

---

# HTML CSS Javascript

정종욱 jongukjeong@gmail.com

---

---

## **Part I**

# 웹표준이란?

## **Part II**

# HTML & CSS

- 1. HTML & CSS** 기초
- 2. CSS-**박스 모델
- 3. CSS-**글꼴 및 텍스트의 이해
- 4. CSS-**레이아웃설계

## **Part III**

# JavaScript

- 1. JavaScript Basic**
- 2. jQuery**
- 3. AJAX**

# Part I

## 웹 표준이란?

1. 웹 표준 등장배경
2. 웹 표준 정의
3. 웹 접근성과 웹표준

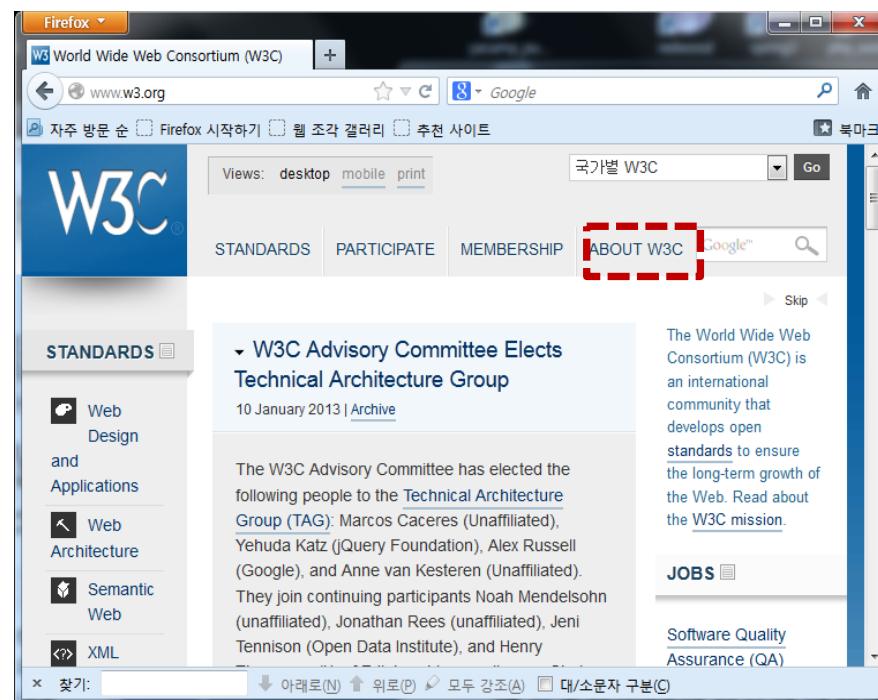
## 1. 웹 표준 등장배경

- 초창기 웹은 넷스케이프가 독주하는 상황이었지만, 윈도우98부터 IE브라우저가 기본 탑재되어 IE6.0는 브라우저 시장의 90%를 점유 했었다.
- IE의 ActiveX (자사 기술 독점)에 따른 웹 환경의 폐해
- IE이외의 사파리, 크롬, 파이어폭스, 오페라 등 다양한 브라우저가 데스크톱 환경에서 영향력이 커졌다
- 특히, 모바일 환경에서는 사파리, 크롬, 파이어폭스가 주도권을 잡았다.
- 특정 웹 브라우저에 종속되는 웹 페이지는 더 이상 설 자리가 없음을 의미
- 웹 페이지 제작 기술에 표준의 필요성이 대두

## 2. 웹 표준 정의

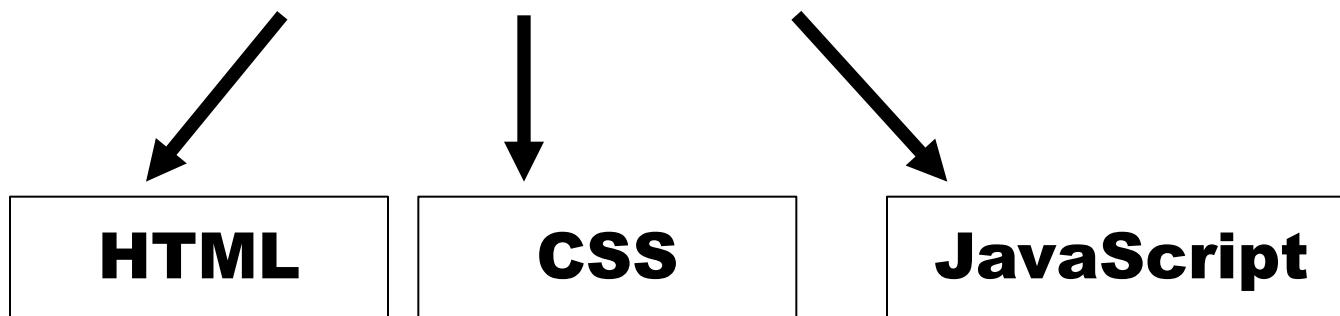
### □ 웹 표준을 주도하는 W3C( World Wide Web Consortium )

- <http://www.w3.org>
- 웹 기술의 표준화를 담당하는 기구
- HTML, CSS, XML 등 웹과 관련된 다양한 기술에 대해 논의하며 관련기술 표준 재정
- 웹 표준은 W3C의 목표와 비전에 따라 연구한 결과를 웹에 반영시키기 위한 권고 사항



## 2. 웹 표준 정의

- 특정 브라우저에서만 사용하는 비표준화된 기술은 배제
- W3C에서 정한 표준 기술
- 웹문서의 **구조**와 **표현** 그리고 **동작**을 구분해서 사용(제작)



- 요약

- HTML과 CSS를 웹의 표준으로 사용하자
- HTML은 구조를 잡고, CSS는 디자인을 담당하도록 분리하자

## 2. 웹 표준 정의

---

### □ HTML 과 CSS를 웹의 표준으로 사용하자

- 웹 표준에 따라 웹 페이지를 작성하게 되면 표준을 따르는 브라우저에서는 모두 같은 정보를 얻게 되므로 사용자가 가장 큰 혜택을 보게 된다.
- IE, 파이어폭스, 크롬, 사파리등 서로 다른 브라우저라도 같은 화면을 보여 주는 것, 즉 브라우저의 벽을 허물어 주는 개념을 ‘크로스 브라우징’이라 한다.

### □ HTML은 구조를 잡고, CSS는 디자인을 담당하도록 분리하자

- 정보의 구조는 HTML이 담당하고 CSS는 레이아웃과 디자인을 담당하도록 분리
- 하나의 콘텐츠 소스는 다양한 장비와 브라우저에서 사용할 수 있다는 의미

### 3. 웹 접근성과 웹 표준

---

#### □ 웹 접근성

- 어떤 장비, 어떤 브라우저를 사용하던지 해당 사이트를 얼마나 볼 수 있는냐를 의미
- 일반인이건 장애인이건 어떤 사용자도 해당 사이트 사용에 어려움이 없어야 한다는 의미
- 웹 표준을 지켜 작성한 HTML 문서는 웹 접근성을 준수하는 데 비표준 문서들 보다 쉽다.
- 표현을 CSS로 분리 시킨 HTML 문서는 웹 접근성을 준수하는 데 표현과 정보가 섞여 있는 문서들 보다 쉽다.

# Part II

## HTML & CSS

### 1. HTML & CSS 기초

1. HTML Basic
2. 시맨틱 마크업 (Semantic Markup)
3. CSS Basic

# 1. HTML Basic

---

## □ 웹언어 (= HTML)

- 웹페이지를 만들기 위해 **HTML(Hyper Text Markup Language)**를 사용하여 파일을 생성하고, 웹 서버에 그 파일을 올려 놓아야 한다.
- **HTML**은 페이지를 보여주기 위해서 알아야 할 필요가 있는 모든 것을 브라우저에게 알려준다.

## □ 실습예제1 ( wp\_ch2/ex1.html )

간단한 HTML 살펴보기 – SKP 라운지

# 1. HTML Basic

- SMU 라운지 HTML에 있는 태그를 브라우저가 어떻게 해석하는지 확인해 보기

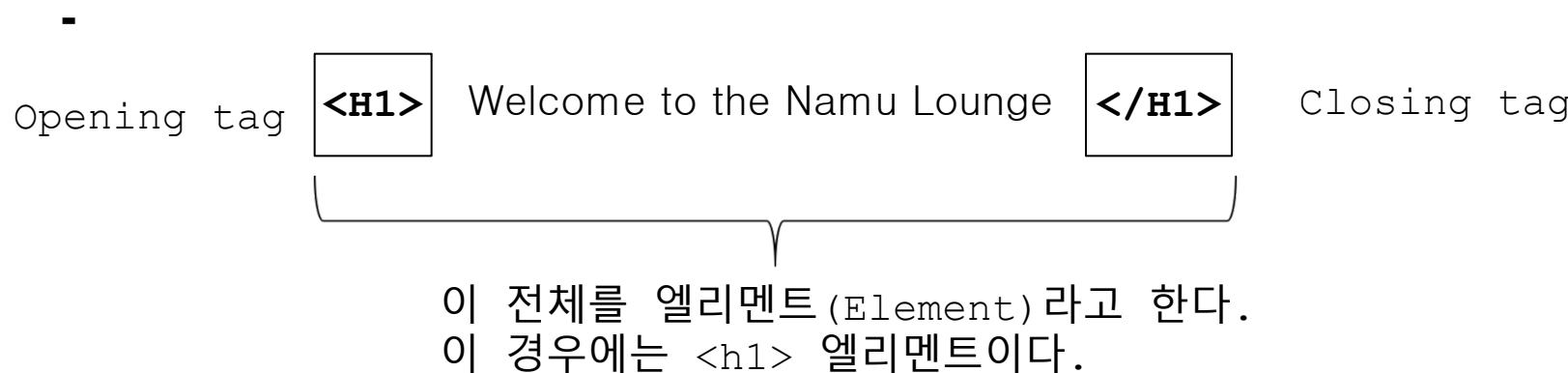
```
<html>
<head>
<title>SMU Lounge</title>
<meta http-equiv="Content-Type"
      content="text/html; charset=UTF-8">
</head>
<body>
<h1>Welcome to the SMU Lounge</h1>
  
  <p>
    소프트웨어관 라운지에서 시원한 건강 음
   료, 친구들과의 대화로
    하루 스트레스를 확 날려 버리세요.
    편안한 음악</em>도 감상하세요.
    무선인터넷에도 언제든지 접속 가능합니다.
  </p>
  <h2>오는 길</h2>
  <p>
    소프트웨어관 7층에서 바로 찾을 수 있어요.
  </p>
</body>
</html>
```



# 1. HTML Basic

## □ 태그

- < > 사이에 오는 단어나 문자 ex) `<head>`, `<p>`, `<h1>`
- 브라우저에게 작성한 텍스트의 구조와 의미에 관해 알려준다.



- 웹페이지의 구조를 브라우저가 알 수 있게 하기 위해서는, 콘텐츠를 둘러싼 태그들의 쌍을 사용
- 엘리멘트 = 시작(opening) 태그 + 콘텐츠 (내용) + 종료(closing) 태그

# 1. HTML Basic

## □ 엘리먼트에 스타일 적용해 보기

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>SMU Lounge</title>
    <style type="text/css">
      body {
        background-color:#d2b482;
        margin-left:20%;
        margin-right:20%;
        border: 1px dotted gray;
        padding: 10px 10px 10px 10px;
        font-family: 굴림
      }
    </style>
  </head>
  <body>
    <h1>Welcome to the SMU Lounge</h1>
    
    <p>
      소프트웨어관 라운지에서 시원한 <a href="/wp_ch2/beverage/2013_0220/elixir.html">건강 음료</a>, 친구들과의 대화로
      하루 스트레스를 확 날려 버리세요.
      <em>편안한 음악</em>도 감상하세요.
      무선인터넷에도 언제든지 접속 가능합니다.
    </p>
    <h2>오는 길</h2>
    <p>
      소프트웨어관 7층에서 바로 찾을 수 있어요. ㅋㅋ
    </p>
  </body>
</html>
```

## 2. HTML Basic

---

### □ 하이퍼텍스트 (HT<sub>ML</sub>)

- 웹 전체의 기초가 되는 것
- 단일 페이지에서 벗어나 다른 페이지와 연결할 수 있게 해주는 것
- <a> 엘리멘트에 의해 구현
- <a href='>콘텐츠</a> 이런 식으로 링크가 걸리며, href 속성은 목적지를 명시

### □ 실습예제2

건강음료 상세 페이지가 추가 되었다.

ex1.html에 두 페이지의 링크 을 작성해 보세요.

## 2. HTML Basic

---

### □ 실습예제3

라운지 체계화 하기

1. **beverages** 그리고 **images** 폴더를 생성합니다.
2. **elixir.html** 파일을 **beverages** 폴더로 이동
3. 모든 그림파일은 **images** 폴더로 옮김
4. **ex1.html** 파일을 로드하고 링크 부분 클릭, 링크가 꺼져있으면 링크연결

## 2. 시맨틱 마크업

### □ 시맨틱 검색

'시맨틱 검색'은 검색로봇이 검색어 의미를 스스로 분석하고 추리해 원하는 정보를 더 정교하게 찾아주는 검색방식

### □ 시맨틱 마크업

'시맨틱 마크업'이란 HTML의 태그를 사용하여 문서 안의 내용이 담고 있는 의미가 무엇인지 표현할 수 있도록 구조를 작성하는 것을 말함

내용이 담고 있는 의미가 무엇인지 표현할 수 있도록 작성

HTML과 CSS를 사용하여 '구조'와 '표현'을 구분

## 2. 시맨틱 마크업

- '엘리먼트'는 HTML문서의 개별적인 구성요소
- '태그'는 부등호기호(<>)로 둘러 쌓여진 엘리먼트를 부르는 이름  
예) <p>내용</p>

An HTML element is an individual component of an HTML document.

- ▶ 엘리먼트는 HTML문서의 개별적인 구성요소이다.

Tags are composed of the name of the element, surrounded by angle brackets.

- ▶ 태그는 부등호기호 (<>)로 둘러 쌓여진 엘리먼트를 조합해서 부르는 이름이다.

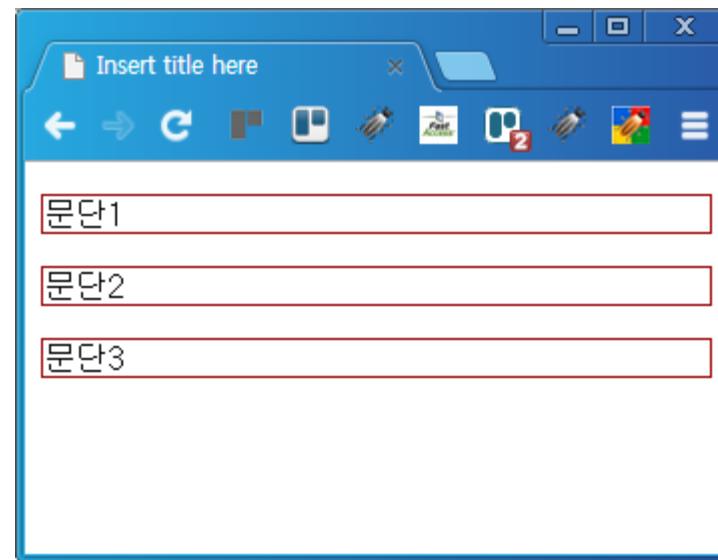
## 2. 시맨틱 마크업

### □ '블록레벨' 엘리먼트는 브라우저의 안에 블록처럼 쌓여 가면서 줄바꿈을 하여 표시

- 항상 새로운 줄로 줄 바꿈 하여 표시됨
- 너비값은 브라우저 화면에 100%로 꽉 차게 표시됨

### □ 실습예제 4.

블록 엘리먼트 < p > 를 사용해서 다음 그림과 같은 결과 값이 나오게 HTML를 작성하고 블록 엘리먼트의 특성을 확인해 보세요.



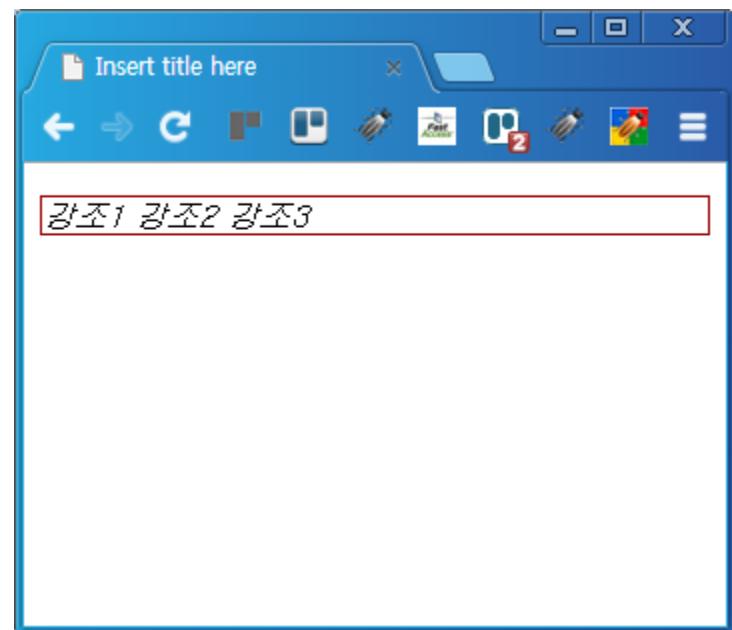
## 2. 시맨틱 마크업

□ '인라인레벨' 엘리먼트는 이미 표시된 엘리먼트에 이어서 같은 줄안에 표시되고 너비는 내용만큼만 차지함

- 같은 줄 안에서 이어서 표시됨
- 너비 값은 요소가 가지고 있는 값 자체로 표시됨

### □ 실습예제 5.

인라인 엘리먼트 <em> 를 사용해서 다음 그림과 같은 결과가 나오게 HTML를 작성하고 인라인 엘리먼트의 특성을 확인해 보세요.



## 2. 시맨틱 마크업

### □ 기본 마크업 태그

- <http://www.w3.org/TR/html401/index/elements.html>

W3C Recommendation

W3 Index of the HTML 4 Elements

www.w3.org/TR/html401/index/elements.html

previous next contents attributes index

### Index of Elements

Legend: Optional, Forbidden, Empty, Deprecated, Loose DTD, Frameset DTD

Name	Start Tag	End Tag	Empty	Depr.	DTD	Description
A						anchor
ABBR						abbreviated form (e.g., WWW, HTTP, etc.)
ACRONYM						
ADDRESS						information on author
APPLET				D	L	Java applet
AREA		F	E			client-side image map area
B						bold text style
BASE		F	E			document base URI
BASEFONT		F	E	D	L	base font size
BDO						I18N BiDi over-ride
BIG						large text style
BLOCKQUOTE						long quotation
BODY	O	O				document body
BR		F	E			forced line break

## 2. 시맨틱 마크업

### □ 기본 마크업 태그 h1

- <h1>은 가장 중요한 헤더정보에
- <h6>은 가장 낮은 중요도의 헤더정보를 정의
- <h1>첫번째로 중요한 제목</h1>
- <h2>두번째로 중요한 내용</h2>
- <h2>두번째로 중요한 내용</h2>
- <h3>세번째로 중요한 내용</h3>

**Heading** : <h1>,<h2>,...,<h6>

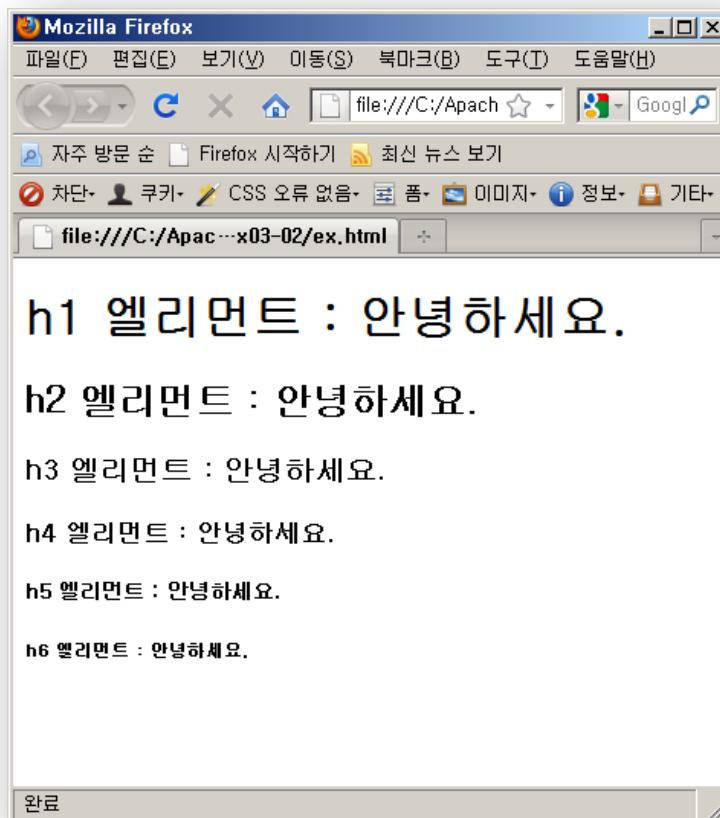
▶ <h1>부터 <h6>까지 헤더정보를 정의합니다.

## 2. 시맨틱 마크업

### □ 기본 마크업 태그 **h1 ( cont'd )**

### □ 실습예제 6

**heading** 태그 6개를 전부 사용해서 아래 그림과 같은 결과를 화면에 출력하세요.



## 2. 시맨틱 마크업

---

### □ 기본 마크업 태그 **p**

- p는 문단을 정의
- <p>첫 번째 문단입니다.</p>

### □ 기본 마크업 태그 **em**

- em은 강조를 나타냄
- <em>강조된 키워드</em>

### □ 기본 마크업 태그 **strong**

- strong은 더 강한 강조를 나타냄
- <strong>제일 중요한 키워드</strong>

## 2. 시맨틱 마크업

---

### □ 기본 마크업 태그 **address**

- 문서의 저자나 소유자를 위한 연락처 정보를 정의

예)

<address>

이름 : <a href='mailto:jongukjeong@gmail.com'>정종욱</a><br>

주소 : 서울 구로구 구로동<br>

전화번호 : 010-2074-9317<br>

</address>

## 2. 시맨틱 마크업

### □ 기본 마크업 태그 **div**

- CSS 스타일을 적용하여 디자인을 입힐 때 많이 사용
- 블록레벨 엘리먼트들을 그룹으로 묶음

예)

```
<div style='background-color: #008000'>  
  <h1>제목</h1>  
  <p>문장</p>  
</div>
```

- ▶ **div** 엘리먼트는 **html** 문서에서 영역을 정의합니다.
- ▶ **div** 엘리먼트는 블록레벨 엘리먼트들을 그룹으로 묶어서 **css** 스타일을 적용하여 디자인을 입힐 때 많이 사용합니다.

## 2. 시맨틱 마크업

### □ 기본 마크업 태그 **span**

- **span**은 html문서에서 인라인레벨 엘리먼트를 그룹으로 묶을 때 사용

예)

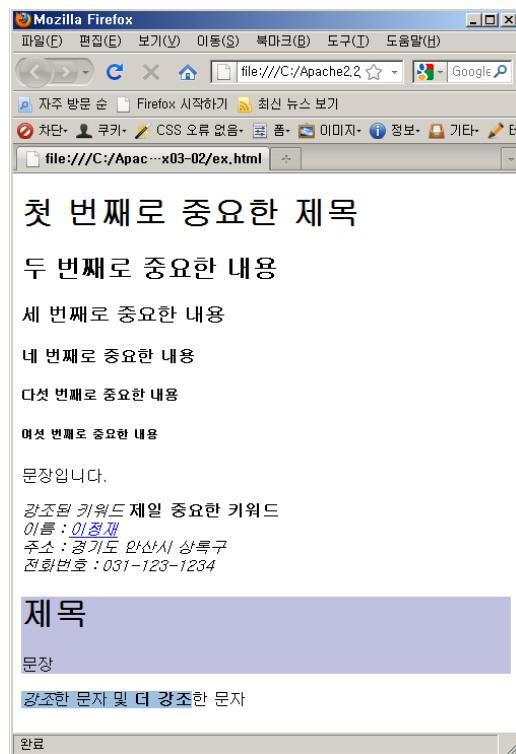
```
<p>
  <span style='background-color: #FFFF00'><em>강조</em>한 문자 및
  <strong>더 강조</strong></span>한 문자
</p>
```

- ▶ **span** 엘리먼트는 html문서에서 인라인레벨 엘리먼트를 그룹으로 묶을 때 사용합니다.
- ▶ **span** 엘리먼트로 묶여진 그룹은 css 스타일을 입히거나 자바스크립트로 처리할 때 사용합니다.

## 2. 시맨틱 마크업

### □ 실습예제 7

기본 마크업 태그를 사용하여 아래 그림과 같은 화면이 출력되도록 하세요.



## 2. 시맨틱 마크업

---

### □ DOCTYPE

- DOCTYPE은 html의 태그는 아니지만 웹페이지에서 맨 처음에 선언되어 어떤 종류의 html을 사용할지 웹브라우저에게 알려줄 목적으로 사용
- 브라우저는 DOCTYPE 선언을 확인하고 브라우저 모드를 결정한다.
- 브라우저 모드는 표준 호환 모드, 비표준 호환 모드
- 웹 표준에서는 웹 페이지를 제대로 표현하기 위해서는 올바른 문서 형태를 정의 해주어야 한다.
- 가장 많이 사용되고 있는 HTML 버전은 HTML4.01 과 XHTML1.0
- 최신 버전인 HTML5를 문서형식으로 사용하는 웹사이트도 증가 추세

## 2. 시맨틱 마크업

### □ W3C의 doctype 추천 페이지 (<http://www.w3.org/QA/2002/04/valid-dtd-list.html>)

The screenshot shows a Microsoft Internet Explorer window displaying the W3C QA Recommended DTD List. The page lists various document type declarations (DTDs) categorized by version and type.

- HTML 4.01:**
  - Strict

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
 "http://www.w3.org/TR/html4/strict.dtd">
```
  - Transitional

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
 "http://www.w3.org/TR/html4/loose.dtd">
```
  - Frameset

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
 "http://www.w3.org/TR/html4/frameset.dtd">
```
- XHTML 1.0:**
  - Strict (quick reference)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```
  - Transitional

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```
  - Frameset

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```
- XHTML 1.1 - DTD:**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
 "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```
- XHTML Basic 1.1 (quick reference):**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.1//EN"
 "http://www.w3.org/TR/xhtml-basic/xhtml-basic11.dtd">
```
- HTML 5 [NOT a standard yet]**

```
<!DOCTYPE HTML>
```
- MathML Doctype Declarations**
- MathML 2.0 - DTD:**

```
<!DOCTYPE math PUBLIC "-//W3C//DTD MathML 2.0//EN"
```

## **2. 시맨틱 마크업**

---

### **□ 실습예제 8 - HTML 4.01 DOCTYPE**

**ex1.html**에 **html 4.01 DOCTYPE**를 적용시키고 페이지에 어떤 변화가 있는지 확인해 보세요.

## 2. 시맨틱 마크업 (수정)

### □ HTML 4.01 DOCTYPE

- 3가지 형태의 DOCTYPE 지원
- 이전 버전으로 제작된 HTML의 지원을 위해

```
<!DOCTYPE HTML PUBLIC '-//W3C//DTD HTML 4.01//EN' 'http://www.w3.org/TR/html4/loose.dtd'>
```

- 정확한 표준모드로 사용하기 위해(W3C 권고)

```
<!DOCTYPE HTML PUBLIC '-//W3C//DTD HTML 4.01//EN' 'http://www.w3.org/TR/html4/strict.dtd'>
```

- 많이 사용하지 않지만, 보통 매뉴얼을 작성하거나 관리자 페이지에서 많이 쓰는 프레임 셋을 이용한 웹사이트에서

