<HTML5 강좌 리스트>

Contents

[HTML5 강좌 및 동영상 목록]

[HTML5 강좌] 1. HTML5 개요

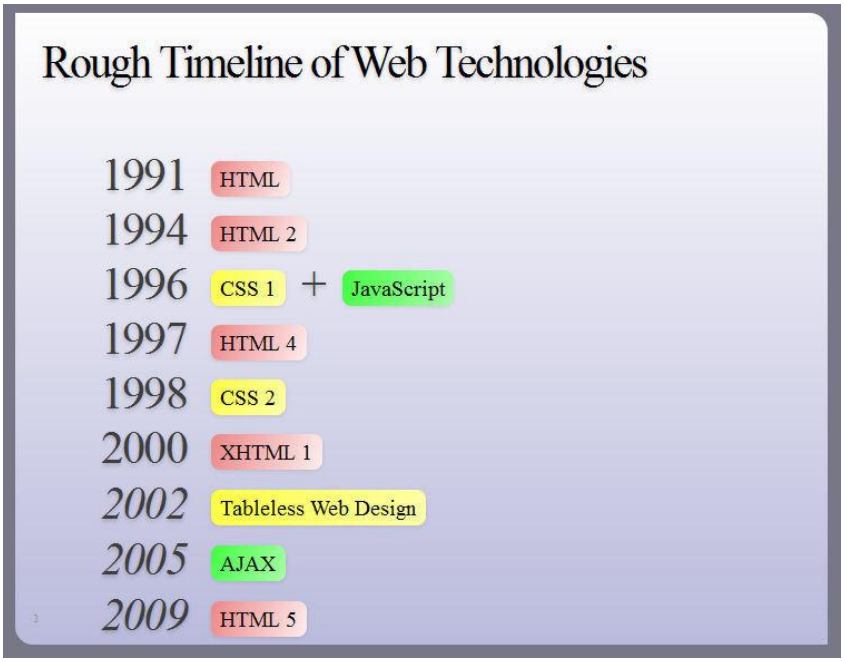
"HTML 5 is really the second coming of this Web stuff — of the Web"

- **Dion Almaer**
(co-founder of the Ajaxian Web site and co-director of developer tools at Mozilla)

<HTML 의 역사>

HTML 5 을 정리하면서, HTML 역사에 대해서도 살펴보려한다.

- 1991 년에 시작해서 현재까지의 HTML.



이미지 참고 : http://www.scoroncocolo.com/Html5.html

1991 년부터 1997 년까지 HTML 은 빠르게 발전했다. CSS 가 포함이 되었고, JavaScript 가 추가되었다. HTML 4.0 이후에도 W3C 에 의해서 HTML 은 HTML 4.01, XHTML 1.0, 1.1, 2.0 등으로 진화시키기 위해서 많은 노력을 해왔다.

그 과정에서 W3C 는 HTML 의 Version Up 을 중단하고 XHTML 로 개발의 방향을 변경하였다. XHTML 2.0 은 하위호환성을 고려하진 않는 새로운 언어로 디자인하기 시작했다.

그런 XHTML 2.0 은 W3C 의 이상과 현실의 차이로 인해서 XHTML 은 브라우저들에게서 외면당해졌고, W3C 와는 생각이 다른, 몇개의 브라우져회사가 모여 WHATWG 라는 Working Group 을 설립하여, 웹 개발현실을 반영하면서도 하위버젼의 HTML 과도 호환되는 발전된 HTML 을 정의하기 시작했다.

WHATWG 를 인정하지 않던 단호한 W3C 도 XHTML 2.0 을 포기하고 2009 년 10 월, WHATWG 를 인정하였으며, 그들과 방향성을 공유하기 시작했다.
그리고, "HTML 5" 라는 이름을 가지고 새로운 HTML 을 디자인하기 시작했다.

<HTML 의 한계>

HTML, 참... 오랜 기간동안 사용된 언어임에 틀림없다.
그리고 HTML 4 에 대한 그때의 지식으로 오늘날까지도 HTML 을 읽고 있는 것을 보면 변화 없이 멈추어버린 언어라고 생각이 될 수도 있겠지만 HTML 의 버전업이 없다고 웹이 발전하지 않았다는 말은 아닐 것이다.
본래 HTML 이 처음 만들어졌을 때의 목적은 인터넷을 통해 문서를 보기 위함이었다.
자신이 가지고 있는 문서들을 웹을 통해서 볼 수 있도록 문서 형태로 표현할 수 있는 그 수단, 즉 Language 가 필요했다.

Dos 의 640K 메모리 장벽이 그러했고, Y2K 밀레니엄 버그가 그러했듯이, HTML 또한 그 당시에는 지금의 웹을 상상도 할 수 없었으리라 생각된다.
단순히 인터넷을 통해 문서를 보기 원했던 것을 뛰어 넘어, 화려하고 인터랙티브한 화면 흐름과 멀티미디어, 예측할 수 없는 사용자들의 요구사항들을 충족시키기 위해 우리들은 HTML 에 갖가지 기술들을 만들어 내고, 포함시키고, 또 그러한 기능들을 사용할 수 있도록 하기위해 사용자들에게 "OK" 버튼을 눌러야만 사용할 수 있다는 협박으로 클릭을 강요하고, 언제, 어떻게 일어날지도 모르는 재앙에 따르는 책임을 사용자들에게 전가시켜왔다.

웹은 그렇게 HTML 버전의 발전은 없었지만 HTML 의 한계를 벗어나기위한 방향으로 발전되어 왔다.
하지만 그렇게 한계를 벗어나기 위한 발전이 반복되면서 한계의 극복은 더욱 불완전한 웹환경으로 몰아가기 시작했다.
세상은 이런 불완전함을 깨뜨리고 순수하고 완전한 존재를 원하게 되었다.

그래서 나타난 것.
"HTML 5" 다.

<HTML 5 의 등장>

그렇게 HTML 은 모든 불완전함을 "순수"라는 이름으로 깨뜨리기 위해 탄생하게 되었다.
HTML 의 역사를 살펴봐서 알수 있듯이 기존의 HTML 은 단순 Markup Language 로서 한계가 명확했으며, 그런 HTML 의 한계는 HTML 5 의 등장으로 기존에 가졌던 HTML 의 기능적 한계를 커버할 수 있다.
발전된 CSS3, JavaScript API 를 통한 기능의 확장과 결합을 통해서 기존의 Markup language 이상의 의미를 가지게 되었다.
HTML 의 한계로 인해서 유행처럼 번졌던 플러그인과 엑티브엑스 컨트롤의 문제들로 인해 웹은 순수한 HTML 만으로는 표현이 불가능한, 너무도 많은 외부기술에 의존하게 되었다.

이로 인해 웹의 접근성은 현저하게 떨어지는 상황이 되었고 HTML 5 의 등장은 플러그인과 엑티브엑스기술에서 웹을 해방시키고 웹의 접근성과 상호운영성 또한 높이게 하는 결과를 가져오게 될 것이다.
JavaScript API 를 이용한 HTML 의 확장은 서버의 부하를 줄이고 그 부하를 클라이언트와 나눔과 함께 Application 으로써의 웹의활용도를 극대화 시킬것이다.

그 결과, HTML 5 는 웹에서 플래쉬, 실버라이트와 같은 플러그인들의 사용을 줄일 것이다.(플러그인의 사용을 아예 없애지는 못할것이라 생각한다. 사용량을 0 - Zero 로 만들려 하지도 않을 것이고. 그러한 플러그인은 플러그인이 반드시 필요한 영역에서 활발히 활동할 것이라 생각한다.)
플러그인과 엑티브엑스의 설치로 인한 취약해진 보안도 걱정하지 않아도 될 것이다.
순수한 웹환경은 현재의 Device 는 물론이고 아직 세상에 있지도 않는, 미래에 나오게될 Device 들의 플랫폼이 될 것이다.
마이크로소프트, 구글을 위시한 기업들이 각 사의 브라우져를 Web OS 로서의 기반 플랫폼으로 구성하고 있고, 각 사들의 서비스들을 HTML 5 로 제작하여 차세대 Web 환경에서의 주도권을 갖기위해 노력하고 있다.
현재 HTML 5 의 표준이 확정되지도 않은 상황인데도 말이다...
어떤가?
HTML 5 를 시작해야하는 이유가 이 정도면 충분한가?

<HTML 5 의 디자인 원칙>
다음은 HTML 5 은 이러한 디자인 원칙을 가지고 만들어지고 있다고 한다.

- 호환성
    - 컨텐츠의 호환성
    - 이전 브라우저와의 호환성
    - 기능의 재사용
    - 이용 방법의 호환성
    - 혁신보다는 발전을 우선함
- 실용성
- 상호 운영성
- 보편적 접근성

(참고: http://www.jopenbusiness.com/mediawiki/index.php/HTML5#HTML5_.ED.91.9C.EC.A4.80.ED.99.94_.EC.9D.BC.EC.A0.95)

위 내용들을 간략하게 살펴보면 아래와 같다.

- 컨텐츠의 호환성 : HTML 5 이전 버전으로 제작한 컨텐츠가 완벽히 정상 작동하리라고는 생각하진 않지만 문제없이 이용가능 해야한다.
- 이전 브라우저와의 호환성 : 말 그대로 HTML 5 가 지원되지 않는 이전 버전의 브라우저에서도 이용가능해야 한다.
- 기능의 재사용 : 이전까지는 각 브라우저 사들이 각자 브라우져에 구현해 놓은 서로다른 기능들을 HTML 5 라는 이름아래 공통된 사양으로 책정하여 브라우저들간의 호환성을 가지게 한다.
- 이용방법의 호환성 : 기존에 사용하던 HTML Tag 의 사용법을 그대로 사용할 수 있게 한다.
- 혁신보다는 발전을 우선함 : HTML 5 이라는 Version UP 이 새로운 MarkUP Language 를 구현하는 것이 아닌 기존에 존재하던 HTML 을 개량한다.
- 실용성 : W3C 가 진행하던 이상적인 MarkUp Language 인 XHTML 2.0 와 같은 HTML 을 제작하는 것이 아닌 웹현장에서 필요로 하는 것들을 중점적으로 진행해야한다.
- 상호운영성 : HTML 5 를 적용한 브라우저라면 동일하게 동작해야한다.
- 보편적 접근성 : 컨텐츠의 소비자가 기계(Device, Search Engine) 와 모든 사람들(장애자를 포함한) 이 접근할 수 있도록 한다.
http://html5test.com/index.html

<HTML5 표준화 일정>

다음은 HTML 5 표준화 일정이다.

- 2006.06 : 웹 하이퍼텍스트 워킹그룹(WHATWG) 출범
    - Web Form 2.0, Web Applications 1.0
- 2007.03 : 새로운 HTML 워킹 그룹 생성
- 2009.10 : W3C 에서 XHTML 전환 실패를 인정함
- 2011.05 : HTML5 최종 초안 (Last Call Working Draft)
- 2012.Q2 : HTML5 후보 표준안 (Candidate Recommendation) - 2 개 이상의 브라우저에서 테스트 완료
- 2014.Q1 : HTML5 제안 표준안 (Proposed Recommendation) - 브라우저 업체의 피드백 반영
- 2014.Q2 : HTML5 최종 표준안 (Recommendation)

(참고:
http://www.jopenbusiness.com/mediawiki/index.php/HTML5#HTML5_.ED.91.9C.EC.A4.80.ED.99.94_.EC.9

D.BC.EC.A0.95)
두서없는 긴글 보시느라 수고하셨습니다.

