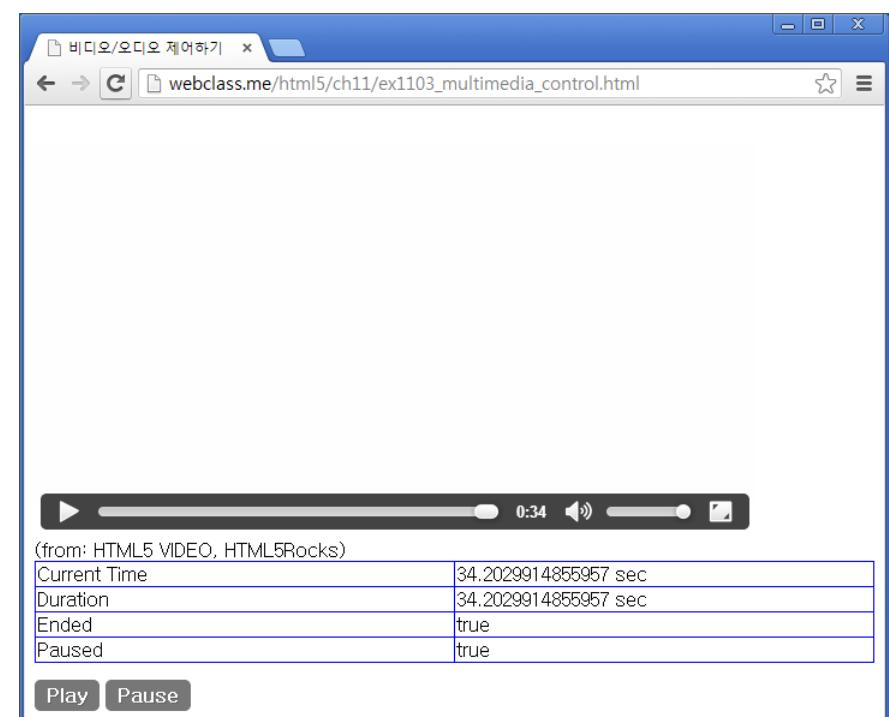


실행결과

■ “Pause”를 선택하여 멈춘 상태



■ 마지막까지 재생한 상태



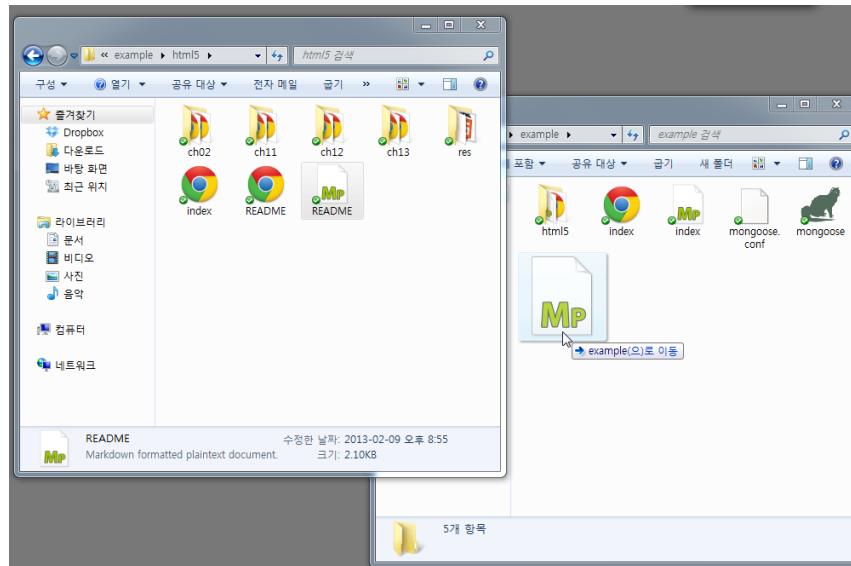
7.3 끌어다 놓기

- 7.3.1 끌기(drag) 이벤트
- 7.3.2 놓기(drop) 이벤트
- 7.3.3 끌어다 놓기 예제

끌어다 놓기(drag & drop)

■ 끌어다 놓기(drag & drop)

- 마이크로소프트의 윈도우, 애플의 맥 OS X과 같은 데스크탑 GUI 운영체제에서 마우스를 사용하여 애플리케이션 간에 파일이나 애플리케이션의 데이터를 전달하는 기능
- 웹 애플리케이션에서는 화면 상에 나타나는 요소를 옮기거나 웹 브라우저와 다른 애플리케이션간에 자료를 전달하기 위해 사용



끌어다 놓기(drag & drop)

■ 끌어다 놓기 처리 과정

- 마우스로 원하는 항목을 선택하여 이동시키는 끌기 과정
- 해당 항목을 목적지에서 마우스 버튼을 떼어서 해당 항목을 전달하는 놓기 과정

■ 놓기 영역

- 놓기 과정의 목적지
- 해당 항목이 처리 가능한지 여부를 이벤트를 통해서 사용자에게 알려줘야 함



관련 이벤트 및 속성

끌기를 할 요소

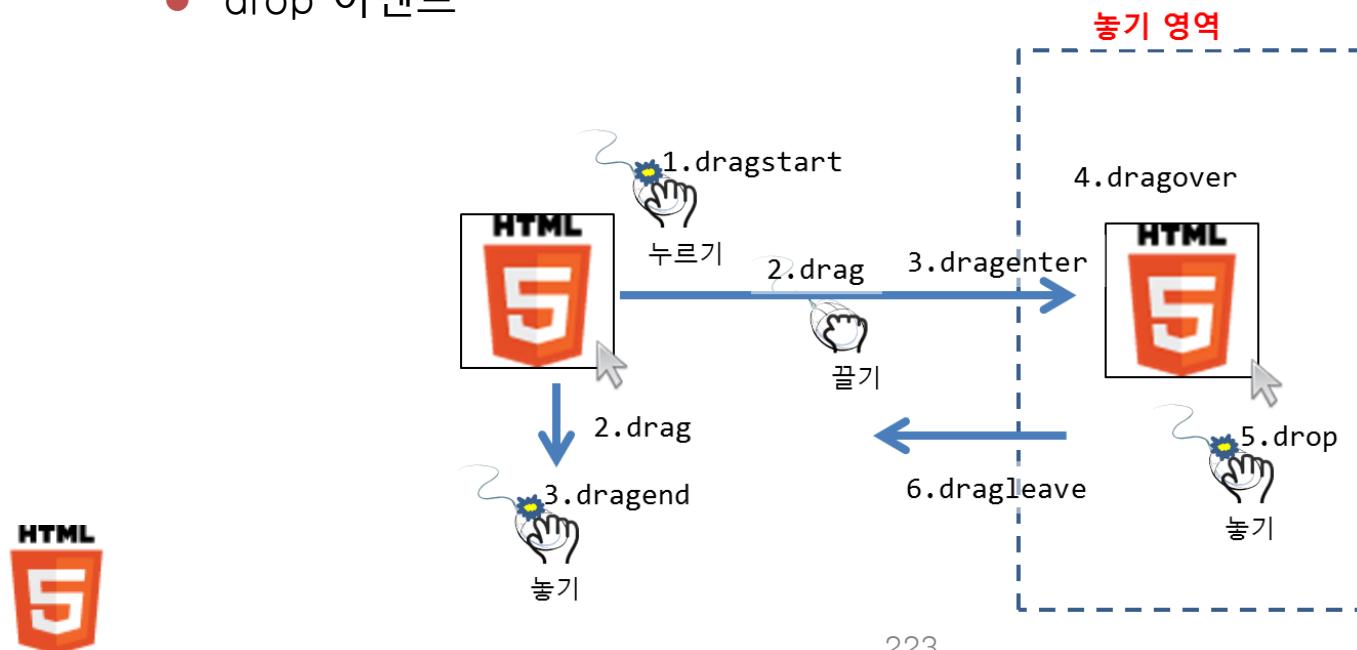
- draggable 속성과 끌기 관련 이벤트인 dragstart 이벤트, dragend 이벤트, drag 이벤트 처리

놓기 영역 요소

- dragenter 이벤트, dragover 이벤트, dragleave 이벤트를 처리

놓기 영역에 요소를 끌어다 놓는 경우

- drop 이벤트



끌기 속성과 이벤트

- 요소의 draggable 속성
 - 요소의 끌기 가능 여부 설정
- 관련 이벤트

종류	이름	설명
이벤트	dragstart	요소를 끌기 시작했을 때 발생
	drag	요소를 끌기 도중에 발생
	dragend	요소를 끌기 끝났을 때 발생

예제: 요소 drag

```
<div id="src">
  
  
  
</div>
<br>
<table>
  <tr><td>Drag Source Id</td><td id="srcId"></td></tr>
  <tr><td>Output</td><td id="o"></td></tr>
</table>

<script type="text/javascript">
var src = document.getElementById("src");
var isDragging;
var srcId;

src.ondragstart = function (e) {
  srcId = e.target.id;
  document.getElementById("srcId").innerHTML = srcId;
  document.getElementById("o").innerHTML = "dragstart: " + srcId;
  isDragging = false;
}
```

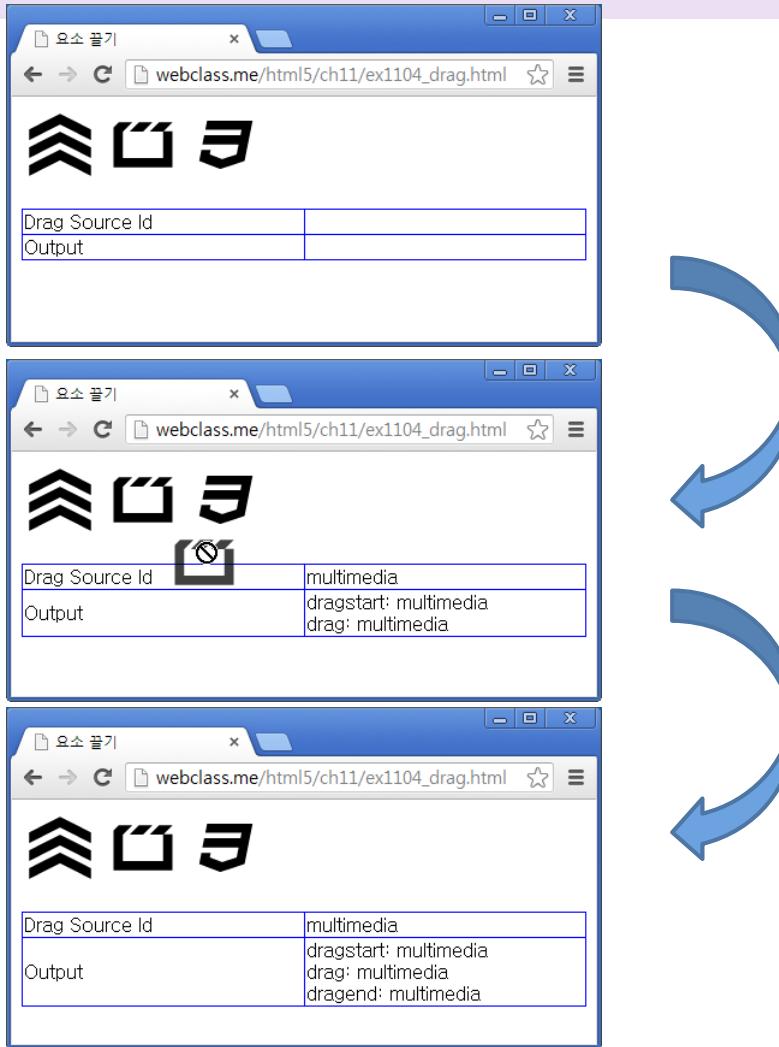
예제: 요소 drag

```
src.ondrag = function (e) {
    if (isDragging == false) {
        document.getElementById("o").innerHTML += "<br>drag: " + srcId;
        isDragging = true;
    }
}

src.ondragend = function (e) {
    document.getElementById("o").innerHTML += "<br>dragend: " + srcId;
}
</script>
```