```
<!DOCTYPE HTML PUBLIC '-//W3C//DTD HTML 4.01//EN' 'http://www.w3.org/TR/html4/frameset.dtd'>
```

## 2. 시맨틱 마크업

### □ XHTML 1.0 DOCTYPE

- 3가지 형태의 DOCTYPE 지원
- 이전 버전으로 제작된 HTML의 지원을 위해

<!DOCTYPE html PUBLIC '-//W3C//DTD XHTML 1.0 Transitional//EN'

' <http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd>'

- 정확한 표준모드로 사용하기 위해(W3C 권고)

<!DOCTYPE html PUBLIC '-//W3C//DTD XHTML 1.0 Transitional//EN'

' <http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd>'

- 프레임 셋을 이용한 웹사이트 만들 때는 다음과 같은 DOCTYPE을 사용하고 Transitional과 동일하게 취급된다.

<!DOCTYPE html PUBLIC '-//W3C//DTD XHTML 1.0 Transitional//EN'

' <http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd>'

## 2. 시맨틱 마크업

---

### □ XHTML 1.1 DOCTYPE

- 1가지 형태의 DOCTYPE 지원
- XHTML 1.0의 문제점을 수정해서 만듬

<!DOCTYPE html PUBLIC '-//W3C//DTD XHTML 1.0 Transitional//EN'

' <http://www.w3.org/TR/xhtml1/DTD/xhtml11.dtd>'>

## 2. 시맨틱 마크업

---

### □ 실습예제 9

여러분 학교의 홈페이지는 어떤 DOCTYPE인가요?

## 2. 시맨틱 마크업

### □ HTML Validator ( 문서 유효성 검증 )

'HTML Validator'는 DOCTYPE에 선언된 데로 태그가 올바르게 작성되었는지 여부를 확인하고 인증해주는 서비스 ( [validator.w3.org](http://validator.w3.org) )

The screenshot shows the homepage of the W3C Markup Validation Service. It features a blue header bar with the W3C logo and the text "Markup Validation Service". Below the header, there are three tabs: "Validate by URI", "Validate by File Upload", and "Validate by Direct Input". A large input field labeled "Validate a document online:" is present, with a placeholder "Address:" and a "Check" button. At the bottom of the page, there is a note about the validator's capabilities and a section for donations.

The screenshot shows the results page after a document has been checked. The top bar indicates "[Valid] Markup Validation of upload://Form Submission - W3C Markup Validator". The main content area displays a green banner stating "This document was successfully checked as XHTML 1.0 Strict!". Below this, detailed validation results are shown, including the source code, encoding, doctype, root element, and namespace. The results indicate "Passed, 1 warning(s)". The page also includes links for "Jump To: Notes and Potential Issues" and "Congratulations - Icons". At the bottom, there is a message about community support and a "I ❤ VALIDATOR" button.

## **2. 시맨틱 마크업**

---

### **□ 실습예제 11**

**1) ex1.html**에 **html 4.01 – strict.dtd DOCTYPE**를 적용시키고

**w3c HTML Validator**에서 테스트를 받고 **DOCTYPE**에 만족하게끔 수정하세요.

**2) ex1.html**에 여러분 학교와 같은 **DOCTYPE**을 적용시키고

**w3c HTML Validator**에서 테스트를 받고 **DOCTYPE**에 만족하게끔 수정하세요.

### 3. CSS Basic

---

#### □ CSS ( Cascading Style Sheet ) 이해

- 워드프로세스의 스타일 적용과 유사
- cascading : ‘계단형’의 의미로 스타일 적용에 특정도, 또는 우선순위가 있고  
우선 순위가 정해지는 것이 계단식 스타일 시트라는 의미
- 필요한 이유 :
  1. HTML에 직접 스타일을 적용함으로써 생기는 HTML문서 자체의 무거움을 줄일 수 있다.
  2. 하나의 스타일로 다수의 페이지에 같은 속성을 적용 함으로써 작업시간 단축
  3. 웹 표준의 원칙 : HTML 마크업을 통해 구조를 잡고, CSS로 디자인을 입힘

### 3. CSS Basic

---

#### □ CSS 의 진화 과정

- W3C을 통해 표준 관리
- 현재 최신 브라우저는 정부 CSS2 규격을 준수
- CSS3 도 표준안이 완성되었고 대부분 브라우저에 지원 ( 완벽하게 지원 X )
- 발전 과정
  1. CSS1 : 1996년 W3C에 의해 공식 발표. 단순한 글꼴 정의, 텍스트 정렬, 마진값에 대한 정의 ( 넷스케이프 4, IE3, 4)
  2. CSS2 : 1998년 발표 거의 모든 브라우저에서 채택. 현재 사용하는 CSS에 모든 규격이 포함. 2006년 CSS2.1 발표로 여러 버그가 수정 되었고 현재 모든 브라우저가 지원하고 있다.
  3. CSS3 : 2005년 부터 개발중에 있으며 아직 W3C 권고안은 발표되지 않았으나 모바일 중심으로 빠르게 채택되어 이미 쓰여지고 있다.

### 3. CSS Basic

---

#### □ CSS 와 HTML의 상호 작용

- 초창기 CSS없는 HTML에서는 태그가 지정한 용도가 아닌 다른 용도(레이아웃 잡거나 표현을 위한 용도로 사용 -> HTML문서 자체가 비대해짐 -> 네트워크 부담, 관리 부담)
- W3C 웹 표준 권고 ( HTML + CSS 분리 )
- 웹 페이지 = HTML + CSS + JavaScript
- 유연한 CSS 와 HTML 상호작용  
HTML 뼈대 (구조) 변경없이 CSS만의 변경을 통해서 전체적인 디자인을 변경이 가능

[ 예시 사이트 ]

<http://www.csszengarden.com>

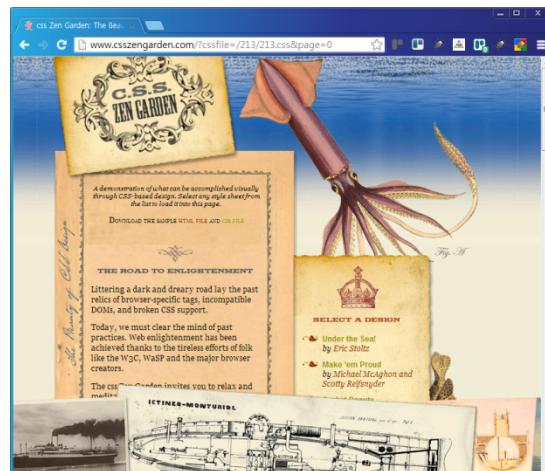
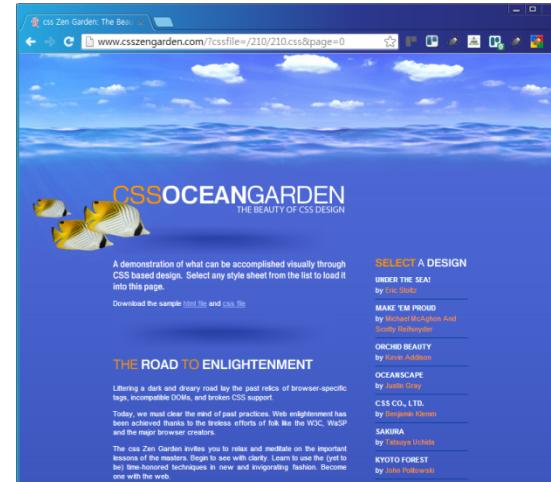
### 3. CSS Basic

## □ CSS 와 HTML의 상호 작용 ( cont'd )

<http://www.csszengarden.com>



<http://www.csszengarden.com/?cssfile=/210/210.css&page=0>



<http://www.csszengarden.com/?cssfile=/213/213.css&page=0>

### 3. CSS Basic

---

#### □ CSS 적용하기 ( 인라인 방식 )

- 절대 권장사항이 아님
- 바로 스타일 적용이 확인 되기 때문에 테스트 용도 또는 웹메일을 전송할 때만 사용해야하고 그 외에는 절대 사용하지 말 것
- 인라인 방식 예시

```
<p style="color:red">.....</p>
```

- 태그에 직접 스타일을 적용하기 때문에 바로 확인이 가능
- 웹 표준 이전 방식과 별 차이가 없음
- 많은 페이지가 있는 사이트는 수정자체가 불가능

### 3. CSS Basic

## □ CSS 적용하기 ( 인라인 방식 )

[실습예제 12]

ex12.html의 h1 엘리멘트에 다음 스타일을 인라인 방식으로 적용해 보세요.

font-size는 24px, font-family는 돋움, color 는 #06f



### 3. CSS Basic

---

#### □ CSS 적용하기 ( 임베디드 방식 )

- 태그에 직접 스타일을 지정하지 않고 <head>와 </head> 사이에 스타일을 지정하는 방식
- HTML 문서 내부에 따로 스타일을 지정
- 임베디드 방식 예시

```
<style type='text/css' media='screen'>
    p { color:#ddd }
</style>
```

- 스타일 형식을 지정 해주고 미디어 타입을 지정
- 스타일 형식 ‘text/css’ 는 고정
- media타입은 화면에 보여줄 때 screen, 프린트 출력에는 print 그리고 두 가지 모두에 적용 되는 공통 스타일이면 all 등이 올 수 있다.
- CSS 코드가 길어지면 관리하기가 힘들어 진다.

### 3. CSS Basic

---

#### □ CSS 적용하기 ( 임베디드 방식 )

[실습예제 13]

실습예제12의 인라인 방식을 임베디드 방식으로 바꾸어 ex13.html에 적용해 보세요.

[실습예제 14]

<h1>태그로 감싼 구문이 하나 더 있는 예제 ex14-1.html 과 ex14-2.html 를 브라우저에서 열어 보고 차이점을 생각해 보세요.

### 3. CSS Basic

#### □ CSS 적용하기 ( 외부파일 )

- 사이트 관리가 쉽기 때문에 권고하는 방식
- 확장자가 .css 파일에 CSS스타일을 만들어 외부 파일로 저장하고 HTML에 @import, link 를 통해 스타일을 불러 오는 방식
- 임베디드 방식과 마찬가지로 <head> 와 </head> 사이에 특정 CSS 파일을 불러오도록 지정한다.

##### 1) @import 방식

```
<style type='text/css' media='screen'>
    @import url(main.css)
</style>
```

##### 2) link 방식

```
<link href='main.css' rel='stylesheet'
      type='text/css' media='screen' />
```

- link방식을 선호

구버전 브라우저에서 @import를 인식 못함  
속도 측면에서 link방식이 조금 빠르다고 함.

### 3. CSS Basic

---

#### □ CSS 적용하기 ( 외부 파일 )

[실습예제 15]

ex15.html에 body 전체에 다음 스타일이 적용되게 해보세요.

padding 값은 TRBL 모두 0

margin 값은 TRBL 모두 10px

font-size는 12px

font-family는 Arial 또는 Helvetica, sans-serif

color는 #333

단, 스타일 정의는 main.css 파일에 별도로 하고 ex15.html에서 link방식으로 가져와야 합니다.

### 3. CSS Basic

#### □ CSS 적용하기 ( @import 활용 )

- 하나의 CSS파일 내부에서 다른 CSS 파일을 불러올 수 있다.



- main.css : 모든 문서들의 기본(공통)이 되는 속성을 지정
- company.css : 회사 소개 페이지인 company.html의 디자인 속성이 정의
- member.css : 회원가입등 회원 관련 페이지의 디자인 속성이 정의
- 각각의 css파일이 공통 속성의 main.css를 @import를 사용해서 임포트 하면 된다.

### 3. CSS Basic

---

#### □ CSS 적용하기 ( @import 활용 )

[실습예제 16]

ex16.html에 다음 스타일이 적용되도록 해보세요.

1.main.css에 정의 해놓은 모든 속성은 그대로 적용

2.body의 bg를 #ff6으로 합니다.

3.<h1>의 color만 #000로 변경합니다.

단, 새로운 스타일 정의는 ex16.css 파일에 별도로 하고 ex16.html에서 link방식으로 가져와야 합니다.

### 3. CSS Basic

---

#### □ 선택자 ( selector )

**선택자 { 속성: 속성값; }**

- 다양한 선택자는 CSS 핵심
- 선택자 : 기본적으로 태그, 클래스, 아이디 선택자 등이 있으며 그 외에 종속, 하위, 전체, 그룹 선택자 등이 웹 개발 실무에서 많이 사용

### 3. CSS Basic

---

#### □ 태그 선택자

- 말그대로 HTML 태그를 선택자로 사용하는 것
- [실습예제 17]

페이지내

모든 <h1> 태그에 font-size: 20px, color: #f00, margin:10px 적용

모든 <h2> 태그에 font-size: 14px, color: #36f, margin:5px 적용

모든 <p> 태그에 font-size: 12px, color: #333, margin:5px 적용

ex17.html를 작성하세요.

스타일링 방식은 임베디드로 하세요.

### 3. CSS Basic

---

#### □ 클래스 선택자

- 사용자가 직접 이름(class)을 만들어 속성을 지정.
- 클래스 선택자 정의

```
.CLASSNAME { 속성이름: 속성값; }
```

- .(점)으로 시작하고 첫문자는 영문자로 시작해야 하며, 대소문자 구별
- 태그 선택자가 페이지내 모든 태그에 적용되었다면 클래스 선택자는 클래스 속성에서 같은 클래스 이름을 가진 엘리멘트들이 적용 대상이 된다.

### 3. CSS Basic

---

#### □ 클래스 선택자

- [실습예제 18]

headline 이란 클래스 이름의 스타일 속성은 다음과 같습니다.

color: #333, border:1px solid #999, margin:25px, padding:10px

이 속성을 ex17.html 문서의 다음 문단에 적용해 보세요.

“윈도우 7은 전작인 윈도우 비스타의 불편함을 개선하고, 또한 사용자 편의를 위해서 이전 버전에 비해 많은 부분이 보강되었습니다.”

### 3. CSS Basic

---

#### □ 아이디 선택자

- 사용자가 직접 이름(id)을 만들어 속성을 지정.
- 아이디 선택자 정의

```
#ID { 속성이름: 속성값; }
```

- #(샵)으로 시작하고 첫문자는 영문자로 시작해야 하며, 대소문자 구별
- 기본적으로 HTML 엘리멘트의 id는 유일해야 하기 때문에 하나의 엘리멘트는 하나의 유일한 id를 가진다. 따라서 클래스 선택자와 차이는 페이지내 특정 하나의 엘리멘트에만 적용할 수 있다.

### 3. CSS Basic

---

#### □ 아이디 선택자

- [실습예제 19]

head 란 아이디 이름의 스타일 속성은 다음과 같습니다.

height: 50px, background-color:#3cf, padding:10px, border:1px solid #09f,  
color: #fff

이 속성을 ex17.html 문서의 <h1> 태그에 적용해 보세요.

### 3. CSS Basic

---

#### □ 종속 선택자

- 태그, 클래스, 아이디 선택자가 결합된 형태의 선택자
- 종속 선택자 예

```
h1#head { ... }
```

```
.headline.selected { ... }
```

```
input#user-id.focused { ... }
```

```
p.title { ... }
```

- 태그에 결합된 형태는 태그중에 특정 아이디, 특정 클래스에만 적용
- 클래스와 아이디에 모두 적용해서 스타일을 적용할 수 있지만, 너무 복잡한 조합은 피하는 것이 좋다.

### 3. CSS Basic

---

#### □ 종속 선택자

- [실습예제 20] ex20.html

txt1 이란 클래스 이름의 스타일 속성은 다음과 같습니다.

font-weight: normal; color: #F60;

1) txt1 클래스 속성을 다음 두 문장에 적용하세요.

“최근 아이폰이 스마트폰의…… 원도우 7입니다.” <p>…

“맥오에스텐은 현재스노우레오퍼드 …… 용량이 감소되었습니다.” <div>…

2) txt1 클래스가 적용된 태그중 <p> 태그에만 다음 스타일을 적용해 보세요.

color:#36F; font-weight: bold;

### 3. CSS Basic

---

#### □ 하위 선택자

- 선택자 내부의 자식 선택자에 속성을 지정하는 방식이다.

- 하위 선택자 예

body h1, body h2, body p { ... }

p .txt1 { ... }

-> ex20.html

.headline span

-> ex20.html

- [Quiz] ex21.html에서 다음은 어떤 엘리멘트의 속성을 지정 한 것일까?

ul.list1 li { ... }

ul.list2 li { ... }

p a { ... }

ul li ul a { ... }

### 3. CSS Basic

---

#### □ 하위 선택자

- 엘리멘트 개별로 클래스 또는 아이디를 주지 않아도 스타일을 적용할 수 있다.

[실습 예제 21]

ex21.html에서

- 1) <P> 태그 안의 <a> 태그에 다음 스타일을 적용하세요.

text-decoration:underline, font-weight: bold, color: #F60

- 2) 클래스 list1 안의 <a> 태그에 다음 스타일을 적용하세요.

font-weight: bold, color: #F00, text-decoration: underline

- 3) 클래스 list2 안의 <a> 태그에 다음 스타일을 적용하세요.

font-weight: bold, color: #39F, text-decoration: none;

### 3. CSS Basic

---

#### □ 그룹 선택자

- 각각의 선택자를 그룹으로 지어 속성을 부여하는 것
- 선택자들 간에 공통적인 속성이 있는 경우 일괄 적용으로 편리하게 사용
- 그룹 선택자 예

```
body h1, body h2, body p { ... }
```

```
.right_box, .left_box { ... }           -> ex21.html
```

### 3. CSS Basic

---

#### □ 그룹 선택자

[실습 예제 22]

ex22.html에서

<h1>, <h2>, 그리고 txt-box 클래스에 텍스트 아래에 밑줄을 긋는 스타일을 그룹 선택자를 사용해서 적용해 보세요.

### 3. CSS Basic

---

#### □ 스도 선택자 (Psuedo Selector)

- 선택자로 바로 사용되는 것이 아니고 선택자와 함께 사용되어 선택자를 보조 하는 역할
- 그 역할에 따라 몇가지가 정해져 있다.

:hover → 마우스의 커서가 올라가 있는 상태

:active → 마우스 커서를 클릭한 순간부터 놓기 직전까지 상태

:link → 링크를 클릭하지 않은 그냥 링크만 되어 있는 상태

:visited → 링크를 눌러서 방문한 후 상태

:before → 문장이 시작되기 전

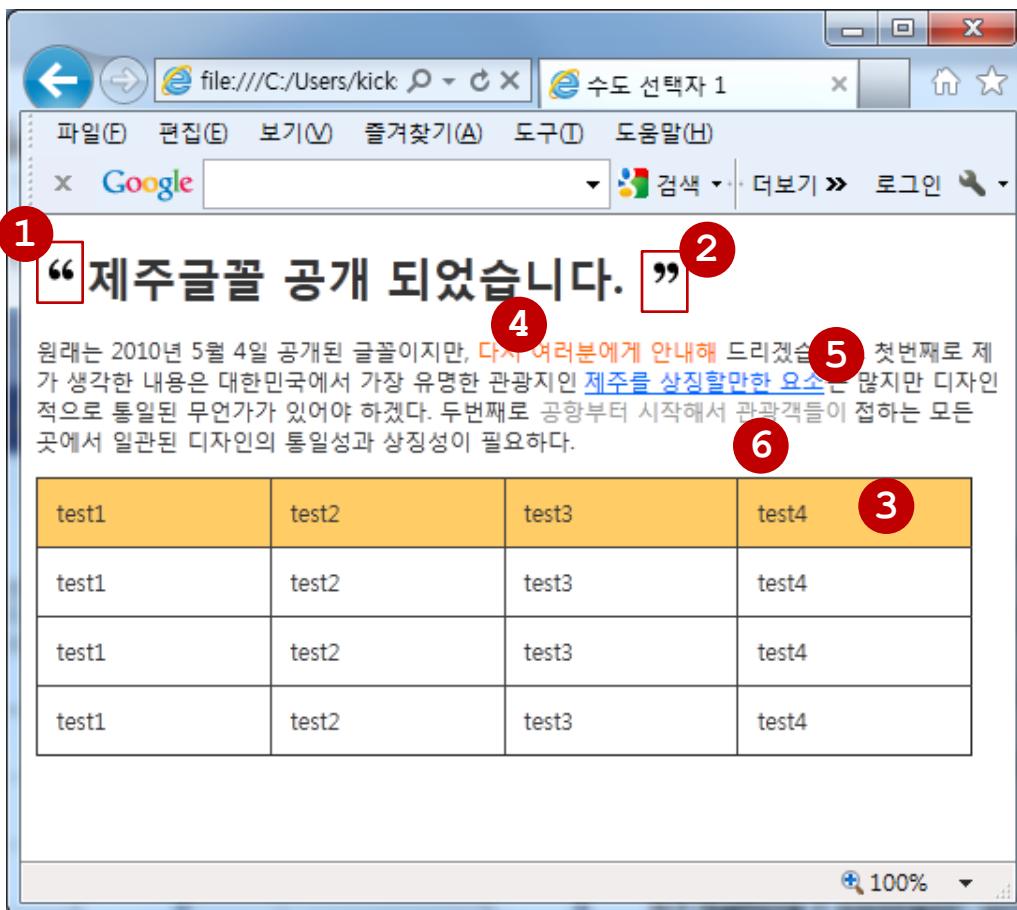
:after → 문장이 끝난 다음

- :hover, :active, :link, :visited 는 a 태그와 함께 링크를 데코레이션 할 때 많이 사용된다

### 3. CSS Basic

## □ 스도 선택자 (Psuedo Selector)

[예제23] ex23.html를 브라우저에서 열어 확인해 보세요



```
a:link { color: #F60; text-decoration: none; }  
a:hover { color: #06F; text-decoration: underline; }  
a:visited { color: #999; }  
a:active { color: #F00; text-decoration: line-through; }
```

```
h1:before { content: url(bull1.gif); }  
h1:after { content: url(bul2.gif); }  
table { border-collapse: collapse; width: 500px; }  
td { padding: 10px; }  
tr:hover { background-color: #FC6; }
```

### 3. CSS Basic

---

#### □ 전체 선택자

- 말그대로 전체 엘리멘트를 뜻한다. (와일드 카드)
- 스타일이 적용되는 모든 엘리멘트에 일괄 적용하고자 할 때 사용한다.

[예제] 브라우저 별로 모든 엘리멘트는 기본적인 margin값과 padding값을 가지고 있다  
그런데 브라우저별로 이 값이 틀리기 때문에 디폴트로 0로 만들고 시작하자

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

- 하위 선택자에 적용된 경우,