다음
HTML 4 vs HTML 5 에서 뵙겠습니다.

[HTML5 강좌] 2. HTML4 vs HTML5 (1)

HTML 4 Vs HTML 5

둘의 차이점을 알기위해서 반드시 먼저 짚고 넘어가야할 주제가 있다.

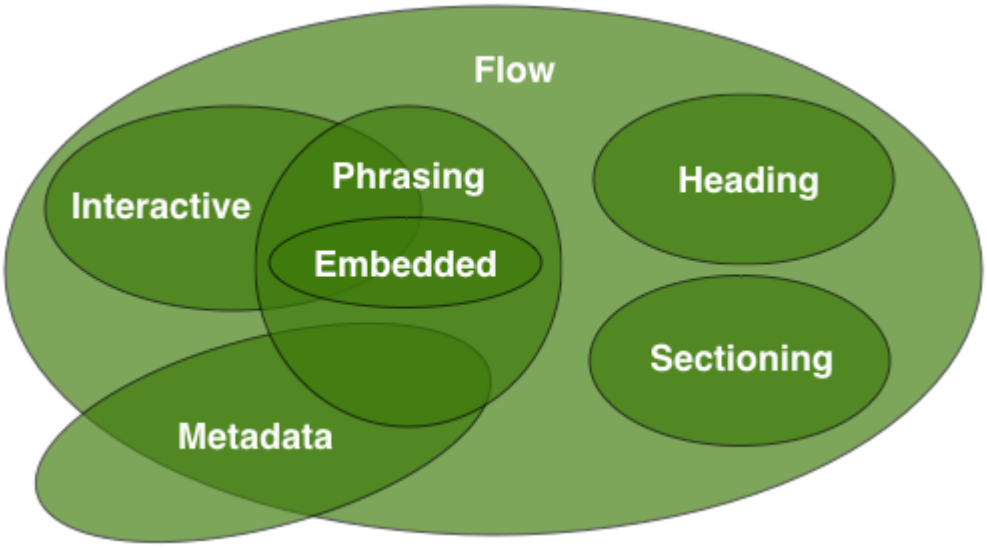
그것은...
바로 **Contents Model** 이다.

HTML 5 이전의 HTML 문서는 , <p>, <h1>, <h2>... Tag 와 같은 **Block Level Element** 와 <img>, <input>, <i>, <b> Tag 와 같은 **Inline Element** 로 구분 되어지는 Tag 들로 구성되어 두 종류의 Tag 들을 적절히 사용하여 표현하였다.
그리고 이러한 Tag 들은 Tree 형태의 계층구조를 이루어 문서의 OutLine 을 구성하고 그 구조를 따라 Contents 에 접근할 수 있다.(JavaScript 나 CSS 에서 특정 Element 에 접근하는 방식을 떠올린다면 이해가 쉬울 것입니다.)

HTML 5 문서 또한 다르지 않다.
하지만 HTML 5 는 "Contents Model" 이라는 새로운 개념을 추가하여 문서의 Outline 을 잡고 Contents 를 구성하게 된다.
HTML 5 는 기존의 Tag 에, 한마디로 HTML 5 는 구성에 중점을 둔 구조라고 할 수 있겠다.
따라서 기존 DOM Tree 계층구조에서의 Contents 접근이 아닌 구성에 따라서 접근이 가능하게 되었다.(<Header>, <Footer>, <article> Tag 를 떠올려 보세요. DOM Tree 를 따라갈 필요없이 곧바로 Contents 에 접근이 가능하다.-<Header>, <Footer>, <article> 과 같은 Tag 를 아직 모르시는 분께는 죄송하지만 곧바로 다음에서 다루겠다. 일단은 그런게 있구나 라고 알아 두세요.)
이런 이유로 HTML 5 의 Element 들은 추가되었고, 기존의 HTML 보다 훨씬 더 Sementic 한 Web 을 구성할 수 있게 되었다는데에 집중해야 할 것이다.

따라서, 이제 우리는 HTML 5 의 각 Tag(Element)가 갖는 역할과 의미를 확실히 이해하여 Content Model 에 적합한 HTML 문서를 작성할 수 있도록 한층 더 많은 생각을 해야할 것이다.

HTML 5 에서는 아래 그림과 같이 Contents 의 종류에 따라 Element 들을 분류하고 있다.



(참고 : http://www.whatwg.org/specs/web-apps/current-work/multipage/content-models.html)

Contents 는 다음과 같이 분류한다.

1. Metadata Contents
2. Flow Contents
3. Sectioning Contents
4. Heading Contents
5. Phrasing Contents
6. Embedded Contents
7. Interactive Contents

다음은 위 분류에 대해 간단히 설명해 보았다.

Category	**설명**	**Tag**
Metadata Content	나머지 Contents 의 Presentation 이나 behavior 를 설정하거나 현재문서와 다른 문서와의 관계를 설정. 또는, 기타 "Out Of Band"정보를 전달한다.	base, command, link, meta, noscript, script, style, title
Flow Content	문서와 Application 의 Body 에서 사용되는 대부분의 Element 들은 Flow Content 로 분류된다.	a, abbr, address, area (map 요소의 자손인 경우) , article, aside, audio, b, bdi, bdo, blockquote, br, button, canvas, cite, code, command, datalist, del, details, dfn, div, dl, em, embed, fieldset, figure, footer, form,

		h1, h2, h3, h4, h5, h6, header, hgroup, hr, i, iframe, img, input, ins, kbd, keygen, label, map, mark, math, menu, meter, nav, noscript, object, ol, output, p, pre, progress, q, ruby, s, samp, script, section, select, small, span, strong, style (scoped 속성이 있으면) , sub, sup, svg, table, textarea, time, ul, var, video, wbr, Text
Sectioning Content	Headings 와 Footers 의 범위를 정의한다.	article, aside, nav, section
Heading Content	Section 의 Header 를 정의한다.	h1, h2, h3, h4, h5, h6, hgroup
Phrasing Content	문서의 Text 이다. 또한 그 Text 를 intra-paragraph Level 로 Markup 하는 Element 이다.	a (구문 컨텐츠만을 포함하는 경우) , abbr, area (map 요소의 자손인 경우) , audio, b, bdi, bdo, br, button, canvas, cite, code, command, datalist, del (구문 컨텐츠만을 포함하는 경우) , dfn, em, embed, i, iframe, img, input, ins (구문 컨텐츠만을 포함하는 경우) , kbd, keygen, label, map (구문 컨텐츠만을 포함하는 경우) , mark, math, meter, noscript, object, output, progress, q, ruby, s, samp, script, select, small, span, strong, sub, sup, svg, textarea, time, var, video, wbr, Text
Embedded Content	문서에 다른 Resource 를 삽입하는 Content 이다.	audio, canvas, embed, iframe, img, math, object, svg, video
Interactive Content	사용자의 상호작용을 위해 특별하게 의도된 Content 이다.	a, audio (controls 속성이 있으면) , button, details, embed, iframe, img (usemap 속성이 있으면) , input (type 속성이 Hidden 상태가 "아니면") , keygen, label, menu (type 속성이 toolbar 상태면) , object (usemap 속성이 있으면), select, textarea, video (controls 속성이 있으면)

자세한 내용은 http://www.whatwg.org/specs/web-apps/current-work/multipage/content-models.html
를 참고하시길...

다음은 위 내용을 바탕으로 변화된 HTML 5 의 모습을 살펴보기로 하겠다.

<참고>
앞서가는 디자이너와 퍼블리셔를 위한 HTML5 & CSS
철저분석 HTML 5
http://www.whatwg.org/specs/web-apps/current-work/multipage/index.html#contents
http://www.clearboth.org/html5/

[HTML5 강좌] 3. HTML4 vs HTML5 (2)

HTML4 vs HTML5

이번엔 두 버전 간의 차이점을 살펴보려 한다.

W3C **친절하게도** 두 버전간의 차이점에 대한 문서를 오픈해 놓았다.
http://www.w3.org/TR/2010/WD-html5-diff-20100304/
"친절하게도" 라고 쓴 이유는 HTML 4 인 경우 HTML 에 대한 GuideLine 아주 간단히 제공했었다고 한다.

이 문서만 봐도 HTML 4 와 HTML 5 의 차이는 바로 알 수 있으실테니 한번 꼭 살펴보시길 권장합니다.

소개할 내용은 위 링크를 통해 살펴보게될 내용을 바탕으로 진행을 하도록 하겠다.

위 문서는 두 버전간의 차이점을 크게 세가지로 구분해 놓았다.

1. Syntax (구문)
2. Language (언어)
3. API

이 세가지는 다시 여러 항목으로 나뉘는 데 그 사항들은 후에 다시 자세히 다룰 예정이니 이 글에서는 간단히 살펴보고 넘어가려한다.

1. Syntax

먼저 간단한 HTML 5 예제를 보자.

```
<!DOCTYPE HTML><html>
  <head>
    <meta charset="UTF-8">
    <title>HTML 5 예제</title>
  </head>  <body>
    <!-- 아시다시피 여기서부터 Body Contents 가 들어가겠죠? -->
    <p>안녕하시렵니까? HTML 5?</p>
  </body>
</html>
```

위에서부터 한번 쭈욱 훑어보면 알 수 있으실 것입니다.

1-1. DOCTYPE

먼저 **DOCTYPE** 을 볼 수 있다.
너무도 간결하지 않습니까?
HTML 5 문서라면 **최상단에**(Enter-개행문자 도 들어가면 안된다.) **반드시** 넣어줘야한다. ^^
DOCTYPE 은 "HTML" 과 "XHTML" 두 가지가 있다.
HTML 5 문서를 작성할 때 HTML 로 작성할지, XHTML 로 작성할지 첫 부분에 선언해 준다는 것으로 HTML 문법을 따르도록 할 것인지 아니면 XML 문법을 따르게 할 것인지를 결정하는 것이라고 보면 되겠다.
참고로 HTML 5 가 따르는 HTML 문법을 "HTML5", HTML 5 가 따르는 XML 문법을 "XHTML5" 라 부른다.

위 문서는 "탁" 보면 "착" 알 수 있듯이.
HTML5 문법을 따르도록 작성할 것이라는 것을 알 수 있다.

1-2. Encoding

그 다음 보이는 것이 Charset. 바로 Encoding 에 대한 사항이다.
기존 HTML 문서에서 볼 수 있는

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
```

와는 다르게

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

무척이나 간단해진 것을 알 수 있을 것이다.

1-3. SVG, MathML

그리고 아래 글상자와 같이 HTML 문서안에 SVG 나 MathML 을 사용할 수 있다.