실행결과



이미지 끌기

이미지 끌기 끝

drop 이벤트

관련 이벤트

종류	이름	설명
이벤트	dragenter	끌기한 요소가 놓기 영역으로 들어왔을 때 발생
	dragover	끌기한 요소가 놓기 영역에 있을 때 발생
	dragleave	끌기한 요소가 놓기 영역을 떠날 때 발생
	drop	끌기한 요소를 놓기 할 때 발생

놓기 가능 여부 처리

- 놓기 영역에서 항상 놓기가 가능한 것이 아님

```
target.ondragover = function (e) {  
    e.preventDefault();  
    ...  
}
```



예제: 요소 놓기

```
<div id="src">
    
    
    
</div>
<br>
<div id="target">
    <span id="drop" class="button">Drop here</span>
</div>
<br>
<table>
    <tr><td>Drag Source Id</td><td id="srcId"></td></tr>
    <tr><td>Drag Target Id</td><td id="targetId"></td></tr>
    <tr><td>Output</td><td id="o"></td></tr>
</table>

<script type="text/javascript">
var src = document.getElementById("src");
var target = document.getElementById("target");
var isDraggingOver;
var srcId, targetId;
```

```
src.ondragstart = function (e) {
    srcId = e.target.id;
    document.getElementById("srcId").innerHTML = e.target.id;
}

target.ondragenter = function (e) {
    targetId = e.target.id;
    document.getElementById("targetId").innerHTML = targetId;
    document.getElementById("o").innerHTML += "dragenter: " + targetId;
    isDraggingOver = false;
}

target.ondragover = function (e) {
    e.preventDefault();

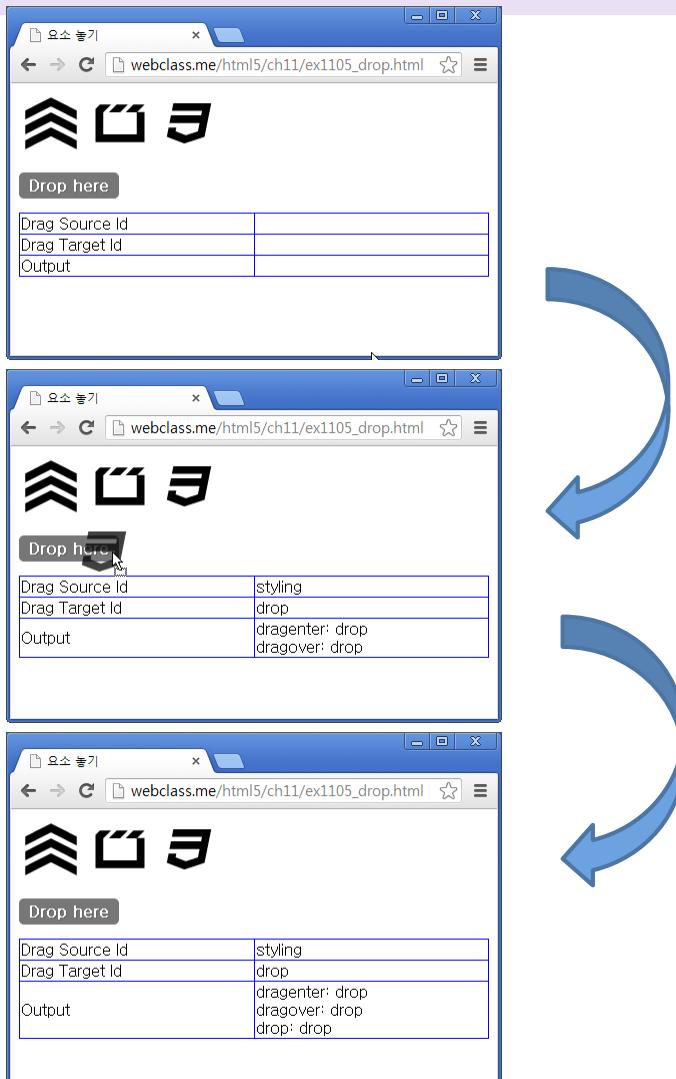
    if (isDraggingOver == false) {
        document.getElementById("o").innerHTML += "<br>dragover: " + targetId;
        isDraggingOver = true;
    }
}

target.ondragleave = function (e) {
    document.getElementById("o").innerHTML += "<br>dragleave: " + targetId;
}

target.ondrop = function (e) {
    e.preventDefault();
    document.getElementById("o").innerHTML += "<br>drop: " + targetId;
}
</script>
```



실행결과



놓기 영역으로 끌기

놓기

예제: 끌어다 놓기 처리하기

```
<div id="src">
  
  
  
</div>
<br>
<div id="target">
  <span id="dropzone" class="button">Drop here</span>
</div>
<br>

<script type="text/javascript">
var srcId;

src.ondragstart = function (e) {
  srcId = e.target.id;
}

target.ondragover = function (e) {
  e.preventDefault();
}
```

예제: 끌어다 놓기 처리하기

```
target.ondrop = function (e) {
    e.preventDefault();

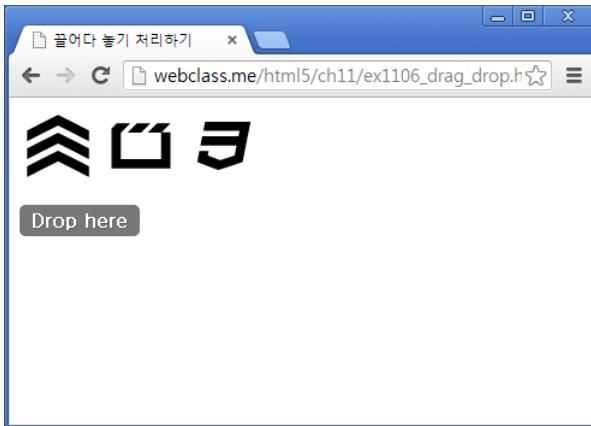
    var newE = document.getElementById(srcId).cloneNode();

    var target = document.getElementById("target");
    var dropzone = document.getElementById("dropzone");
    target.innerHTML = "";
    target.appendChild(dropzone)
    target.appendChild(document.createElement("br"));
    target.appendChild(newE);
}
</script>
```



실행결과

■ 처음 실행화면



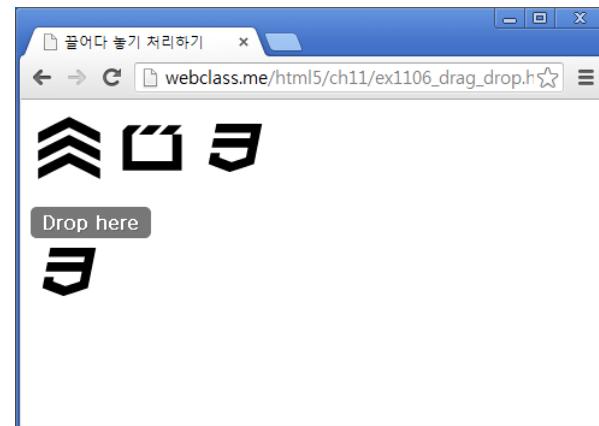
■ 이미지 끌기 화면 (놓기 영역 밖)



■ 이미지 끌기 화면 (놓기 영역 안)



■ 이미지 끌어다 놓기 화면



8장. HTML5 저장 및 기타 API

- 8.1 웹스토리지 사용하기
- 8.2 파일 다루기
- 8.3 그외 저장 관련 API
- 8.4 기타 HTML5 API

8.1 웹스토리지 사용하기

8.1.1 로컬스토리지

8.1.2 세션스토리지

8.1.3 스토리지 변경 이벤트

8.1.4 로컬스토리지와 세션스토리지

웹스토리지와 로컬스토리지

■ 웹스토리지(Web storage)

- 웹브라우저에 자료를 저장하기 위한 기능
- 이전에 웹브라우저에서 저장할 수 있는 방법
 - ▶ 쿠키, 서버에 저장
- 로컬스토리지, 세션스토리지
 - ▶ 자료의 보존 기간의 차이

■ 로컬스토리지(local storage)

- 값을 기기에 저장하기 위해서 사용
- 각 항목을 구분하기 위한 키와 보존할 값의 쌍으로 저장
- `window.localStorage` 객체
- Storage 이벤트와 Storage 이벤트의 인자인 StorageEvent 객체



window.localStorage 객체

■ window.localStorage 객체의 메소드와 속성

종류	이름	설명
메소드	clear()	모든 자료를 삭제
	getItem()	지정된 키의 자료를 반환
	key()	지정된 인덱스의 키를 반환. 값이 없으면 null 반환
	removeItem()	지정된 키의 자료를 삭제
	setItem()	지정된 키에 자료를 저장
속성	length	스토리지에 저장된 항목의 개수
	[] 연산자	키에 해당하는 항목을 반환. 반환된 항목에 값을 저장하거나 읽을 수 있음

예제: 로컬스토리지 사용하기

```
<body onload="showAll()">
    키: <input id="k" type="text">
    값: <input id="v" type="text">
    <button onclick="saveItem()">저장</button>
    <button onclick="removeItem()">삭제</button>
    <hr>
    <select id="entries" size=5 onchange="onEntrySelected()"> </select>
    <button onclick="showAll()">다시 표시</button>

    <script type="text/javascript">
        var localStorage = window.localStorage;

        if (!localStorage) {
            alert("로컬스토리지를 지원하지 않습니다.");
        }

        var key = document.getElementById("k");
        var value = document.getElementById("v");
        var entries = document.getElementById("entries");
```



예제: 로컬스토리지 사용하기

```
// 스토리지 내용 모두 표시
function showAll() {
    // 목록 clear
    entries.innerHTML = "";

    // 루프, 스토리지 내용 확인
    for ( var i = 0; i < localStorage.length ; i++ ) {
        var k = localStorage.key(i);
        entries.options[entries.options.length] =
            new Option(k + ":" + localStorage[k], k);
    }
}

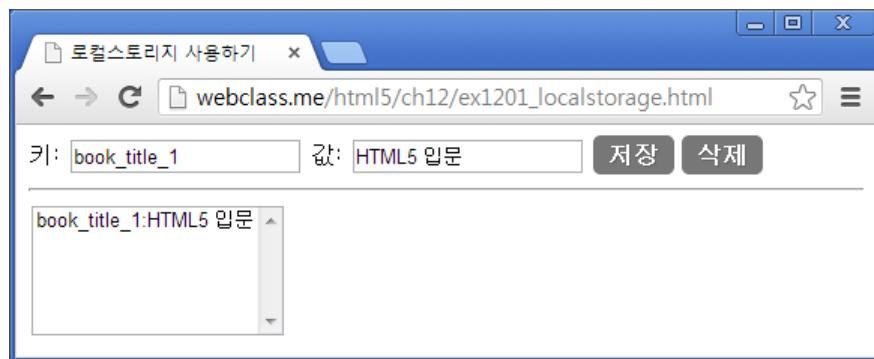
// 스토리지에 저장
function saveItem() {
    localStorage.setItem(key.value, value.value);
    showAll();
}

// 스토리지로부터 값을 삭제
function removeItem() {
    localStorage.removeItem(key.value);
    showAll();
}
```