```
#idname * 또는 .classname *
```

아이디가 idname 인 엘리멘트 내부의 모든 자식 엘리멘트에 해당 속성들이  
적용될 것이다.

클래스 이름이 classname 인 엘리멘트들의 내부의 모든 자식 엘리멘트들에게  
해당 속성들이 적용될 것이다.

### **3. CSS Basic**

---

#### **□ CSS 주석**

- 다른 프로그래밍 언어와 마찬가지로 주석을 사용할 수 있다.
- /\* \*/ 만 사용할 수 있다.

# Part II

## HTML & CSS

### 2. css - 박스모델

1. 마진과 패딩
2. border
3. 백그라운드 이미지
4. float를 이용한 박스모델 정렬

## 1. 마진과 패딩

---

- 가장 자주쓰는 속성
- 마진은 컨텐츠의 테두리를 기준으로 외부공간 지정
- 패딩은 컨텐츠의 테두리를 기준으로 내부공간 지정
- 지정 방법

방법1. 4개의 방향을 각각 지정

**margin-top:10px;**

**margin-right:20px;**

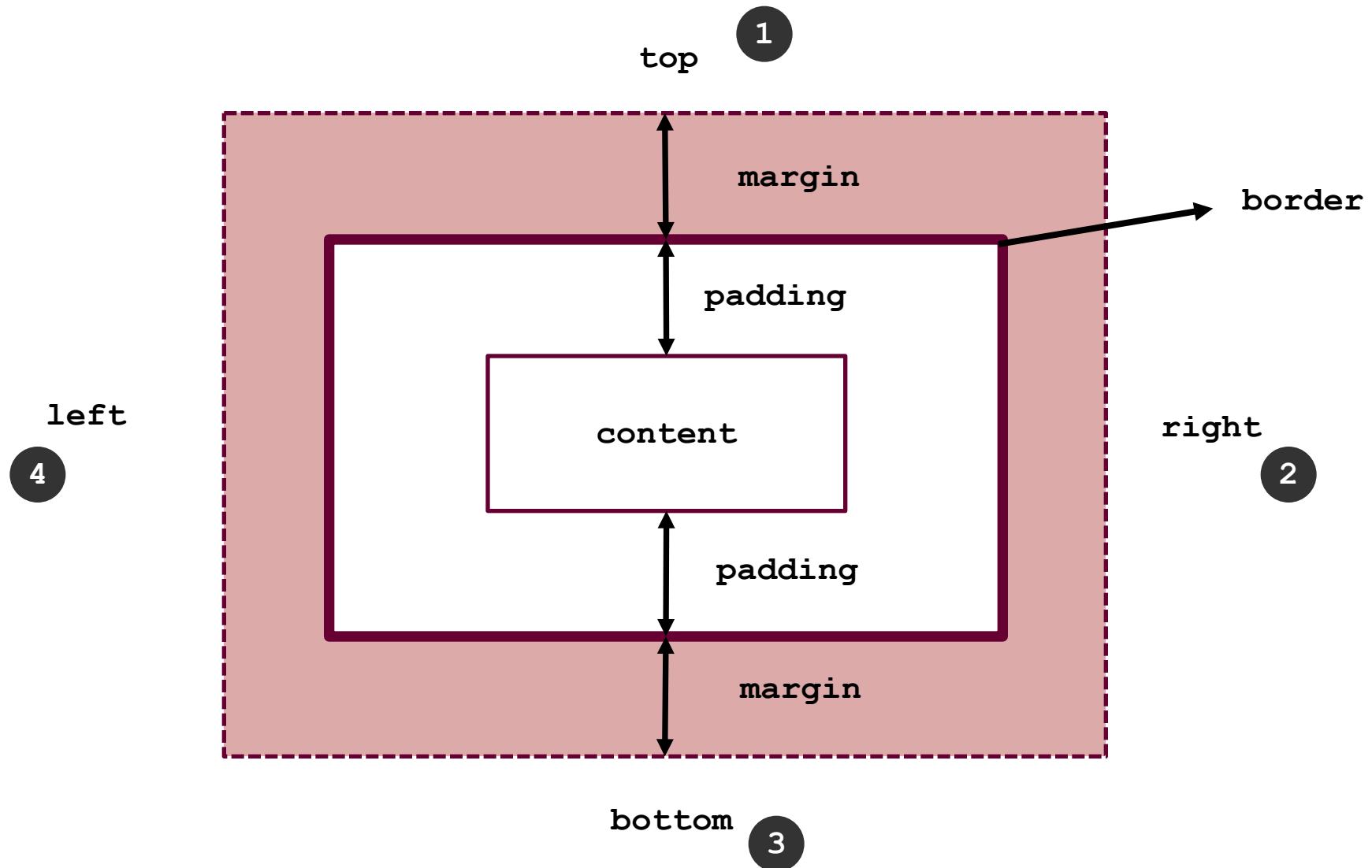
**margin-bottom:30px;**

**margin-left:40px;**

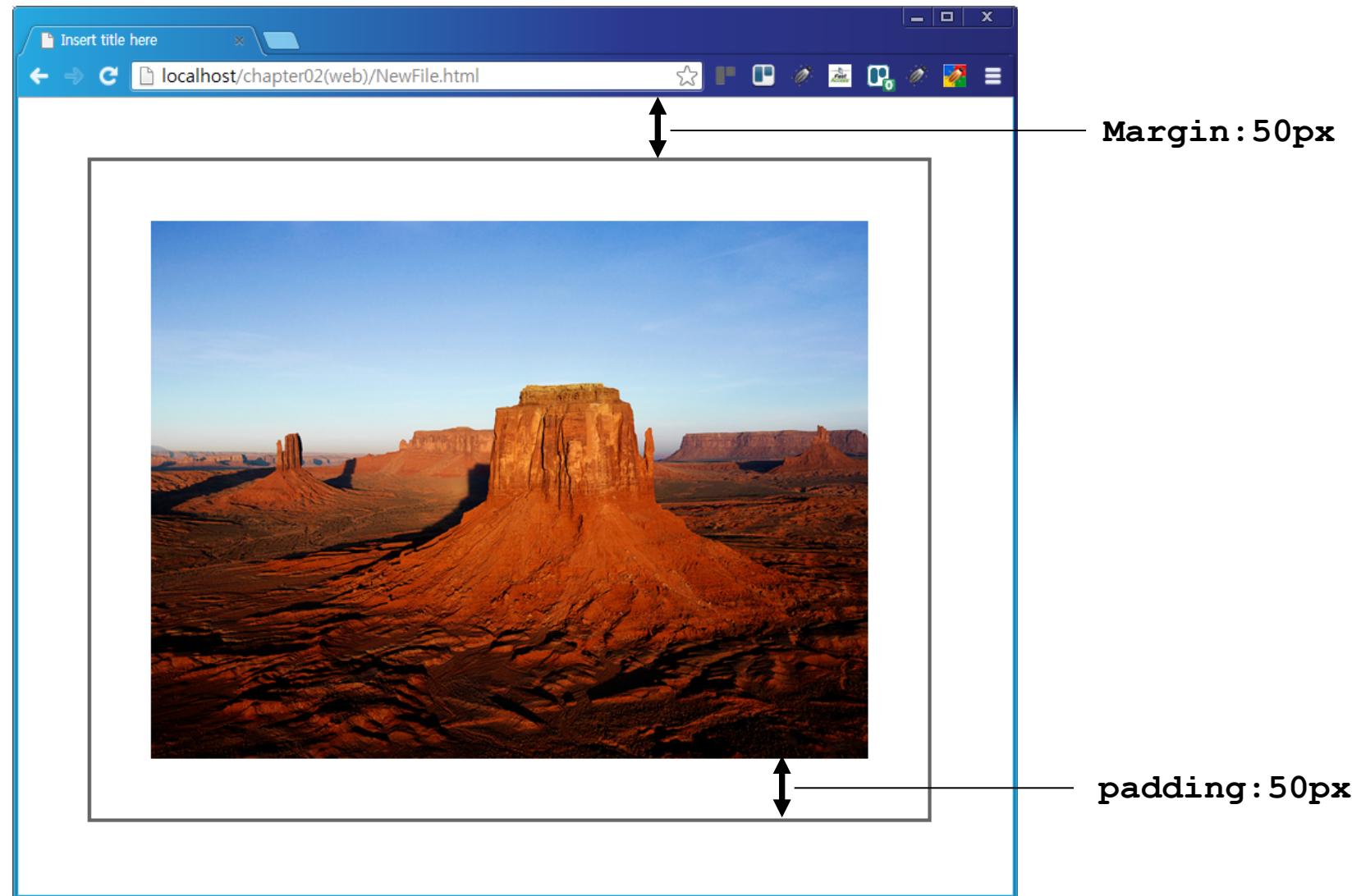
방법2. 각 방향으로 한꺼번에 지정하는 단축형

**margin: 10px 20px 30px 40px;**

# 1. 마진과 패딩



# 1. 마진과 패딩



## 1. 마진과 패딩

- 보통 모든 브라우저마다 마진 값이 기본적으로 지정되어 있다. 그리고 브라우저마다 그 값이 다르다. 그래서 초기화 해주는 것이 페이지 레이아웃 잡을 때 편리

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

- px(픽셀), %(퍼센트), em(엠) 이렇게 세가지 단위를 가장 많이 사용

- [예제 1]

- ex1.html에서 기본 마진 값을 초기화 하고 기본 마진 값을 확인 해 보세요.
- ex1.html에서 화면에 보이는 박스의 margin, padding 값을 각각 10px 작용해서 결과를 확인해 보세요.

- [예제 2]

- ex2.html 아래 박스에 margin, padding 값을 각각 20px로 지정하고 결과를 확인해 보세요.

## 1. 마진과 패딩

---

- **margin** 값은 음수가 될 수 있다.

**ex1.html**에서 **margin-top:-20px**를 적용해서 결과를 확인 해 보세요.

- **width** 와 **height** 지정

**width:** 속성값

**height:** 속성값

- **width** 와 **height**의 속성값은 %(퍼센트) 또는 px(픽셀 사용)

- 최대(최소) **width** 와 최대(최소) **height** 지정

**max-width:** 속성값

**min-width:** 속성값

**max-height:** 속성값

**min-height:** 속성값

## 1. 마진과 패딩

---

- 최대(최소) **width** 와 최대(최소) **height** 는 화면해상도에 맞는 페이지 작성할 때 유용
- [예제 3]

중앙의 박스에 다음 속성을 적용하고 브라우저 크기가 변할때 마다 어떤 변화가 있는지 확인해 보세요.

**max-width: 980px**

**min-width: 300px**

**max-height: 500px**

**min-height: 300px**

## 2. border

---

- 박스의 선을 긋는 속성으로 가장 많이 사용되는 속성 중 하나이다
- **border** 지정 방법

### 방법1. 일반형

border-width: 속성값 (두께지정)

border-style : 속성값 ( dashed, dotted, double, inset, outset, ridge, solid, none)

border-color : 색상지정;

### 방법2.

border: 두께 스타일 컬러;

### 방법3.

border-top: 두께 스타일 컬러;

border-right: 두께 스타일 컬러;

border-bottom: 두께 스타일 컬러;

border-left: 두께 스타일 컬러;

## 2. border

---

### □ 예제4.

**ex4.html**에서 박스 속성을 확인해보세요.

브라우저별로 각각 열어서 확인해 보고 차이점을 확인해 보세요.

### 3. 백그라운드 이미지

- 아름다운 페이지를 표현하는 제일 강력한 속성
- 백그라운드 이미지를 사용하면 스타일만 제거했을 때 불필요한 이미지들이 대부분 사람 짐 -> 프린트할 때 유용
- 표준 웹 페이지는 백그라운드 이미지로 이미지를 대부분 표현한다.
- 백그라운드 이미지 지정방법

방법1 :

**background:** 속성값

속성값에는

**color:** 색상을 지정,

**image:** 배경 이미지 지정 URL('.....')

**repeat:** 배경이미지의 반복 **repeat, no-repeat, repeat-x, repeat-y**

**position:** 백그라운드 이미지의 위치 x,y 축을 기준으로 픽셀 값으로 지정하거나  
**top, center, bottom, left, right** 를 복합적, 또는 하나만 사용

### 3. 백그라운드 이미지

---

#### □ 예제 6

ex6.html에 백그라운드 이미지 속성을 다음과 같이 적용하고 확인해 보세요.

**color: #022250**

**image: image/back\_image2.jpg**

**repeat: no-repeat;**

**position: bottom center**

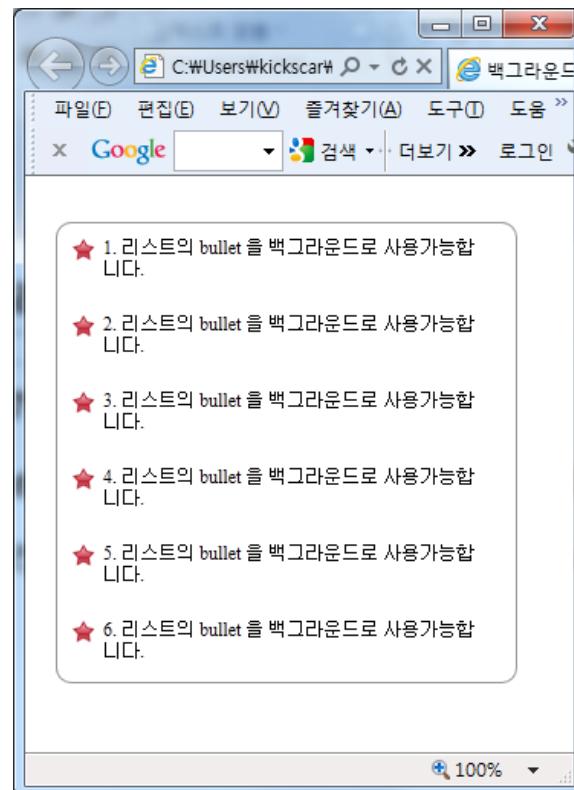
단축형으로 스타일을 지정해 보세요.

### 3. 백그라운드 이미지

#### □ 예제 7

ex7.html은 백그라운드 이미지 응용 예제중 리스트의 블릿에 적용된 예제 입니다.

백그라운드 이미지가 적용된 것을 확인해 보세요.

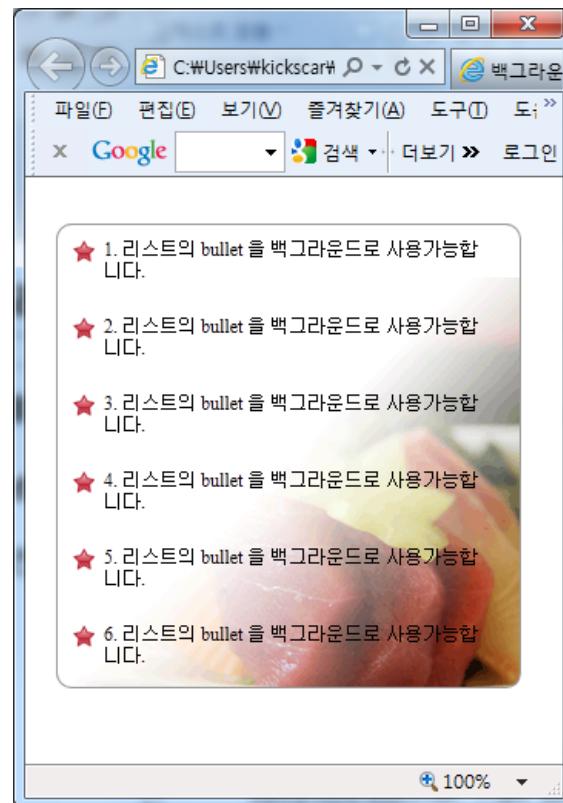


### 3. 백그라운드 이미지

#### □ 예제 8

예제7에서 리스트 전체에 백그라운드 이미지를 적용하는 실습예제입니다.

다음 결과 화면처럼 백그라운드 이미지를 적용해 보세요.



백그라운드 이미지 파일  
images/back2.gif

## 4. **float**을 이용한 박스모델 정렬

---

- 박스 모델의 위치를 잡을 때 사용
- 부유라는 의미로 박스 모델을 물에 띠워 오른쪽이나 왼쪽으로 보낸다는 의미
- 기본 사용법

**float:** 속성값

속성값에는

**right(오른쪽), left(왼쪽), none(없음)**

## 4. float을 이용한 박스모델 정렬

### □ 예제9

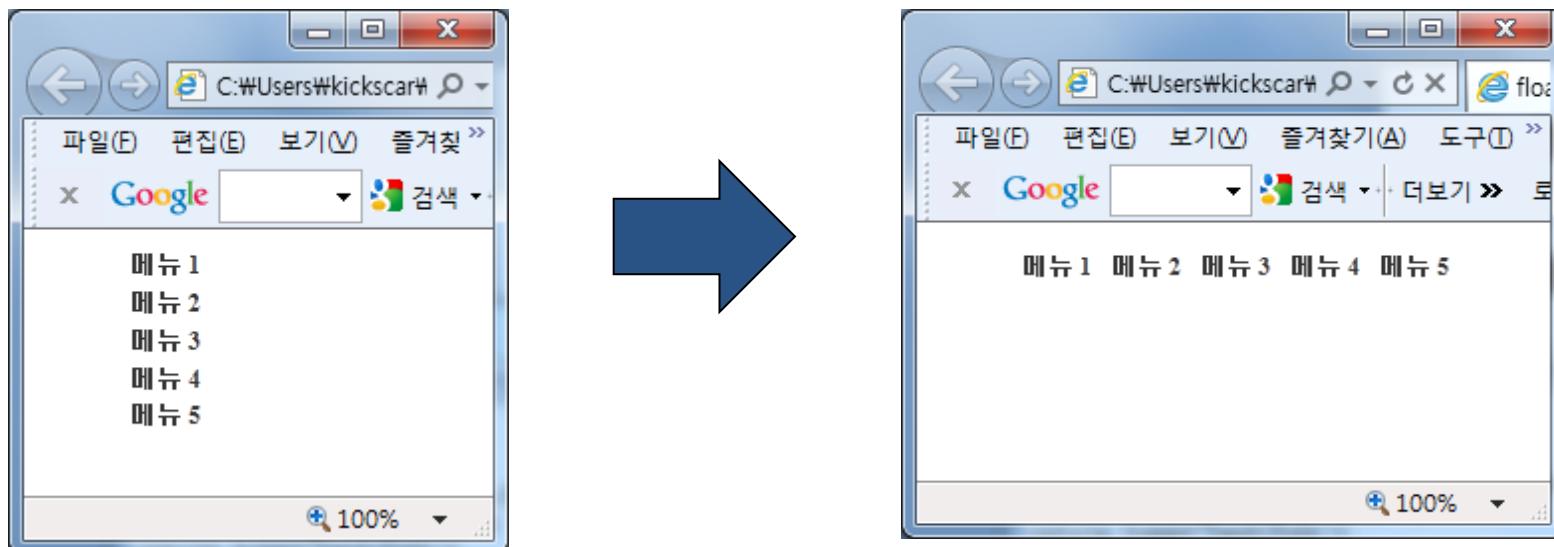
그림처럼 ex9.html의 클래스 이름 **illust** 가 지정된 엘리멘트는 현재 **floating** 되어 있지 않은 상태입니다. 왼쪽과 오른쪽으로 **float** 시켜서 결과를 각각 확인해 보세요.



## 4. float을 이용한 박스모델 정렬

### □ 예제10

float를 이용한 list의 수평 정렬 실습문제입니다. 왼쪽 그림과 같은 수직 리스트를 오른쪽 그림과 같은 수평 정렬로 바꿔 보세요.



## Part II

# HTML & CSS

### 3. css – 글꼴 및 텍스트의 이해

1. 글꼴의 이해
2. 텍스트 처리방법

## 1. 글꼴의 이해

---

- 기본 글꼴은 브라우저에서 세팅할 수 있음.
- 웹 페이지의 글꼴을 지정하면 OS에 설치된 글꼴이라면 지정한 글꼴로 웹페이지가 브라우저에 타나난다.
- 윈도우 환경 (굴림체, 돋움체, 바탕체, 궁서체)
  - + 맑은 고딕 (윈도우 비스타 이후 버전 또는 오피스 2007)
- 우리나라에서는 90% 이상이 윈도우 환경이라는 가정 하에 웹페이지에서 굴림 또는 돋움을 기본 글꼴로 지정하여 사용한다.
- 글꼴 이름 지정  
**font-family: “폰트 이름”;**
- 글꼴 이름은 여러 개 지정하여 넣을 수 있다.  
**font-family: “맑은 고딕”, 돋움;**

# 1. 글꼴의 이해

## □ 영문 글꼴 조합

**Verdana, Geneva, sans-serif**

**Georgia, "Times New Roman", Times, serif**

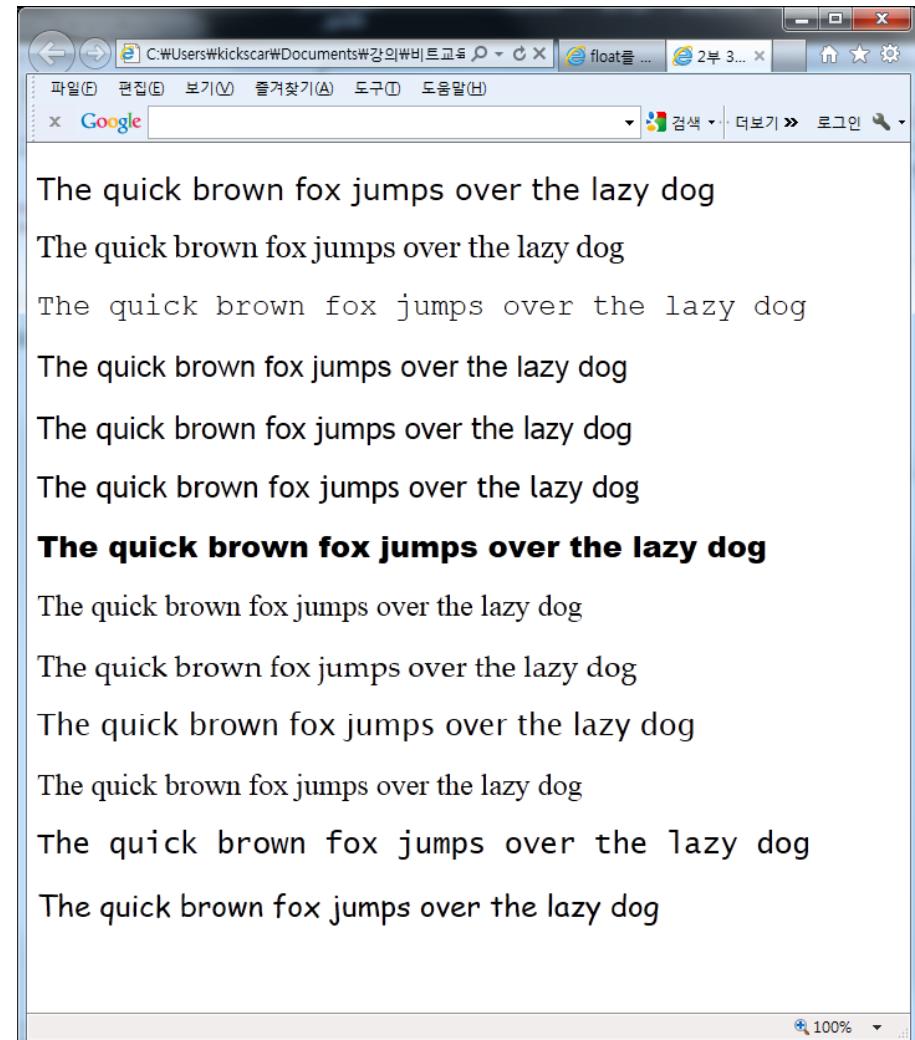
**"Courier New", Courier, monospace**

**Arial, Helvetica, sans-serif**

**Tahoma, Geneva, sans-serif**

**"Trebuchet MS", Arial, Helvetica, sans-serif**

등 13개 조합이 가능하다.



예제 1. 영문 글꼴 조합

# 1. 글꼴의 이해

---

## □ 글꼴 색상 지정

color: 색상

- 색상은 대부분 16진 수로 표시 : #ff0022 ( ff->Red, 00->Green, 22->Blue )
- 10진수 RGB로 표시 : rgb(158, 61, 74) 또는 RGB( 50%, 50%, 50% )
- 16진수 색상표 (웹 색상 표) : [http://ko.wikipedia.org/wiki/웹\\_색상](http://ko.wikipedia.org/wiki/웹_색상) 참고

## 1. 글꼴의 이해

### □ 글꼴 사이즈 지정

**font-size:** 사이즈값

- 글꼴 사이즈를 나타내는 단위

**px**(픽셀), **%**(백분율), **em**(엠), pica(피카), in(인치), cm(센치미터)  
mm(밀리미터)

- $1\text{px} = 72\text{dpi}$  이다. 보통 인쇄할 때는  $300\text{ dpi}$
- 모든 브라우저의 기본 글꼴의 크기는  $16\text{px}$ 를 기준으로 한다.  
따라서  $1\text{em} = 16\text{px}$

- 웹에서 가장 많이 사용하는 단위는 px(픽셀)이다.

## 1. 글꼴의 이해

---

### □ 글꼴 사이즈 지정 ( cont'd )

- **font-size: 12px** 과 **font-size : 75%** 는 같다.

기본 16px 의 75%는 12px 이다.

- % 보다는 em를 많이 쓴다.

- **font-size: 12px** 과 **font-size : 0.75em** 은 같다.

$$0.75 = 12 / 16$$

[quiz] font-size: 2em 은 몇 픽셀일까요?

- 우리나라에서는 웹 페이지에서 보통 12px를 기준으로 잡는다.

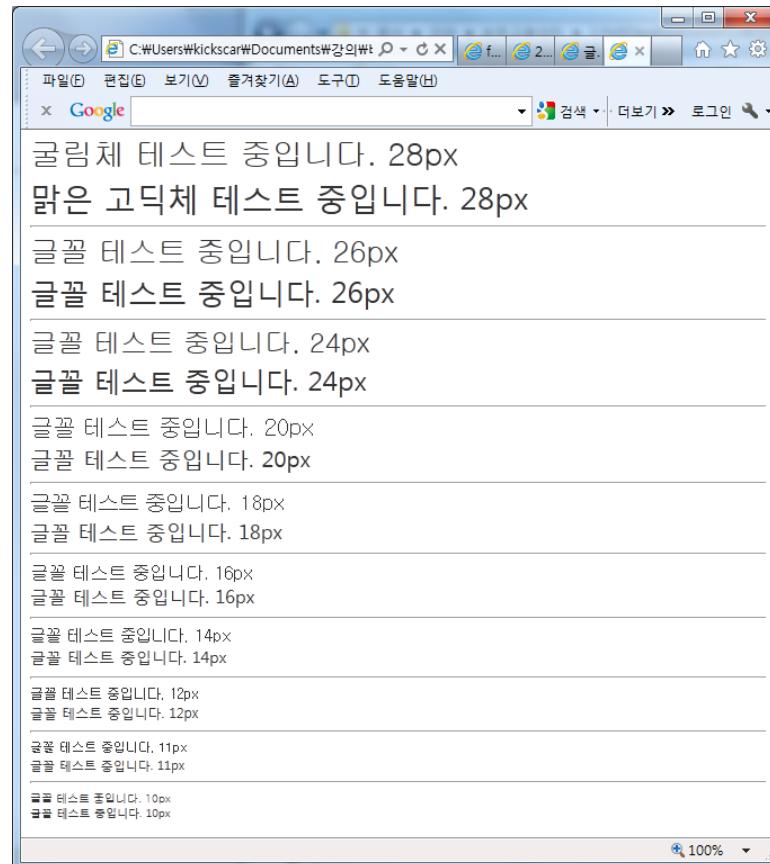
- 보통 10px이하 28px 이상은 사용하지 않는다.

# 1. 글꼴의 이해

## □ 예제 2

굴림체와 맑은 고딕체의 사이즈별 비교 예제입니다.

10px 이하와 28px 이상을 추가해보고 가독성을 판단해 보세요.



## 1. 글꼴의 이해

---

### □ px(픽셀) 보다 %, em 를 써야 하는 이유

px를 사용해서 강제적으로 글꼴의 크기를 정하는 경우, 브라우저에서 글꼴의 크기를 조정하더라도 글꼴이 변하지 않는다. ( 접근성에 좋지 못함 )

### □ 예제 3

ex3.html를 브라우저에서 열고 메뉴 > 보기 > 텍스트 크기에서 크기를 변화 시켰을 때 변화에서 강제적 글꼴 크기 지정시 문제점을 확인해 보세요.

## 2. 텍스트 처리방법

---

- 단락과 단락, 문장 내부 등 실제 웹 문서의 내용을 정리
- 글자와 글자 사이의 간격, 문장과 문장 사이의 간격 등 여러 속성을 정의해서 가독성이 높은 문서를 만든다.
- 문장과 문장 사이의 간격 (**line-height**)
  - 모든 브라우저의 기본적인 **line-height**의 값은 120%
  - 기본적인 사용법

**line-height:** 속성값 ( px, %, em )

- 가능하면 px로 설정하지 않도록 한다.

### □ 예제4

- 1) **line-height** 속성을 변경해가면서 브라우저 창에서의 글 간격을 확인해 보세요.
- 2) 브라우저에서 글꼴크기를 변경해 가면서 **line-height**를 고정으로 했을 때의 문제점을 확인해 보세요

## 2. 텍스트 처리방법

---

- 텍스트의 오른쪽, 왼쪽 등으로 정렬하기 위해 **text-align**를 사용

**text-align: 속성값( right, left, center)**

- 텍스트에 밑줄을 주거나 글자에 다른 느낌을 주기 위해 **text-decoration** 사용

**text-decoration: 속성값( blink, line-through, none, overline, underline )**

- 문자를 들여쓰기를 하고 싶을 때 **text-indent** 보통, 첫 번째 글자를 들여쓰기 할 때 사용

**text-indent: 속성값**

- 소문자 , 대문자 전환를 위해서 **text-transform** 를 사용

**text-transform: 속성값( capitalize, lowercase, uppercase )**

- 문장의 세로 정렬 지정

**vertical-align: 속성값( auto, baseline, central, top, middle, bottom)**

## 2. 텍스트 처리방법

---

- 단어와 단어 사이의 간격

**word-spacing:** 속성값

- 문자와 문자 사이의 간격

**letter-spacing:** 속성값

- 예제 5

ex5.html의 Style Sheet 내용을 변경시켜 텍스트 처리에 어떤 변화가 생겼는지

테스트하면서 텍스트 처리 관련된 속성들을 익혀 보세요.

# Part II

## HTML & CSS

### 4. css – 레이아웃설계

1. Table 태그의 정확한 사용
2. 리스트 형식 정의하기
3. 리스트를 이용한 메뉴 만들기
4. 폼 제작하기

# 1. **Table** 태그의 정확한 사용

## □ **table**의 가장 기본적인 코드