```
<!doctype html>
<title>SVG in text/html</title>
<p>
 A green circle:
 <svg> <circle r="50" cx="50" cy="50" fill="green"/> </svg>
</p>
```

(참고 : http://www.w3.org/TR/2010/WD-html5-diff-20100304/)

그리고 Tag 이름은 XHTML Type 으로 작성하든 HTML Type 으로 작성하던간에 대소문자를 가리지 않는다는 점 정도면 될 것 같습니다.

2. Language

이제 본격적으로 HTML 4 와 HTML 5 간의 특징들을 살펴보겠다.
새로 만들어진 Element 와 Attribute, 변경된 Element 와 Attribute, 없어져 버린 Element 와 Attribute 등으로 나눠서 알아보자.

2-1. 새롭게 추가된 Element

이번에 추가된 Element 에는 특히나 Sementic 요소들이 많이 추가되었다.
아래 표는 이번에 추가된 Sementic Element 이다.

Element	설명
Header	문서의 Header 를 나타낼 때 사용한다.
Footer	문서의 Footer 를 나타낼 때 사용한다.
Nav	문서내에 Navigation 요소가 있을 때 사용한다.
Section	문서의 영역을 구성하며, 문서 구조를 구성하는 H1~H6 와 함께 사용한다.
Article	뉴스기사나 블로그 article 과 같은 독립된 Contents 를 표시할 때 사용한다.
Aside	주요 컨텐츠 이외의 참고가 될 수 있는 컨텐츠를 구성할때 사용한다.
Hgroup	각 Section 의 Header 를 나타낼때 사용한다.
Figure	그림, 비디오와 같은 포함된 컨텐츠의 Caption 을 표시할때 사용한다.
Figcaption	캡션에 사용한다.

새로운 엘리먼트는 어떻게 사용할까?

간단히 그림을 그려봤다...

이러한 새로운 Sementic Element 들로 인해서 기존의 Div Tag 들로 나뉘어 졌단 막연하고 무의미했던 Contents 들이 한층더 유용한 정보들로 탈바꿈하게 되었다.
또 이런 변화는 우리들에게 Contents 구성에 대한 커다란 숙제를 안겨주고 있다.

이 밖에 몇가지 Element 가 추가되었다.

Element	**설명**
Audio, Video	HTML 5 Element 중 관심이 집중되는 Element 중 하나다. 멀티미디어 컨텐츠를 표시하는 데 사용한다.
Embed	Plugin 컨텐츠를 표시할 때 사용한다.
Mark	별도로 표시한 문자를 표시하는데 사용한다.
Progress	작업 진행상황을 나타낼 때 사용한다.
Meter	측정값을 표시할 때 사용한다.
Time	날짜, 시간을 표시할 때 사용한다.
Ruby, rt, rp	Ruby 언어를 나타낼 때 사용한다.
Canvas	Bitmap Graphic 을 표시할 때 사용한다.
Command	사용자가 호출할 수 있는 명령어를 표시하는데 사용한다.
details	사용자 요청에 따라 얻은 콘트롤이나 추가적인 정보를 표시하는데 사용한다.
Datalist	List Attribute 와 함께 사용하여 ComboBox 를 만들때 사용한다.
Keygen	키쌍(Key pair)을 생성할 때 사용한다.
Output	스크립트 계산 결과 같은 실행결과를 표시하는데 사용한다.

2-2. 새롭게 추가된 Attribute

새로 추가된 속성들 중에 몇 가지만 소개하려 한다.

Attribute	**설명**
contenteditable	편집가능한 Area 임을 나타낸다.
contextmenu	작성자가 제작한 Context Menu 지정하는데 사용할 수 있다.
data-*	접두사 "data-" 를 가진 속성으로 추후 HTML 버전 충돌없이 사용자 태그로 이용하거나 추후 브라우져가 지원하게 되었을때 사용할 수 있다.
draggable	새로운 Drag & Drop API 에서 사용할 수 있다.
hidden	element 가 아직 없거나 없을 때 사용한다.
role, aria-*	보조기능에 가리키는데 사용할 수 있다.
spellcheck	맞춤법 검사기능을 제공할지 여부를 지정한다.

2-3. 변경된 Element

Element	설명
a	href 속성이 없이 사용하면 "placeholder link" 를 나타낸다.
address	sectioning 의 새로운 개념으로 범위를 나타낸다. 어떤 Contents 부분관 관련된 연락처 정보인지 알 수 있다.
b	문자열 강조 뿐만아니라 제품소개 중 제품명 키워드 같이 특별히 중요하지는 않으나 일반적으로 강조하기 위해 사용한다.
hr	단락 수준의 주제 바꿈에 사용한다.
i	기존 기능처럼 문자의 기울임 뿐만이 아니라 음성, 분위기, 분류명, 기술용어, 다른언어의 숙어구, 생각, 선박명 등을 표현할 때 사용한다.
label	기본 플랫폼 User interface 표준이 아닐 경우 Label 에서 Control 로 Focus 를 이동시키면 안된다.
menu	툴바와 Context Menu 용으로 사용한다.
small	세부 주석과 법적 인쇄물 같은 작은 인쇄를 나타낼때 사용한다.
strong	기존 Strong 의 기능보다 더 중요한 것을 표시할때 사용한다.

2-4 변경된 Attribute

아래 속성은 사용하지 않기를 권장하며 꼭 필요한 곳에서만 사용하도록 한다.

Attribute	설명
img 의 border	border 값은 0 일때만 사용하고 CSS 를 사용할 수 있다.
a 의 name	name 속성을 id 속성으로 바꾸어 쓸 수있다.
table 의 summary	다른 대안을 정의하고 있다.
script 의 language	language 값은 JavaScript 에만 사용하고 type 속성과 함께 쓰지 않고 생략할 수있다.

2-5. 제거된 Element

없어진 이유를 보면 크게 세가지로 나뉜다.

1) CSS 로 처리할 수 있는 Element

2) Frame 관련 Element

3) 거의 사용되지 않는 Element

2-5-1. CSS 로 처리할 수 있는 Element

다음 Element 는 순전히 보여지는 것에만 사용되는 Element 들이다.

basefont, big, center, font, s, strike, tt, u

2-5-2. Frame 관련 Element

다음 Element 는 Frame 과 관련된 Element 로 이들 Element 의 사용은 사용성과 접근성에 부정적인 영향을 끼치 때문이다.
frame, frameset, noframes

2-5-3. 거의 사용되지 않는 Element

다음 Element 는 자주 사용되지 않고, 혼란스럽게하거나 다른 Element 로 기능을 대체할 수 있기 때문이다.
acronym, applet, isindex, dir

2-6. 제거된 Attribute

나열하기에 많아서 좀 애매하다...
http://www.w3.org/TR/2010/WD-html5-diff-20100304/#absent-attributes
문서에서 3.6 Absent Attributes 를 참고 하시길...

3. API

이젠 새로 사용할 수 있게된 API 들을 설명하려 한다.

- **Video/Audio API** : Video, Audio Element 와 함께 Video, Audio 를 재생하는데 사용수 있다.
- **Offline Web Application** : 말 그대로 Offline 을 지원하는 API 이다.

Web Application 이 특정 **프로토콜 또는 Media Type 을 등록**할 수 있는 API
새로운 Grobal Attribute 인 "Contenteditable" Attribute 와 함께 사용하는 **편집 API**
"draggable" Attribute 와 함께 사용하는 **Drag & Drop API**
Navigation 시 사용하는 **History 정보를 노출하는 API**

그리고 그 사항들은 아래와 같은 사항들을 확장하였다..

3-1 HTMLDocumnet 확장사항들

DOM Level2 HTML 의 HTMLDocument interface 를 확장했다.
- **getElementsByClassName()** : Class Name 으로 Element 를 선택할 수 있다.
- **innerHTML** : HTML 또는 XML 문서를 분석하고 직렬화하는 쉬운 방법이다.
- **activeElement, hasFocus** : Element 가 현재 Focus 되어 있는지, "Document" 가 Focus 되어있는지 여부를 확인한다.
- **getSelection()** : 현재 선택되어있는 Object 를 반환한다.

3-2 HTMLElenemt 확장사항들

HTML 5 의 HTMLElement interface 는 몇 가지 확장사항들이 있다.

- getElementsByClassName()

- innerHTML :

- classList 는 ClassName 에 편리하게 접근할 수 있는 접근자다. Element 의 Class 들을 조작하기 위해서 has(), add(), remove(), toggle() 과 같은 함수를 노출한다.

a, area, link Element 는 relList 라는 유사한 Attribute 를 가지고 있으며 rel Attribute 에 같은 기능을 제공한다.

HTML 4 와 HTML 5 간의 차이점에 대한 어느 정도 필요한 정보들을 간단히 나열해 보았다.

4. HTML 5 작성 Tip

4-1. 이전 Browser 에서 새로운 요소들 사용.

HTML 5 에서 사용하는 Tag 들은 이전 버전의 브라우져에서는 지원하지 않는다.

하지만 이전 Browser 에게 약간의 편법을 쓰면 HTML 5 Tag 들을 지원하는 것처럼 하게 할 수 있다.

아래 Javascript 를 추가해보자.

```
<script type="text/javascript">
        document.createElement('address');
        document.createElement('article');
        document.createElement('aside');
        document.createElement('figure');
        document.createElement('footer');
        document.createElement('header');
        document.createElement('hgroup');
        document.createElement('menu);
        document.createElement('nav');
        document.createElement('section ');
</script>
```

그리고 아래 구문을 CSS 에 추가에 추가한다.

이로써 아래 Tag 들에 대해서 Block 요소로 인식하게 해준다.

```
address, article, aside, figure, footer, header, hgroup, menu, nav, section { display:block;}
```

4-2. 사용하는 Browser 의 HTML 5 기능 지원여부 확인

현재 사용하는 Browser 에 HTML 5 기능이 적용되어 있는지 알아보기 위해서 다음 페이지를 통해서 확인하기 바란다.

http://a.deveria.com/caniuse/

현재 브라우져가 특정 기능을 지원하는지 Code 상에서 확인하기위해 필요한 Script 라이브러리가 있다. 그 것은 Modernizr (http://www.modernizr.com) 이다.

아래내용은 간단히 GeoLocation 기능을 지원하는 지를 확인하는 JavaScript 코드이다.

```
if (Modernizr.geolocation){
  navigator.geolocation.getCurrentPosition(function(position) {
    // pass the lat and long values to an application
    // e.g. a setUserLatandLong() function may find the closest bodega
    setUserLatandLong(position.coords.latitude,position.coords.longitude);
  });
}
```

(참고 : http://www.modernizr.com)

이와 같이 HTML 5 JavaScript API 기능의 지원 여부를 손쉽게 확인할 수 있도록 도와준다.

4-3. 구문 검증

다음은 우리가 작성한 HTML 문서를 검증하기 위한 Site 를 소개한다.

http://validator.w3.org

http://html5.validator.nu

http://gsnedders.html5.org/outliner/ (이 Site 는 Contents Model 에 기반을 두어 작성한 HTML 5 문서의 Outline 을 검증합니다.)

두편의 Article 을 통해 이전 버전의 HTML 과 HTML 5 와의 차이점과 변경된 사항들을 간단히 살펴보았다.

지금까지 나열된 사항들은 추후 이어질 아티클에서 다시 조금더 자세한 내용들로 언급이 될 것이다.

[HTML5 강좌] 4. Sementic Element (1)

이번 주제는 Sementic Element 이다.
HTML 4 vs HTML 5 (1), (2) 주제의 Contents Model 과 Language 에서 언급되었던 이야기를 조금 더 자세히, 특히 아웃라인인과 과 관련된 엘리먼트를 좀 더 살펴보려고 한다.

아래 내용들은 본인의 생각도 물론 많이 포함되어 있지만

http://www.w3.org/TR/2011/WD-html5-20110525/

웹혁명을 꿈꾸다 HTML5 & API 입문
웹표준 가이드 HTML5 CSS3

앞서가는 디자이너와 퍼블리셔를 위한 HTML5 & CSS
철저해설 HTML 5
구글개발자가 들려주는 HTML5 활용

등을 참고하고 재구성 했음을 미리 말씀드립니다.