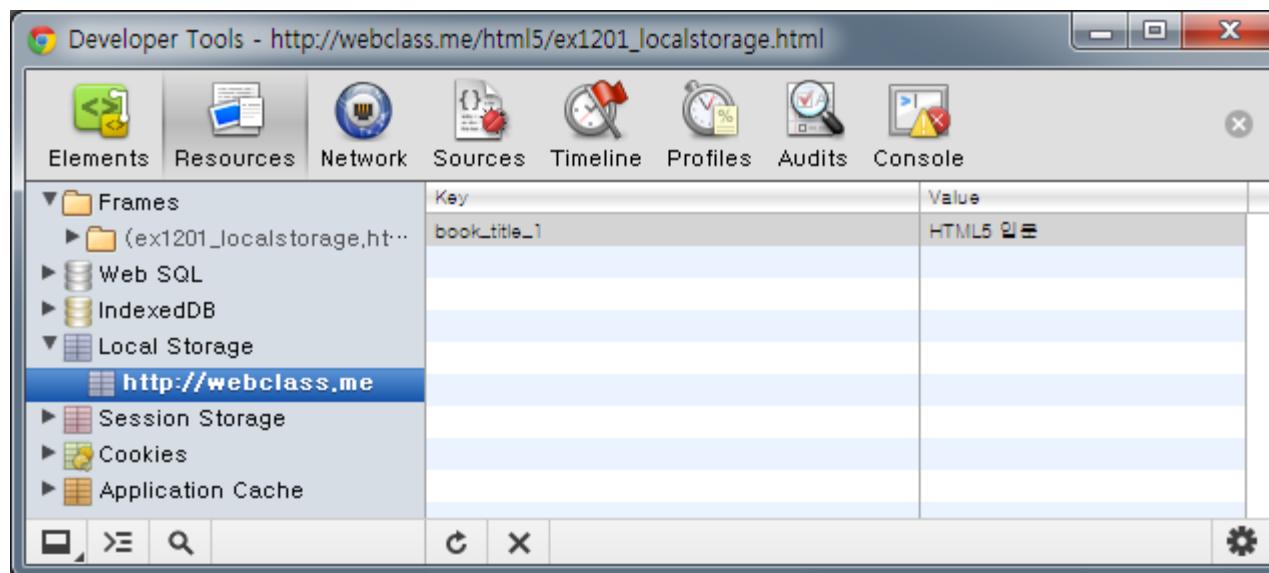
실행결과

```
// 선택된 엔트리를 텍스트 필드에 표시
function onEntrySelected() {
    var selectedOption = entries.options[entries.selectedIndex];
    key.value = selectedOption.value;
    value.value = localStorage.getItem(selectedOption.value);
}
</script>
</body>
```



웹스토리지 내용 확인

■ 크롬 개발자도구



세션스토리지

■ 세션스토리지(session storage)

- window.sessionStorage 객체를 사용
 - ▶ 메소드, 속성, 이벤트는 앞에서 살펴본 localStorage 객체와 동일
- 로컬스토리지의 차이점
 - ▶ 로컬스토리지: 브라우저 세션 종료와 무관하게 자료를 보관
 - ▶ 세션스토리지: 세션이 종료되면 저장된 자료가 사라짐



예제: 세션스토리지 사용하기

```
<body onload="showAll()">
    키: <input id="k" type="text">
    값: <input id="v" type="text">
    <button onclick="saveItem()">저장</button>
    <button onclick="removeItem()">삭제</button>
    <hr>
    <select id="entries" size=5 onchange="onEntrySelected()"> </select>
    <button onclick="showAll()">다시 표시</button>

    <script type="text/javascript">
        var sessionStorage = window.sessionStorage;

        if (!sessionStorage) {
            alert( "세션스토리지를 지원하지 않습니다." );
        }

        var key = document.getElementById("k");
        var value = document.getElementById("v");
        var entries = document.getElementById("entries");
```



예제: 세션스토리지 사용하기

```
// 스토리지 내용 모두 표시
function showAll() {
    // 목록 clear
    entries.innerHTML = "";

    // 루프, 스토리지 내용 확인
    for ( var i = 0; i < sessionStorage.length ; i++ ) {
        var k = localStorage.key(i);
        entries.options[entries.options.length] =
            new Option(k + ":" + sessionStorage[k], k);
    }
}

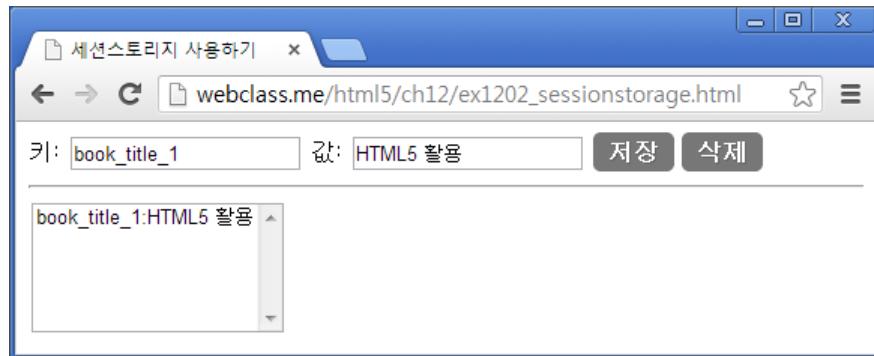
// 스토리지에 저장
function saveItem() {
    sessionStorage.setItem(key.value, value.value);
    showAll();
}

// 스토리지로부터 값을 삭제
function removeItem() {
    sessionStorage.removeItem(key.value);
    showAll();
}
```



실행결과

```
// 선택된 엔트리를 텍스트 필드에 표시
function onEntrySelected() {
    var selectedOption = entries.options[entries.selectedIndex];
    key.value = selectedOption.value;
    value.value = sessionStorage.getItem(selectedOption.value);
}
</script>
</body>
```



스토리지 변경 이벤트

■ Storage 이벤트

- 웹스토리지가 변경될 때 발생
- 이벤트 처리기에 StorageEvent 객체를 인자로 전달
 - ▶ StorageEvent 객체: 변경에 대한 세부 정보

종류	이름	설명
속성	key	변경된 항목의 키
	oldValue	변경 전의 값
	newValue	변경 후의 값
	url	이벤트가 발생한 출처

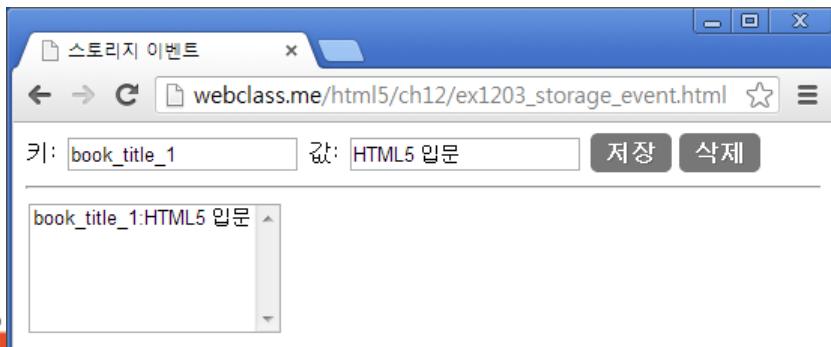
예제: 스토리지 이벤트

```
// StorageEvent를 표시
function displayStorageEvent(e) {
    var msg = "key=" + e.key + "\n" +
              "oldValue=" + e.oldValue + "\n" +
              "newValue=" + e.newValue + "\n" +
              "url=" + e.url + "\n" +
              "storageArea=" + e.storageArea;

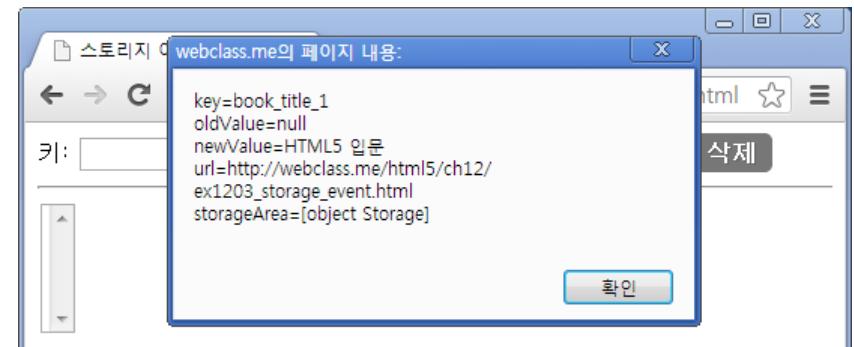
    alert(msg);
    showAll();
}

window.addEventListener("storage", displayStorageEvent, true);
```

웹브라우저 1



웹브라우저 2



로컬스토리지와 세션스토리지

■ 보안적 고려사항

- Web Storage에 저장된 Data는 public이다.
- 보안이 요구되는 Data는 저장하지 말아야 한다.
- 계속적으로 Storage를 삭제해 주어야 한다.

로컬스토리지	세션스토리지
<ul style="list-style-type: none">- 저장 기간에 제한이 없고 사용자가 지우지 않는 한 영구적으로 저장.- 도메인이 다르면 서로의 로컬스토리지에 접근할 수 없음.- 같은 도메인에 속한 웹페이지는 모두 같은 로컬스토리지를 공유함	<ul style="list-style-type: none">- window 객체와 같은 유효범위와 생존기간을 가짐. (같은 도메인이라도 윈도우마다 따로 생성.)- 윈도우 복제로 생성된 경우, 스크립트를 이용해 새 창을 연 경우 같은 값을 가진 세션스토리지가 "복제" 됨. (공유가 아님)- 새로 생성된 윈도우와 기존 윈도우의 세션스토리지는 서로 영향을 주지 않음.

8.2 파일 다루기

8.2.1 파일 객체

8.2.2 파일읽기 객체

파일 다루기

■ 파일 다루기

- HTML5에서는 로컬 컴퓨터에 있는 파일을 웹 애플리케이션이 읽을 수 있는 기능을 제공
- 파일 내용을 텍스트 형식 뿐만 아니라 이진(binary) 형식으로도 읽음

■ FileList 객체

- File Object의 리스트이다.
- FileList를 가져오기 위한 2가지 방법이 있음
 - 1) file input type을 이용
 - 2) drag and drop의 DataTransfer에 있는 file속성을 이용

■ FileList의 속성과 메서드

- length : FileList의 파일 개수
- Item(index) : FileList에 저장된 File객체를 가져옴



파일 객체

■ 파일 객체

- 파일 정보를 알아내기 위한 File 객체

종류	이름	설명
속성	name	파일 이름
	type	파일 종류
	lastModifiedDate	파일 마지막 변경일
	size	이벤트가 발생한 출처
메소드	slice()	파일 일부를 반환

예제: 파일 정보 가져오기

```
<input class="button" type="file" id="files" name="files[]" multiple />
<br>
<table>
    <tr><td id="list"></td></tr>
</table>

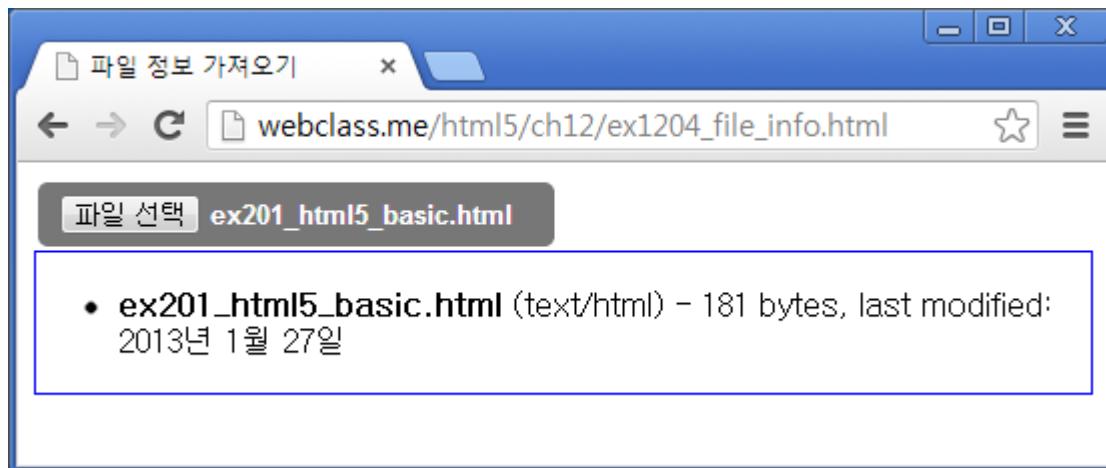
<script>
function handleFileSelect(evt) {
    var files = evt.target.files; // FileList object

    // files is a FileList of File objects. List some properties.
    var output = [];
    for (var i = 0, f; f = files[i]; i++) {
        var lastDate = f.lastModifiedDate;
        output.push(
            '<li><strong>', escape(f.name),
            '</strong> (', f.type || 'n/a', ') - ',
            f.size, ' bytes, last modified: ',
            lastDate ? lastDate.toLocaleDateString() : 'n/a',
            '</li>');
    }
    var listE = document.getElementById('list')
    listE.innerHTML = '<ul>' + output.join('') + '</ul>';
}

var filesE = document.getElementById('files');
// filesE.onchange = handleFileSelect;
filesE.addEventListener('change', handleFileSelect, false);
</script>
```