```
<table>
  <tr>
    <td>
      컨텐츠 부분
    </td>
  </tr>
</table>
```

## □ 헤더가 추가된 **table**의 기본적인 코드

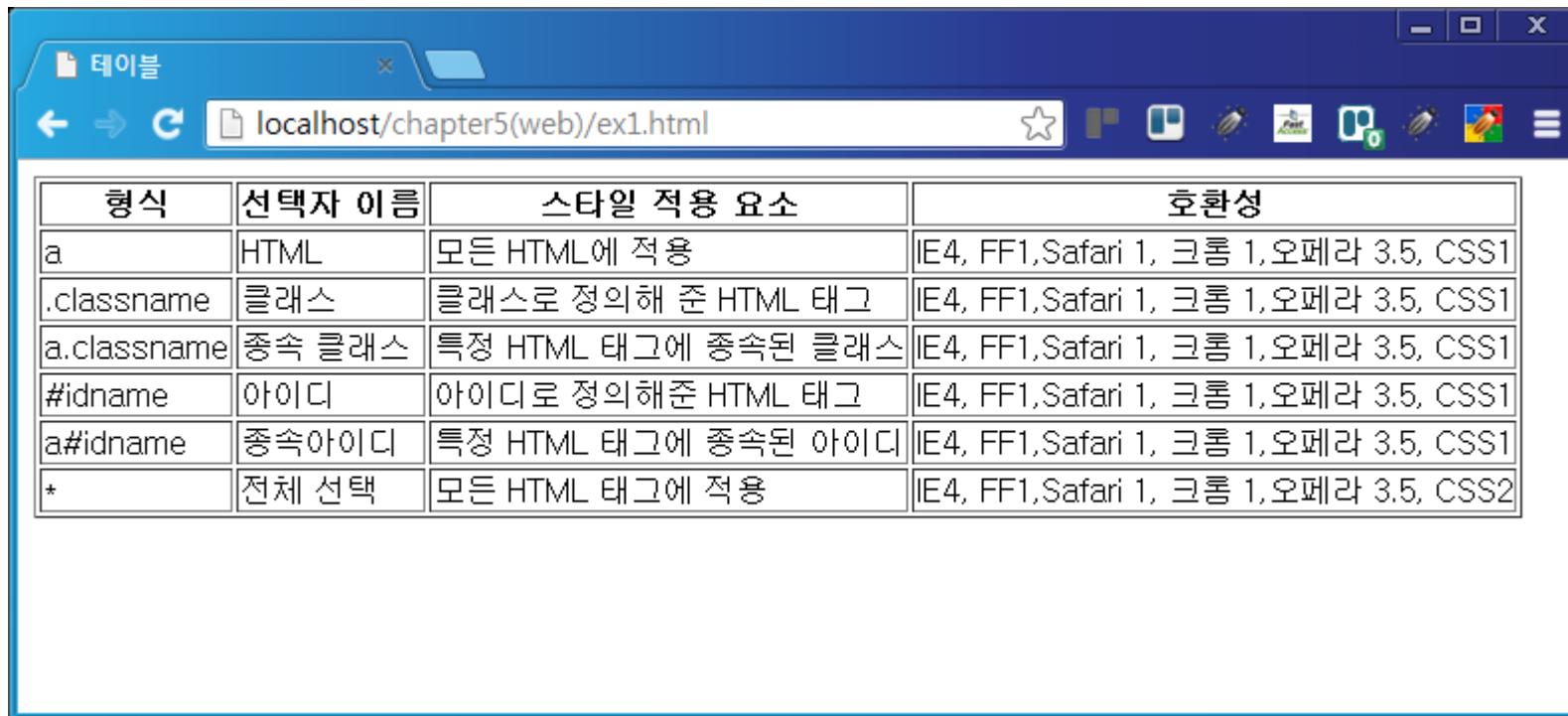
```
<table>
  <tr>
    <th>
      제목부분
    </th>
  </tr>
  <tr>
    <td>
      컨텐츠 부분
    </td>
  </tr>
</table>
```

tr : Table Row
th : Table Header
td : Table Data

# 1. Table 태그의 정확한 사용

## □ [ 실습문제 1 ] 기본적인 테이블 소스 ( practice1.html )

- 가장 기본적인 테이블의 형태입니다.
- 기본 코드에서 <th>, <td> 의 차이점은 무엇인가요?
- 다음 예제부터 단계별로 기본적인 HTML 코드는 그대로 두고 CSS만 변경하여 멋진 테이블로 변환합니다.



The screenshot shows a web browser window titled "테이블" (Table) displaying a table of CSS selectors. The table has four columns: "형식" (Format), "선택자 이름" (Selector Name), "스타일 적용 요소" (Style Application Element), and "호환성" (Compatibility). The data is as follows:

형식	선택자 이름	스타일 적용 요소	호환성
a	HTML	모든 HTML에 적용	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
.classname	클래스	클래스로 정의해 준 HTML 태그	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
a.classname	종속 클래스	특정 HTML 태그에 종속된 클래스	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
#idname	아이디	아이디로 정의해준 HTML 태그	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
a#idname	종속아이디	특정 HTML 태그에 종속된 아이디	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
*	전체 선택	모든 HTML 태그에 적용	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS2

# 1. **Table** 태그의 정확한 사용

## □ [ 실습문제1 – 단계1 ]

1. 기본 마진값과 패딩값을 없앱니다.
2. body에 다음 속성을 적용해서 하위 엘리멘트 모두가 그 속성을 상속 받도록 해보세요.  
기본글꼴 : “맑은 고딕”, “돋움”  
기본글꼴 사이즈 : 0.75em( 12px )  
텍스트 색상 : #333
3. 테이블에 class 이름 tbl-ex를 줍니다.
4. 그리고 테이블에 top margin 10px 과 중앙에 오도록 right-left 마진을 자동으로 적용합니다.

The screenshot shows a web browser window with a blue header bar. The title bar says "테이블" and the address bar shows "localhost/chapter5(web)/ex1.html". Below the address bar is a toolbar with various icons. The main content area displays a table with the following data:

형식	선택자 이름	스타일 적용 요소	호환성
a	HTML	모든 HTML에 적용	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
.classname	클래스	클래스로 정의해 준 HTML 태그	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
a.classname	종속 클래스	특정 HTML 태그에 종속된 클래스	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
#idname	아이디	아이디로 정의해준 HTML 태그	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
a#idname	종속아이디	특정 HTML 태그에 종속된 아이디	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
*	전체 선택	모든 HTML 태그에 적용	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS2

# 1. Table 태그의 정확한 사용

## □ [ 실습문제1 – 단계2 ]

table 태그에 보면 속성으로 border=1 준 것을 이제는 CSS로 옮겨야 합니다.

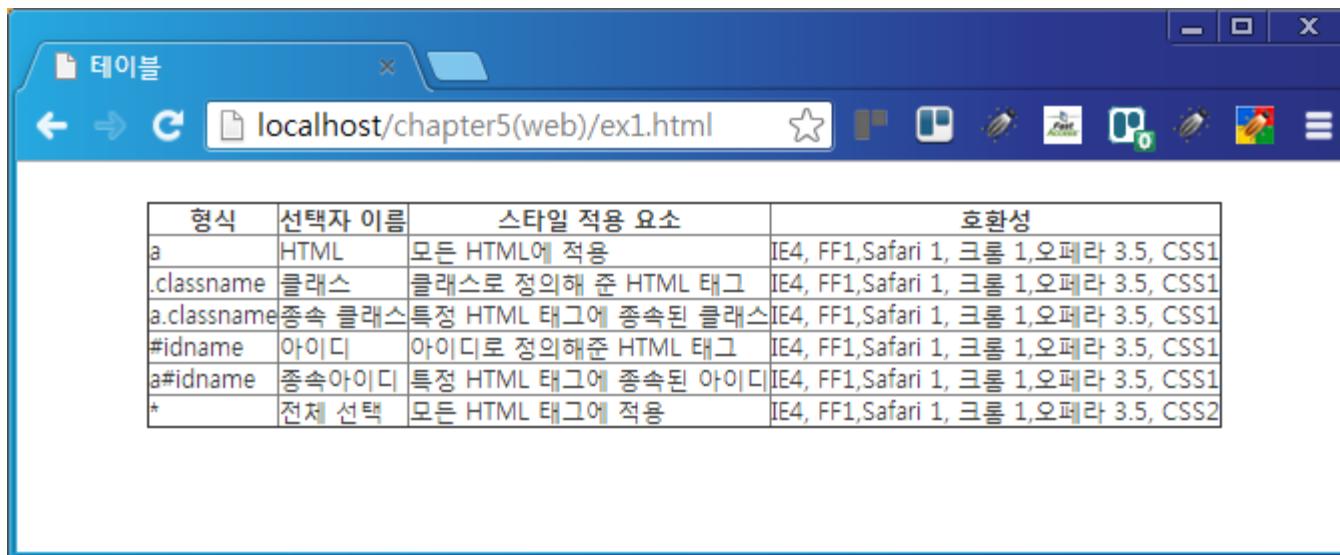
- 1) table 태그안의 속성 border 를 없애고 클래스 tbl-ex에 solid 타입의 #333 색상의 solid 타입의 border를 그어 보세요.
- 2) 1)번 처럼 하면 table 외각에만 선이 그어지고 내부 cell에는 선이 그어지지 않습니다. th, td에 같은 속성값의 border를 적용해 보세요. 두 종류의 태그 엘리멘트 들에 같은 속성을 적용하므로 그룹 선택자를 적용해 보세요.
- 3) 최종 결과가 단계1 와 같은가요?



# 1. Table 태그의 정확한 사용

## □ [ 실습문제1 – 단계3 ]

테이블의 스타일 속성중 중요한 속성이 border-collapse입니다. border-collapse 속성값이 collapse이면 셀 간격이 없어집니다. 마치, table 속성의 cellspacing = 0 와 같습니다. 확인해 보세요.



The screenshot shows a web browser window titled "테이블" (Table) displaying a table of CSS selectors and their properties. The table has four columns: 형식 (Type), 선택자 이름 (Selector Name), 스타일 적용 요소 (Style Application Element), and 호환성 (Compatibility). The browser's address bar shows "localhost/chapter5(web)/ex1.html".

형식	선택자 이름	스타일 적용 요소	호환성
a	HTML	모든 HTML에 적용	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
.classname	클래스	클래스로 정의해 준 HTML 태그	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
a.classname	종속 클래스	특정 HTML 태그에 종속된 클래스	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
#idname	아이디	아이디로 정의해준 HTML 태그	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
a#idname	종속아이디	특정 HTML 태그에 종속된 아이디	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
*	전체 선택	모든 HTML 태그에 적용	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS2

# 1. Table 태그의 정확한 사용

## □ [ 실습문제1 – 단계4]

th 와 td 안의 콘텐츠가 너무 꽉 차 있습니다. 내부 콘텐츠의 간격을 벌리기 위해 th 와 td의 padding 값을 TRBL 모두 8px 정도 줍니다.

th 와 td에 공통적인 속성값 설정이므로 그룹 선택자로 속성값을 부여 해보고 결과를 확인해 보세요.

형식	선택자 이름	스타일 적용 요소	호환성
a	HTML	모든 HTML에 적용	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
.classname	클래스	클래스로 정의해 준 HTML 태그	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
a.classname	종속 클래스	특정 HTML 태그에 종속된 클래스	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
#idname	아이디	아이디로 정의해준 HTML 태그	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
a#idname	종속아이디	특정 HTML 태그에 종속된 아이디	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
*	전체 선택	모든 HTML 태그에 적용	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS2

# 1. Table 태그의 정확한 사용

## □ [ 실습문제1 – 단계5 ]

테이블에 헤더 부분의 타이틀을 꾸며 볼려고 합니다.

- 1) th에 백그라운드 색을 #999로 설정해 보세요.
- 2) th의 폰트 크기를 1.1em( 17.6px )로 설정해 보세요.
- 3) th의 텍스트 색상을 #fff(white)로 설정해 보세요.

형식	선택자 이름	스타일 적용 요소	호환성
a	HTML	모든 HTML에 적용	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
.classname	클래스	클래스로 정의해 준 HTML 태그	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
a.classname	종속 클래스	특정 HTML 태그에 종속된 클래스	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
#idname	아이디	아이디로 정의해준 HTML 태그	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
a#idname	종속아이디	특정 HTML 태그에 종속된 아이디	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
*	전체 선택	모든 HTML 태그에 적용	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS2

# 1. Table 태그의 정확한 사용

## □ [ 실습문제1 – 단계6 ]

테이블 모양이 어느정도 갖추어 졌지만, 게시판용으로 쓰기에는 아직 미흡합니다.

일반적으로 게시판 테이블은 세로라인이 없습니다. 다음과 같이 적용합니다.

- 1) table의 왼쪽, 오른쪽 border를 0으로 설정해 보세요.
- 2) th, td의 왼쪽, 오른쪽 border를 0으로 설정해 보세요.

형식	선택자 이름	스타일 적용 요소	호환성
a	HTML	모든 HTML에 적용	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
.classname	클래스	클래스로 정의해 준 HTML 태그	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
a.classname	종속 클래스	특정 HTML 태그에 종속된 클래스	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
#idname	아이디	아이디로 정의해준 HTML 태그	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
a#idname	종속아이디	특정 HTML 태그에 종속된 아이디	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
*	전체 선택	모든 HTML 태그에 적용	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS2

# 1. Table 태그의 정확한 사용

## □ [ 실습문제1 – 단계7 ]

테이블 헤드의 border를 더 굵게 표시해서 타이틀을 더 강조하고 싶습니다.

th의 top, bottom의 border 굵기를 2px로 더 굵게 만들어 보세요.



The screenshot shows a web browser window with a title bar "4장 테이블" and a URL bar "localhost/chapter5(web)/ex1.html". The main content area displays a table with six rows, each representing a CSS selector and its properties. The table has four columns: "형식" (Format), "선택자 이름" (Selector Name), "스타일 적용 요소" (Style Applied Element), and "호환성" (Compatibility). The rows are separated by horizontal lines.

형식	선택자 이름	스타일 적용 요소	호환성
a	HTML	모든 HTML에 적용	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
.classname	클래스	클래스로 정의해 준 HTML 태그	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
a.classname	종속 클래스	특정 HTML 태그에 종속된 클래스	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
#idname	아이디	아이디로 정의해준 HTML 태그	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
a#idname	종속아이디	특정 HTML 태그에 종속된 아이디	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
*	전체 선택	모든 HTML 태그에 적용	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS2

# 1. Table 태그의 정확한 사용

## □ [ 실습문제1 – 단계8 ]

타이틀을 제외한 게시물 리스트에서 게시물 간의 구분은 점선으로 하는 것이 더 보기 좋을 듯 싶습니다.  
td의 border 스타일을 dotted로 적용해보세요.  
( 문제가 발생하면, th와 td의 border를 따로 적용해 보세요 )

아래 그림처럼 나오면 게시판이 완성 되었습니다.



The screenshot shows a Microsoft Internet Explorer browser window. The title bar says "4장 테이블". The address bar shows "localhost/chapter5(web)/ex1.html". The content area displays a table with a dotted border between rows. The table has four columns: 형식 (selector), 선택자 이름 (name), 스타일 적용 요소 (style), and 호환성 (compatibility). The rows are:

형식	선택자 이름	스타일 적용 요소	호환성
a	HTML	모든 HTML에 적용	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
.classname	클래스	클래스로 정의해 준 HTML 태그	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
a.classname	종속 클래스	특정 HTML 태그에 종속된 클래스	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
#idname	아이디	아이디로 정의해준 HTML 태그	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
a#idname	종속아이디	특정 HTML 태그에 종속된 아이디	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS1
*	전체 선택	모든 HTML 태그에 적용	IE4, FF1, Safari 1, 크롬 1, 오페라 3.5, CSS2

# 1. Table 태그의 정확한 사용

## □ [ 실습과제 1 ]

완성 된 게시판 리스트에 다음 스타일을 적용해 보세요.

<tr> 태그에 (게시물에) 마우스가 올라가면 백그라운드 색상이 #fc6 되게 스타일을 추가해 보세요.

또 커서의 이미지가 손 모양이 되도록 스타일을 추가해 보세요.

( 힌트 : 수도 선택자 :hover를 사용하면 됩니다. )



The screenshot shows a web browser window with a title bar "4장 테이블" and a tab "localhost/chapter5(web)/ex1.html". The main content area displays a table comparing different CSS selector types:

형식	선택자 이름	스타일 적용 요소	호환성
a	HTML	모든 HTML에 적용	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
.classname	클래스	클래스로 정의해 준 HTML 태그	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
a.classname	종속 클래스	특정 HTML 태그에 종속된 클래스	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
#idname	아이디	아이디로 정의해준 HTML 태그	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
a#idname	종속아이디	특정 HTML 태그에 종속된 아이디	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
*	전체 선택	모든 HTML 태그에 적용	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS2

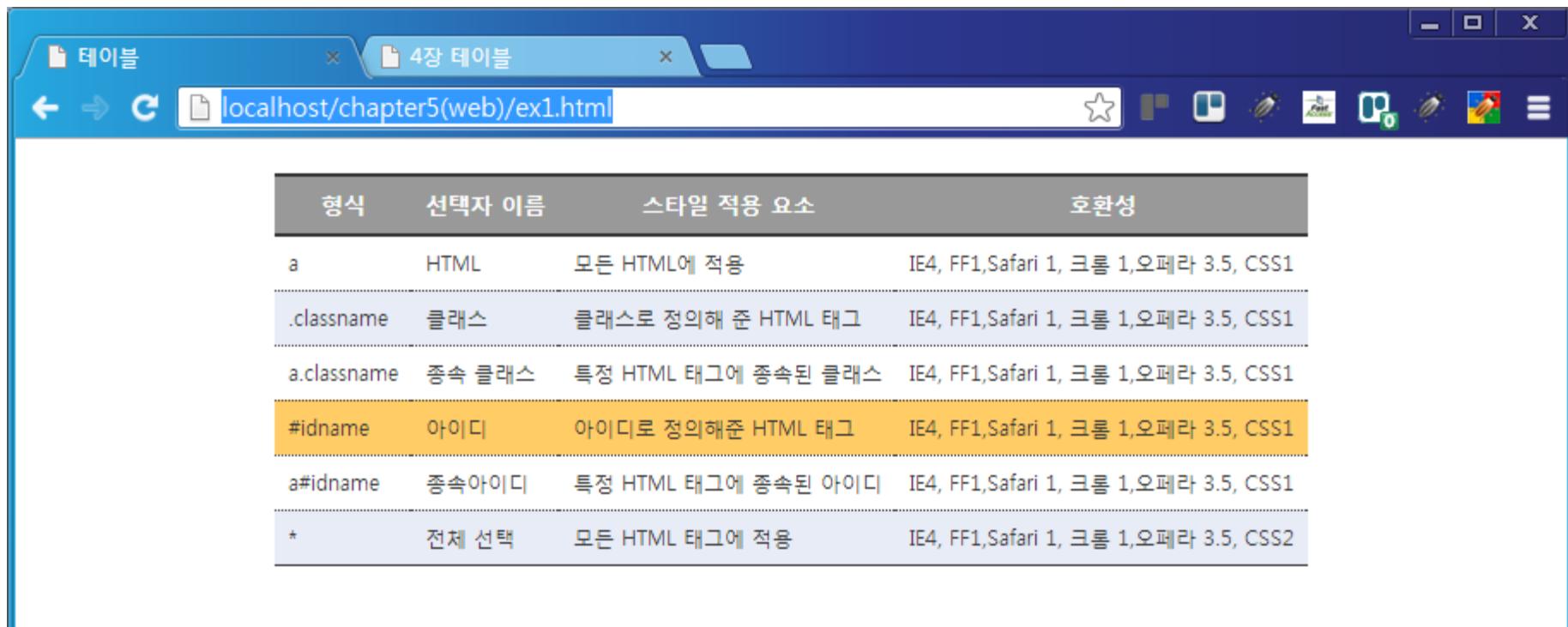
# 1. Table 태그의 정확한 사용

## □ [ 실습과제 2 ]

그림과 같이 짹수번째 <tr>에는 기본 백그라운드 색상을 #E8ECF6 로 적용하고 싶습니다. 그림을 참고 해서 적용해 보세요.

( 힌트 : 짹수 번째 <tr>에 같은 클래스 이름을 주고 그 클래스에 공통으로 백그라운드 색상 속성을 주면 됩니다.)

실습과제1번과 충돌이 나면 해결해야 합니다. 백그라운드가 적용된 짹수번째 <tr>도 실습문제 1번이 요구한대로 적용되어야 합니다.



The screenshot shows a web browser window titled "4장 테이블" with the URL "localhost/chapter5(web)/ex1.html". The page displays a table of CSS selectors:

형식	선택자 이름	스타일 적용 요소	호환성
a	HTML	모든 HTML에 적용	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
.classname	클래스	클래스로 정의해 준 HTML 태그	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
a.classname	종속 클래스	특정 HTML 태그에 종속된 클래스	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
#idname	아이디	아이디로 정의해준 HTML 태그	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
a#idname	종속아이디	특정 HTML 태그에 종속된 아이디	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS1
*	전체 선택	모든 HTML 태그에 적용	IE4, FF1,Safari 1, 크롬 1,오페라 3.5, CSS2

## 2. 리스트 형식 적용하기

- List 형식은 목록 나타낼 때 사용
- 웹 페이지에서는 목록뿐만 메뉴를 만들 때도 사용

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

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

- <ul> 은 ● 과 같은 형식으로 순서가 없는 목록을 만들 때 사용
- <ol> 은 1, 2, 3과 같은, 또는 A, B, C 또는 a, b, c 그리고 로마자 I, II, III 등으로 시작 할 때 사용

## 2. 리스트 형식 적용하기

---

### □ [예제 1] ex1.html

1) <ul>에 **list-style-type** 속성에 다음 값들을 각각 적용해 보세요.

**square, disc, circle**

2) <ol>에 **list-style-type** 속성에 다음 값들을 각각 적용해 보세요.

**decimal, lower-alpha, lower-roman, upper-alpha, upper-roman**

3) <ul>에 적용했던 **list-style-type** 값을 <ol>에 적용해보세요.

반대로 <ol>에 적용했던 **list-style-type** 값을 <ul>에 적용해보세요.

결과는 ?

## 2. 리스트 형식 적용하기

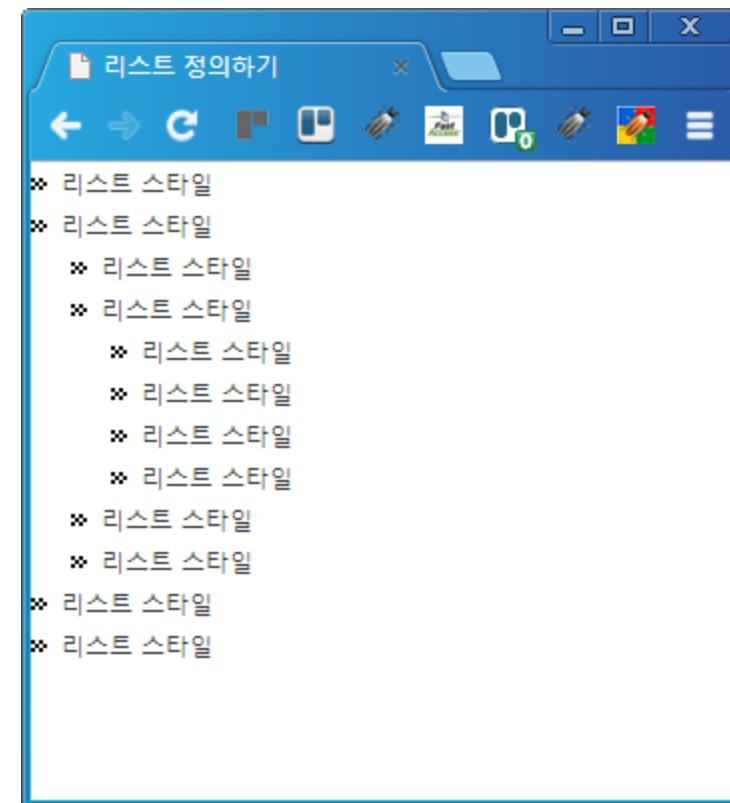
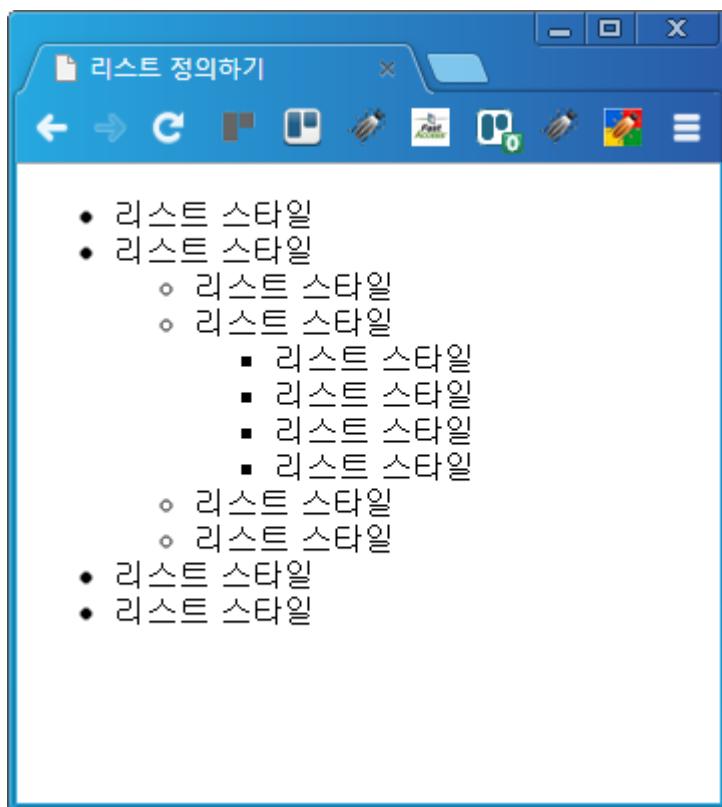
---

- <ul>, <ol>은 CSS를 적용해 스타일 타입을 결정해 주면 구분 자체가 무의미
- 보통은 <ul> 를 많이 선호하고 사용하고 있음

## 2. 리스트 형식 적용하기

### □ [ 실습문제 2 ] 기본적인 3depth 리스트 ( practice2.html )

– background-image로 기본 리스트 type를 대체하는 실습문제입니다.



## 2. 리스트 형식 적용하기

### □ [ 실습문제 2 – 단계1 ]

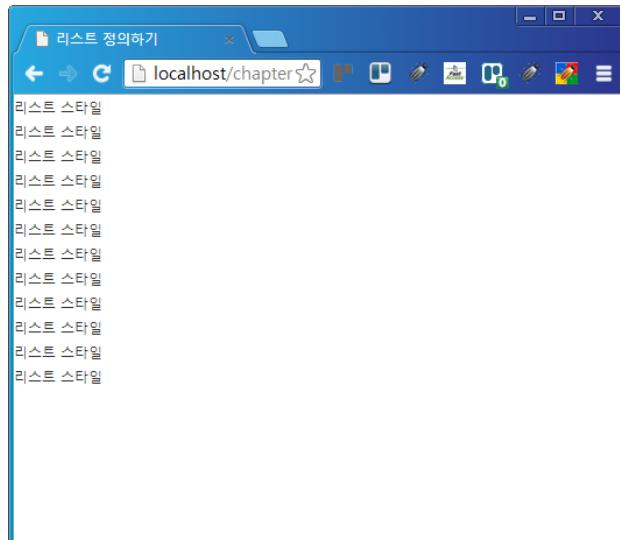
1. 기본 마진값과 패딩값을 없앱니다.
2. body에 다음 속성을 적용해서 하위 엘리멘트 모두가 그 속성을 상속 받도록 해보세요.

기본글꼴 : “맑은 고딕”, “돋움”

기본글꼴 사이즈 : 0.75em( 12px )

텍스트 색상 : #333.

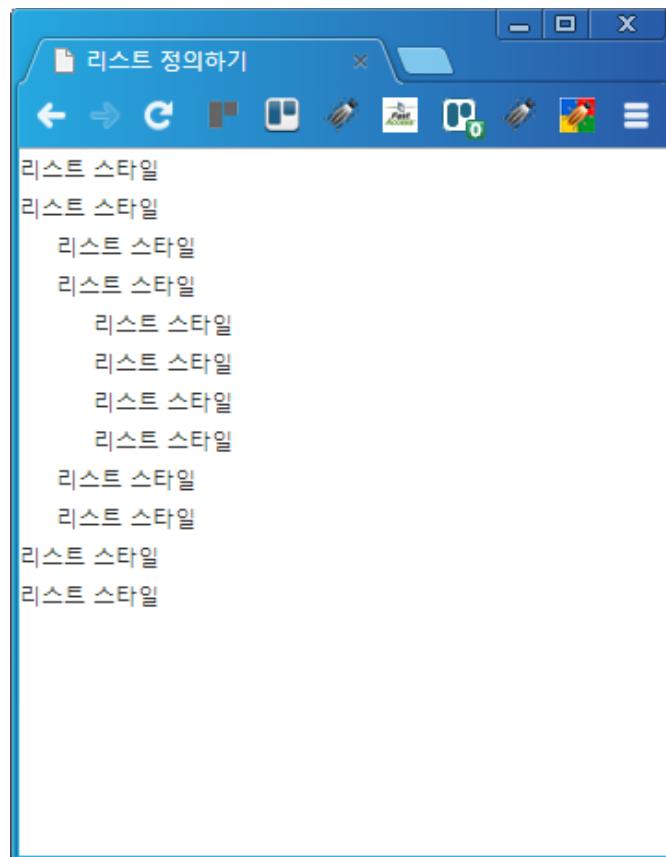
3. 리스트 li의 간격을 조정하기 위해 line-height 값을 1.8em ( 28.8px) 줍니다.



## 2. 리스트 형식 적용하기

### □ [ 실습문제 2 – 단계2 ]

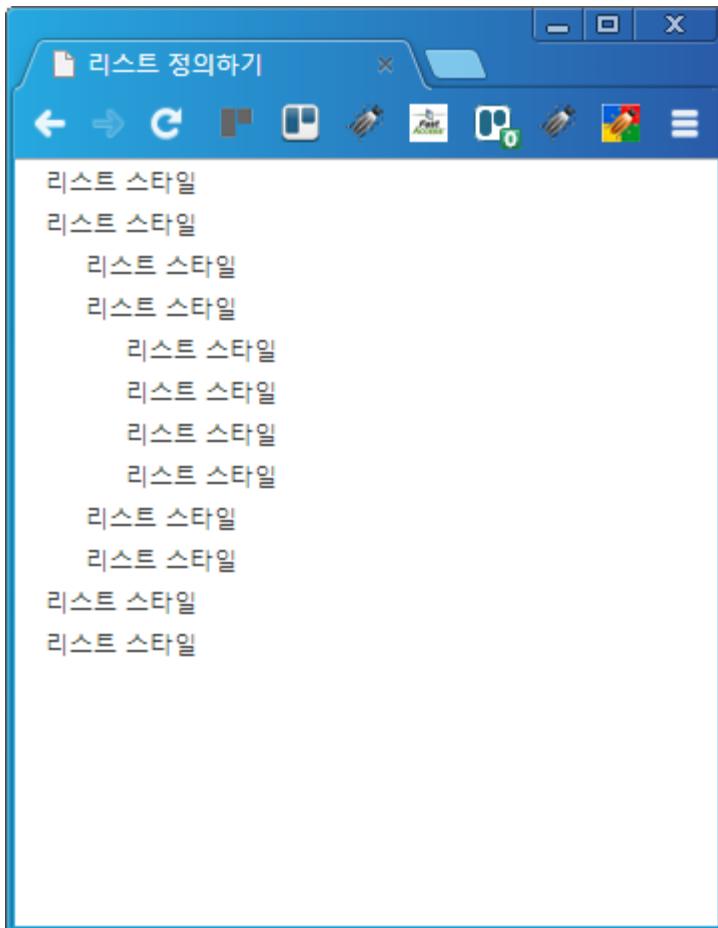
- 모든 서브 <ul> 태그(엘리먼트)에 left margin값 20px을 주어 자연스럽게 depth가 생기도록 합니다.
- <ul>에 list-style-type을 주어 기본 블릿이 생기는 것을 막습니다.



## 2. 리스트 형식 적용하기

### □ [ 실습문제 2 – 단계3 ]

list item <li>에 이미지를 앞에 넣기 위해 left padding 값을 15px 줍니다.



## 2. 리스트 형식 적용하기

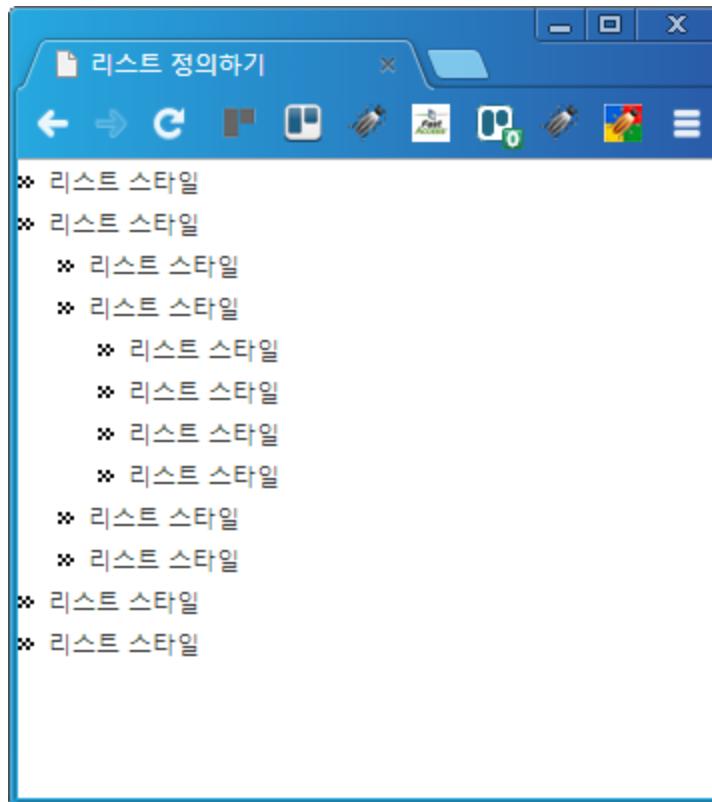
### □ [ 실습문제 2 – 단계4 ]

<li>에 블릿 이미지를 다음과 같이 설정합니다.

image url : images/*bullet11.jpg*

repeat: no-repeat

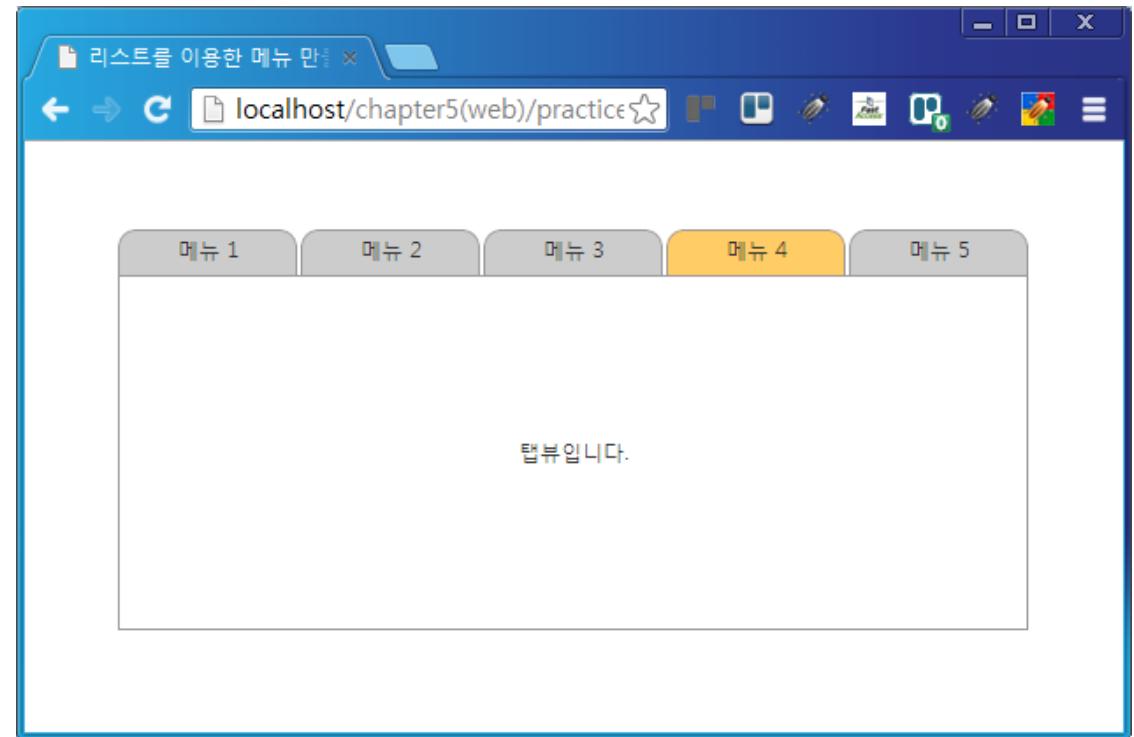
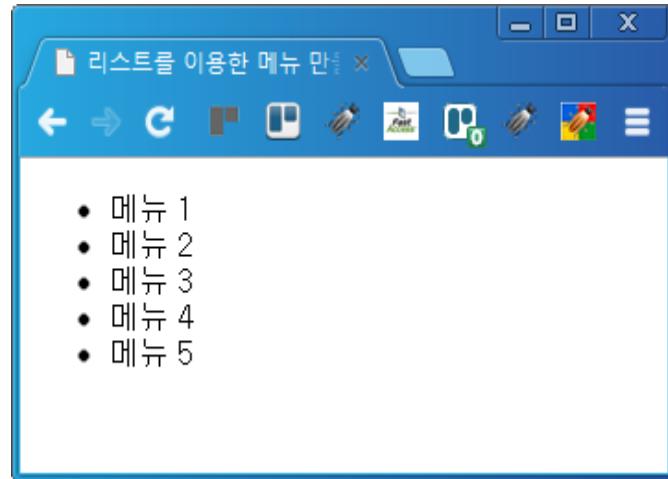
position: 0 8px



### 3. 리스트를 이용한 메뉴 만들기

#### □ [ 실습문제 3] 기본적인 리스트 ( practice3.html )

- 리스트는 메뉴를 만들 때도 사용합니다.
- 단계별로 CSS설정하여 그림과 같이 기본적인 리스트 메뉴에서 탭메뉴로 변환합니다.



### 3. 리스트를 이용한 메뉴 만들기

#### □ [ 실습문제 3 – 단계1 ]

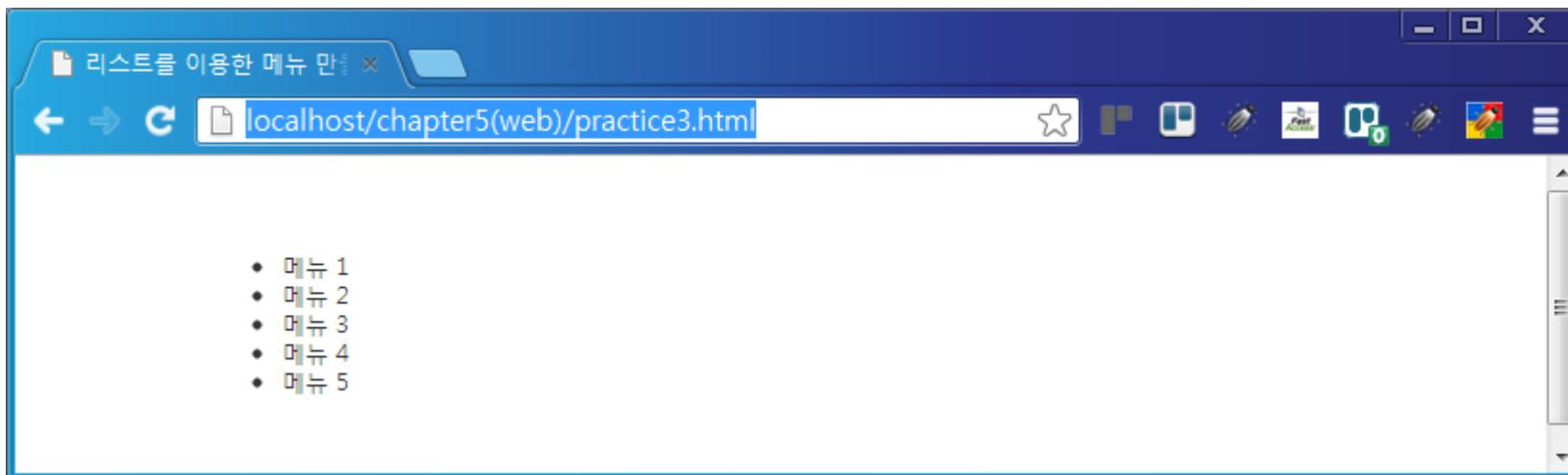
1. 기본 마진값과 패딩값을 없앱니다.
2. body에 다음 속성을 적용해서 하위 엘리멘트 모두가 그 속성을 상속 받도록 해보세요.

기본글꼴 : “맑은 고딕”, “돋움”

기본글꼴 사이즈 : 0.75em( 12px )

텍스트 색상 : #333

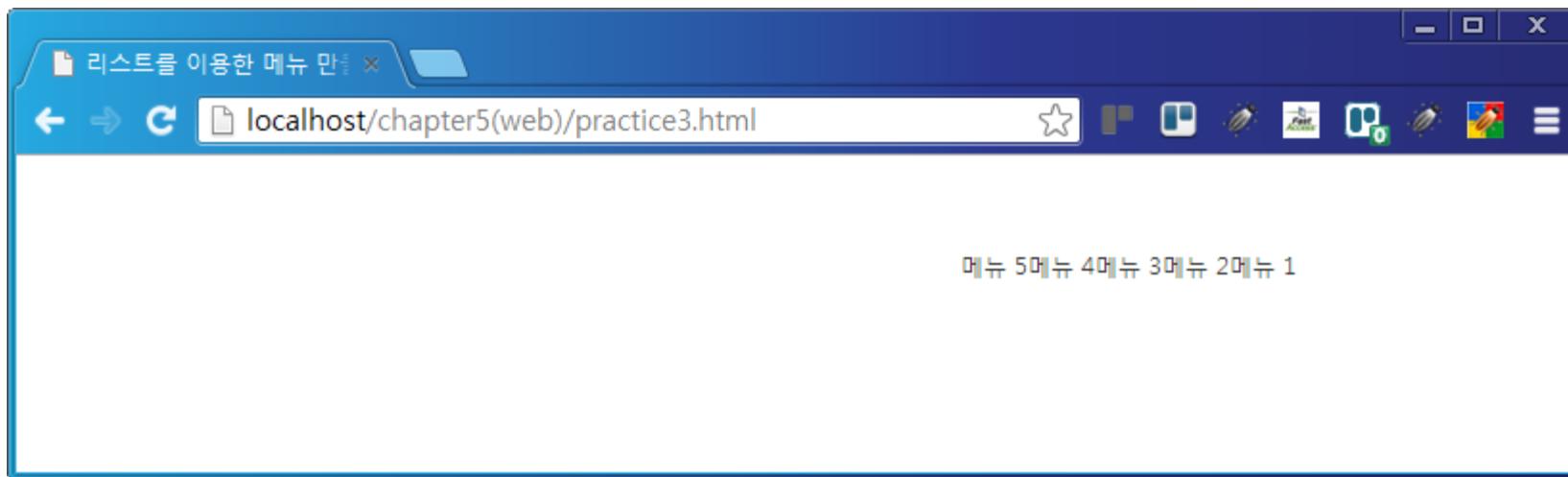
3. 그리고 리스트를 둘러 싸고 있는 <div>에 class “tab-box”를 부여 합니다.
4. 탭박스에 margin-top 20px, margin-left , margin-right를 자동으로 하여 중앙에 위치 시킵니다.
5. 탭박스의 넓이를 지정해 줍니다. 520px



### 3. 리스트를 이용한 메뉴 만들기

#### □ [ 실습문제 3 – 단계2 ]

1. 탭메뉴에 해당되는 <ul>를 list-type-style 를 none으로 설정합니다.
2. 탭메뉴 아이템 <li> 를 right로 float 시킵니다.



### 3. 리스트를 이용한 메뉴 만들기

#### □ [ 실습문제 3 – 단계3 ]

1. 탭메뉴 아이템 <li>를 탭으로 만듭니다.

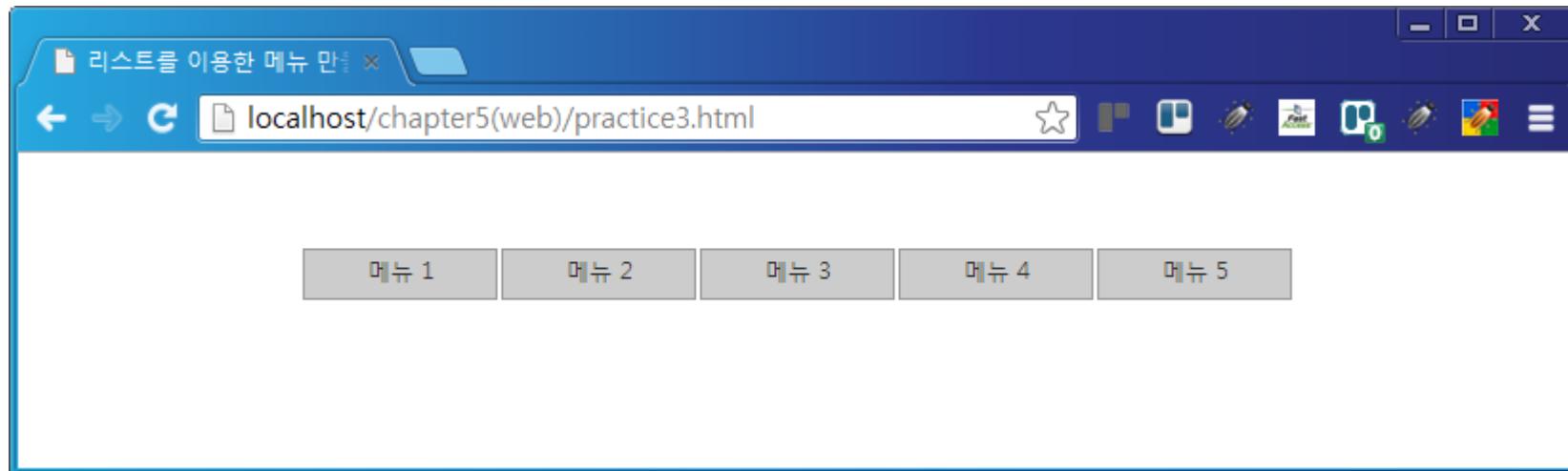
width:100px height:22px; border:1px solid #999, background-color:#ccc

2. 탭 꾸미기

**text-align:center, padding-top: 3px**

3. 탭 아이템 간 간격 벌리기

</li> 외 margin-right:2px 를 적용합니다.



### 3. 리스트를 이용한 메뉴 만들기

#### □ [ 실습문제 3 – 단계4 ]

1. 모서리가 둥근 탭을 만듭니다. ( css3 )

**border-top-left-radius** 과 **border-top-right-radius**를 각각 10px 씩 적용합니다.

2. 탭 리스트 <ul> 의 높이를 적용합니다.

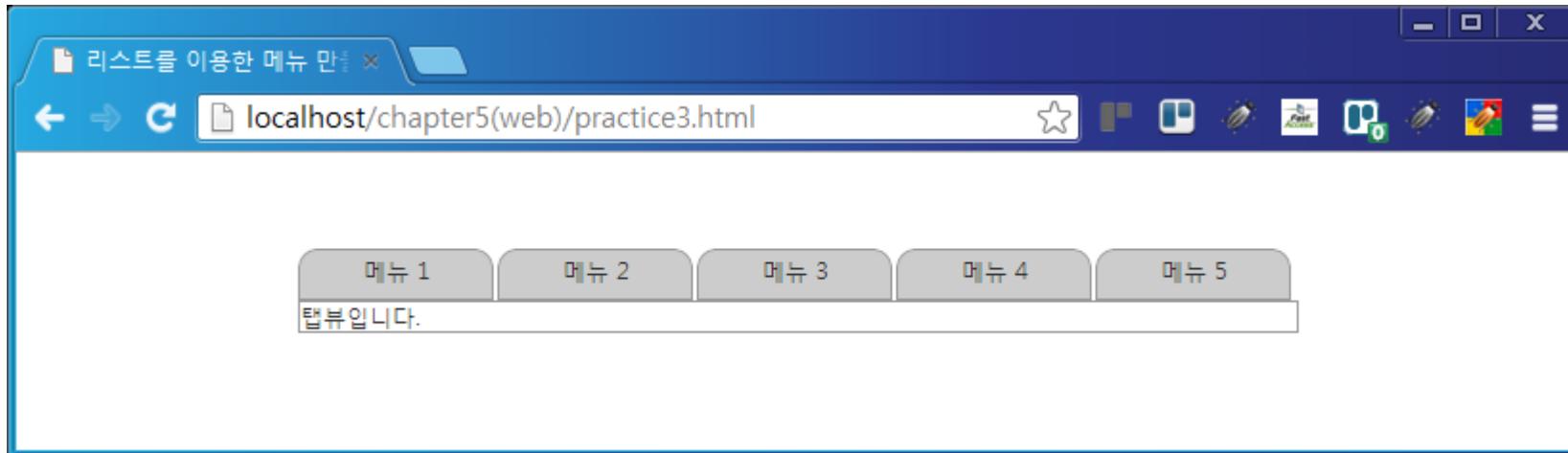
$22 + 2 + 3 = 27\text{px}$  ( 탭 아이템 높이 + 테두리 선 + 내부 padding 값 )

3. 탭 리스트 아래에 다음 속성으로 탭 뷰 엘리먼트

<div> 탭뷰입니다. </div>

를 추가 합니다.

*width:100px, border:1px solid #fff*



### 3. 리스트를 이용한 메뉴 만들기

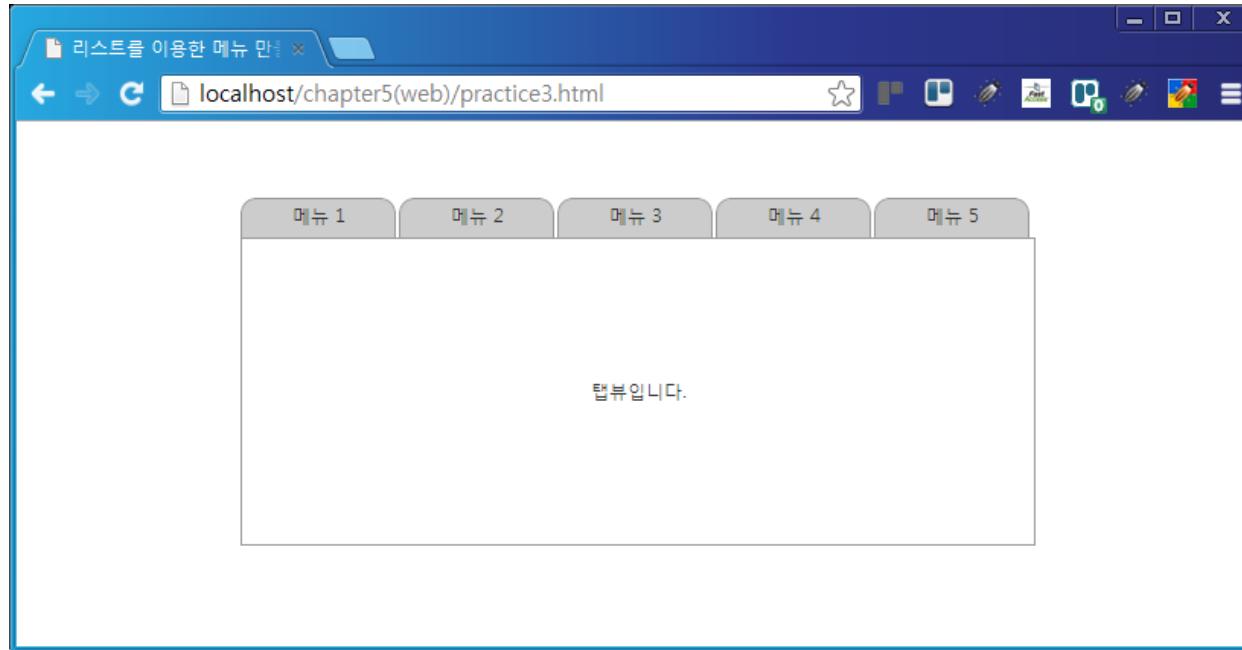
#### □ [ 실습문제 3 – 단계5 ]

1. 탭뷰의 위치를 조정해서 탭과의 수평라인을 맞춥니다.

**margin-top: -1px**

2. 테스트를 위해 탭뷰의 스타일을 조정합니다.

**text-align:center;  
line-height:200px;**



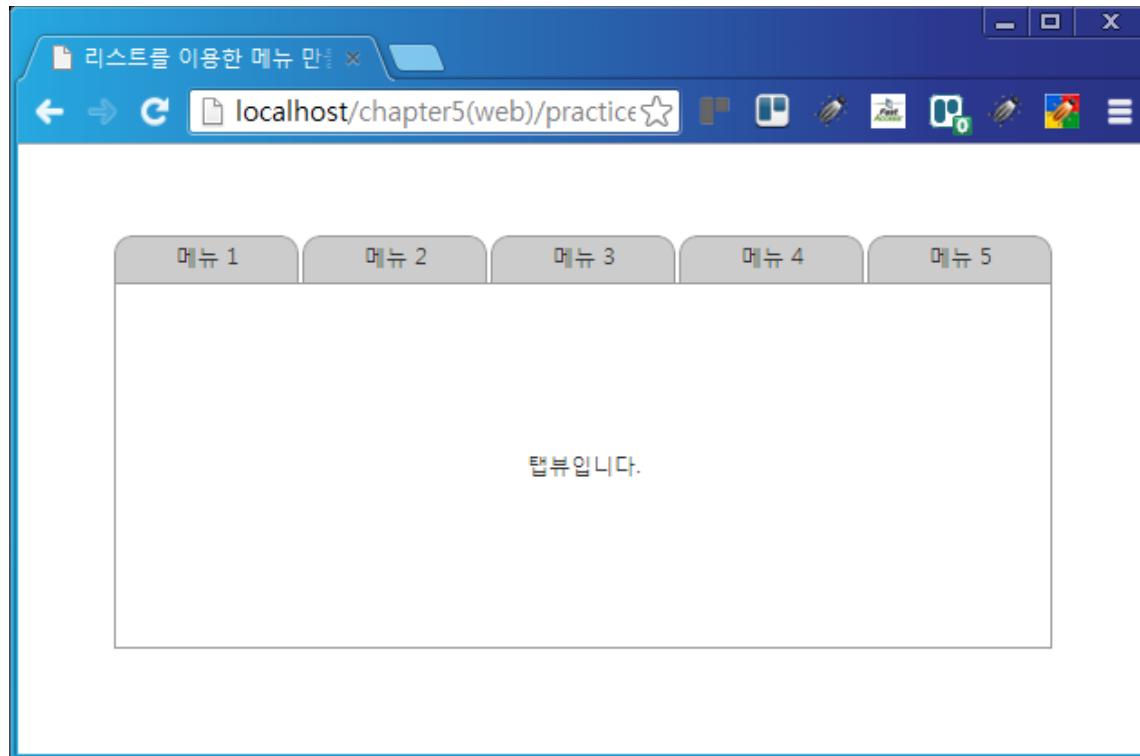
### 3. 리스트를 이용한 메뉴 만들기

#### □ [ 실습문제 3 – 단계6 ]

1. 탭뷰의 길이를 조정해서 탭과의 길이를 맞춥니다. ( 현재 100%가 왜 틀립니까? )

**width: 516px**

2. 완성 되었습니다. 추가해야 하는 부분은 탭의 선택 그리고 작동하게끔 하는 자바스크립트 코드입니다.



### 3. 리스트를 이용한 메뉴 만들기

#### □ [ 실습과제 3 ]

HTML 소스를 보면 “메뉴4”라는 텍스트가 있는 <li>에 “selected”라는 클래스 이름이 지정되어 있습니다. 이 클래스 이름으로 그림과 같은 선택 효과를 내보세요.



### 3. 리스트를 이용한 메뉴 만들기

#### □ [ 실습과제 4 ]

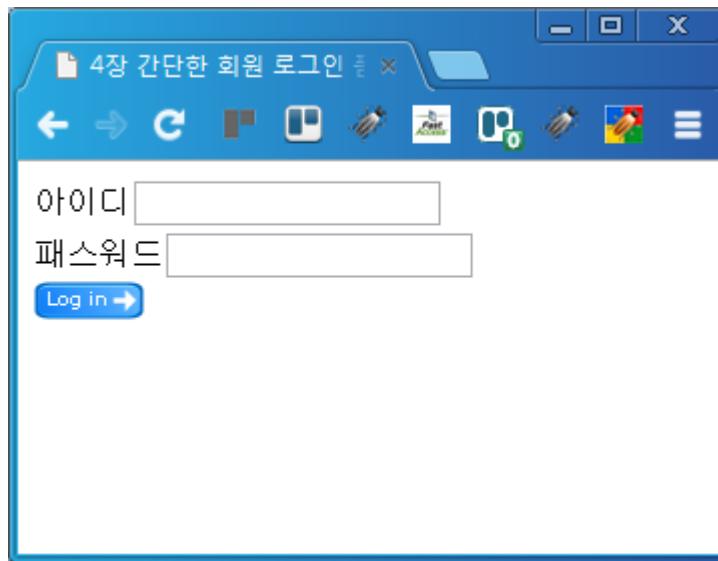
마우스를 눌렀을 때 탭이 선택되고 탭 뷰안의 텍스트가 변하는 것을 구현해보세요.



## 4. 폼제작 하기

### □ [ 실습문제 4 ] 기본 form으로 폼 제작 (practice4.html)

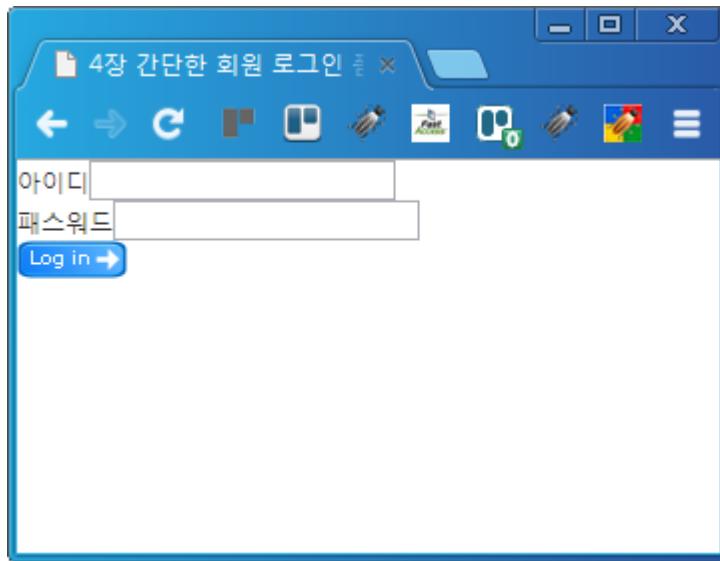
- 로그인 폼을 CSS적용해 보다 정교한 폼으로 만드는 실습문제 입니다.



## 4. 품제작 하기

### □ [ 실습문제 4 - 1단계]

1. 기본 마진값과 패딩값을 없앱니다.
2. body에 다음 속성을 적용해서 하위 엘리멘트 모두가 그 속성을 상속 받도록 해보세요.  
기본글꼴 : “맑은 고딕”, “돋움”  
기본글꼴 사이즈 : 0.75em( 12px )  
텍스트 색상 : #333
3. form의 id를 login-form이라고 설정합니다.



## 4. 폼제작 하기

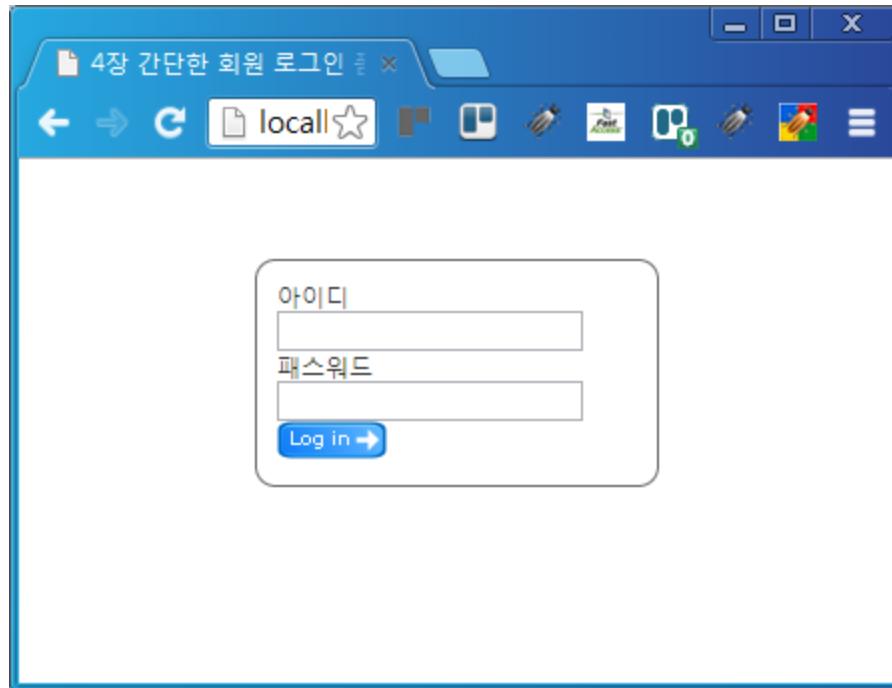
### □ [ 실습문제 4 - 2단계]

1. login-form 의 width 및 테두리 길이등에 스타일을 적용합니다.

width:180px, padding:10px, border 1px solid #666 이며

border-radius: 10px (CSS3) 둥근모서리로 바꿉니다.

margin 은 left right는 자동으로 잡아 주고 height는 대략 50px쯤 잡아 줍니다.



## 4. 폼제작 하기

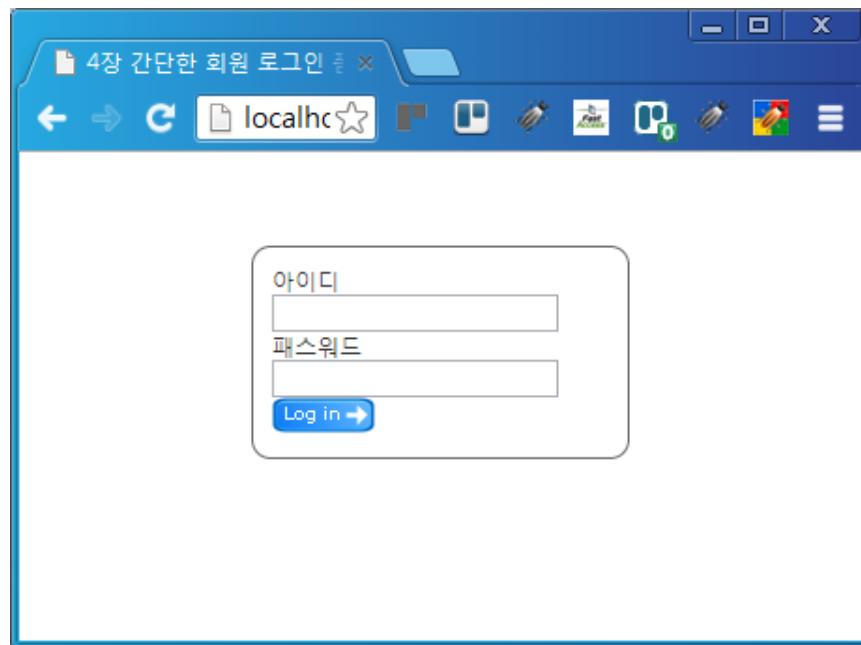
### □ [ 실습문제 4 - 3단계]

1. label은 input 앞에서 라벨링을 합니다.
2. label은 원래 인라인 태그입니다.

실습문제와 같이 <br>태그 없이 아래로 떨어지기 위해서는  
label을 블록 엘리멘트로 고쳐줘야 합니다.