```
<!DOCTYPE HTML>
<html lang="ko">
 <head>
  <meta charset="UTF-8">
 <title>HTML 5 예제</title>
 </head>  <body>
  <!-- 아시다시피 여기서부터 Body Contents 가 들어가겠죠? -->
  <p>안녕하시렵니까? HTML 5?</p>
 </body>
</html>
```

위 예제가 생각나시는지 모르겠다.
^^
위 요소들을 몇 가지 살펴보려한다.

1. 문서타입과 Root 요소

1-1. 문서타입.

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

이 문서타입은 Microsoft 가 Mac 용 internet explorer 5 를 개발하면서 예전에 만든 HTML 페이지들이 제대로 표시되지 않는 심각한 문제가 발생하여, 이를 해결하기 위해서 문서의 상단에 HTML DocType 을 추가하였다.
Mac 용 Internet explorer 의 경우 이 DocType 을 확인하도록 하여 이 DocType 이 포함되지 않은 기존 HTML 페이지는 옛날방식으로 렌더링하도록 하였다.
그래서 새 명세를 적용하려면 HTML Element 앞에 적절한 Doc Type 만 지정하면 됐다.

그런데 아이디어가 남용되면서 명세를 최대한 지키는 표준모드와 표준을 따르지 않는 비표준모드로 나뉘고, 또 몇몇 요소에서 표준을 따르지 않는 준 표준모드까지 나오게 되었다.

위 DocType 은 여러 표준 Doc type 중 하나다 나머지는 이곳(http://www.w3.org/QA/2002/04/valid-dtd-list.html) 에서 확인할수 있다.

여러분이 생각해도 정리를 해야할 것 같다는 생각이 들지 않는가?
그래서 확~! 줄여서 나온 DocType !!!

```
<!DOCTYPE HTML>
```

인 것이다.

그리고 모든 브라우져가 표준모드로 작동할 것이다.
한가지 주의할 점은 맨 첫줄에 써주어야 한다는 것이다. 빈줄이 있으면 안된다는 것.
그렇지 않으면 비표준모드로 랜할 수 있다.

그리고 또 한가지...
기존에 HTML 4.01 이나 XHTML 1.0, 1.1 로 작성한 페이지를 HTML 5 로 변환하는 작업의 시작 이라는 것!

1-2. Root Element

HTML 페이지의 루트 엘리먼트인 <HTML> 는 기존에 XHTML 에서 사용하는 namespace 가 포함되어 있었는데 이제 HTML 5 에서는 Lang 속성만 있다.(manifast 속성은 추후에 다시 알아보기로 하자...)
그래서...
<html lang="ko"> 만 적으시면 된다.

1-3. <head> Element

head Element 는 변화 요소가 거의 없는데 문서의 Metadata 를 담는 Element 이며 Title Element 가 필수로 들어가고 Meta Element, Link Element, Script Element 가 들어간다는 것은 기본적인 사항이라 모두들 알고 있을 것이다.

1-4. Meta Element

Meta Element 는 Content 속성을 사용해서 나타내는데 이번에 새로 Charset 속성을 사용하여 문자 Encoding 을 나타낼 수 있게되었습니다.
HTTP Header 에 문자 Encoding 이 포함되어 있다면 Meta Element 의 문자 Encoding 보다 우선한다는 것도 알아두시길...

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
```

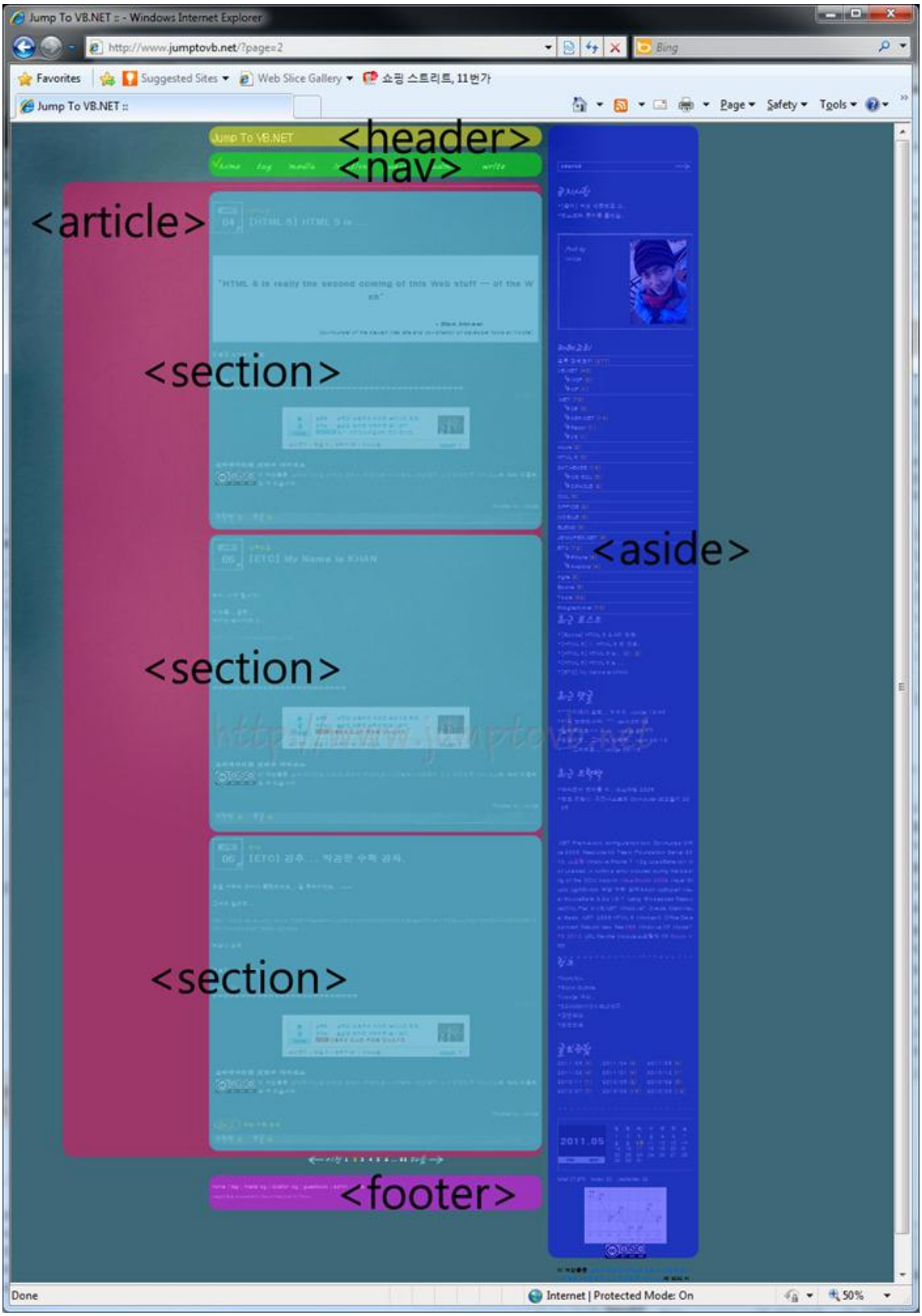
위와 같이 적긴하지만 다른 요소들은 생략하시더라도 아래와 같이 charset 을 포함한 Meta Element 를 포함하기 바란다.

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

2. 아웃라인 구성 Element 들

이번엔 이번에 추가된 아웃라인을 구성하는 Element 들을 살펴보려한다.
이전 글에서 소개했듯이 HTML 5 에 문서의 구조화와 관련된 요소들이 많이 추가되었다.

아래와 그림과 같이 간단히 나타낼 수 있는데 하나하나 중요하게 생각해야 하는 사항들만 짚어 보도록 하자.


Jump To VB.NET :: - Windows Internet Explorer
http://www.jumptovb.net/?page=2
Favorites
Suggested Sites
Web Slice Gallery
Jump To VB.NET ::
Page
Safety
Tools
Jump To VB.NET
<header>
<nav>
<article>
<section>
<aside>
<section>
<section>
<footer>
Done
Internet | Protected Mode: On
50%

2-1. Section Element

Outline 을 구성하는 Element 들을 설명하기 위해서는 먼저 Section Element 를 알아봐야 할 것이라 생각한다.

Section Element 는 일반적인 Section 을 나타낸다.

그런데 의미 자체로만 보게되면 기존에 사용하던 Div Element 와 사용하기에 혼동하기 쉽다.

Section Element 는 장이나 절 단위를 나타낸다고 생각하면 된다. 스타일을 적용하기위해서 Section Element 를 사용해서는 안되며 그런 용도라면 Div Element 를 사용하도록 해야한다.

Section Element 는 장이나 절을 나타내므로 제목과 본문을 하나로 묶을 때 사용합니다.

만약에 아래와 같이 제목과 본문에 스타일을 적용하고 싶다거나 구분을 지으려 한다면 Div Element 를 사용해야 할 것이다.


```
<section class="title">
        <h1>제목</h1>
        <section class="content">
                <p>본문</p>
        </section>
</section>
```

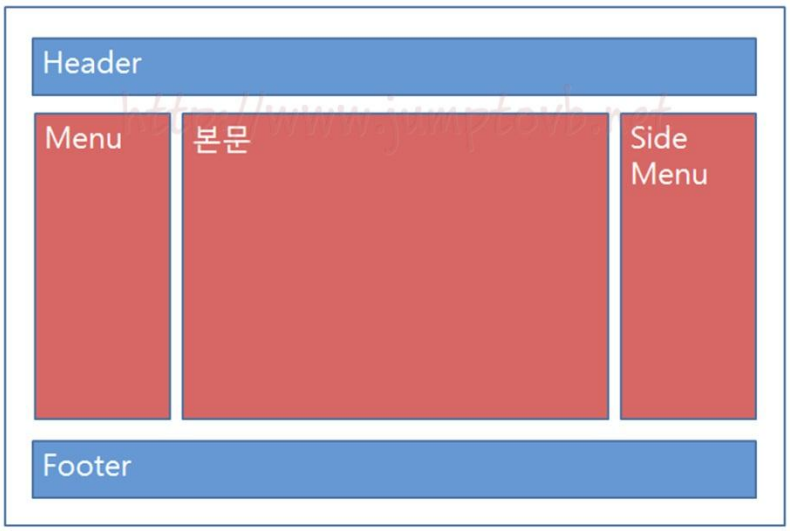
위와 같이 Element 를 구성하게되면 제목과 본문을 다른 Section 으로 해석하게 될 것이다.

본문의 Level 이 하나 더 내려가게 되겠죠?

그래서 위그림과 같은 구조에서는 아래와 같이 Mark up 해야 할 것이다.

```
<section class="title">
        <h1>제목</h1>
        <div class="content">
                <p>본문</p>
        </div>
</section>
```

또한 아래와 같은 구성에서도 각각을 모두 Section Element 로 나누는 것이 아닌 HTML 5 의 OutLine 구성 Element 들을 적절히 사용하여 구성해야 할 것이다.




```
<body>
        <header>제목</header>
        <div class="left">
                <nav>...
                </nav>
        </div>
        <div>
                <article>...
                </article>
        </div>
        <div>
                <aside>....
                </aside>
        </div>
        <footer>....</footer>
</body>
```

이와 같이 말이다...

2-2. nav Element

nav Element 는 Navigation Section 을 나타냅니다. 페이지상의 Header 나 좌측, 우측에 표현하는 Link 모음이고 간혹 Footer 에도 표현하기도 하지요.

모든 Link 에 nav Element 를 사용하면 안되겠고, 다시한번 이야기 하지만 스타일을 적용하기 위해서 nav Element 를 사용하면 안될 것이다.