실행결과



파일읽기 객체

■ 파일읽기 객체

- 파일로부터 내용을 읽는 FileReader 객체

종류	이름	설명
속성	result	읽기 작업 결과를 가짐
	error	읽기 작업 중에 나타나는 오류 정보를 가짐
메소드	readAsText()	파일을 텍스트 형식으로 읽음
	readAsBinaryString()	파일을 이진 형식으로 읽음
	readAsDataURL()	파일을 DataURL 형식으로 읽음
이벤트	load	파일 읽기 작업이 완료되었을 때 발생
	error	파일 읽기 작업 중 오류가 나타났을 때 발생

예제: 파일 읽기

```
<input class="button" type="file" id="files" name="files[]" />
<script>
    function handleFileSelect(evt) {
        var files = evt.target.files; // FileList object
        var f = files[0];

        var reader = new FileReader();

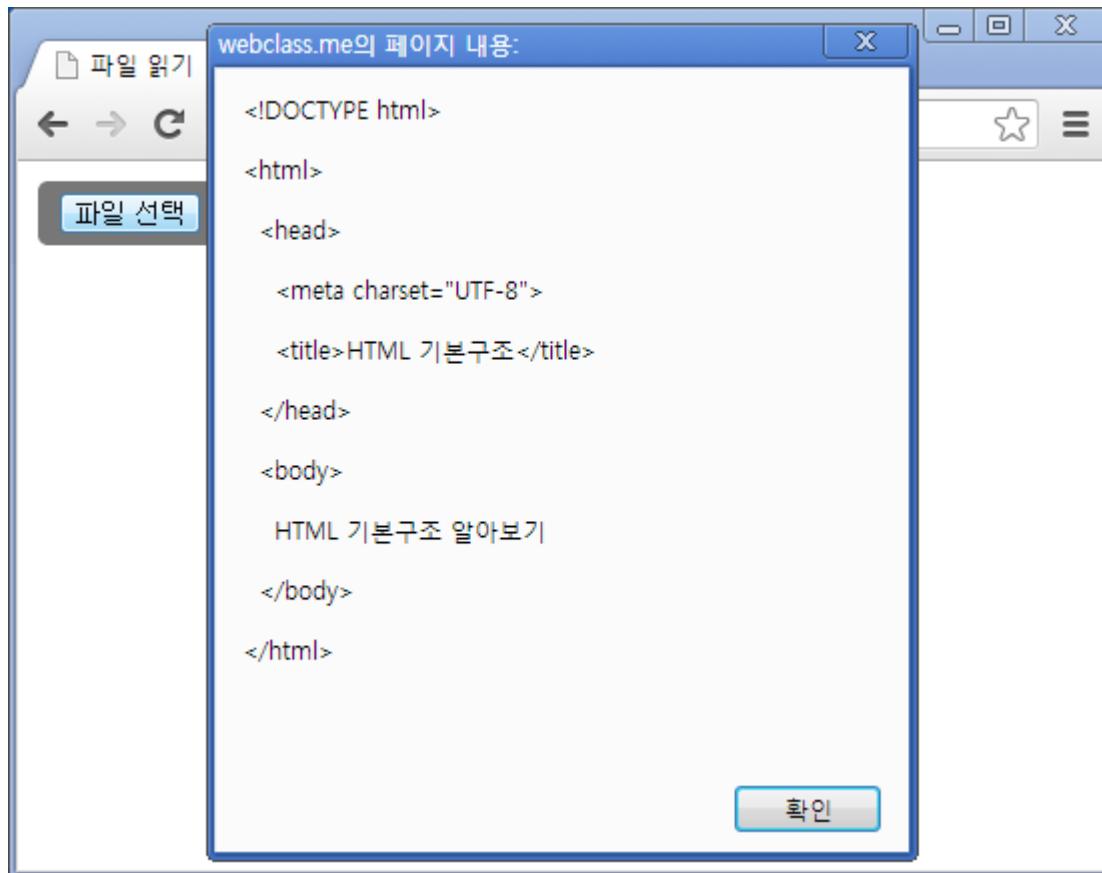
        reader.onload = function(e) {
            alert(e.target.result);
        };

        // Read in the file as a text.
        reader.readAsText(f);
    }

    document.getElementById('files').addEventListener('change',
                                                handleFileSelect, false);
</script>
```



실행결과



8.3 그외 저장 관련 API

8.3.1 오프라인 웹 애플리케이션

8.3.2 인덱스드 데이터베이스

8.3.3 웹SQL

오프라인 웹 애플리케이션

■ 오프라인 웹 애플리케이션

- 네트워크 연결이 없는 상태에서도 작동 가능한 웹 애플리케이션
- 구성요소
 - ▶ 매니페스트 파일(manifest) : 웹 애플리케이션과 관련된 여러 파일 중 어떤 파일을 PC와 같은 기기에 저장해야 하는지를 지정하는 파일
 - ▶ 애플리케이션 캐시(application cache) : 저장 가능



오프라인 웹 애플리케이션

■ 매니페스트(manifest) 파일의 구성 부분

- 캐시(cache) 부분: 여러 파일 중에서 기기에 저장할 파일을 지정
- 네트워크(network) 부분: 서버에서 항상 가져와야 하는 파일을 지정
- 폴백(fallback) 부분: 파일을 가져오지 못하는 경우 대신 사용할 파일을 지정

```
// CACHE 부분  
CACHE MANIFEST  
README.html  
// NETWORK 부분  
NETWORK:  
api.css  
// Fallback 부분  
FALLBACK:  
index.html README.html
```

■ 매니페스트(manifest) 파일 지정

```
<!DOCTYPE html>  
<html manifest="example.manifest">  
...  
</html>
```



인덱스드 데이터베이스

■ 인덱스드 데이터베이스(Indexed Database)

- 웹응용에서 자료를 클라이언트에 저장하기 위한 기능
- 웹스토리지와 다른 점: 저장되는 값의 자료형이 문자열형 만 가능
- 인덱스드 데이터베이스 특징
 - ▶ 자바스크립트객체
 - ▶ 저장되는 키와 자료에 인덱스를 줄 수 있어, 키 또는 인덱스로 자료를 읽고 쓸 수 있음
 - ▶ 데이터베이스에서 지원하는 커서 개념을 지원

웹SQL

■ 웹SQL

- 웹브라우저에서 데이터 관리를 위해 SQL을 지원하는 데이터베이스 API
- 관련 함수
 - ▶ openDatabase() 함수와 transaction() 함수, executeSql() 함수

```
var db = openDatabase('bookstoredb', '1.0', 'Bookstore DB', 2 * 1024 * 1024);
db.transaction(function (tx) {
    tx.executeSql('CREATE TABLE IF NOT EXISTS BOOKS (isbn unique, title)');
    tx.executeSql(
        'INSERT INTO BOOKS (isbn, title) VALUES ("8970505962", "자바입문")');
    tx.executeSql(
        'INSERT INTO BOOKS (isbn, title) VALUES ("897050415X", "C#입문")');
});
db.transaction(function (tx) {
    tx.executeSql('SELECT * FROM BOOKS', [], function (tx, results) {
        var len = results.rows.length, i;
        document.write("<p>자료개수: " + len + "</p>");
        document.write("<ul>");
        for (i = 0; i < len; i++){
            document.write("<li>" + results.rows.item(i).title + "</li> ");
        }
        document.write("</ul>");
    }, null);
});
```



8.4 기타 HTML5 API

8.4.1 웹소켓

8.4.2 웹RTC

8.4.3 웹워커

8.4.4 웹GL

웹소켓

■ 웹소켓(Web Socket)

- 웹에서 양방향 통신을 지원하기 위해 만든 기능
- 클라이언트인 웹브라우저 뿐만 아니라 서버인 웹서버도 웹소켓을 지원해야 함
- URL 형식: ws:///* 형식과 wss:///* 형식
- window.WebSocket 객체 사용

종류	이름	설명
메소드	send()	데이터를 전송
	close()	연결 종료
속성	URL	웹소켓 연결 URL
	readyState	웹소켓 연결 상태를 나타냄
	bufferedAmo unt	전송되지 않고 버퍼된 데이터 양을 나타냄
이벤트	open	연결이 되었을 때 발생
	message	데이터를 수신했을 때 발생
	close	연결이 종료되었을 때 발생



기타 API

웹RTC

■ 웹RTC(Web Real Time Communication)

- 웹에서 플러그인 없이 마이크로소프트의 스카이프나 구글 플러스의 행아웃과 같은 음성통화, 비디오 채팅, 영상통화, P2P 통신을 할 수 있도록 해주는 기능
- 구성 기능
 - ▶ 카메라와 마이크와 같은 장치로부터 입력을 받을 수 있는 미디어 캡처 API
 - ▶ 피어통신(peerconnection), 데이터채널관리와 같은 통신 기능

```
<video id="sourceid"></video>
<script>
var video = document.getElementById('sourceid');
navigator.getUserMedia('video', success, error);
function success(stream) {
    video.src = window.URL.createObjectURL(stream);
}
</script>
```



웹워커

■ 웹워커(Web Worker)

- 웹 애플리케이션에서 병렬처리를 지원하기 위해 만든 기능
- Worker 객체 사용

종류	이름	설명
메소드	postMessage()	워커 쓰레드에 자료 전송
	terminate()	워커 쓰레드 종료
이벤트	message	워커 쓰레드로부터 자료를 받았을 때 발생



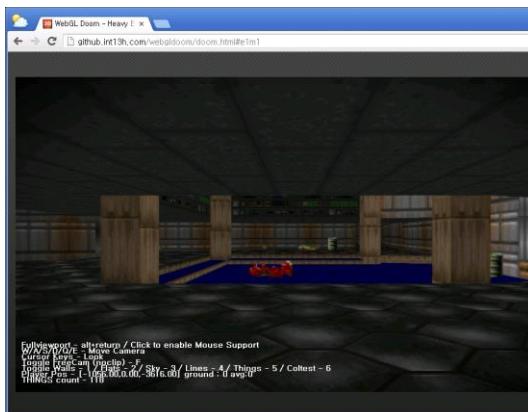
웹GL

■ 웹GL(WebGL)

- OpenGL ES 2.0에 기반한 웹을 위한 3D 그래픽 자바스크립트 인터페이스

```
// Get A WebGL context
var canvas = document.getElementById("canvas");
var gl = canvas.getContext("experimental-webgl");

// setup a GLSL program
var vertexShader = createShaderFromScriptElement(gl, "2d-vertex-shader");
var fragmentShader = createShaderFromScriptElement(gl, "2d-fragment-shader");
var program = createProgram(gl, [vertexShader, fragmentShader]);
gl.useProgram(program);
```