```
label { display:block }
```

3. HTML 소스에서 <br> 모두 없애고 label를 테스트 해보세요



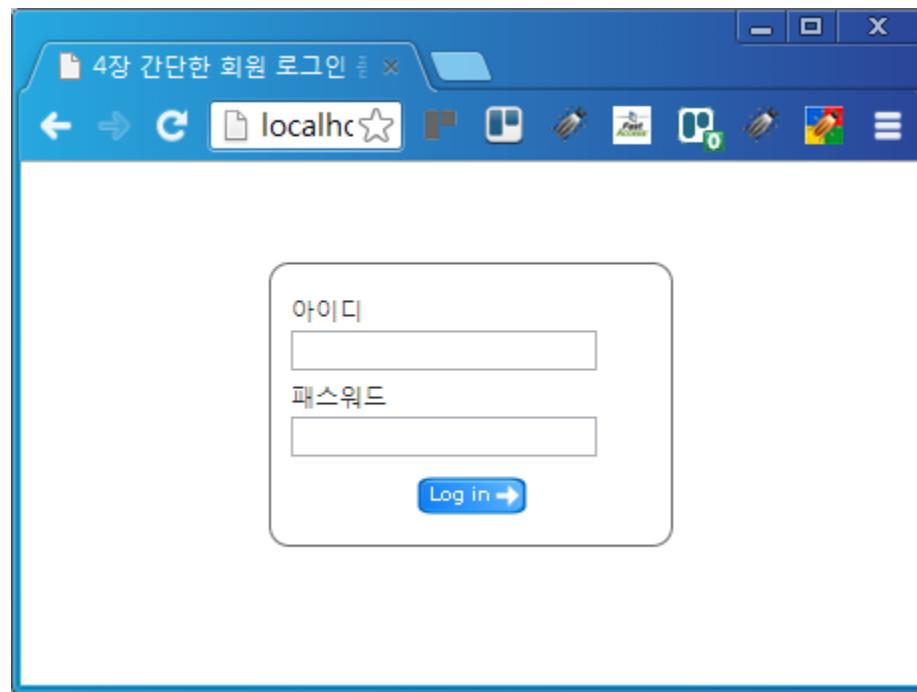
## 4. 폼제작 하기

### □ [ 실습문제 4 - 4단계]

1. label은 input과 너무 붙어 있으므로 top margin 과 bottom margin을 각각 5px 3px 정도 적용해 보세요.
2. login button 도 위치 조정이 필요 합니다.

input 엘리먼트는 inline 엘리먼트로 label과 마찬가지로 display:block 속성을 주고 이미지의 width 속성을 준후, 적당히 margin 값을 조절해 보세요.

input[type=image] 로 input은 type 지정으로 선택할 수 있으니 적용해 보세요.



# Part III

## 자바스크립트

### 1. JavaScript Basic

1. 개요
2. 기본 문법
3. 객체
4. 이벤트

## 1. 개요 - 역사

- 네스케이프의 라이브 커넥트라는 서버측 기술에서 연동이 필요한 클라리언트측의 스크립트 언어였던 라이브스크립트에서부터 시작
- 네스케이프사가 **SUN**사와의 제휴로 이름이 자바스크립트로 변경하고 **HTML** 보완 언어로 발표 (1995년)
- **MS**의 독자적 **J**스크립트 발표등 통일되지 않은 스펙과 호환성 문제 발생
- 네스케이프사가 1996년 자바스크립트 스펙을 **ECMA**국제회의에 제출, 첫 버전 **ECMA-262** 발표.
- 지금은 **ECMA-262** 세 번째 버전이 나와 있으며 이를 기반으로 한 자바스크립트 1.6이 대부분 브라우저에서 지원되고 있다.

# 1. 개요 – Hello World

## □ [실습1] Hello World

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  
         "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  
<html xmlns="http://www.w3.org/1999/xhtml" lang="ko">  
<head>  
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />  
<title>Hello World</title>  
<script type="text/javascript">  
    document.write( "Hello World" );  
</script>  
</head>  
<body>  
</body>  
</html>
```

**document.write( "Hello World" )** 대신 **alert( "Hello World" )**로 바꾸고 브라우저에서 확인보세요.

## 1. 개요 - 실행과 특징

---

- 브라우저 내에 내장된 자바스크립트 실행엔진 (해석기)를 통해 실행되어지고  
브라우저 화면에 반영된다.
- 객체지향이라기 보다는 **객체 기반 스크립트 언어**
- 객체를 생성하느 클래스 개념은 지원하고 있지 않지만 **prototype** 이라  
는 객체 생성 지원 개념이 있다.