또 한가지!! Site Map Page 를 nav Element 로 표현해서도 안될 것이다. Site Map 의 Link 는 nav 특성보다는 Main Contents 이기 때문이다.

주로 nav Element 내에 ul Element 를 사용하여 목록으로 Navigation Link 를 Markup 한다.

이번엔 nav Element 를 그룹화하는 방법을 몇 가지 알아보죠.

위 이미지는 제 블로그의 우측에 있는 링크들이다.

붉은 부분만 떼어놓고 본다면 아래와 같이 Markup 할 수 있을 것이다.

```
<nav>
        <h3>카테고리</h3>
        <ul>
                <li><a href="...">분류전체보기</a></li>
                <li><a href="...">VB.NET</a></li>
                ...
        </ul>
</nav>
<nav>
        <h3>글보관함</h3>
        <ul>
                <li><a href=">2011/05(8)</a></li>
                <li><a href=">2011/04(4)</a></li>
                ...
        </ul>
</nav>
```

한 가지 더 생각해 보자면 위 그림상에 있는 컨텐츠의 내용은 aside element 에 속하는 것으로 아래와 같이 nav Element 들을 aside Element 내에 포함시킬 수 있을 것이다.

```
<aside>
        <nav>
                <h3>카테고리</h3>
                <ul>
                        <li><a href="...">분류전체보기</a></li>
                        <li><a href="...">VB.NET</a></li>
                        ...
                </ul>
        </nav>
        <nav>
                <h3>글보관함</h3>
                <ul>
                        <li><a href=">2011/05(8)</a></li>
                        <li><a href=">2011/04(4)</a></li>
                        ...
                </ul>
        </nav>
</aside>
```

이렇게 말이다.

[HTML5 강좌] 5. Sementic Element (2)

2-3. article Element

article Element 는 블로그에 포스트된 글들, 뉴스사이트의 기사들의 묶음 같은 독립된 하나의 컨텐츠를 나타내며, 게시판 글이나 블로그 포스트의 댓글들, 위젯, 가젯영역 또한 article Element 의 영역이라 할 수 있겠다.

아래 그림을 참고하자.

각 포스트들은 Section 으로 나뉘고 Section 들을 하나의 article Element 로 감싸고 있는 모습을 볼 수 있다. 단순히 페이지의 메인 컨텐츠라고 해서 전체를 article Element 로 감싸서는 안되므로 해당 컨텐츠가 영역의 성격에 적합한지 항상 다시 한번 체크해 보아야 할 것이다.

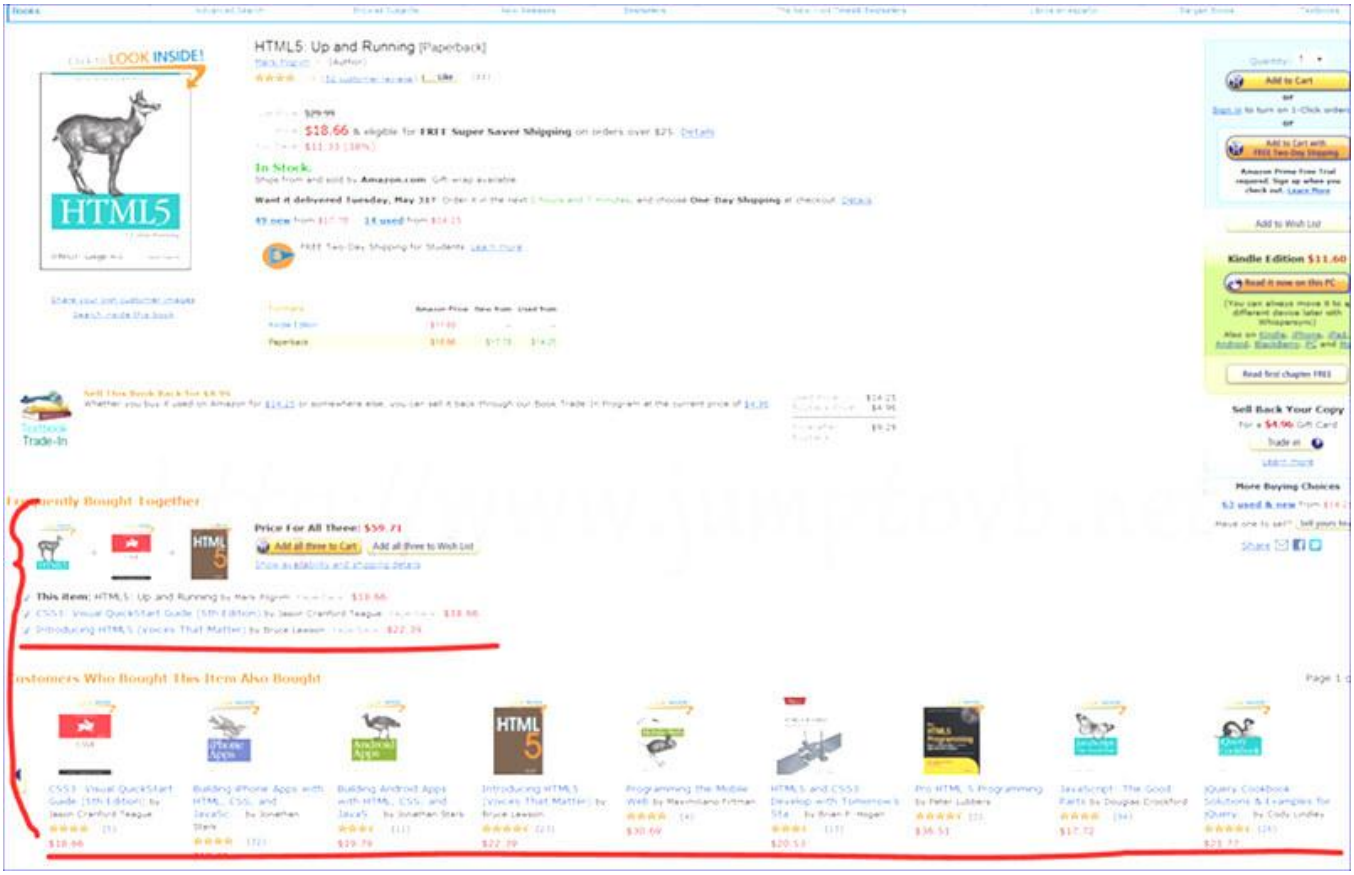
2-4. aside Element

aside Element 는 보충기사, 사이드바, 광고 와 같이 Main 컨텐츠와는 관련이 적어 분리될 수 있다고 생각되는 것에 사용한다.

또한 이전 글의 nav Element 에서 살펴봤듯이 nav Element 들을 Group 화 할때도 사용할 수 있다.

컨텐츠에 직접적으로 관련이 되어 있어서 만약 삭제되면 컨텐츠내용이 구성되지 않는 내용이 있다면 그 부분은 aside Element 를 사용하면 안된다.

위 이미지의 경우는 main Content 로서 내용을 설명하고 있으므로 aside Element 를 사용하면 안되겠지만. 위 그림의 aside 부분이나 아래 그림의 도서쇼핑몰의 추천 도서과 같은 내용은 Main Contents 와는 직접적인 관계는 없으므로 aside Element 를 사용하기에 적합하다고 할 수 있다.



aside Element 의 경우 컨텐츠의 내용과 제작자의 의도에 따라서 다른 Element 로 사용할 수 있으므로 Outline 구성 Element 중에서 조심스럽게 사용해야하는 Element 중 하나다.

2-5. hgroup Element

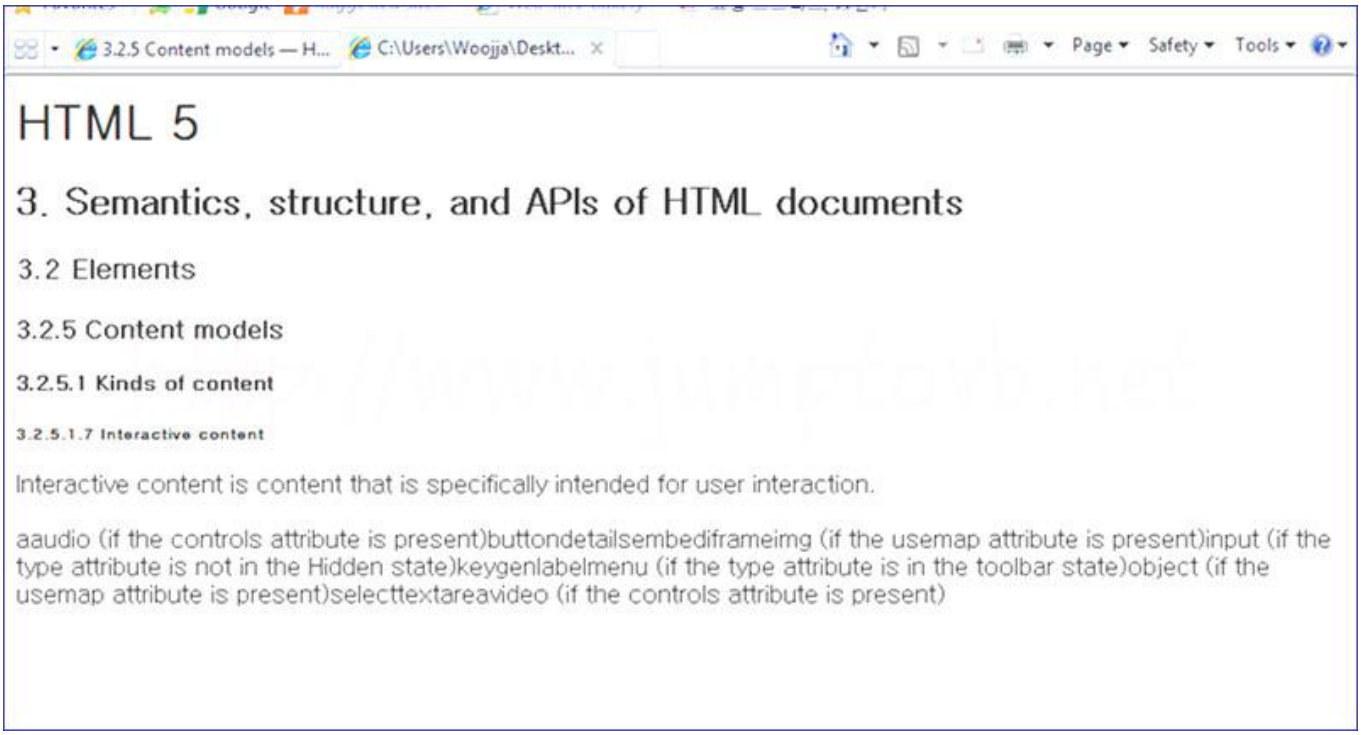
hgroup Element 는 제목과 이에 따르는 소제목이나 부제, Catchphrase 를 그룹화하기 위해 사용하는데, h1~h6 Element 만 포함할 수 있다.