9장. 캔버스

목차

- 9.1 캔버스 이해하기
- 9.2 캔버스 기본 API 사용하기
- 9.3 캔버스 고급 기능 사용하기



9.1 캔버스 이해하기

9.1.1 캔버스의 특징

9.1.2 캔버스와 컨텍스트 객체

HTML5 캔버스

■ 자바스크립트를 이용해서 웹 문서상에 그림 그리는 기능

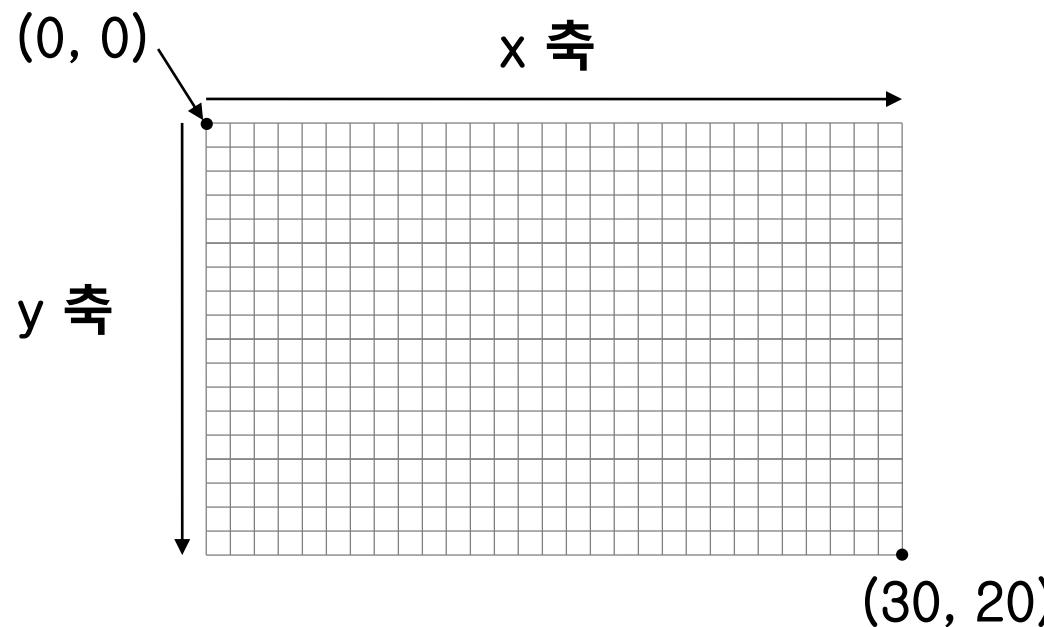
- HTML5 이전
 - ▶ 직접 이미지 파일을 태그를 이용해서 문서상에 포함
 - ▶ 자바 애플릿 이용
 - ▶ 플래시 이용
- HTML5 캔버스
 - ▶ 자바스크립트만을 이용해서 그림을 그릴 수 있다
 - ▶ 별도의 플러그인이나 프로그램 설치 없이 가능
 - ▶ 이미지나 그림을 합성, 변환 조작도 가능



캔버스 좌표계

■ 사각 평면의 2차원 좌표계 사용

- 이차원 (2D) 이미지 표현
- x, y 2개의 축으로 구성
- 왼쪽 상단 모서리가 원점 (0, 0)



비트맵 그래픽

■ 픽셀(pixel)

- 좌표계 상의 각각의 정사각형 네모 칸
- 이미지를 구성하는 점이며 색상을 가진다.
- 각 픽셀의 색상 값을 바꾸어 다양한 이미지를 표현

■ 비트맥 그래픽

- Bitmap graphics
- 픽셀만으로 이미지를 표현하고 저장하는 형태

■ 캔버스의 도형이나 그림, 글씨 등 2차원 비트맵으로 저장

- 이미 그려진 도형이나 그림을 확대하는 등은 작업은 불가능

캔버스로 그림 그리기 준비

■ 캔버스 요소

- <canvas> 태그를 이용해서 캔버스 요소 추가
- width와 height 속성을 이용해 캔버스 좌표계의 크기 지정
- DOM을 통한 접근을 위해 id 지정

```
<canvas id="myCanvas" width="30" height="20"></canvas>
```

■ 컨텍스트(context) 객체

- 캔버스에 내용을 채우기 위한 객체
- 캔버스 요소 객체의 getContext() 메소드를 이용
 - ▶ <canvas> 요소 객체에 접근한 후 getContext("2d") 메소드 실행

```
var canvas = document.getElementById("myCanvas");
var context = canvas.getContext("2d");
```



9.2 캔버스 기본 API 사용하기

9.2.1 기본 도형 그리기

9.2.2 기본 도형 꾸미기

9.2.3 이미지와 글자 그리기

캔버스 기본 도형 그리기

■ 캔버스 기본 API

- 컨텍스트 객체의 메소드를 호출함으로써 그림이 그려짐

■ 캔버스 컨텍스트의 선 그리기 메소드

- 선긋기, 경로, 곡선 등
- 현재 시작 지점에서 다음 지점까지 선을 연결하는 방식

▶ 현재위치 이동

- `moveTo(x,y)`: 현재 시작 지점을 이동시키는 메소드
- 선을 그린 마지막 지점으로 현재 위치 이동

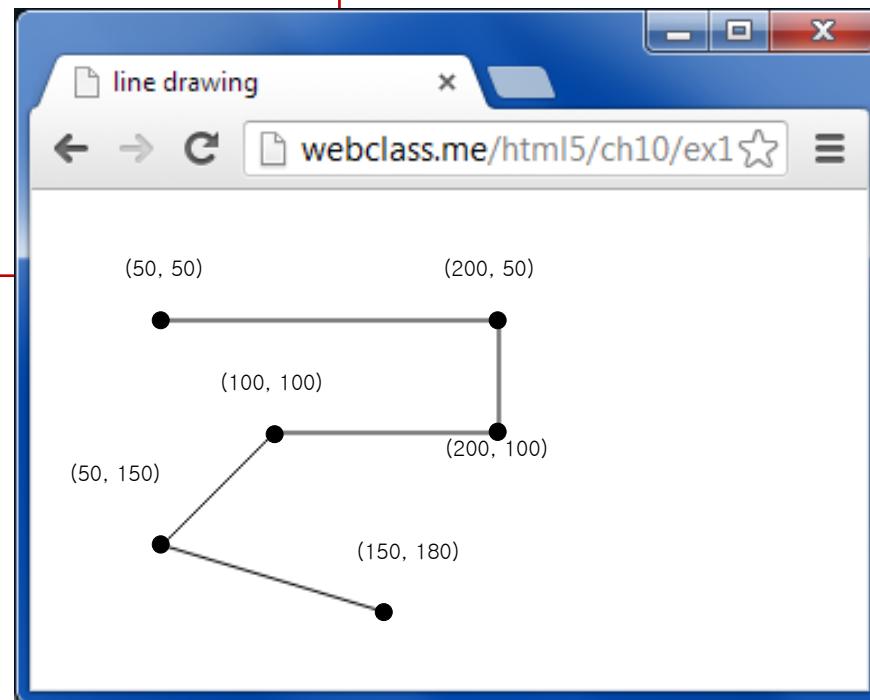
캔버스 기본 도형 그리기

■ 기본 도형 그리기 메소드

캔버스 컨텍스트 메소드	기능 및 설명
<code>context.moveTo(x, y)</code>	선의 시작 지점을 (x, y) 좌표로 이동시킨다.
<code>context.lineTo(x, y)</code>	현재의 시작점에서 (x, y) 지점까지 선을 그린다.
<code>context.rect(x, y, width, height);</code>	왼쪽 위 모서리를 (x, y) 지점으로 하고 가로와 세로 변의 크기가 각각 width, height인 사각형을 그린다. 현재의 시작점을 (x, y)로 이동시킨다.
<code>context.stroke();</code>	현재 지정된 색상과 선 끝 모양으로 선을 그린다. <code>stroke()</code> 메소드를 실행하지 않으면 선이 그려지지 않는다. 기본 색상은 검정색이다.

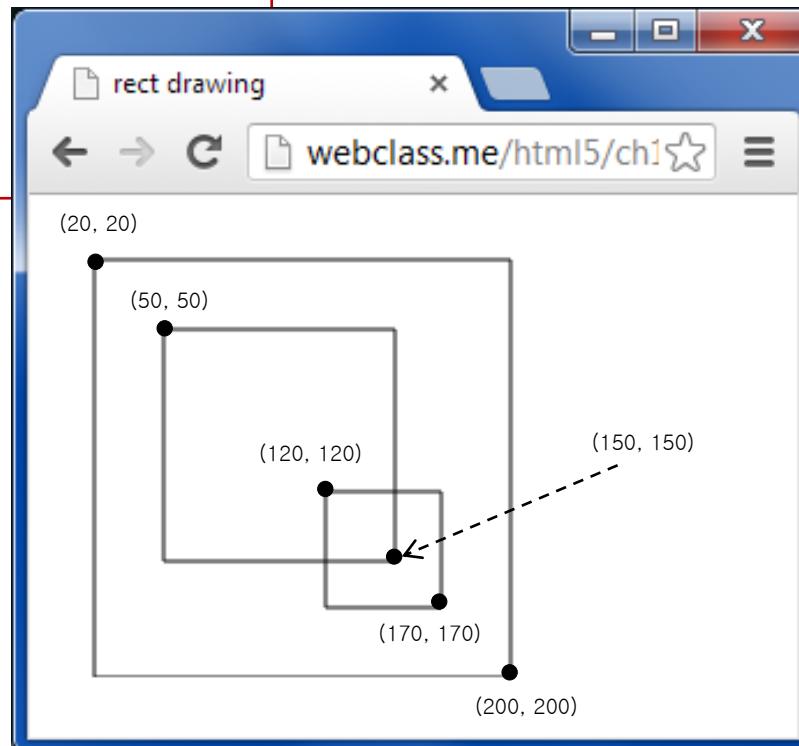
직선 그리기 예제

```
1 var canvas = document.getElementById("myCanvas");
2 var context = canvas.getContext("2d");
3
4 context.moveTo(50, 50);
5 context.lineTo(200, 50);
6 context.lineTo(200, 100);
7 context.lineTo(100, 100);
8 context.lineTo(50, 150);
9 context.lineTo(150, 180);
10 context.stroke();
```



사각형 그리기 예제

```
1 var canvas = document.getElementById("myCanvas");
2 var context = canvas.getContext("2d");
3
4 context.rect(50, 50, 100, 100);
5 context.rect(20, 20, 180, 180);
6 context.rect(120, 120, 50, 50);
7 context.stroke();
```



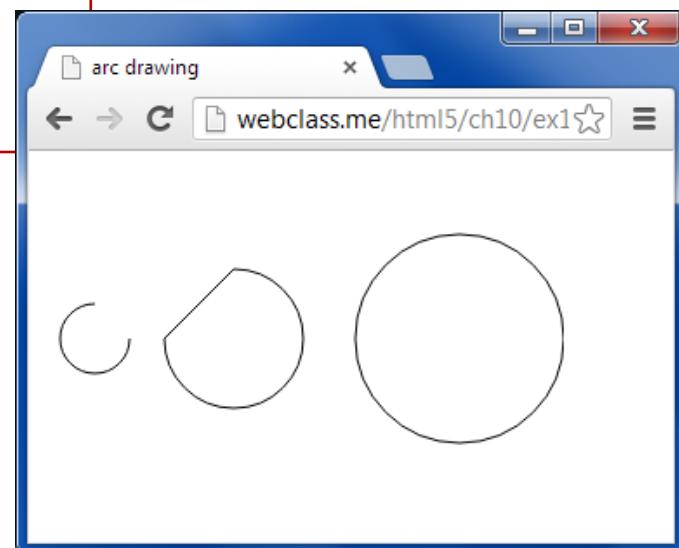
원호와 곡선 그리기

- `stroke()` 메소드를 호출하지 않으면 실제로 캔버스에 선이 그려지지 않음에 유의

캔버스 컨텍스트 메소드	기능 및 설명
<code>context.arc(x, y, r, startAngle, endAngle, antiClockwise)</code>	(x, y)를 원점으로 하고 반지름 r인 원호를 그린다. 시작 각도와 끝 각도를 지정하여 원호를 그린다. <code>antiClockwise</code> 값을 <code>false</code> 로 설정하면 시계방향으로 원호를 그린다. 기본값은 시계방향이다.
<code>context.quadraticCurveTo(cx, cy, x, y);</code>	하나의 제어점을 가지는 곡선을 그린다. 시작점은 현재 위치이며 끝 점은 (x, y)이다. (cx, cy)이 제어점이 된다.
<code>context.bezierCurveTo(cx1, cy1, cx2, cy2, x, y);</code>	두개의 제어점을 가지는 곡선을 그린다. 시작점은 현재 위치이며 끝 점은 (x, y)이다. 두개의 제어점은 (cx1, cy1)과 (cx2, cy2)로 지정한다.