## 1. 개요 - 실행과 특징

### □ HTML 문서 내에서 자바스크립트 사용하기

```
<html>
<body>
...
<script
type="text/javascript">
    자바스크립트 소스코드
</script>
...
</body>
</html>
```

```
<html>
<head>
<script type="text/javascript">
    자바스크립트 소스코드
</script>
</head>
<body>
...
</body>
</html>
```

보통 헤드안에 삽입되어 실행 되나 본문 어디에도 삽입되어 실행될 수 있다

## 1. 개요 - 실행과 특징

### □ HTML 문서 내에서 자바스크립트 사용하기

```
<html>
<head>
<script type="text/javascript" src="hello.js"></script>
</head>
<body>
...
</body>
</html>
```

- 외부 파일을 불러서 실행할 수 있다 ( 보통 이방식으로 자바스크립트를 실행하고 관리한다 )

### □ [ 실습2 ]

실습1를 외부파일에서 불러와 실행 시키는 방식으로 수정하고 확인해 보세요.

## 2. 기본문법 - 변수 와 데이터 타입

### □ 기본 데이터 타입

Number, String, Boolean

```
<script type="text/javascript">

var number = 5
var anotherNumber = new Number(5)
var pi = 3.14
var string = "Hello, I'm a string!"
var anotherString = new String( "Hello, I'm a string!" );
var used = false;

alert( typeof number)
alert( typeof anotherNumber )
alert( typeof pi )
alert( typeof string )
alert( typeof anotherString )
alert( typeof used )

</script>
```

## 2. 기본문법 - 변수 와 데이터 타입

### □ null 과 undefined

```
<script type="text/javascript">

var myVar, myVar2 = null;
alert( myVar + " , " + myVar2 );
alert( myVar == myVar2 );
alert( myVar === myVar2 );

</script>
```

## 2. 기본문법 - 변수 와 데이터 타입

### □ 변수 이름 규칙

변수의 이름은 알파벳(대문자 A ~ Z, 소문자 a ~ z), 밑줄(\_)이나 달라(\$)로 시작될 수 있으며, 다음에는 알파벳, 밑줄, 달라 기호에 추가로 숫자(0 ~ 9)까지 사용할 수 있다.

### □ 선언없이 대입시에 변수 타입이 결정된다.

```
<script type="text/javascript">

v = "This is Test";
alert( typeof v );

v = 20;
alert( typeof v );

</script>
```

## 2. 기본문법 - 변수 와 데이터 타입

### □ 변수 범위

```
<script type="text/javascript">

function func() {
    v1 = "hello javascript";
}

func();
alert( v1 );

</script>
```

□ v1 앞에 var를 붙히게 되면?

□ var와 함께 변수에 값을 대입하는 것이 좋다.

## 2. 기본문법 - 구문

- 세미콜론을 반드시 구문끝에 붙혀야 하는 것은 아니다.

```
<script type="text/javascript">  
  
var str = "This is Test"  
document.write( str )  
  
</script>
```

- 한 줄로 붙혀 쓸 수는 없다.

```
<script type="text/javascript">  
var str = "This is Test" document.write( str )  
</script>
```

- 엔진이 한줄 씩 일거나 ;(세미콜론) 으로 분리된 구문을 읽어 해석해 나가기때문에  
문장사이에는 \n(개행) 이나 ;(세미콜론) 으로 분리

## 2. 기본문법 - 구문

- 자바와 마찬가지로 Camel 표기법이 기본

```
<script type="text/javascript">  
  
    //// CamelCase is the norm  
    if(fooBar == bazBar) {}  
  
</script>
```

- 객체 속성, 메서드에 접근 할 때에는 . 으로 접근한다.

```
<script type="text/javascript">  
someObject.someMethod();  
</script>
```

- 주석은 한줄 주석 // , 여러줄 주석 /\* \*/ 모두 가능하다.

## 2. 기본문법 - 조건문

- **If문, If~else, if~else if** 등의 조건문은 다른 언어들과 다르지 않다.

```
<script type="text/javascript">

var something = false;
if( something ) doSomething();

</script>
```

```
<script type="text/javascript">

if(something == foobar) {
    alert("equals foobar!");
} else if(something == bazbat) {
    alert("equals bazbat!");
} else {
    alert("equals neither!");
}

</script>
```

## 2. 기본문법 - 조건문

### □ Switch 문도 사용이 가능하다.

```
<script type="text/javascript">

switch( something ) {
    case "foobar":
        // if something == "foobar"
        alert( "foobar!" );
        break
    case "barfoo":
        // if something == "barfoo"
        alert("Barfoo!");
        break
    case "fallthru":
        // without a break, results will cascade
        // the result? [alert] Falling through...
        // [alert] fallen through
        alert("Falling through...");
    case "fellthru":
        alert("fallen through.");
        break
    default:
        // if there is no case "*" match, execute this code
        alert("Case not found... here's a default")
}
</script>
```

## 2. 기본문법 - 반복문

- **for, while, do~while** 문도 다른 언어들과 다르지 않다.

```
<script type="text/javascript">

for(var i = 0; i < 3; i++) {
    document.write( i + "<br>" );
}

</script>
```

- [ 실습3 ] 1단에서 9단까지 구구단 출력해 보세요.

- [ 실습4 ]

\*\*\*\*\*

\*\*\*\*

\*\*\*

\*\*

\*

### 3. 객체 - 정의 / 생성

- 정보를 관리하기 위해 의미를 부여하고 분류하는 논리적 단위
- 클래스의 인스턴스(instance) -> 구체화, 실체화
- 객체는 속성(attribute)과 함수(function)을 가지고 있다.
- [실습 5] 객체의 생성

```
var employee1 = new Object();
employee1.name = "홍길동";
employee1.title = "과장";
employee1.showInfo = function() {
    document.write( "이름 : " + this.name );
    document.write( "<br>" );
    document.write( "직책 : " + this.title );
}

employee1.showInfo();
```

### 3. 객체 - 정의 / 생성

- JSON ( JavaScript Object Notation )
- 자바스크립트에 객체생성을 위한 표기하는 방법
- 어떤 객체인지 표기할 수 있고 바로 생성 가능
- [실습 6 - 1] 객체의 생성

```
var employee1 = {};
employee1.name = "홍길동";
employee1.title = "과장";
employee1.showInfo = function() {
    document.write( "이름 : " + this.name );
    document.write( "<br>" );
    document.write( "직책 : " + this.title );
}

employee1.showInfo();
```

### 3. 객체 - 정의 / 생성

#### □ [실습 6 - 2] 객체의 생성(JSON)

```
var employee1 = {
    name: "홍길동",
    title: "과장",
    showInfo: function() {
        document.write( "이름 : " + this.name );
        document.write( "<br>" );
        document.write( "직책 : " + this.title );
    }
}

employee1.showInfo();
alert( employee1.name + " " + employee1.title );
```

### 3. 객체 - 정의 / 생성

#### □ [실습 6 - 3] 객체의 생성(JSON) – 콤마 사용에 조심

```
var foo = {  
    name: "bar",  
    nick: "buzz",  
    aNumber: 5,  
    doStuff: function() {  
        alert( "I'm" + this.name );  
    }  
}
```

어디에서 에러가 있는지 찾고 바르게 수정해 보세요.

### 3. 객체 - 정의 / 생성

#### □ 생성자 사용

```
var Foo = function() {
    this.name = "bar";
    this.nick = "buzz";
    this.aNumber = 5;
    this.doFoo = function() {
        alert( "I'm " + this.name );
    };
}

var foo = new Foo();
foo.doFoo();
```

#### □ [ 실습 6 – 4 ]

Foo 생성자에서 name, nick 를 초기화 시킬수 있도록 수정해 보세요.

### 3. 객체 - 정의 / 생성

#### □ prototype 기반의 상속을 통해 객체지향 구현 ( 속성, 함수 공유 )

```
var Foo = function( name, nick ) {
    this.name = name;
    this.nick = nick;
}

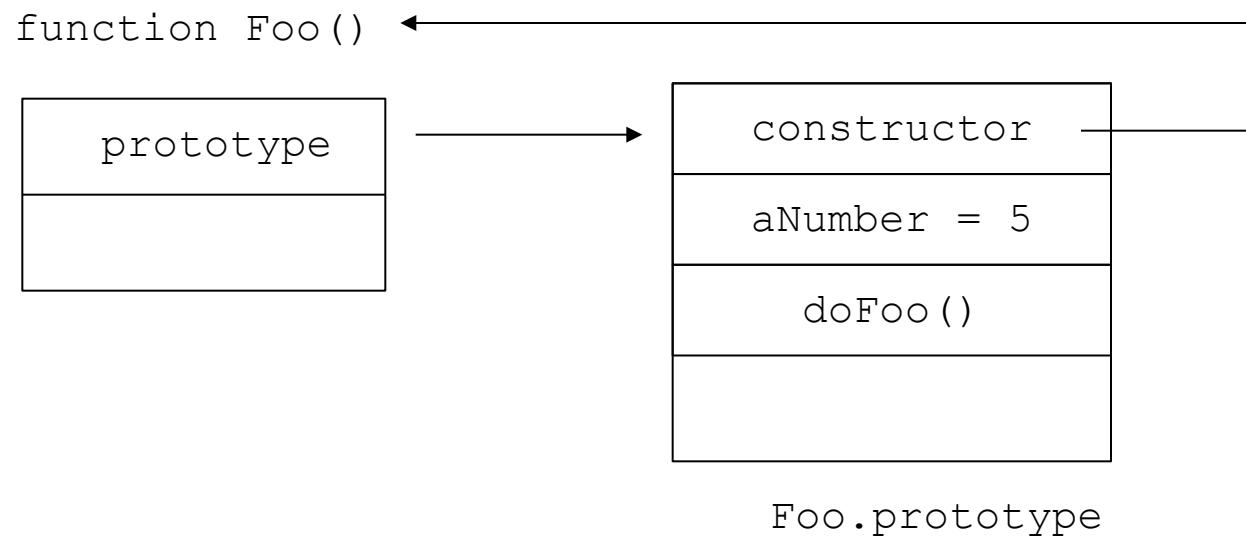
Foo.prototype.aNumber = 5;
Foo.prototype.doFoo = function() {
    alert( "I'm " + this.name );
};

var fool = new Foo( "foo1", "nick1" );
fool.doFoo();

var foo2 = new Foo( "foo2", "nick2" );
foo2.doFoo();
```

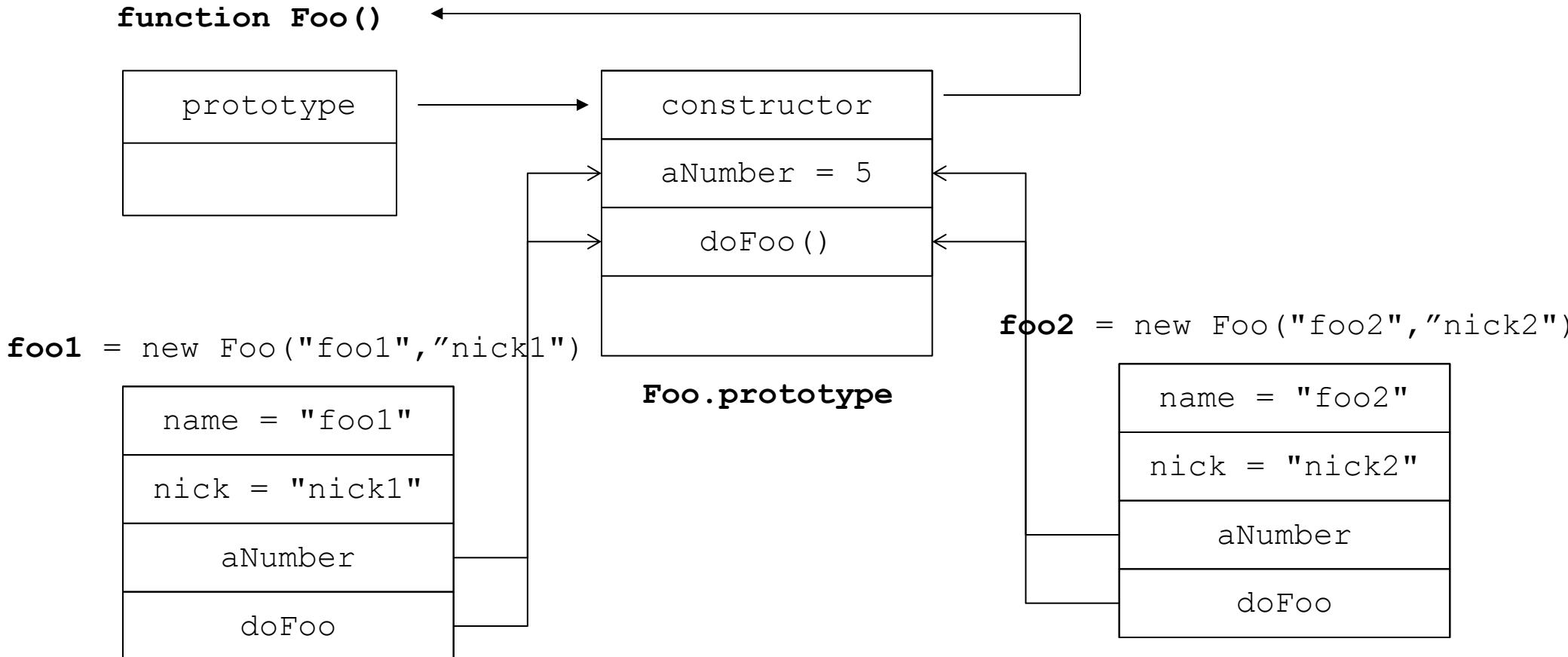
### 3. 객체 - 정의 / 생성

#### □ prototype 기반의 상속



### 3. 객체 - 정의 / 생성

#### □ prototype 기반의 상속



### 3. 객체 - 정의 / 생성

---

#### □ [ 실습예제 6 - 5 ]

다음 속성과 함수(메서드)를 가지고 있는 **Class** 개념을 **function**과 **프로토타입**을 사용하여 구현해 보세요.

1) 생성자 함수 **Rectangle** ( **x1, y1, x2, y2, color** )

2) **LeftTop** 좌표 **x1, y1**

3) **RightBottom** 좌표 **x2, y2**

4) **color** ( **#000** )

5) **show** 함수 : 화면에 사각형을 표시

"[width:100, height:200, color:#000]" 인 사각형을 그렸습니다."

### 3. 객체 - 정의 / 생성

---

#### □ [ 실습예제 6 - 6 ]

- 1) 실습예제 6-5에 CSS를 사용해서 실제로 그려보기
- 2) **position: absolute** 개념 익히기.
- 3) 동적으로 객체에 속성 삽입. (**border**)

### **3. 객체 – 내장 객체 **Array****

---

#### **□ 배열 생성**

- 1) var a = new Array(10);**
- 2) var a = new Array();**
- 3) var a = new Array( 1, "ABC", true) ;**

#### **□ 배열 생성 (리터럴)**

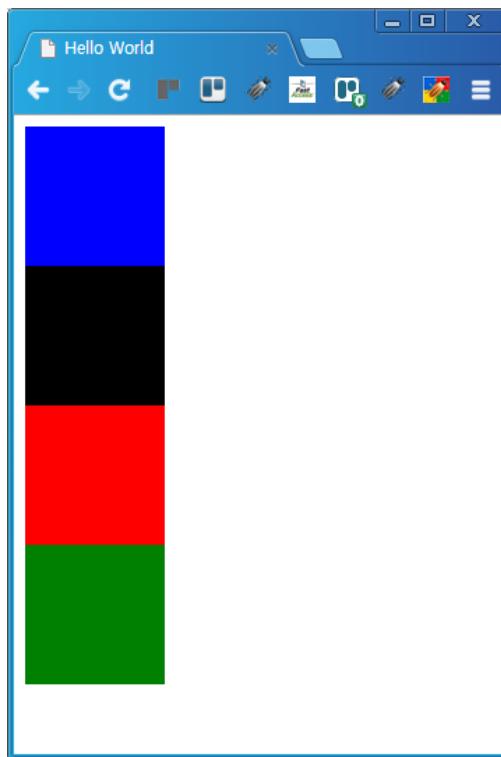
**var a = [];**

### 3. 객체 – 내장 객체 **Array**

□ **length** 속성 : 배열의 size을 담고 있는 속성

□ [ 실습예제 6-7 ]

배열을 사용해서 다음과 같은 결과가 나오도록 자바스크립트 코드를 작성해 보세요.



[힌트] 색상 배열 선언에 다음 코그를 사용합니다.

```
var colors = ["blue", "black", "red", "green"];
```

### 3. 객체 - 내장 객체 **Array**

#### □ 배열과 객체의 관계

속성 접근시 실습예제와 같이 배열처럼 접근할 수 있다

#### [ 실습예제 6-8 ]

```
var employee1 = {  
    name: "홍길동",  
    title: "과장"  
}  
  
alert( employee1["name"] + " " + employee1["title"] );
```

### 3. 객체 - 내장 객체 **Array**

#### □ 주요함수

종류	설명
concat(array1,...)	배열을 하나로 합친다.
join(str)	배열 전체를 str구분자를 가지는 하나의 문자열로 만든다.
pop()	배열의 맨 마지막 변수를 삭제한다.
push(item1,..)	배열의 마지막에 변수들을 추가한다.
reverse()	배열의 순서를 뒤집는다.
shift()	배열의 맨 처음 값을 삭제한다.
slice()	배열의 일부분만을 추출하여 새로운 배열을 만든다.
sort(func)	배열을 정렬한다.

### 3. 객체 - 내장 객체 **Array**

#### □ 주요 함수 예제

```
var hege = [ "Cecilie", "Lone" ];
var stale = [ "Emil", "Tobias", "Linus" ];
var children = hege.concat( stale );

console.log( children );
```

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
var energy = fruits.join();

console.log( energy );
```

### 3. 객체 - 내장 객체 **Array**

#### □ 주요 함수 예제

```
var fruits = [ "Banana", "Orange", "Apple", "Mango" ];  
  
fruits.push( "Kiwi" );  
console.log( fruits );  
  
console.log( fruits.pop() );  
console.log( fruits.pop() );  
  
console.log( fruits );
```

### 3. 객체 - 내장 객체 **Array**

#### □ 주요 함수 예제

```
var fruits = [ "Banana", "Orange", "Apple", "Mango" ];  
  
fruits.reverse();  
console.log( fruits );  
  
fruits.shift();  
console.log( fruits );  
  
var citrus = fruits.slice(1, 3);  
console.log( citrus );  
  
fruits.sort();  
console.log( fruits );
```

### 3. 객체 - 내장 객체 **String**

- 배열처럼 다룰 수 있다.

#### [예제 6-9]

문자열을 배열처럼 접근해서 사용하는 예제입니다.

IE에서 잘 작동하는지 확인해 보고 잘 작동하도록 수정해 보세요.

```
var str = "Hello, I'm a string!";
console.log( "문자열 길이는 : " + str.length );

for(var i = 0; i < str.length; i++) {
    console.log( str[i] );
}
```

### 3. 객체 - 내장 객체 **String**

#### □ 합치기 와 자동 형 변환

##### [예제 6- 10 ]

문자열 합치기 예와 그 때, 자동 변환되는 경우들을 확인해 보세요.

```
console.log( "첫 번째 문자열 " + " 두 번째 문자열" );

var str = "number" + 5;
console.log( str + " : " + typeof( str ) );

str = 5 + "5";
console.log( str + " : " + typeof( str ) );
```

### 3. 객체 - 내장 객체 **String**

#### □ [ 예제 6-11 ] 주요함수 사용 예 1

```
var str = "string1 string2 string3"  
  
alert( str.length );  
  
var start = str.indexOf( 'string2' );  
alert( start );  
alert( str.substr( start ) );  
  
// 간단히,  
alert( str.substr( str.indexOf( 'string2' ) ) );  
  
// str은 변하지 않는다.  
alert( str );
```

### 3. 객체 – 내장 객체 **String**

#### □ [ 예제 6-12 ] 주요함수 사용 예 2

```
var str = "string1 string2 string3"  
  
// 배열로 분리한다.  
var a = str.split(' ');  
  
// 배열을 확인하는 코드를 직접 작성해 보세요.
```

### 3. 객체 - 내장 객체 **String**

#### □ [ 예제 6-13 ] Escaping HTML, URLs, etc.

```
//에러
"<h3>Here's a headline!</h3>".escape();

// escape 함수는 전역함수로 제공
var escaped = escape("<h3>Here's a headline!</h3>");
var unescaped = unescape(escaped);

// URL 엔코딩
var url = "http://mysite.com/?stuff=\"안 대혁! &bar=";
var encodedURL = encodeURI(url);
var decodedURL = decodeURI(encodedURL);
```

### 3. 객체 - 내장 객체 **Date**

□ Date 객체는 날짜와 시간을 다루는 객체이다.

□ 기본 사용법

```
var d = new Date(); // 현재 시간  
document.write( d );
```

□ [예제 6-14]

다음 Date 객체 생성자를 사용해 Date 객체를 생성하고 document.write 를 이용해 결과를 확인해 보세요.

1) Date( year, month, day )

2) Date( yyyy, mm, dd, hh, mi, ss)

3) Date( milliseconds )

### 3. 객체 - 내장 객체 **Date**

#### □ 관련 함수

종류	설명
getYear() / setYear()	년도
getMonth() / setMonth()	월(0:1월, 1:2월, ......., 11:12월)
getDate() / setDate()	일(1일 ~ 31일)
getDay() / setDay()	요일(0:일요일, 1:월요일, ..., 6:토요일)
getHours() / setHours()	시간(0시 ~ 23시)
getMinutes() / setMinutes()	분(0 ~ 59)
getSeconds() / setSeconds()	초(0시 ~ 59)
getMilliseconds() / setMilliseconds()	시간(0시 ~ 23시)
getHours() / setHours()	시간(0시 ~ 23시)

### 3. 객체 - 내장 객체 **Date**

#### □ [ 실습예제 6-15 ] Date 객체 함수 사용

```
var d = new Date(2013, 0, 28); //2013년 1월 28일
document.write(
  "년도: " + (d.getFullYear() + 1900) + "<BR>" +
  "월: " + (d.getMonth() + 1) + "<BR>" +
  "일은: " + d.getDate() + "<BR>" +
  "요일은: " + d.getDay() + "<BR>" +
  "시는: " + d.getHours() + "<BR>" +
  "분은: " + d.getMinutes() + "<BR>" +
  "초는: " + d.getSeconds() + "<BR>" +
  "밀리초: " + d.getMilliseconds() + "<HR>");

d.setYear(2014); // 2014년 세팅
document.write(d + "<HR>");

d.setMonth( 11 ); // 12월 세팅
document.write(d + "<HR>");
```

### 3. 객체 - 내장 객체 **Function**

---

□ 함수도 객체로 간주

□ 함수 생성 방식

1) var sum = new Function( "a", "b", "return a+ b" );

2)

```
function sum ( a, b ) {  
    return a+b;  
}
```

3)

```
var sum = function( a, b ) {  
    return a+ b;  
}
```

### 3. 객체 - 내장 객체 **Function**

#### □ 함수 argument

#### □ [실습예제 6-16]

함수의 **argument**는 값과 객체뿐만 아니라 함수도 될 수 있음을 다음 예제로 확인해 보세요.

```
function myFunction( arg1, arg2, arg3 ) {  
  
    // 값  
    alert("I have an argument! " + arg1);  
  
    // 객체  
    alert(arg2.bar);  
  
    // 함수  
    arg3();  
}  
  
myFunction( "foo", { bar: "baz" }, function(){ alert("Victory!") } );
```

### 3. 객체 - 내장 객체 **Function**

#### □ [실습예제 6-16]

함수의 **argument**로 함수가 넘어 갈 경우 함수의 본문이 길어 질 경우,

```
var f = function() {  
    alert( "victory!" );  
}  
  
myFunction( "foo", { bar: "baz" }, f );
```

직접 예제에 적용해서 확인해 보세요.

### 3. 객체 - 내장 객체 **Function**

#### □ [실습예제 6-16]

함수의 **argument**는 함수내부에서 **argument** 객체로 참조할 수 있다.

```
function myFunction() {  
    // 값  
    alert("I have an argument! " + arguments[0] );  
  
    // 객체  
    alert( arguments[1].bar );  
  
    // 함수  
    arguments[2]();  
}
```

직접 예제에 적용해서 확인해 보세요.

## 4. 이벤트

- HTML DOM은 다음과 같은 javascript 가 이벤트에 반응할 수 있도록 하고 있다.
  - 사용자의 마우스 클릭
  - 웹 페이지의 로딩 완료 되었을 때
  - 이미지가 로딩 되었을 때
  - HTML element에 마우스가 움직이거나 오버되었을 때
  - Input 필드가 변경 되었을 때
  - HTML form 이 submit 될 때
  - 사용자의 key 누름
- 이벤트에 대한 반응 처리 ( HTML Event Attribute 에 javascript 를 추가한다 )

**onclick = JavaScript**

## 4. 이벤트

### [ 실습예제 6 – 17 ]

```
<!DOCTYPE html>
<html>
<body>
<h1 onclick="this.innerHTML='Ooops!'">Click on this text!</h1>
</body>
</html>
```

- 이벤트를 처리할 수 있는 함수 ( Event Handler )로 처리할 수 있다.

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function changetext(id) {
  id.innerHTML="Ooops!";
}
</script>
</head>
<body>
<h1 onclick="changetext(this)">Click on this text!</h1>
</body>
</html>
```

## 4. 이벤트

- HTML Element에 이벤트를 맵핑할 때는 이벤트속성(Event Attribute)을 사용

### [ 실습예제 6-18 ]

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript">
function displayDate() {
    document.getElementById("demo").innerHTML = Date();
}
</script>
</head>
<body>
<p>버튼을 클릭하면 <em>displayDate()</em> 함수가 실행 됩니다.</p>
<button onclick="displayDate()">Try it</button>
<p id="demo"></p>
</body>
</html>
```

## 4. 이벤트

---

### [ 실습예제 6-19 ]

[ 실습예제 6-18 ] 에 이벤트 속성 추가해 보기

1) 마우스오버 : **onmouseover**

**onclick** 반응과 같도록 같은 이벤트 핸들러를 사용합니다.

2) 마우스 아웃 : **onmouseout**

날짜가 사라지게끔 이벤트 핸들러를 새로 추가합니다.

## 4. 이벤트

- 자바스크립트로 특정 Element 이벤트 맵핑하기

### [ 실습예제 6 – 20 ]

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript">
document.getElementById("myBtn").onclick = function() {
    displayDate()
};

function displayDate() {
    document.getElementById("demo").innerHTML = Date();
}

</script>
</head>
<body>
<p>버튼을 클릭하면 <em>displayDate()</em> 함수가 실행 됩니다.</p>
<button id="myBtn">Try it</button>
<p id="demo"></p>
</body>
</html>
```

## 4. 이벤트

---

[ 실습예제 6-21 ]

[ 실습예제 6-19 ] 의 이벤트속성을 사용해 이벤트 핸들러와 연결했던 것을  
자바스크립트를 사용해 이벤트 핸들러와 연결( 매피ング ) 해 보세요.

## 4. 이벤트

---

- **onload** 는 사용자가 특정 페이지에 입장 했을 때 발생한다.
- 반대로 **onunload** 가 특정 페이지를 떠나면 발생한다.
- **onload**에서 페이지의 최초작업 초기화 작업을 할 수 있을 것이다.

## 4. 이벤트

### [ 실습예제 6 – 22 ]

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript">
function checkCookies() {
    if( navigator.cookieEnabled==true ){
        alert("쿠키를 사용할 수 있습니다.")
    } else {
        alert("쿠키를 사용할 수 없습니다.")
    }
}
</script>
</head>
<body onload="checkCookies()">
<p>브라우저에서 쿠키 사용여부를 어떻게 설정했는 지 알수 있습니다.</p>
</body>
</html>
```

## 4. 이벤트

---

### [ 실습예제 6 – 23 ]

**BrowserDetect** 객체를 사용해서 페이지에 방문하는 사용자의 브라우저 종류, 버전 그리고 OS를 알아 내는 코드를 **.onload** 이벤트 핸들러에 추가해 보세요.

### [참고]

```
<javascript type="text/javascript" src="browser-detect.js"></script>
```

## 4. 이벤트

- **onchange** 는 **Input , select** 값이 변하면 발생한다.

[ 실습예제 6 – 24 ]

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript">
function myFunction() {
    var x=document.getElementById("fname");
    x.value=x.value.toUpperCase();
}
</head>
<body>
아이디 입력: <input type="text" id="fname" onchange="myFunction()">
<p>INPUT에 포커스가 없어지면, 이전 값이 변했는 지 판단해서 항상 대문자로 만드는 예
제입니다.</p>
</body>
</html>
```

## 4. 이벤트

- **onchange** 는 **Input , select** 값이 변하면 발생한다.

[ 실습예제 6 – 24 ]

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript">
function myFunction() {
    var x=document.getElementById("fname");
    x.value=x.value.toUpperCase();
}
</head>
<body>
아이디 입력: <input type="text" id="fname" onchange="myFunction()">
<p>INPUT에 포커스가 없어지면, 이전 값이 변했는 지 판단해서 항상 대문자로 만드는 예
제입니다.</p>
</body>
</html>
```

## 4. 이벤트

### □ onmousedown, onmouseup 그리고 onclick

[ 실습예제 6-25 ]

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script>
function mDown( obj ) {
    obj.style.backgroundColor="#1ec5e5";
    obj.innerHTML="버튼을 떼주세요.";
}
function mUp( obj ) {
    obj.style.backgroundColor="#D94A38";
    obj.innerHTML="클릭하세요!";
}
</script>
</head>
<body>
<div onmousedown="mDown(this)" onmouseup="mUp(this)" style="background-color:#D94A38; width:90px; height:20px; padding:40px;">클릭하세요!</div>
</body>
</html>
```

## 4. 이벤트

---

### [과제]

더 많은 이벤트 예제들은 다음 페이지에서 참고할 수 있습니다.