HTML 5 의 hgroup Element 를 사용하기 이전, HTML 4.01 에서 아래와 같은 index 를 가진 Page 를 제작하기 위해서는

간단히 만든다하면...

```
<h1>HTML 5</h1>
<h2>3. Semantics, structure, and APIs of HTML documents</h2>
<h3>3.2 Elements</h3>
<h4>3.2.5 Content models</h4>
<h5>3.2.5.1 Kinds of content</h5>
<h6>3.2.5.1.7 Interactive content</h6>
<p>
Interactive content is content that is specifically intended for user interaction.</br></br> aaudio (if the controls attribute is present)buttondetailsembediframeimg (if the usemap attribute is present)input (if the type attribute is not in the Hidden state)keygenlabelmenu (if the type attribute is in the toolbar state)object (if the usemap attribute is present) selecttextareavideo (if the controls attribute is present)
</p>
```

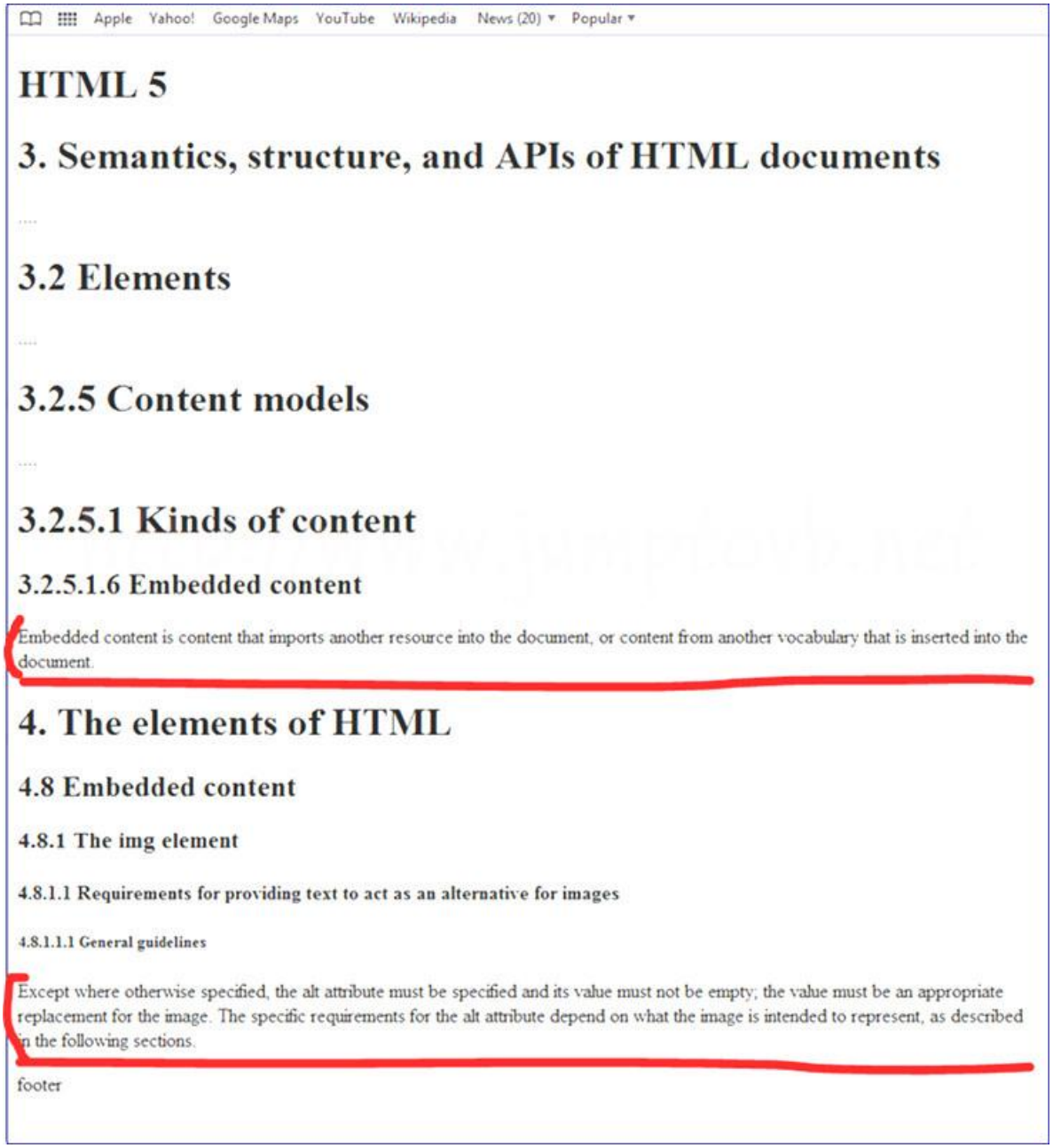
아래 같이 표현이 될 것이다.



그런데, 만약에 한 Depth 더 내려가야한다면 어떻게 표현해야 할까?

HTML 4.01 까지는 한계가 있었다.
HTML 5 부터는 Outline 구성 Element 에는 HeadLine Element Level 의 높고 낮음에 따라 Outline 이 결정되며 H1 이나 H2 와 같은 높은 Level 의 Headline Element 들도 이제는 각 Section 이나 Article 같은 Outline 구성 Element 어디에서나 사용할 수 있게되었다.

그럼, 이번엔 다음 이미지를 보자



이 HTML 문서의 붉은 색 줄
Headline Tag 다음 P Tag 를 이용해서 Headline 을 설명해주고 있다.
겉으로 보기엔 차이가 나지 않는다.
하지만 아래 Mark UP 을 보면 둘의 차이를 알 수 있는데 hgroup 의 용도를 알 수있을 것이라 생각한다.

```
<!DOCTYPE HTML>
<html lang="ko">
<HEAD>
<meta charset="UTF-8">
 <title>HTML 5 예제</title>
</HTAD>
<body>
 <header>
  <h1>HTML 5</h1>
 </header>
 <section>
  <h1>3. Semantics, structure, and APIs of HTML documents</h1>
  <p>....</p>
  <section>
   <h1>3.2 Elements</h1>
   <p>....</p>
   <section>
    <h1>3.2.5 Content models</h1>
    <p>....</p>
    <section>
     <h1>3.2.5.1 Kinds of content</h1>
     <h2>3.2.5.1.6 Embedded content</h2>
     <p>
      Embedded content is content that imports another resource into the document, or content from another vocabulary that is inserted into the document.
     </p>
    </section>
   </section>
  </section>
  <section>
    <hgroup>
     <h1>4. The elements of HTML</h1>
     <h2>4.8 Embedded content</h2>
     <h3>4.8.1 The img element</h3>
     <h4>4.8.1.1 Requirements for providing text to act as an alternative for images</h4>
     <h5>4.8.1.1.1 General guidelines</h5>
    </hgroup>
    <p>
     Except where otherwise specified, the alt attribute must be specified and its value must not be empty; the value must be an appropriate replacement for the image. The specific requirements for the alt attribute depend on what the image is intended to represent, as described in the following sections.      </p>
   </section>
 </section>
 <footer>
 footer
 </footer>
</body>
</html>
```

위 파란색 Section 의 <P> Tag 는 H2 Headline 의 내용임을 나타낸다.
하지만 붉은색 Section 의 <P> Tag 는 Hgroup 에 해당되는 내용임을 나타낸다. 이와 같이 Headline 을 여러개로 묶을 필요성이 있을때 Hgroup Element 를 사용하게 된다.

아래 내용의 경우 위 Headline 을 Hgroup 으로 묶을 수 있을 것이다.


'혹독한 5월' 두산, 이성열 역전 결승타에 웃었다
이성열 3타점 맹활약...한화에 6-3 역전승, 연패 탈출 성공
11.05.30 08:19 | 최종 업데이트 11.05.30 08:19
이용재 (kirinbeer)
TAG 두산 베어스, 홍상삼, 이성열 태그등록
혹독한 5월을 보내고 있는 두산 베어스가 이성열의 짜릿한 역전 결승타에 힘입어 연패 탈출에 성공했다.
두산은 29일 잠실구장에서 벌어진 프로야구 한화와의 홈경기에서 6-3으로 승리해 4연패 사슬을 끊고 분위기 전환의 발판을 마련했다. 이성열은 결승타를 포함해 4타수 3안타 3타점을 올리며 맹타를 휘둘러 승리의 주역이 되었다.