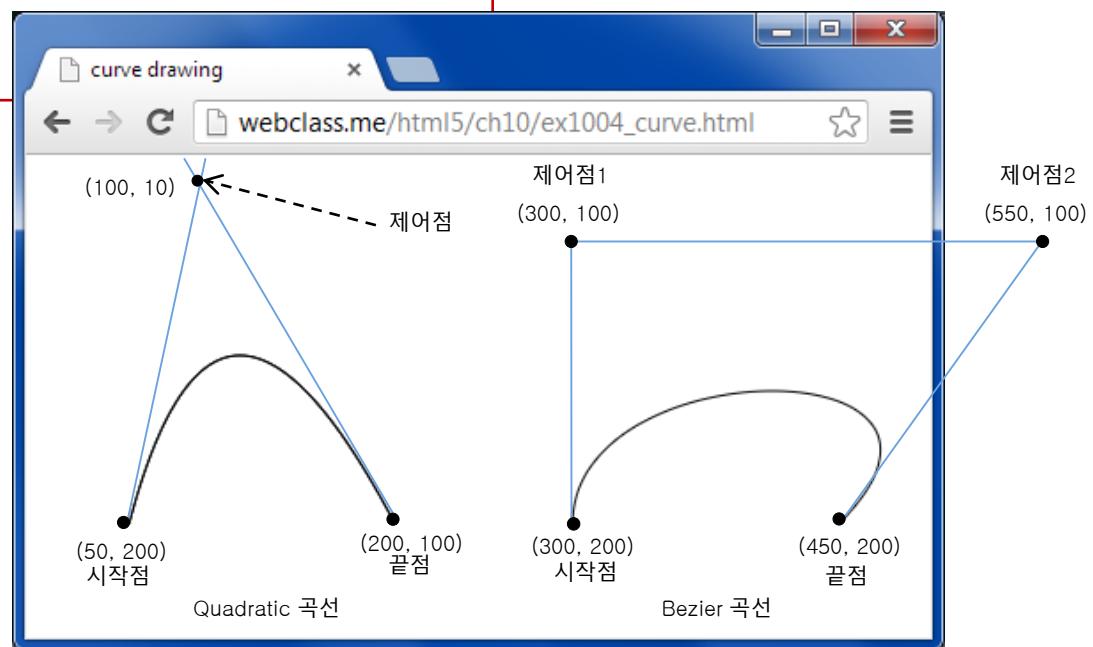
원호 그리기 예제

```
1 var canvas = document.getElementById("myCanvas");
2 var context = canvas.getContext("2d");
3
4 context.beginPath();
5 context.arc(30, 100, 20, 0, 1.5*Math.PI); // Math.PI 상수를 이용해 각도지정
6 context.stroke();
7
8 context.beginPath();
9 context.arc(110, 100, 40, 1*Math.PI, 1.5*Math.PI, true); // 반시계방향 원호
10 context.closePath(); // 경로 시작점까지 직선으로 연결하여 경로를 종료한다.
11 context.stroke();
12
13 context.beginPath();
14 context.arc(240, 100, 60, 0, 2*Math.PI); // 360도 원호를 그려 원 그리기
15 context.stroke();
```



곡선 그리기 예제

```
1 var canvas = document.getElementById("myCanvas");
2 var context = canvas.getContext("2d");
3
4 context.moveTo(50, 200);
5 context.quadraticCurveTo(100, 10, 200, 200);
6
7 context.stroke();
8
9 context.moveTo(300, 200);
10 context.bezierCurveTo(300, 100, 600, 100, 450, 450);
11
12 context.stroke();
```



경로 그리기

■ 연속된 선 그리기를 통한 경로 그리기

- beginPath()
 - ▶ 경로의 시작 설정
- closePath()
 - ▶ 경로 지정을 종료
 - ▶ 처음 경로 시작 지점으로 선을 연결하여 경로를 완성

캔버스 컨텍스트 메소드	기능 및 설명
context.beginPath();	경로 지정을 시작하는 메소드이다.
context.closePath();	경로 지정의 종료를 의미하는 메소드이며 현재까지 그려진 경로의 마지막 위치에서 경로의 시작점까지 직선으로 연결한다. 그리고, 현재 위치는 경로의 시작점으로 이동 시킨다.

기본 도형 꾸미기

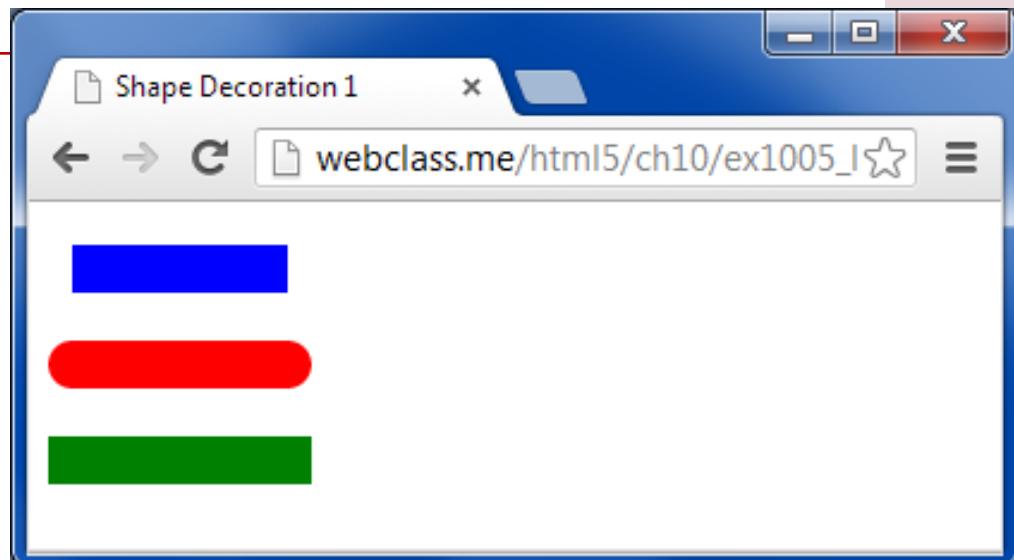
■ 선 꾸미기와 색칠하기

캔버스 컨텍스트 속성 및 메소드	기능 및 설명
context.lineWidth	선의 두께를 픽셀 개수로 설정한다.
context.strokeStyle	선의 색상을 지정한다. 색상을 지정하는 방법은 일반적인 웹 문서에서와 동일하다. 예) "blue" 혹은 "#0000ff" 등
context.lineCap	선의 양쪽 끝 모양을 지정한다. 지정할 수 있는 형태는 "butt", "round", "square"이며 기본 값은 "butt"이다.
context.lineJoin	선이 깍이는 모서리 지점에서의 모양을 지정한다. "miter", "round", "bevel" 세가지 중의 한가지 값으로 지정할 수 있다. 기본값은 "miter" 스타일이다.
context.fillStyle	경로, 원, 사각형 등의 도형의 내부를 색칠할 색상 값을 지정한다. 스타일값으로 그라데이션이나 패턴을 지정할 수도 있다. 색상은 웹 문서와 동일하게 지정할 수 있다. 예) "red" 혹은 "#ff0000" 등
context.fill();	현재 지정된 fillStyle 색상으로 도형을 채운다. 색칠할 도형은 fill() 메소드를 실행하기 이전에 그려지 모든 도형들이다.



선 꾸미기 예제

```
1 context.beginPath();
2 context.moveTo(10, 20);
3 context.lineTo(100, 20);
4 context.lineWidth = 20;
5 context.strokeStyle = "blue";
6 context.lineCap = "butt";
7 context.stroke();
8
9 context.beginPath();
10 context.moveTo(10, 60);
11 context.lineTo(100, 60);
12 context.strokeStyle = "red";
13 context.lineCap = "round";
14 context.stroke();
15
16 context.beginPath();
17 context.moveTo(10, 100);
18 context.lineTo(100, 100);
19 context.strokeStyle = "green";
20 context.lineCap = "square";
21 context.stroke();
```

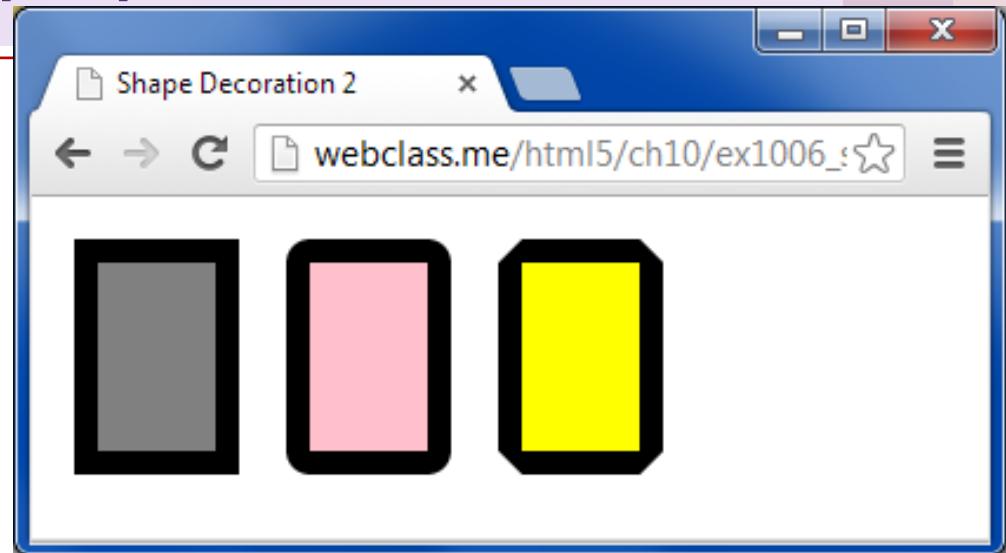


도형 꾸미기 예제

```
1 context.beginPath();
2 context.rect(20, 20, 50, 80);
3 context.lineWidth = 20;
4 context.strokeStyle = "black";
5 context.lineJoin = "miter";
6 context.fillStyle = "grey";
7 context.stroke();
8 context.fill();

9
10 context.beginPath();
11 context.rect(110, 20, 50, 80);
12 context.strokeStyle = "black";
13 context.lineJoin = "round";
14 context.fillStyle = "pink";
15 context.stroke();
16 context.fill();

17
18 context.beginPath();
19 context.rect(200, 20, 50, 80);
20 context.strokeStyle = "black";
21 context.lineJoin = "bevel";
22 context.fillStyle = "yellow";
23 context.stroke();
24 context.fill();
```



이미지 그리기

- 기존에는 이미지를 그리기 위해서는 태그를 이용
- 캔버스에 이미지 그리기
 - 사이즈 조정, 크롭(crop) 등의 기능도 가능
 - Image 객체를 이용
 - ▶ Image() 생성자를 이용해서 생성