### HTML DOM Events

[http://www.w3schools.com/jsref/dom\\_obj\\_event.asp](http://www.w3schools.com/jsref/dom_obj_event.asp)

이 문서에서

**Mouse Event,**

**Keyboard Event,**

**Form Event**

의 예제들을 확인해 보세요

# Part III

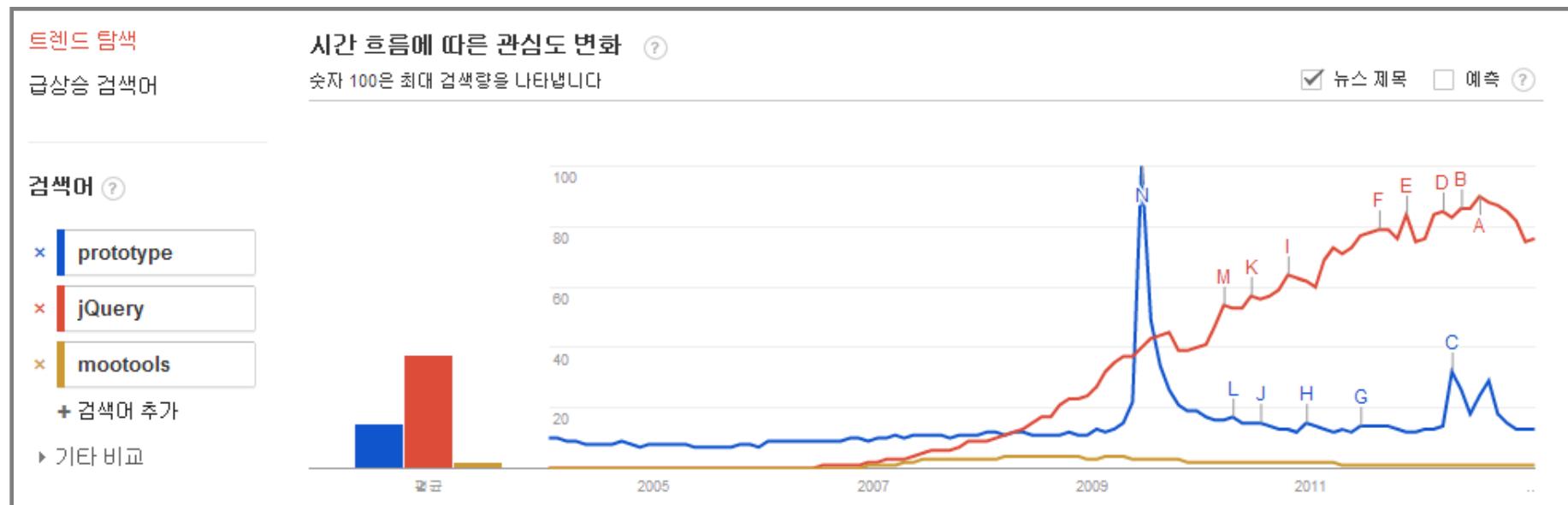
## 자바스크립트

### 1. jQuery

1. 개요
2. jQuery Basic

## 1. 개요 - 소개

- 2006년 Mozilla의 자바스크립트 에반젤리스 Jhon Resig에 의해 개발 / 공개
- 여러 자바스크립트 라이브러리 ( **prototype.js**, **Mootool.js** 등) 중에 가장 주목 받고 있다.
- **jQuery**로 코딩하면 자바스크립트 코드가 간결해 진다.
- 가볍다 ( 90KB )
- IE6.0 이상, Firefox2.0 이상, Safari 3 이상, Opera 9이상, Google Chrome등의 주요 브라우저를 지원하여 클로스브라우징을 가능케 한다.



## 1. 개요 - 사용준비

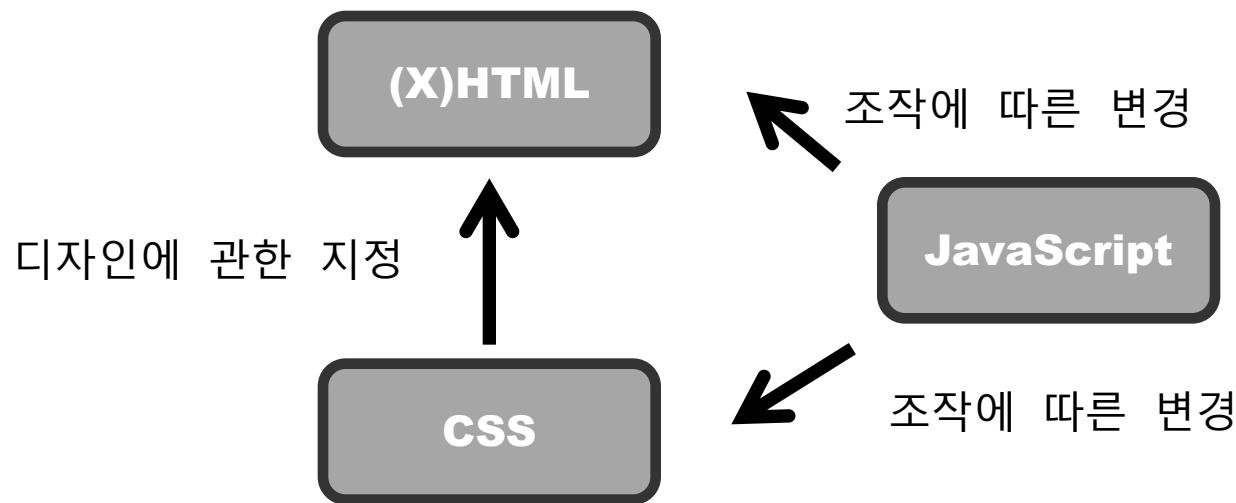
- 다운로드 (<http://jquery.com/download/>, 최신 버전 1.9.0)
- 개발시에는 **uncompressed** 버전( jquery-1.9.0.js )으로 개발
- 릴리즈시에는 **compressed** 버전 ( jquery-1.9.0.min.js )으로 릴리즈한다.

### [실습예제 1] jQuery 설치 및 버전 확인

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript" src=".//jquery/jquery-1.9.0.js"></script>
<script>
alert( $( ) . jquery );
</script>
</head>
<body>
</body>
</html>
```

## 2. jQuery Basic

### □ (X)HTML + CSS + JavaScript ( jQuery )



## 2. jQuery Basic

### □ ready 함수

#### [실습예제 2 - 1] javascript의 실행 타이밍.

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript" src="./jquery/jquery-1.9.0.js"></script>
<script>
    alert("hello jquery");
</script>
</head>
<body>
<p>
    이 문장이 보이고 Hello World가 뜨면, 자바스크립트에서 HTML 엘리멘트에 접근할 수
    있는 것입니다.
</p>
</body>
</html>
```

## 2. jQuery Basic

### □ ready 함수

#### [실습예제 2 - 2] javascript의 실행 타이밍.

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript" src="./jquery/jquery-1.9.0.js"></script>
<script>
$( document ).ready( function() {
    alert( "Hello jQuery" );
})
</script>
</head>
<body>
<p>
이 문장이 보이고 Hello World가 뜨면, 자바스크립트에서 HTML 엘리멘트에 접근할 수
있는 것입니다.
</p>
</body>
</html>
```

## 2. jQuery Basic - 선택자

---

### □ 선택자 ( selector )

- HTML Element를 선택하는 역할을 한다.
- 문서에서 Element 를 가져온 후, 반환된 객체 함수를 사용하여 Element를 조작하게 된다.

```
var object = $( "selector" );  
object.func();
```

- 1) CSS 에서 자주 사용하는 셀렉터
- 2) CSS2 셀렉터
- 3) CSS의 속성 셀렉터
- 4) jQuery 자체 필터

## 2. jQuery Basic - 선택자

---

- CSS에서 자주 이용되는 셀렉터 - 태그
  - 특정 HTML 태그를 컨트롤하기 위해 사용

[실습예제 3] ex3.html

1) li 엘리먼트 색상 바꾸기 예제 입니다.

2) ready function안을 비웠을 때와 비교 확인 해 보세요.

3) css() 함수의 사용법도 익혀 보세요.

## 2. jQuery Basic - 선택자

---

### □ CSS에서 자주 이용되는 셀렉터 - ID

- 특정 id 속성을 가진 HTML 태그를 컨트롤하기 위해 사용
- ID값에 #(hash)를 붙힌다.

### [실습예제 4] ex4.html

- 1) 특정 li 엘리먼트 색상 바꾸기 예제 입니다.
- 2) 아이디를 바꾸어 가며 테스트 해보세요.

## 2. jQuery Basic - 선택자

---

### □ CSS에서 자주 이용되는 셀렉터 - 클래스

- 특정 **class** 속성을 가진 **HTML** 태그를 컨트롤하기 위해 사용
- **.(dot)**에 **class** 속성값을 지정하여 선택.

### [실습예제 5] ex5.html

- 1) 특정 클래스를 가진 **li** 엘리멘트 색상 바꾸기 예제 입니다.
- 2) **class blue**를 가진 **li**엘리멘트도 색상을 바꿔보세요.

## 2. jQuery Basic - 선택자

---

### □ CSS에서 자주 이용되는 셀렉터 - 자손 셀렉터

- 여러 개의 셀렉터를 스페이스로 구분 지어 특정 태그안에 있는 자식 태그까지 컨트롤 한다.

### [실습예제 6] ex6.html

- 1) 특정 클래스를 가진 li 엘리멘트의 **엘리멘트의 색상 바꾸기**  
예제 입니다.
- 2) class blue를 가진 li엘리멘트의 **엘리멘트중 id가 S1인 엘리멘트**  
의 색상을 바꿔보세요.

## 2. jQuery Basic - 선택자

---

### □ CSS에서 자주 이용되는 셀렉터 - 전체 셀렉터

- 전체 태그를 선택할 수 있다.

#### [실습예제 7] ex7.html

- 1) li 엘리멘트 안의 모든 자식 엘리멘트에 색상이 변경되는 예제 입니다.
- 2) 빨간색으로 변하지 않는 부분에 대해 왜 그런지 생각해 보세요.

## 2. jQuery Basic - 선택자

---

### □ CSS에서 자주 이용되는 셀렉터 - 그룹 셀렉터

- 여러 개의 셀렉터를 ,(콤마)로 구분하여 지정할 수 있다.

#### [실습예제 8] ex8.html

1) li 엘리멘트 안에 특정 엘리멘트들을 그룹핑해서 한 번에 색상을 변경하는 예제 입니다.

2) 아이디가 second, fourth 인 <li> 엘리멘트들은 폰트사이즈를 2.0em 으로 그리고 blue 색상으로 또 볼드 처리가 되게 스타일을 변경해 보세요.

## 2. jQuery Basic - 선택자

---

### □ CSS2 셀렉터 - 자식 셀렉터 ( IE6 에서 지원 안함 )

- 특정태그의 바로 아래 위치한 태그를 선택

[실습예제 9] ex9.html

1) li 엘리멘트의 자식 엘리멘트 중 **<strong>** 엘리먼트의 색상을 변경하는

예제 입니다.

2) **<div>** 엘리멘트 의 자식 **<strong>** 엘리먼트도 같은 스타일로 적용해 보세요.

## 2. jQuery Basic - 선택자

---

### □ CSS2 셀렉터 - 인접 셀렉터 ( IE6 에서 지원 안함 )

- 특정 태그의 다음에 있는 태그를 선택할 수 있다.

#### [실습예제 10] ex10.html

- 1) 세번째 <li>엘리먼트를 선택해 색상을 변경하는 예제입니다.
- 2) 네 번째 <li>엘리먼트를 선택해 색상을 변경해 보세요.

## 2. jQuery Basic - 선택자

---

### □ CSS2 셀렉터 - first-child 셀렉터 ( IE6 에서 지원 안함 )

- 특정 태그가 어떤 태그의 첫 엘리먼트인 경우 선택된다

[실습예제 11] ex11.html

- 1) 모든 첫번 째 <li>엘리먼트를 선택해 색상을 변경하는 예제입니다.

## 2. jQuery Basic - 선택자

---

### □ CSS 속성 셀렉터 – [attribute]

- 특정 속성을 가진 태그를 선택 한다.

#### [실습예제 12] ex12.html

- 1) id 속성을 가지고 있는 <li>엘리먼트를 선택해 색상을 변경하는 예제입니다.
- 2) class 속성을 가지고 있는 <li>엘리먼트를 선택해 색상을 blue로 변경해 보세요.

## 2. jQuery Basic - 선택자

---

### □ CSS 속성 셀렉터 – [attribute='value']

- 특정 속성이 특정 값을 가지고 있는 엘리먼트를 선택한다.

[실습예제 13] ex13.html

1) **title** 속성이 “second” 속성을 가지고 있는 <li>엘리먼트를 선택해 색상을 변경하는 예제입니다.

2) **title** 속성이 “fourth” 속성을 가지고 있는 <li>엘리먼트를 선택해 색상을 blue로 변경해 보세요.

## 2. jQuery Basic - 선택자

---

### □ CSS 속성 셀렉터 – [attribute!=‘value’]

- 특정 속성이 특정 값을 가지고 있지 않은 엘리먼트를 선택한다.

#### [실습예제 14] ex14.html

- 1) class가 “blue”가 아닌 속성을 가지고 있는 <li>엘리먼트를 선택해 색상을 변경하는 예제입니다.
- 2) class가 “normal”이 아닌 <li>엘리먼트를 선택해 굵게 나오게 변경해 보세요.

## 2. jQuery Basic - 선택자

---

### □ jQuery의 자체 필터 – first filter / last filter

- 셀렉터 안에서 첫 태그를 “first 필터”, 마지막 태그를 “last 필터”로 지정

#### [실습예제 15] ex15.html

- 1) 첫번째 <li>엘리먼트를 선택해 색상을 변경하는 예제입니다.
- 2) 마지막 <li>엘리먼트를 선택해 blue 색상으로 변경해 보세요.

## 2. jQuery Basic - 선택자

---

### □ jQuery의 자체 필터 – even filter / odd filter

- 짝수 순서로 나타나는 엘리먼트는 **even** 으로 홀수 순서로 나타나는 태그는 **odd** 필터로 지정할 수 있다.

#### [실습예제 16] ex16.html

- 1) 홀수 번째 <li>엘리먼트를 선택해 색상을 변경하는 예제입니다.
- 2) 짝수 번째 <li>엘리먼트를 선택해 **blue** 색상으로 변경해 보세요.

## 2. jQuery Basic - 선택자

---

### □ jQuery의 자체 필터 – **contains** 필터 / **has** 필터

- **contains** 필터는 특정 문자열이 포함되어 있는 엘리멘트를, **has** 필터는 특정 태그가 포함되어 있는 엘리멘트를 선택한다.

#### [실습예제 17] ex17.html

1) 샘플이라는 단어가 들어간 콘텐츠를 가지고 있는 엘리먼트 와 **<strong>** 태그를 포함하고 있는 엘리멘트를 골라 색상으로 변경하는 예제입니다.

## 2. jQuery Basic – HTML / CSS 조작하기

### □ 텍스트의 변경

**.text()** : 파라미터로 문자열을 넘기면 태그안의 텍스트를 문자열로 변경한다.

[실습예제 18] ex18.html

파라미터 내용에 태그를 붙여 넘겨 보세요. “<strong>가나다라마바사아자차카타파하</strong>”

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript" src=". jquery/jquery-1.9.0.js"></script>
<script type="text/javascript">
$( function() {
    $( "#p1" ).text( "가나다라마바사아자차카타파하" );
} );
</script>
</head>
<body>
<p id="p1">이 안의 텍스트를 바꿉니다.</p>
</body>
</html>
```

## 2. jQuery Basic – HTML / CSS 조작하기

### □ 텍스트 가져오기

**.text()** : 파라미터가 없으면 태그에 포함된 텍스트를 가져온다.

#### [실습예제 19] ex19.html

<p id="p1"><strong>안녕하세요</strong></p> 로 바꾸고 결과를 확인해 보세요.

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript" src=". jquery/jquery-1.9.0.js"></script>
<script type="text/javascript">
$( function() {
    alert( $( "#p1" ).text() );
} );
</script>
</head>
<body>
<p id="p1">안녕하세요.</p>
</body>
</html>
```

## 2. jQuery Basic – HTML / CSS 조작하기

### □ HTML의 변경

**.html()** : 파라미터로 HTML문자열을 넘기면 태그안의 내용에 그 HTML이 반영

[실습예제 20] ex20.html

실습예제18 와 비교, 확인해 보세요.

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript" src=".//jquery/jquery-1.9.0.js"></script>
<script type="text/javascript">
$( function() {
    $( "#p1" ).html( "<strong>가나다라마바사아자차카타파하</strong>" );
}
);
</script>
</head>
<body>
<p id="p1">이 안의 HTML을 바꿉니다.</p>
</body>
</html>
```