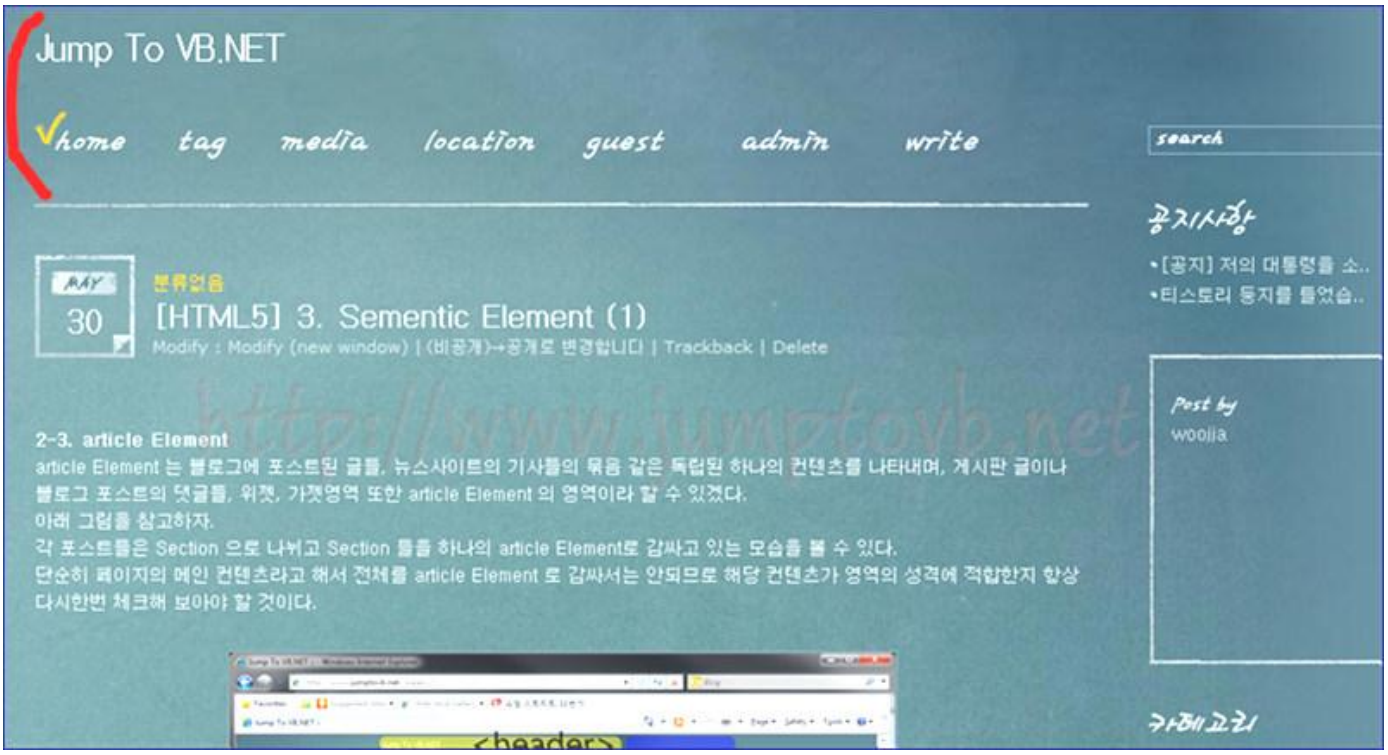
2-6. Header, Footer Element

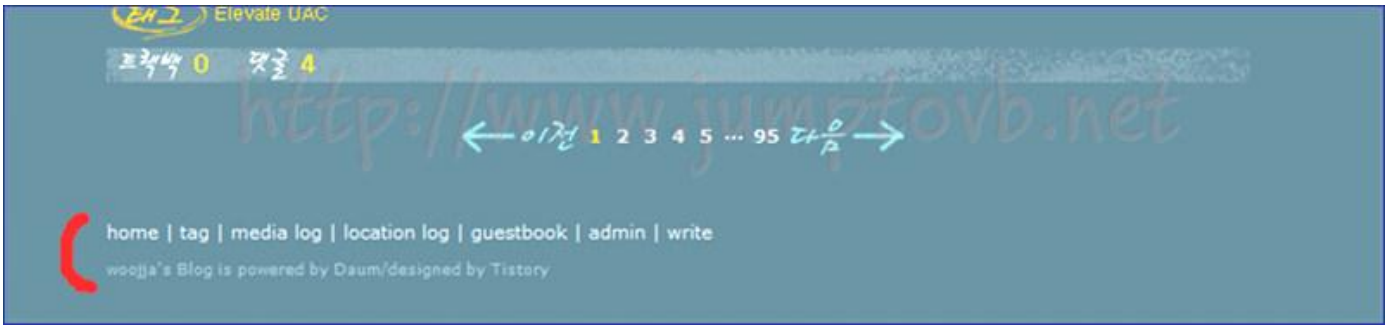
Header Element 는 Section 의 Header 를 나타낸다.
Header Element 에는 HeadLine Element 나 hgroup, nav Element 를 넣는다.
Footer Element 는 Section 이나 Page 의 Footer 를 표현한다.
Section 의 저자정보나, 관련링크, 저작권표기등을 표현하는데 사용한다.
몇번을 사용가능하지만, 사용하는 위치에 따라서 각각의 의미가 달라진다.
page 의 Body Tag 내의 Footer 는 Page 의 Footer, 블로그 포스트등의 article 내의 Footer 는 블로그 포스트에 관한 Footer 정보가 된다.
Footer Element 는 주로 페이지나 Section 의 마지막에만 사용하는 것은 아니다.

아래는 스타일시트가 적용되어 있지는 않아 썰렁하지만, Blog Site 의 Header 와 Footer 를 표현해 보았다.

```
<!DOCTYPE HTML>
<html lang="ko">
<HEAD>
<meta charset="UTF-8">
 <title>Jump To VB.NET</title>
</HTAD>
<body>
 <header>
  <hgroup>
    <h1>Jump To VB.NET</h1>
    <h2>Have a nice day~!</h2>
  <hgroup>
  <nav>
    <ul>
      <li><a href="...">home</a></li>
      <li><a href="...">tag</a></li>
      <li><a href="...">media</a></li>
      <li><a href="...">location</a></li>
      <li><a href="...">guest</a></li>
      <li><a href="...">admin</a></li>
      <li><a href="...">write</a></li>
    </ul>
  </nav>
 </header>
 <section>
 <p>...</p>
 </section>
 <footer>
  <nav>
    <ul>
      <li><a href="...">home</a></li>
      <li><a href="...">tag</a></li>
      <li><a href="...">media log</a></li>
      <li><a href="...">location log</a></li>
      <li><a href="...">guestbook</a></li>
      <li><a href="...">admin</a></li>
      <li><a href="...">write</a></li>
    </ul>
  </nav>
  <p><small>Copyright &#169;2007-2009 woojja All Rights Reserved</small></p>
 </footer>
</body>
</html>
```


Jump To VB.NET
file:///C:/Users/Woojja/Desktop/headerfooter.html
Apple
Yahoo!
Google Maps
YouTube
Wikipedia
News (3
Jump To VB.NET
Have a nice day~!
home
tag
media
location
guest
admin
write
...
home
tag
media log
location log
guestbook
admin
write
Copyright ©2007-2009 woojja All Rights Reserved


Jump To VB.NET
home tag media location guest admin write
search
공지사항
•[공지] 저의 대통령을 소..
•티스토리 둥지를 틀었습..
MAY
30
분류없음
[HTML5] 3. Sementic Element (1)
Modify : Modify (new window) | (비공개)→공개로 변경합니다 | Trackback | Delete
2-3. article Element
article Element 는 블로그에 포스트된 글들, 뉴스사이트의 기사들의 묶음 같은 독립된 하나의 컨텐츠를 나타내며, 게시판 글이나
블로그 포스트의 댓글들, 위젯, 가젯영역 또한 article Element 의 영역이라 할 수 있겠다.
아래 그림을 참고하자.
각 포스트들은 Section 으로 나뉘고 Section 들을 하나의 article Element로 감싸고 있는 모습을 볼 수 있다.
단순히 페이지의 메인 컨텐츠라고 해서 전체를 article Element 로 감싸서는 안되므로 해당 컨텐츠가 영역의 성격에 적합한지 항상
다시한번 체크해 보아야 할 것이다.
Post by
woojja
카테고리

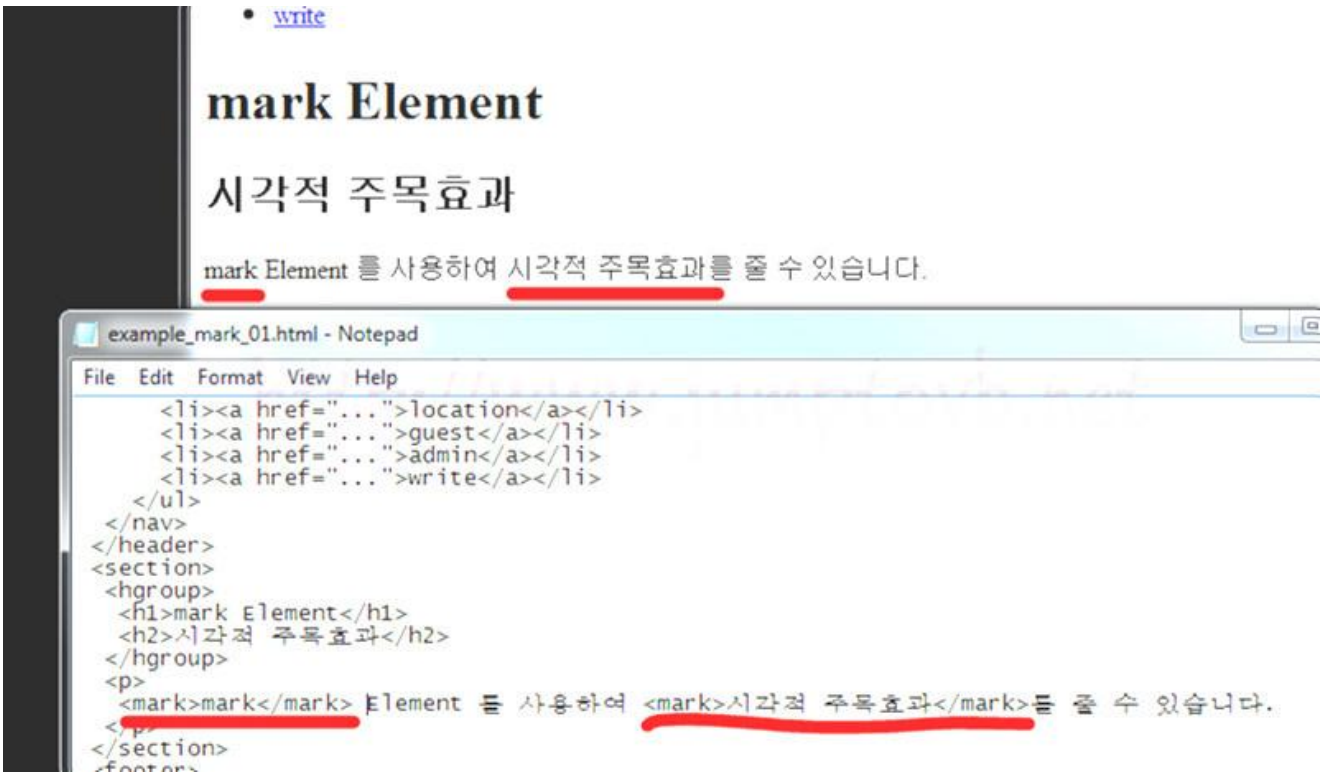

Elevate UAC
트랙백 0 댓글 4
← 이전 1 2 3 4 5 … 95 다음 →
home | tag | media log | location log | guestbook | admin | write
woojja's Blog is powered by Daum/designed by Tistory

3. Text 의미 부여 Element

Text 의미 부여 Elemnet 에는 a, em, strong, small, cite, i, b, span 등 여러 Element 들이 있다. 그 중 HTML 5 에 추가된 대표적인 mark, date Element 에 대해서 알아보도록 하겠다.

3-1. mark Element

mark Element 는 설명을 위해 특정키워드를 강조한다거나 인용문의 일부를 강조하기위해 또는 검색결과의 검색 키워드를 강조하는등의 시각적 주목효과를 Text 에 주려 한다는 것을 표현하기 위해 사용한다.



보는 바와 같이 mark Element 를 사용하여도 외관상 Browser 에는 나타나지 않는다. 외관은 CSS 를 사용하여 표현을 해야할 것이다.

3-2. Time Element

Time Element 는 Browser 나 Robot 등이 시간을 정확히 이해할 수 있어야 한다는 것을 전제로 만든 Element 라고 한다. 그러므로 애매한 시점을 나타내는 표현에 이 Element 를 사용하면 안된다.

```
<time>06:43:21</time>
<time>2011-05-31</time>
<time>2011-05-31T06:43</time>
<time>2011-05-31T06:43:21+09:00</time>
```

Time Element 표현하는 방법은 아래와 같다.
연월일 : 붙임표 ==> 2011-05-31
연월일 시각 구분:대문자 T ==> 2011-05-31T06:43
시분초 구분 : 콜론(:) ==> 2011-05-31T06:43:21
타임존 포맷 추가 : +,- 표시 ==> 2011-05-31T06:43:21+09:00

년도, 월, 월일, 일, 시, 분, 초 만을 나타낼 수는 없다.
Time Element 는 두가지 속성이 있다. datetime 과 pubdate 인데

datetime 속성의 경우 아래와 같이 사용하여

```
<time datetime="2011-05-31">05 월 31 일</time>
<time datetime="04:30">기상시간</time>
```

표현하려는 내용을 컴퓨터가 이해할 수 있는 날짜 형식으로 넣을 수 있다.

pubdate 는 말그대로 published date 다.
블로그 article 을 작성한 시간, 페이지를 작성한 시간등을 나타낼 때 사용한다.
당연히 각 Section 마다 단 한번만 사용할 수 있다.
아래와 같이 사용할 수 있을 것이다.

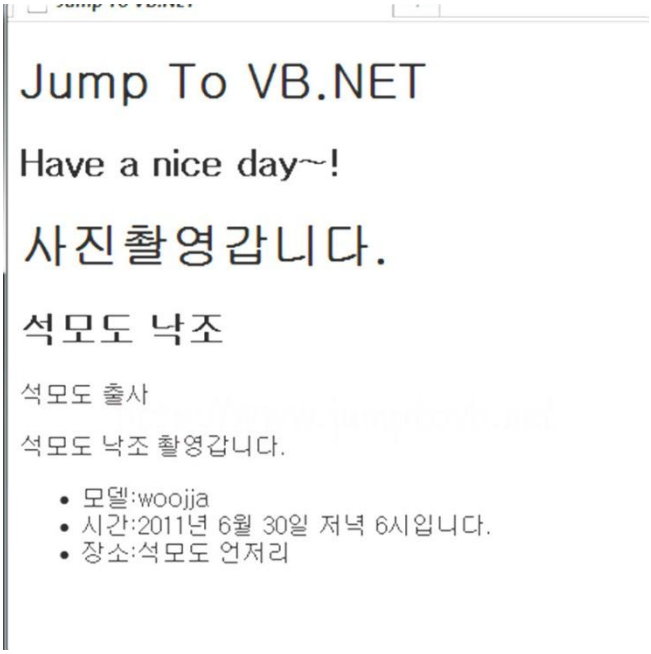
4. 상호작용 Element

4-1. Details Element

Details Element 는 사용자에게 추가적인 상세 정보를 제공할때 사용한다.
Detail Element 에는 제일 처음에 Summary Element 를 넣어 추가정보의 요약이나 설명을 제공하고
Summary Element 다음에 상세한 정보를 입력하는데 그 순서를 반대로 해서는 안된다.
반듯이 Summary Element 를 제일 처음으로 해야 한다.