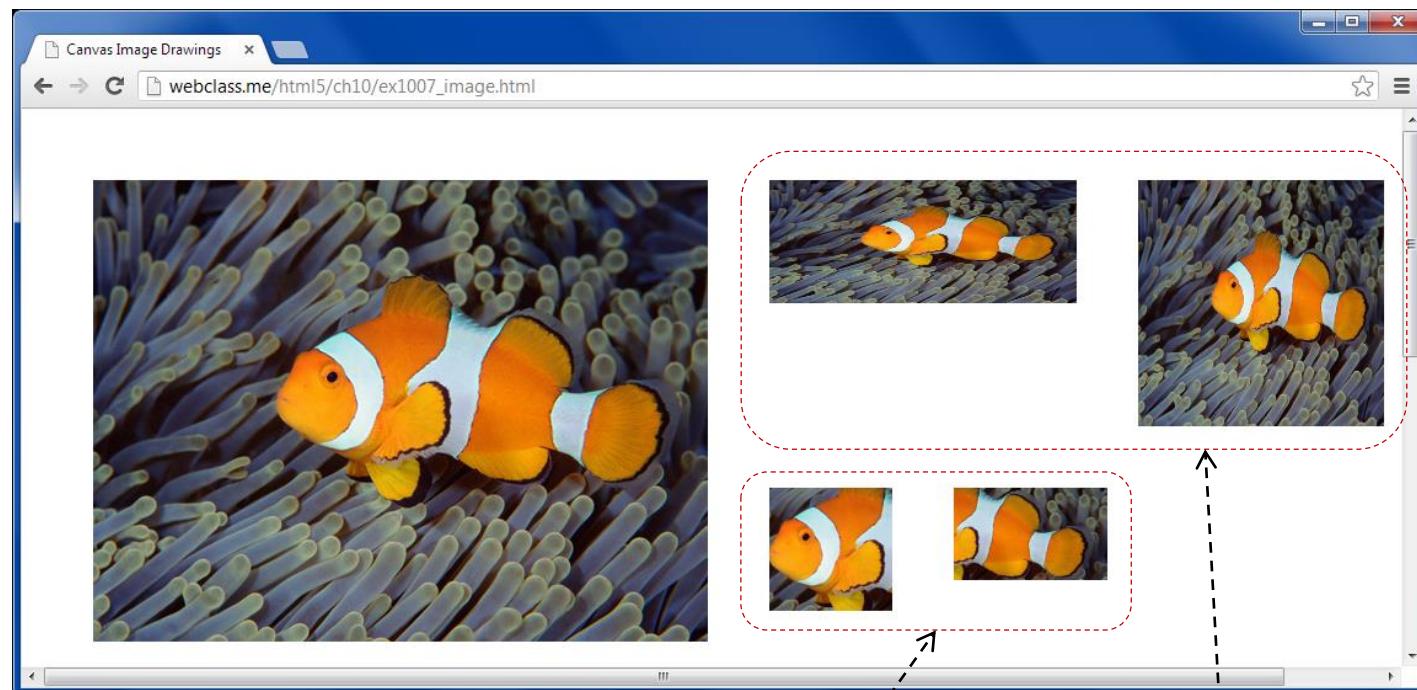
```
var imgObj = new Image();
```

- drawImage() 메소드
 - ▶ 캔버스 컨텍스트에서 이미지를 그리는 메소드

이미지 그리기 예제

```
1 var canvas = document.getElementById("myCanvas");
2 var context2d = canvas.getContext("2d");
3
4 var imgObj = new Image();
5 imgObj.src = "clownfish.jpg";
6
7 imgObj.onload = function() {
8
9     // (50, 50) 지점에 원래 크기 그대로 이미지 그리기
10    context2d.drawImage(imgObj, 50, 50);
11
12    // 크기조정하기: (600, 50) 지점에 250 x 100 크기로 이미지 그리기
13    context2d.drawImage(imgObj, 600, 50, 250, 100);
14
15    // 크기조정하기: (900, 50) 지점에 200 x 200 크기로 이미지 그리기
16    context2d.drawImage(imgObj, 900, 50, 200, 200);
17
18    // 이미지 자르기 후 크기조정:
19    // 1) 원본 이미지 (150, 100) 지점에서 150 x 50 크기의 이미지를 자른다.
20    // 2) Canvas의 (600, 300) 지점에 100 x 75 크리로 그리기
21    context2d.drawImage(imgObj, 150, 100, 150, 150, 600, 300, 100, 100);
22
23    // 이미지 자르기 후 크기조정:
24    // 1) 원본 이미지 (250, 100) 지점에서 250 x 150 크기의 이미지를 자른다.
25    // 2) Canvas의 (750, 300) 지점에 125 x 75 크리로 그리기
26    context2d.drawImage(imgObj, 250, 100, 250, 150, 750, 300, 125, 75);
```

이미지 그리기 예제 실행 결과



이미지 크롭

이미지 사이즈 조정

캔버스에 글자 그리기

■ 비트맵 방식으로 캔버스에 텍스트 그리기

- 삽입된 글자를 수정하거나 크기를 조정하는 것은 불가능
- 텍스트를 그려 넣을기 전에 폰트, 크기, 정렬방법 등을 결정

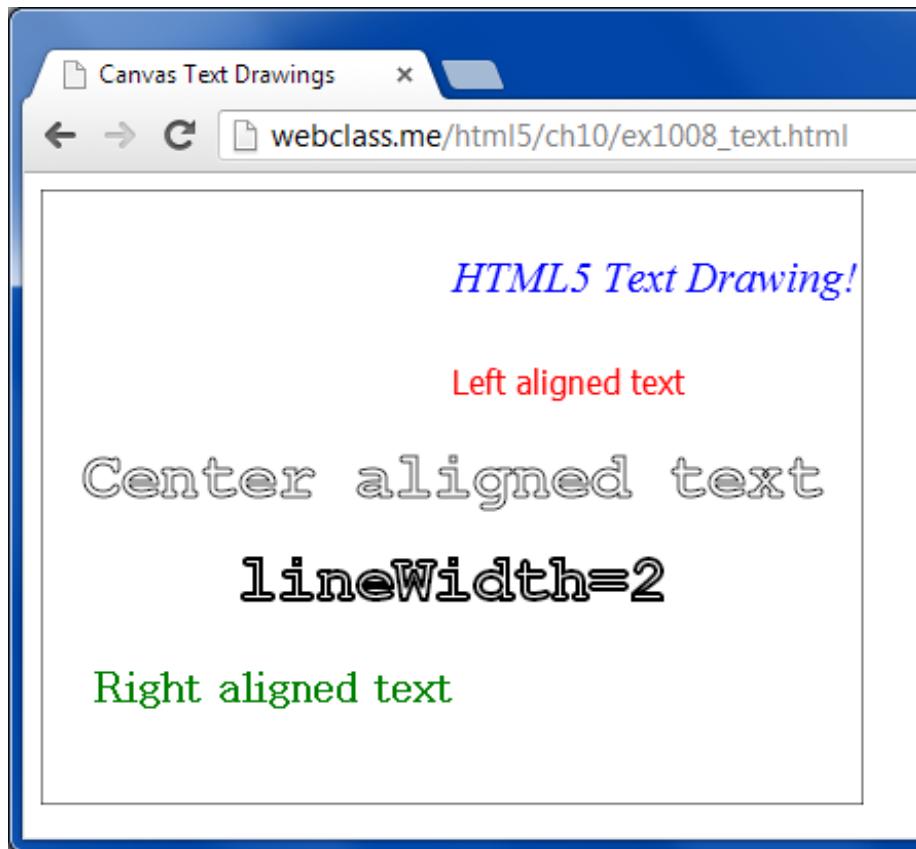
캔버스 컨텍스트 속성 및 메소드	기능 및 설명
context.font	그려 넣을 텍스트의 글자체를 지정한다. 이탤릭체 여부, 글자 크기, 폰트 등을 한번에 지정하게 된다.
context.textAlign	텍스트의 정렬방식을 지정한다. "left", "right", "center", "start", "end"의 값을 가질 수 있다. 기본 값은 "start"이다.
context.fillStyle	글자의 색상을 지정한다. 색상을 지정하는 방법은 일반적인 웹 문서에서와 동일하다. 예) "blue" 혹은 "#0000ff" 등
context.fillText()	현재 지정된 fillStyle 색상으로 캔버스의 지정된 위치에 글자를 를 그려 넣는다. 글자의 왼쪽 위 모서리 지점이 그려넣는 기준점이 된다.

캔버스에 글자 장식하기

■ 색상 및 외곽선 두께 등을 지정

캔버스 컨텍스트 속성 및 메소드	기능 및 설명
<code>context.strokeStyle = 색상값;</code>	글자의 외곽선을 그릴 색상을 지정한다. 색상을 지정하는 방법은 일반적인 웹 문서에서와 동일하다. 예) "blue" 혹은 "#0000ff" 등
<code>Context.lineWidth = 선두께;</code>	글자의 외곽선을 그릴 선의 두께를 지정한다.
<code>context.strokeText(text, x, y);</code>	현재 지정된 <code>strokeStyle</code> 색상으로 캔버스의 (x, y) 위치에 <code>text</code> 의 외곽선을 그려 넣는다. 글자의 외곽선만 그려지게 되므로 내부가 비어있는 형태가 된다. 텍스트의 왼쪽 위 모서리 지점이 텍스트를 그려 넣는 기준점이 된다.

글자 그려넣기 예제



```
1 context.rect(0, 0, 400, 300);
2 context.stroke();
3
4 var text1 = "HTML5 Text Drawing!";
5 var text2 = "Left aligned text";
6 var text3 = "Center aligned text";
7 var text4 = "Right aligned text";
8
9
10 context.font = "italic 16pt Times New Roman";
11 context.fillStyle = "blue";
12 context.fillText(text1, 200, 50);
13
14 context.font = "12pt Tahoma";
15 context.fillStyle = "red";
16 context.textAlign = "left";
17 context.fillText(text2, 200, 100);
18
19 context.font = "bold 24pt Courier New";
20 context.strokeStyle = "black";
21 context.textAlign = "center";
22 context.lineWidth = 1;
23 context.strokeText(text3, 200, 150);
24 context.lineWidth = 2;
25 context.strokeText("lineWidth=2", 200, 200);
26
27 context.font = "bold 16pt Batang";
28 context.fillStyle = "green";
29 context.textAlign = "right";
context.fillText(text4, 200, 250);
```

9.3 캔버스 고급 기능 사용하기

9.3.1 그리기 효과

9.3.2 변환 효과

9.3.3 기타 고급 기능

그리기 효과

■ 합성 (composition) 효과

- 그림자 효과를 줄 수 있는 shadow
- 투명도 조절을 위한 globalAlpha
- 지정한 도형 모양으로 잘라내는 clip(클립)
- 도형간의 연산을 위한 operation

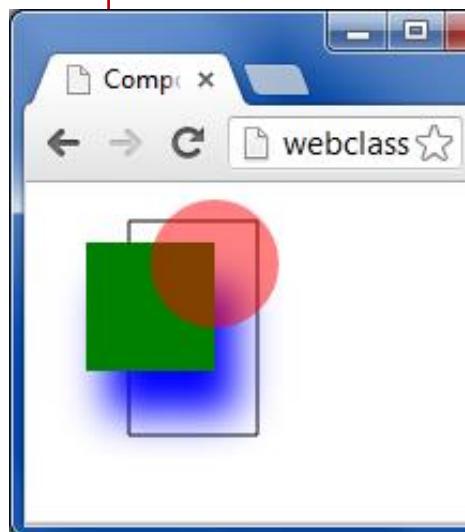
캔버스 컨텍스트 속성 및 메소드	기능 및 설명
context.shadowColor context.shadowBlur context.shadowOffsetX context.shadowOffsetY	그림자 효과를 줄 때 사용하는 속성들이다. 그림자의 색상, 흐림정도, 그림자의 크기를 지정할 수 있다. shadowOffsetX, shadowOffsetY 값을 조정함으로써 그림자의 크기를 조절할 수 있다.
context.globalAlpha	투명도를 조절하기 위해서는globalAlpha 속성값을 조절하면 된다. 0과 1 사이의 실수값을 가져야 하며 0이 완전 투명한 상태, 1이 완전히 불투명한 상태를 뜻한다.
context.clip()	clip() 메소드가 실행되기 바로 이전에 정의된 경로로 도형 자르기를 수행한다. 경로는 path() 등을 이용해 지정하게 된다.