## 2. jQuery Basic – HTML / CSS 조작하기

### □ HTML 가져오기

**.html()** : 파라미터가 없으면 태그에 포함된 html을 그대로 가져온다.

[실습예제 21] ex21.html

실습예제19 와 결과를 비교 확인해 보세요

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript" src=". jquery/jquery-1.9.0.js"></script>
<script type="text/javascript">
$( function() {
    alert( $( "#p1" ).html() );
} );
</script>
</head>
<body>
<p id="p1"><strong>안녕하세요. </strong></p>
</body>
</html>
```

## 2. jQuery Basic – HTML / CSS 조작하기

---

### □ HTML 삽입

- `html()`은 태그안의 내용을 전부 변경
- 기존의 태그안의 내용은 남긴 채, `HTML`을 추가 삽입할 경우
- 다음과 같은 함수를 사용

**prepend()**

**append()**

**before()**

**after()**

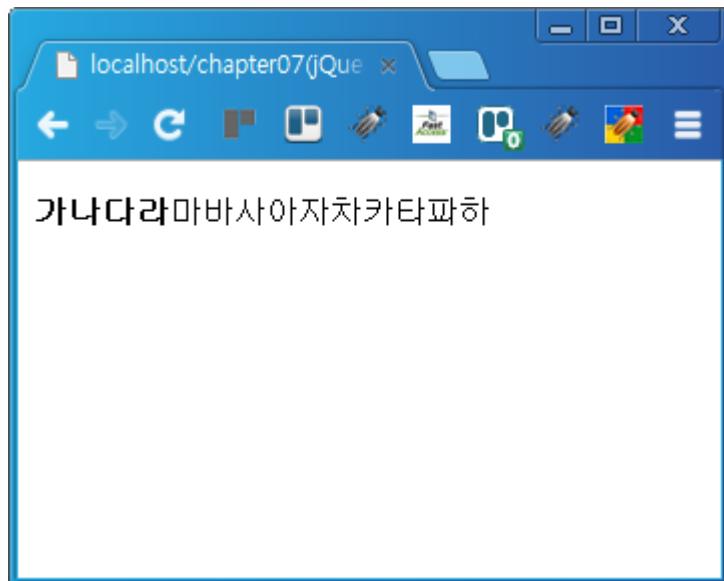
## 2. jQuery Basic – HTML / CSS 조작하기

### □ HTML 삽입 – `prepend()`

- 지정한 태그 안의 내용의 앞에 파라미터 HTML를 삽입한다.

[실습예제 22] ex22.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.



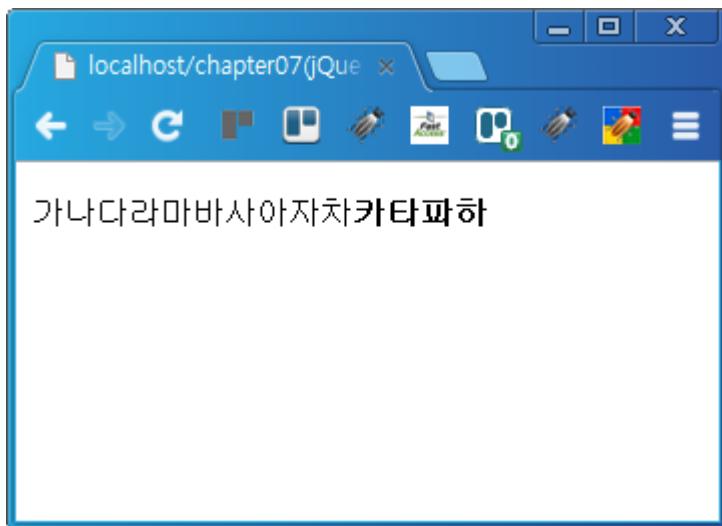
## 2. jQuery Basic – HTML / CSS 조작하기

### □ HTML 삽입 – append( )

- 지정한 태그 안의 내용의 뒤에 파라미터 HTML를 삽입한다.

[실습예제 23] ex23.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.



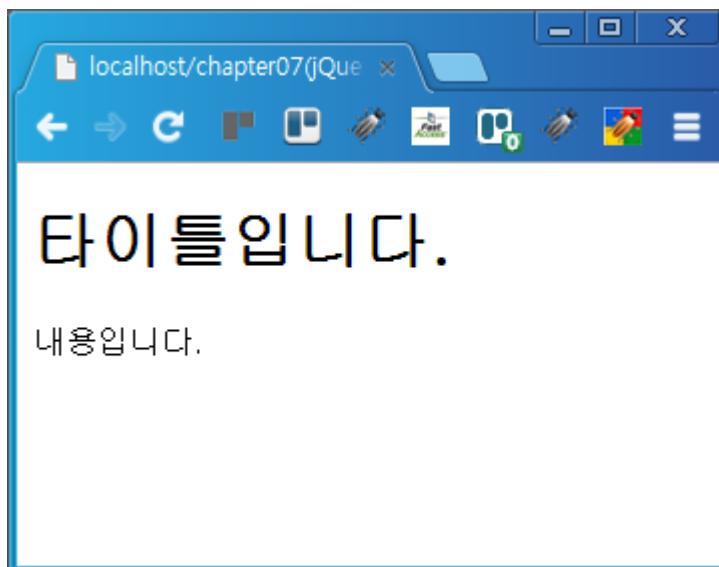
## 2. jQuery Basic – HTML / CSS 조작하기

### □ HTML 삽입 – before( )

- 지정한 태그 **앞**에 파라미터 **HTML**를 삽입한다.

[실습예제 24] ex24.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.



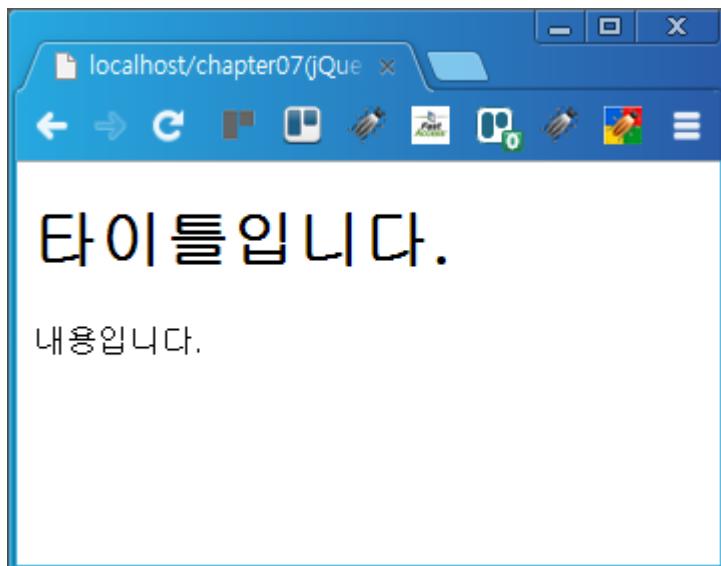
## 2. jQuery Basic – HTML / CSS 조작하기

### □ HTML 삽입 – after( )

- 지정한 태그 뒤에 파라미터 HTML를 삽입한다.

[실습예제 25] ex25.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.



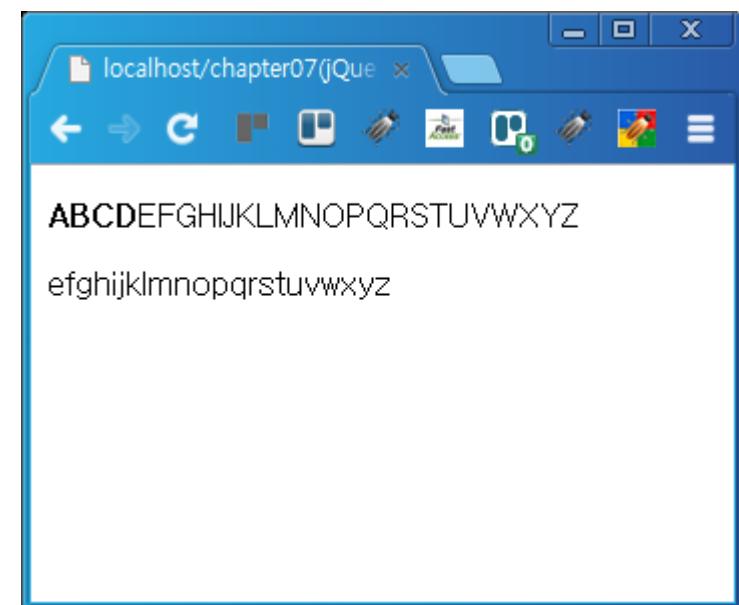
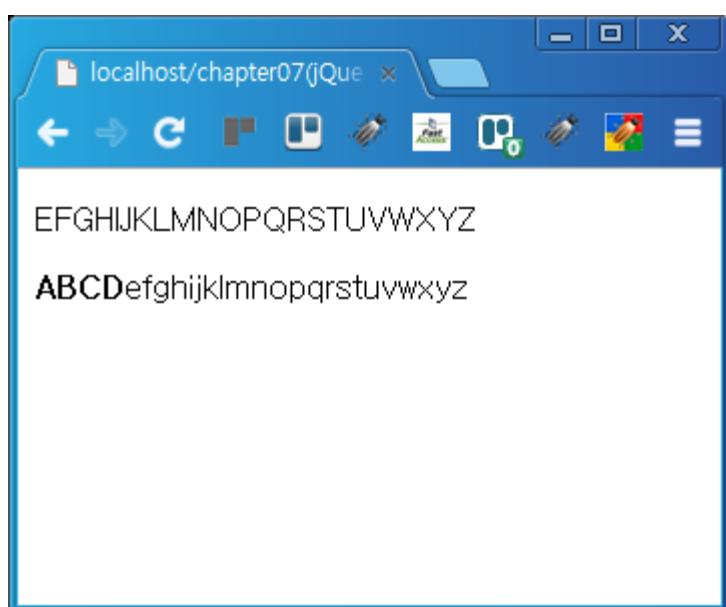
## 2. jQuery Basic – HTML / CSS 조작하기

### □ HTML 이동 – `prependTo( “이동할 곳의 선택자” )`

- 선택자로 지정한 태그를 다른 태그 안에 포함된 텍스트 앞으로 이동

[실습예제 26] ex26.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.



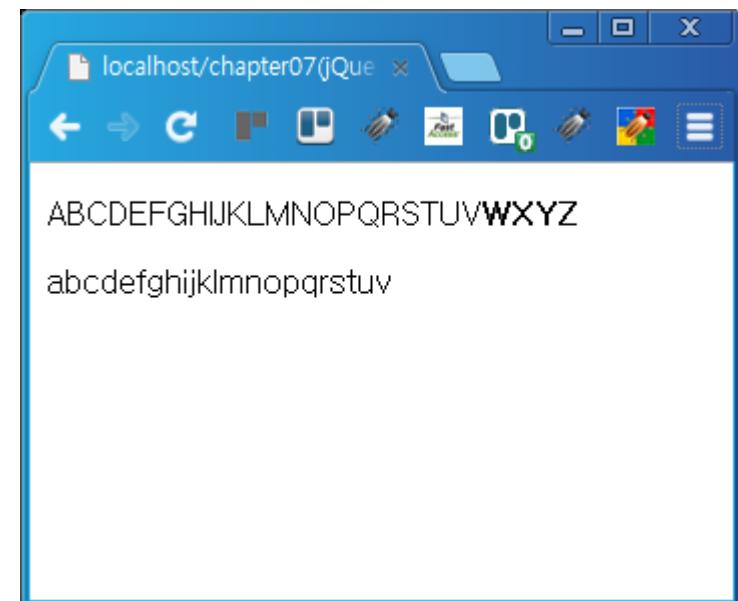
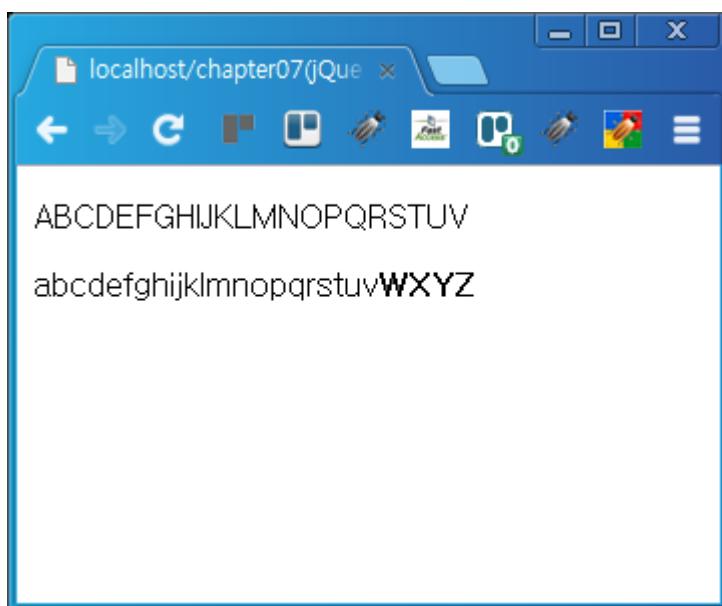
## 2. jQuery Basic – HTML / CSS 조작하기

### □ HTML 이동 – `appendTo( “이동할 곳의 선택자” )`

- 선택자로 지정한 태그를 다른 태그 안에 포함된 텍스트 뒤로 이동

[실습예제 27] ex27.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.



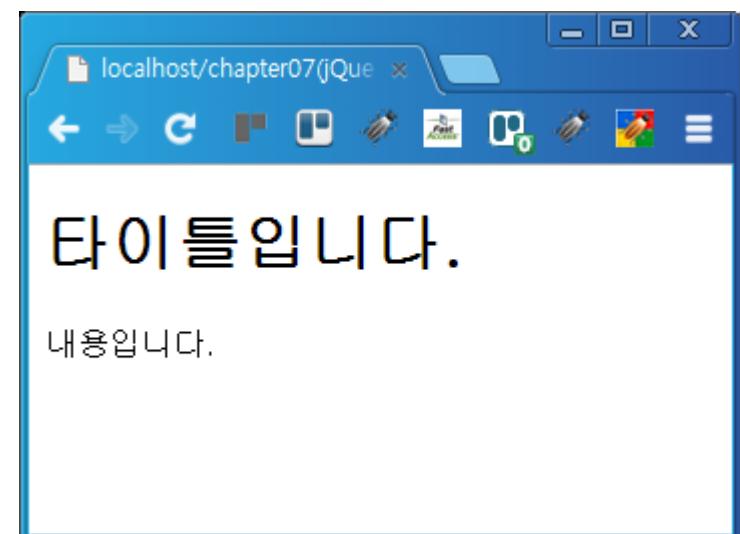
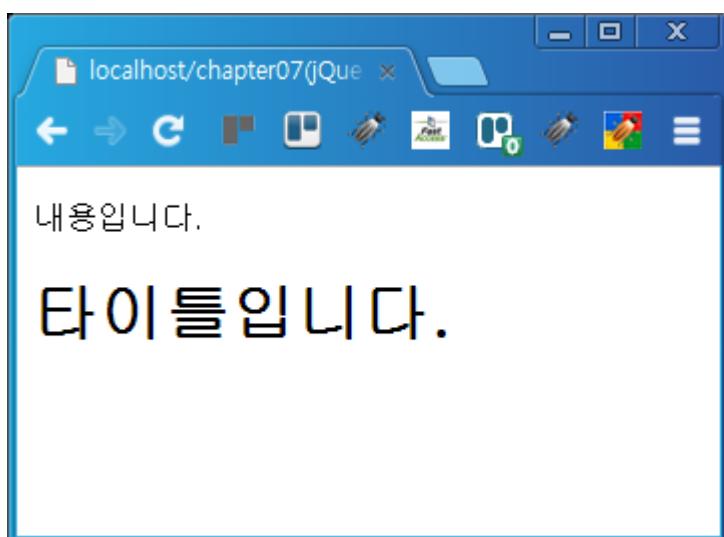
## 2. jQuery Basic – HTML / CSS 조작하기

### □ HTML 이동 – insertBefore( “이동할 곳의 선택자” )

- 선택자로 지정한 태그를 다른 태그 **앞**으로 이동

[실습예제 28] ex28.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.



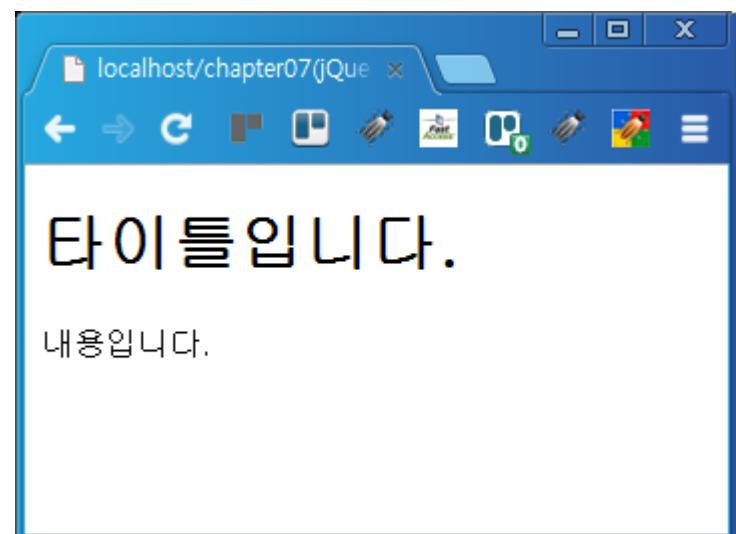
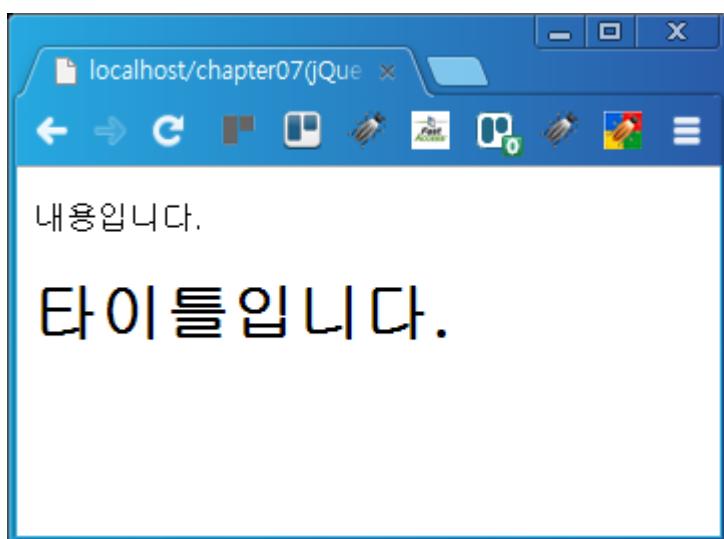
## 2. jQuery Basic – HTML / CSS 조작하기

### □ HTML 이동 – insertAfter( “이동할 곳의 선택자” )

- 선택자로 지정한 태그를 다른 태그 뒤로 이동

[실습예제 29] ex29.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.



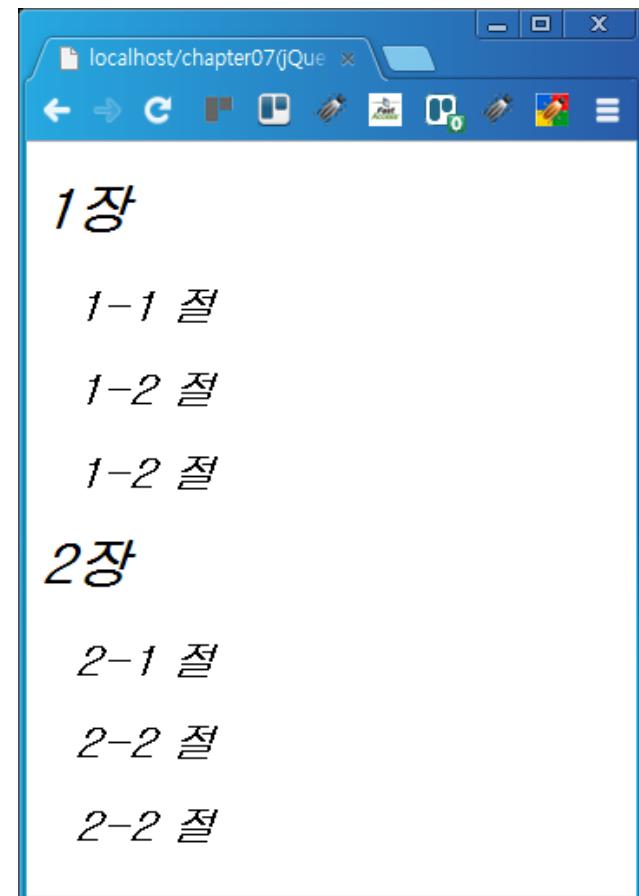
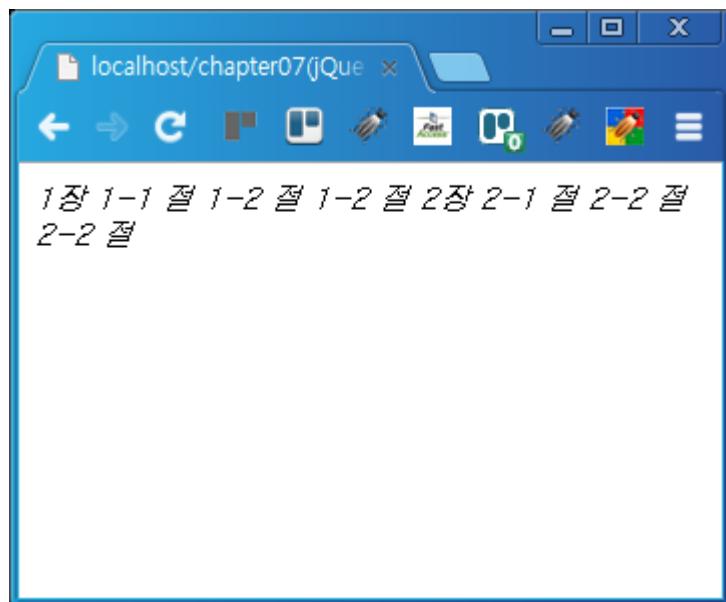
## 2. jQuery Basic – HTML / CSS 조작하기

### □ 다른 태그로 감싸기 – wrap()

- 선택자로 지정된 각각의 엘리먼트를 파라미터에 지정된 태그로 감싼다.

[실습예제 30] ex30.html

다음화면과 같은 결과가 나오도록 코드를  
완성하세요.



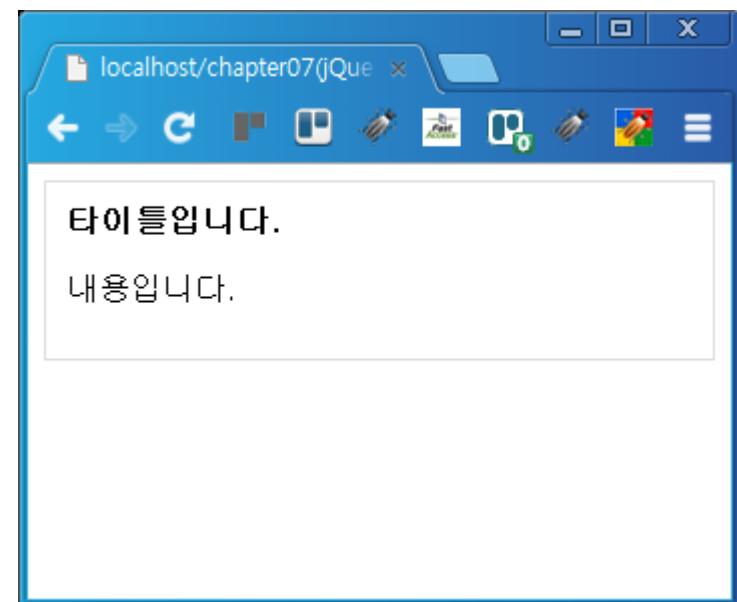
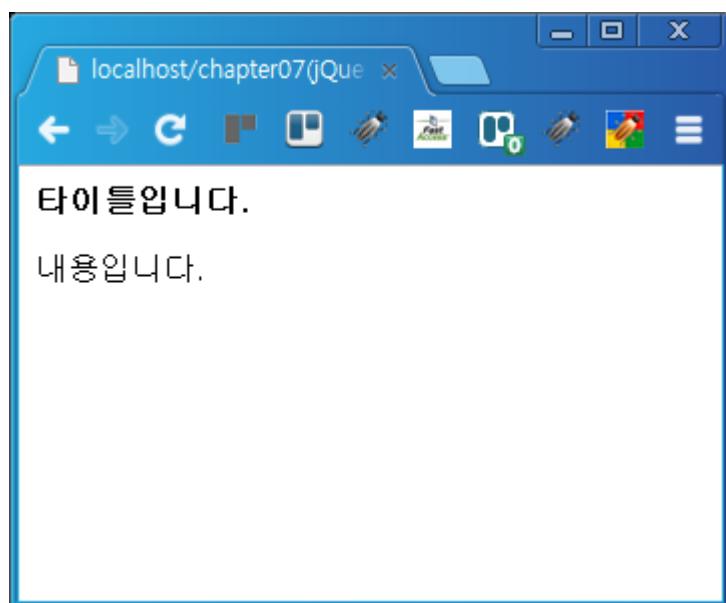
## 2. jQuery Basic – HTML / CSS 조작하기

### □ 다른 태그로 감싸기 – wrapAll()

- 선택자로 지정된 복수의 엘리먼트를 파라미터에 지정된 하나의 태그로 감싼다.

[실습예제 31] ex31.html

다음화면과 같은 결과가 나오도록 코드를  
완성하세요.



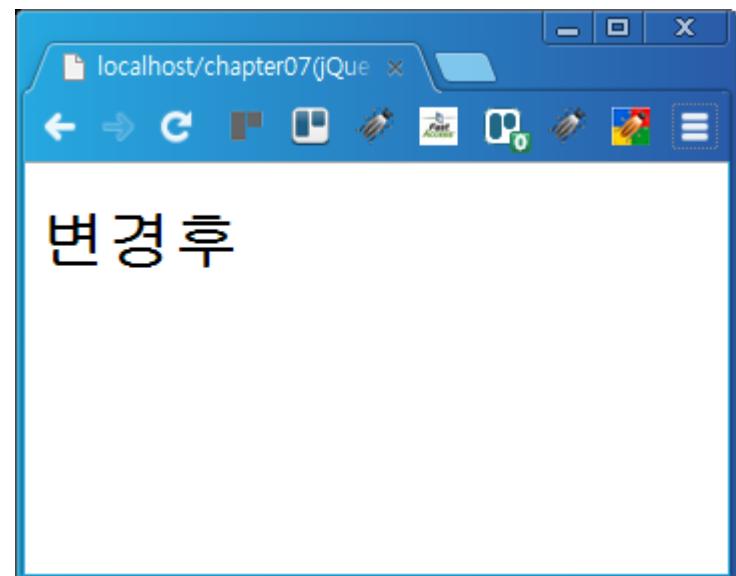
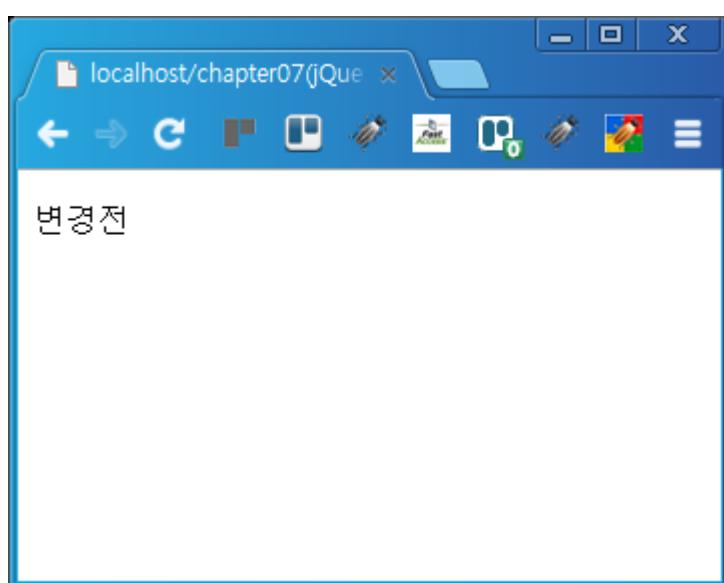
## 2. jQuery Basic – HTML / CSS 조작하기

### □ 태그 변경 – `replaceWith()`

- 지정한 태그를 다른 태그로 바꾸고자 할 때
- 엘리먼트 내용까지 변경된다. (태그만 변경하는 것이 아님)

[실습예제 32] ex32.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.



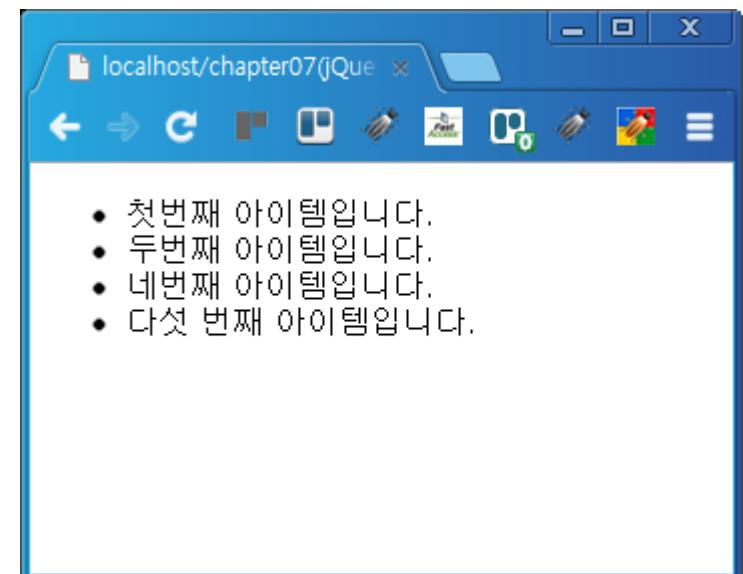
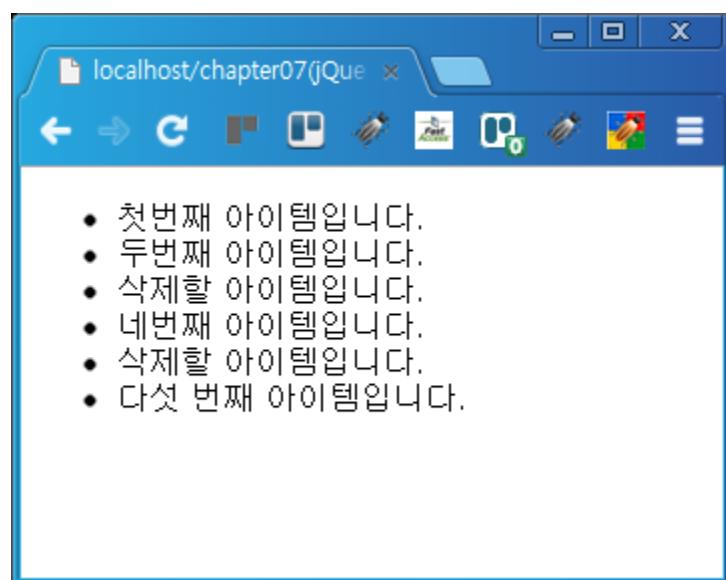
## 2. jQuery Basic – HTML / CSS 조작하기

### □ 태그 제거 – `remove()`

- 선택자로 지정된 태그들을 제거한다.

[실습예제 33] ex33.html

리스트에서 삭제할 아이템들을 삭제해 보세요.



## 2. jQuery Basic – HTML / CSS 조작하기

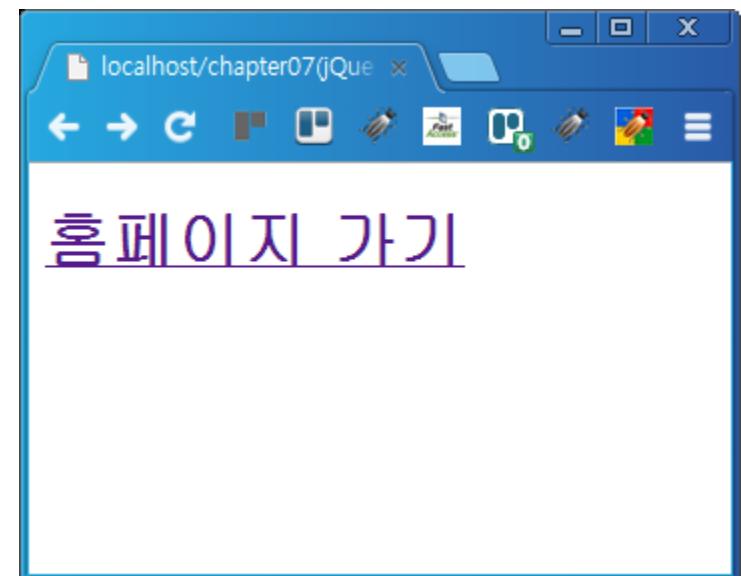
### □ 속성값 변경/가져오기 – attr()

- 태그의 속성값을 변경 할 수 있다.

```
$( “셀렉터” ).attr( “속성명”, “속성값” );
```

### [실습예제 34] ex34.html

- 1) 네이버 홈페이지로 링크를 바꿔 보세요.
- 2) 새 창에서 열리도록 target=\_blank 속성을 추가해 보세요.



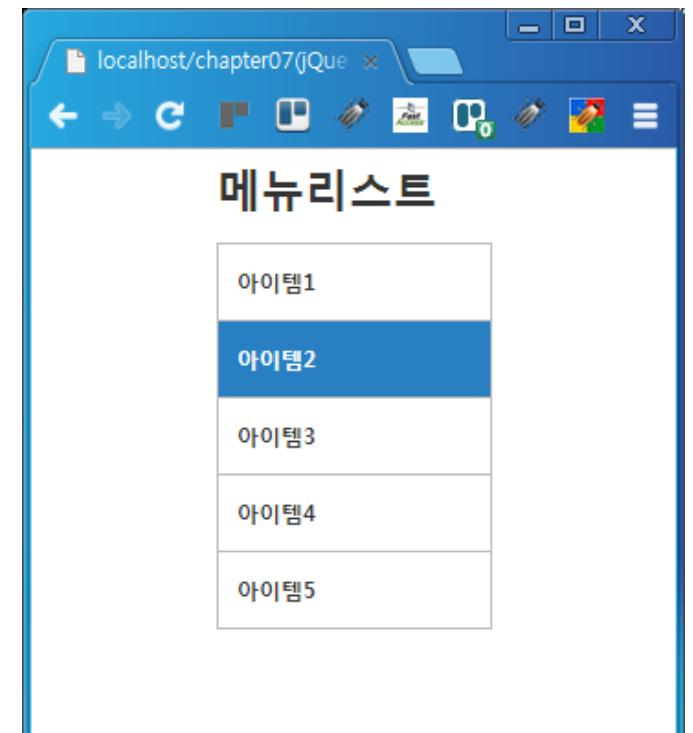
## 2. jQuery Basic – HTML / CSS 조작하기

### □ Class 속성의 추가 와 제거 – `addClass()`, `removeClass()`

```
$( “셀렉터” ).addClass( “클래스 이름”);  
$( “셀렉터” ).removeClass( “클래스 이름”);
```

#### [실습예제 35] ex35.html

- 1) ex35.html 의 CSS 살펴보기
- 2) 각 리스트 아이템에 mouseover 이벤트 받아서 선택된 class이름 추가하기
- 3) 각 리스트 아이템에 mouseout 이벤트 받아서 선택된 class이름 삭제하기



## 2. jQuery Basic – HTML / CSS 조작하기

### □ CSS 제어 – 설정

```
$( “셀렉터”).css( {  
    속성명: “속성값”,  
    속성명: “속성값”,  
    ...  
    ...  
    속성명: “속성값”  
} );
```

#### [실습예제 36] ex34.html

h1에 다음 속성값을 적용해 보세요 ( 폰트 크기 : 1.2em, 폰트 : 맑은 고딕 )  
a:link, a:visited, a:active, a:hover ( color: #333, 밑줄은 없다 )

## 2. jQuery Basic – Event

- HTML 로딩이 완료된 후 이벤트 처리가 되어야 한다.

```
$( document ).ready( function() {  
    /* HTML Element에 접근이 가능하다 */  
});
```



축약

```
$( function() {  
    /* HTML Element에 접근이 가능하다 */  
});
```

## 2. jQuery Basic – Event

### □ click 이벤트

```
$(셀렉터).click( function() {  
    /* 셀렉터로 지정한 태그가 클릭되었을 때 실행하는 처리 */  
});
```

### [ 실습예제 37 ]

그림과 같이 버튼을 누르면 8개의 이미지가 램덤하게 하나씩 화면에 나오는 프로그램을 작성하세요.

### [힌트]

```
var images = [  
    "국화:Chrysanthemum.jpg", "사막:Desert.jpg", "수국:Hydrangeas.jpg", "해파리:Jellyfish.jpg",  
    "코알라:Koala.jpg", "등대:Lighthouse.jpg", "펭귄:Penguins.jpg", "튤립:Tulips.jpg" ]
```

```
var result = Math.floor( Math.random() * ( images.length - 1 ) ) + 1;
```

## 2. jQuery Basic – Event

### □ dblclick 이벤트

```
$(셀렉터).dblclick( function() {  
    /* 셀렉터로 지정한 태그가 더블클릭 되었을 때 실행하는 처리 */  
});
```

#### [ 실습예제 38 ]

실습예제 37에서 화면에 나타난 그림을 더블 클릭하면 img 속성중 alt 속성의 내용을 alert창으로 나오게 하세요.

## 2. jQuery Basic – Event

### □ mousedown(), mouseup() 이벤트

```
$(셀렉터).mousedown( function() {  
    /* 셀렉터로 지정한 태그에 마우스 버튼이 눌렸을 때 실행하는 처리 */  
});
```

#### [ 실습예제 39 ]

실습예제 37에서 화면에 나타난 그림의 **mousedown event**에 그림이 변하는 **click** 이벤트와 동일한 반응이 되도록 처리해 보세요.

# Part III

## 자바스크립트

### 3. AJAX

1. 개요
2. 구현

## 1. 개요

---

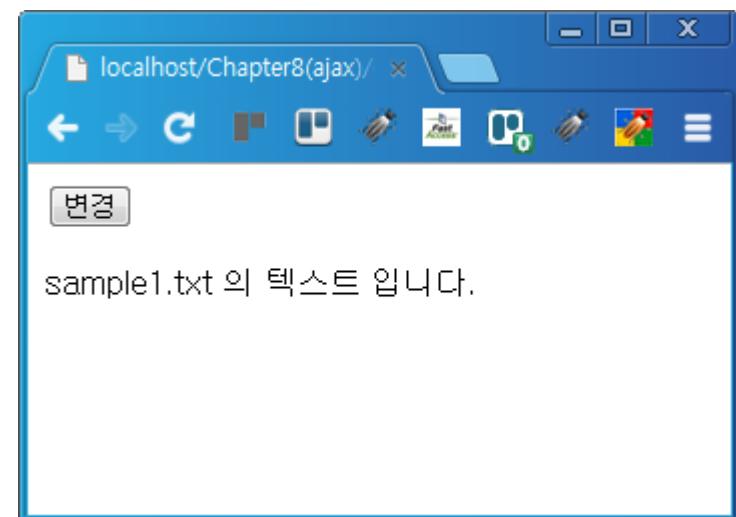
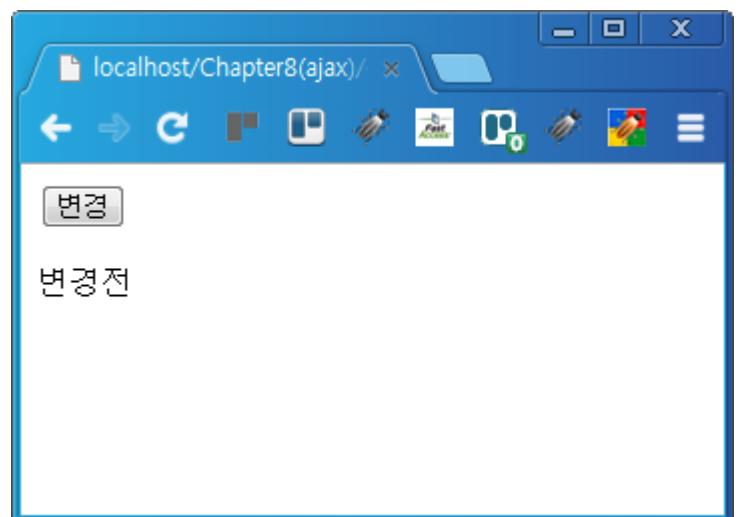
- **AJAX ( Asynchronous Javascript XML )**
- **XML**를 이용한 비동기 통신
- **JavaScript**를 이용해서 서버에서 데이터를 가져와 페이지 전체의 갱신없이 특정부분만 변경
- **XML**를 주고받은 데이터 타입으로 정의 하고 있으나 정해져 있지는 않다.
- **JSON**으로 서버와 데이터를 주로 주고 받는다.

## 2. 구현

- 웹페이지에 텍스트 삽입하기

### [실습예제 1] ex1.html

```
<script type="text/javascript">
$(function() {
    $("button").click(function() {
        $("p").load("./ex1_text.txt");
    });
}) ;
</script>
```



## 2. 구현

### □ 외부 HTML 표시

#### [실습예제 2] ex2.html

```
<script type="text/javascript">
$(function() {
    $("button").click(function() {
        $("div").load("./ex2_load.html");
    });
}</script>
```

<div>변경전</div> → <div><p>ex2\_load.html 의 p태그 입니다.</p></div>

## 2. 구현

### □ JSON 으로 Servlet과 통신

#### [실습예제 2] ex3.html

```
$("button").click(function() {  
    $.ajax( {  
        url : "ex3Json?rnd=" + Math.floor(Math.random() * 999999999),  
        type: "get",  
        dataType: "json",  
        data: "",  
        contentType: 'application/json',  
        success: function(data) {  
            alert( data.name );  
            alert( data.message );  
        },  
        error: function( jqXHR, status, e ){  
            alert( status + " : " + e );  
        }  
    } );  
});
```

## 2. 구현

### □ JSON 으로 Servlet과 통신

#### [실습예제 2] ex3JsonServlet.java

```
response.setContentType( "application/json; charset=utf-8" ) ;

PrintWriter out = response.getWriter() ;
out.print( "{ \"name\": \"test\", \"message\": \"hello\" }" );
```

### JSON Object

```
{
    name: "test",
    message: "hello"
}
```

## 2. 구현

---

### [실습과제]

회원가입에서 아이디 중복체크

1) UserDao 에서는 파라미터로 넘어온 email이 User table에

존재하는지 유무를 확인하고 servlet 에서는 다음과 같은  
JSON 을 브라우저에게 보냅니다.

```
{ result : "exist" } , { result : "not exist" }
```

2) Javascript 에서는 jQuery Ajax 를 사용해 서버에게 파라미터로 email을 보내고 서버에서 보내온 결과 JSON를 적절히 분석해 화면에 나타냅니다.