그리기 효과 예제

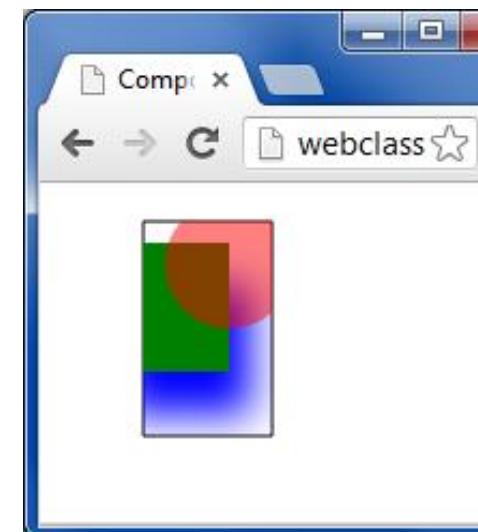
```
1 <script type="text/javascript">
2   var canvas = document.getElementById("myCanvas");
3   var context = canvas.getContext("2d");
4
5   context.beginPath();
6   context.rect(40, 10, 60, 100);
7   context.closePath();
8   context.stroke();
9   context.clip();
10
11  context.beginPath();
12  context.rect(20, 20, 60, 60);
13  context.fillStyle = "green";
14  context.shadowColor = "blue";
15  context.shadowBlur = 30;
16  context.shadowOffsetX = 10;
17  context.shadowOffsetY = 20;
18  context.fill();
19
20  context.beginPath();
21  context.arc(80, 30, 30, 0, 2*Math.PI);
22  context.fillStyle = "red";
23  context.globalAlpha = 0.5;
24  context.shadowColor = "transparent";
25  context.fill();
26 </script>
```

클립 효과 사용시 유의 사항

- 잘라내고자 하는 그림을 그리기 이전에 clip() 메서드를 실행해 함
- clip() 메서드 실행 이전에 그려진 그림은 자르기 효과가 적용 안됨



(a) `clip()` 메소드를 실행
하지 않은 경우



(b) `clip()` 메소드를 실행
한 경우

도형간 연산

■ 연속해서 그려 넣는 도형간의 연산을 통해 합성을 수행

- globalCompositeOperation 속성값을 이용
 - ▶ 본 속성값을 지정한 시점의 전과 후에 대해 연산을 수행
 - ▶ 속성 지정 이전이 source, 속성 지정 이후가 destination
 - ▶ 기본 연산: source-over 연산

캔버스 컨텍스트 속성	가능한 속성 값
context.globalCompositeOperation	'source-atop' 'source-in' 'source-out' 'source-over' 'destination-atop' 'destination-in' 'destination-out' 'destination-over' 'lighter' 'darker' 'xor' 'copy';



변환 효과

■ 변환(transformation) 효과

- 주로 그림을 그려넣을때 위치 이동, 회전, 대칭 등의 기능을 수행

캔버스 컨텍스트 속성 및 메소드	기능	기능 및 설명
<code>context.translate(x, y);</code>	이동 변환	기준좌표를 (x, y) 만큼 이동시켜 도형이나 그림의 위치를 이동시킨다.
<code>context.scale(x, y);</code>	크기 변환	도형의 크기를 조절한다. 가로 세로 방향의 배율을 (x, y)값으로 조절가능하며 (1, 1)이 기준 값이며 1보다 크면 도형의 크기가 커지며 1보다 작은 값으로 설정하면 작아지게 된다.
<code>context.rotate(회전각도);</code>	회전 변환	도형과 그림을 회전시켜 그려 넣는다. 회전각도는 라디안(radian) 값으로 지정한다. 360° 는 2π 즉 $2 * \text{Math.PI}$ 로 지정할 수 있다. 회전하는 중심점은 context의 왼쪽 위 모서리이다.

상하/좌우 대칭 변환

- `scale()` 메소드의 인자 값을 조정하여 구현

```
// 좌우 대칭  
context.scale(-1,1);  
// 상하 대칭  
context.scale(1,-1);
```

사용자 정의 변환

■ transform() 메소드

```
context.transform(a, b, c, d, e, f);
```

■ 임의의 사용자 정의 변환 행렬을 지정

- $$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

▶ x, y 는 변환되기 이전 좌표

▶ x', y' 는 사용자 정의 변환에 의해 변환된 이후의 좌표 값



사용자 정의 변환 예제

■ 변환식

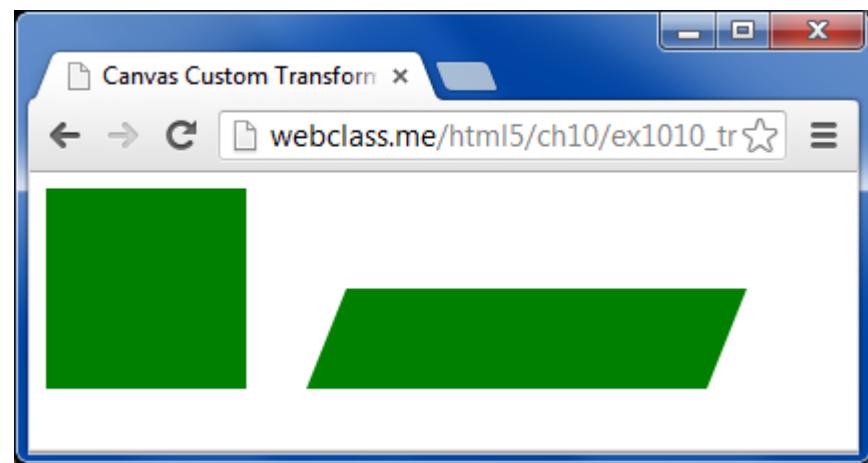
- 크기변환
 - ▶ $x' = 2x - 0.2y, y' = 0.5y$

- 이동 변환
 - ▶ (150, 50) 만큼 이동

■ 유의 사항

- 그림을 그려 넣기 전에 transform() 메소드를 실행해야 함

```
1 // original drawing
2 context.rect(0, 0, 100, 100);
3 context.fillStyle = "green";
4 context.fill();
5
6 // drawing after custom transformation
7 context.transform(2, 0, -0.2, 0.5, 150, 50);
8 context.rect(0, 0, 100, 100);
9 context.fill();
```



변환의 초기화

- 현재까지 지정된 변환이나 사용자 정의 변환 행렬을 초기화
 - `setTransform()` 메소드 이용
 - 아무런 변환을 지정하지 않은 기본 상태로 초기화

```
context.setTransform(1, 0, 0, 1, 0, 0);
```



픽셀 데이터 접근하기

■ 캔버스에 그려진 그림의 픽셀 데이터에 접근

- `getImageData()` 메소드
 - ▶ 리턴 객체의 `data` 속성에 각 픽셀 별 데이터가 1차원 배열로 저장되어 있음
 - `data` 속성 배열의 길이는 캔버스의 가로*세로*4
 - ▶ 픽셀의 구성: red, green, blue, alpha 값의 4개의 요소

```
var imgData= context.getImageData(0, 0, canvas.width, canvas.height);  
var data = imgData.data;
```

- `putImageData()` 메소드
 - ▶ 픽셀 데이터의 값을 수정 후 다시 컨텍스트에 반영

```
context.putImageData(imgData, 0, 0);
```

데이터 URL로 저장하기

- 그림을 PNG(Portable Network Graphics) 등의 형식으로 저장
 - 캔버스의 `toDataURL()` 메소드 이용

- ▶ 그림을 `toDataURL()` 메소드를 이용해서 PNG 형태의 데이터로 변환
 - ▶ 이를 캔버스 요소의 `src` 속성으로 지정하면 파일로 저장이 가능

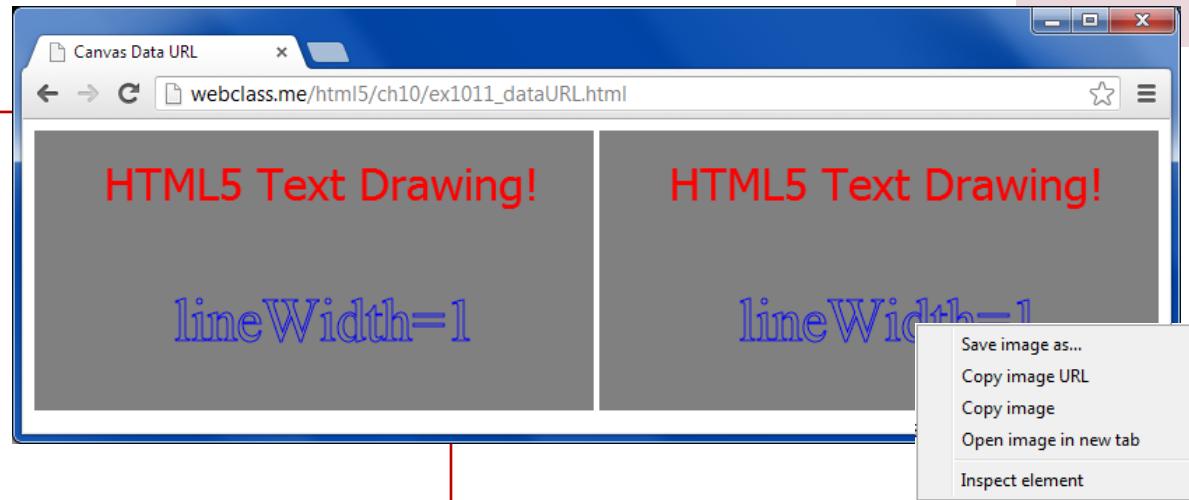
```
var dataURL = canvas.toDataURL();
canvasDom.src = dataURL;
```

- 유의 사항
 - ▶ `toDataURL()` 메소드는 캔버스 컨텍스트의 메소드가 아닌 캔버스 객체의 메소드



데이터 URL 저장 예제

```
1 context.rect(0, 0, 400, 200);
2 context.fillStyle = "grey";
3 context.fill();
4
5 var text1 = "HTML5 Text Drawing!";
6
7 context.font = "24pt Tahoma";
8 context.fillStyle = "red";
9 context.fillText(text1, 50, 50);
10
11 context.lineWidth = 1;
12 context.font = "32pt San Serif";
13 context.strokeStyle = "blue";
14 context.strokeText("lineWidth=1", 100, 150);
15
16 // Canvas 이미지를 data URL로 저장한다. 기본 형식은 PNG 포맷이다.
17 var dataURL = canvas.toDataURL();
18
19 // dataURL을 "canvasImage" 엘리먼트의 src 속성으로 지정하여
20 // 마우스 오른쪽 버튼을 이용하여 PNG file로 저장될 수 있도록 한다.
21 document.getElementById("canvasImage").src = dataURL;
```



캔버스 이미지 (왼쪽)과 데이터 URL 방식으로 저장한 PNG 이미지 (오른쪽)

캔버스 비트맵 초기화

- 캔버스 비트맵을 초기화할 수 있는 가장 간편한 방법은 clearRect() 메소드를 이용
 - (x, y) 위치를 기준으로 width, height의 폭과 높이의 비트맵을 초기화 한다

```
context.clearRect(x, y, width, height);
```



| 판교 | 경기도 성남시 분당구 삼평동 대원판교로 670길 유스페이스2 B동 8층 T. 070-5039-5805
| 가산 | 서울시 금천구 가산동 371-47 이노플렉스 1차 2층 T. 070-5039-5815
| 웹사이트 | <http://edu.kosta.or.kr> 팩스 | 070-7614-3450