또 한 가지 기억해야 할 것은 Details Element 를 보충설명이나 각주등의 용도로 사용해서는 안된다는 것이다.

```
<!DOCTYPE HTML>
<html lang="ko">
<HEAD>
<meta charset="UTF-8">
 <title>Jump To VB.NET</title>
</HTAD>
<body>
 <header>
  <hgroup>
   <h1>Jump To VB.NET</h1>
   <h2>Have a nice day~!</h2>
  <hgroup>  </header>  <section>
  <hgroup>
   <h1>사진촬영갑니다.</h1>
   <h2>석모도 낙조</h2>
  <hgroup>

   <details open="open">
    <summary>석모도 출사 정보</summary>
     <p>석모도 낙조 촬영정보 아래와 같습니다. 참고하세요.</p>
     <ul>
      <li>모델:woojja</li>
      <li>시간:<time datetime="2011-06-30T18:00:00+09:00">2011년 6월 30일 저녁 6시</time>입니다.</li>
      <li>장소:석모도 언저리</li>
     </ul>
   </details>
 </section></body>
</html>
```



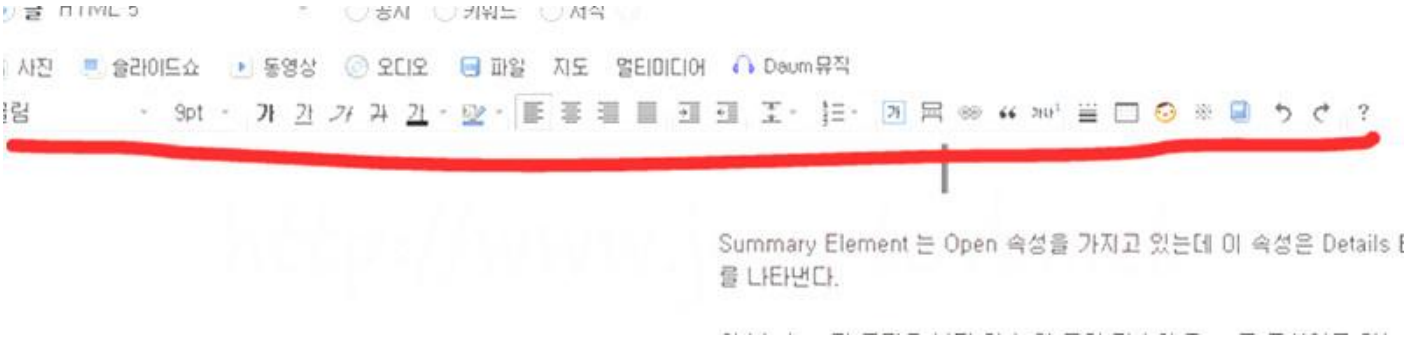
Summary Element 는 Open 속성을 가지고 있는데 이 속성은 Details Element 의 상세정보를 기본적으로 보여줄지, 숨길지 여부를 나타낸다.

위 Markup 과 그림을 보면 알수 있 듯이 단순히 Text 를 표시하고 있는데... 내용을 표시하고 숨기는 기능은 javascript 를 사용하여 제어한다.

아래 Site 를 참고 해보면 어떨까? ^^
http://remysharp.com/demo/details-with-js.html

4-2. Menu Element 와 Command Element

Command Element 는 사용자가 호출할 수있는 명령을 나타낸다.
아래 그림과 같은 명령아이콘을 구성할 수 있을 것이다.



Menu Element 는 Form Control List 나 Command List 를 만들때 사용하는데 Command Element 는 Menu Element 내에 포함시켜 사용해야하며 Radio Button 이나 CheckBox, Button 을 이용하여 명령어 박스를 만들 수 있게 해준다. Command Element 는 nav Element 와는 성격이 조금 달라서 URL 링크나 Form Action 을 처리하지 않는다.
아래는 Command Element 와 Menu Element 에 대한 Markup 예이다.

```
<div>
  <menu type="toolbar">
    <command type="radio" radiogroup="alignment" checked="checked"
       label="Left" icon="icons/alL.png" onclick="setAlign('left')">
    <command type="radio" radiogroup="alignment"
       label="Center" icon="icons/alC.png" onclick="setAlign('center')">
    <command type="radio" radiogroup="alignment"
       label="Right" icon="icons/alR.png" onclick="setAlign('right')">
    <hr>
    <command type="command" disabled
       label="Publish" icon="icons/pub.png" onclick="publish()">
  </menu>
</div>
```

```
<div>
  <menu type="toolbar">
    <li>
      <menu label="File">
        <button type="button" onclick="fnew()">New...</button>
        <button type="button" onclick="fopen()">Open...</button>
        <button type="button" onclick="fsave()">Save</button>
        <button type="button" onclick="fsaveas()">Save as...</button>
      </menu>
    </li>
    <li>
      <menu label="Edit">
        <button type="button" onclick="ecopy()">Copy</button>
        <button type="button" onclick="ecut()">Cut</button>
        <button type="button" onclick="epaste()">Paste</button>
      </menu>
    </li>
    <li>
      <menu label="Help">
        <li><a href="help.html">Help</a></li>
        <li><a href="about.html">About</a></li>
      </menu>
    </li>
  </menu>
</div>
```

(출처:http://www.w3.org/TR/html5/interactive-elements.html)

아래 이미지는 위 menu Element Makrup 을 표현한 화면이다.

(출처:http://www.w3.org/TR/html5/interactive-elements.html)

여기까지 Sementic Element 들에 대해서 간단히 짚어 보았다.

[HTML5 강좌] 6. Strong Web Form

1. input Element 의 새로운 Type

web form 은 10 년이 넘도록 checkbox, radio, password, select, file, submit, text 와 같은 몇개의 입력타입만 가지고 있었다.
이젠 새롭게 몇 가지의 Type 이 추가되었다.

1-1. datetime, week, month, date, time

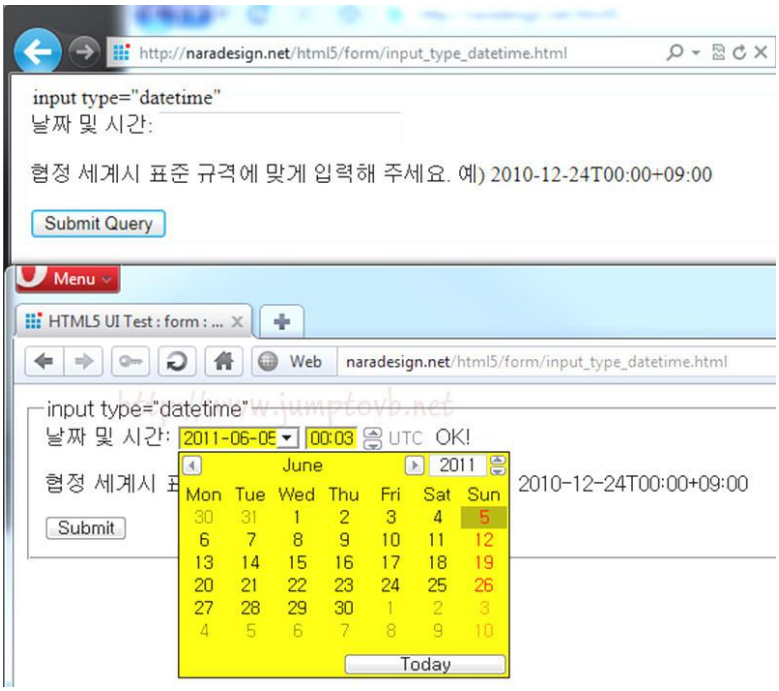
날짜 관련 Type 들이다.
Web Browser 들마다 지원하는 Form 이 다르다.
internet explore 9 과 opera 11 과 비교를 해 보았다.

Markup 은 아래와 같이 볼 수 있다.

```
<input type="datetime">
<input type="week">
<input type="month">
<input type="date">
<input type="time">
<input type="datetime-local">
```

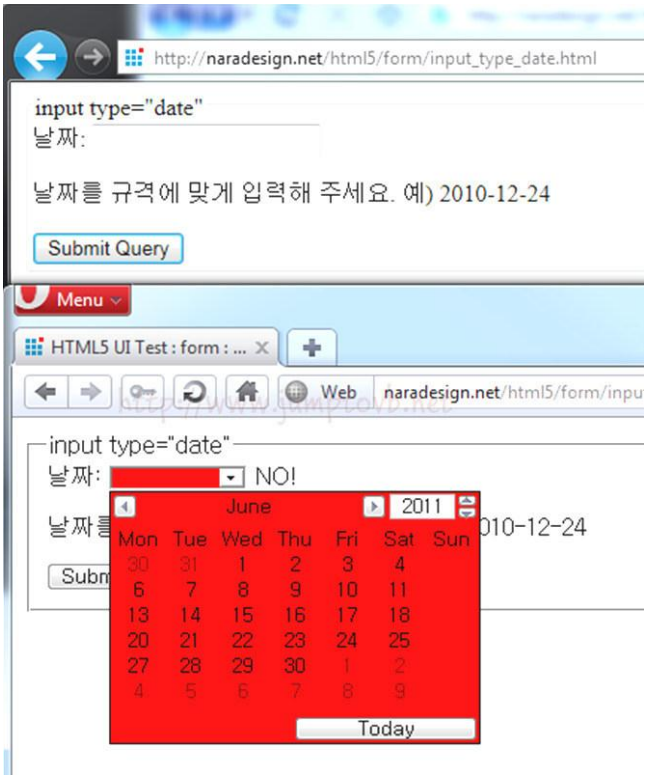
입력한다.

datetime 부터 이미지로 확인해 볼 수 있다.



Opera 에서는 날짜 선택 Control 이 rendering 된다.

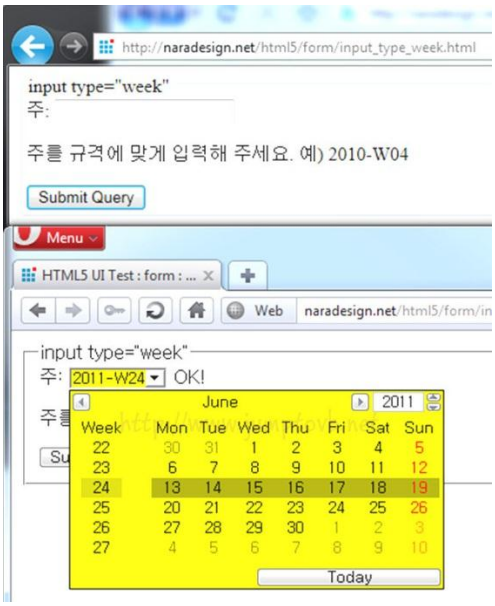
다음은 date Type 이다.



똑같이 날짜 선택 Control 이 나타난다.

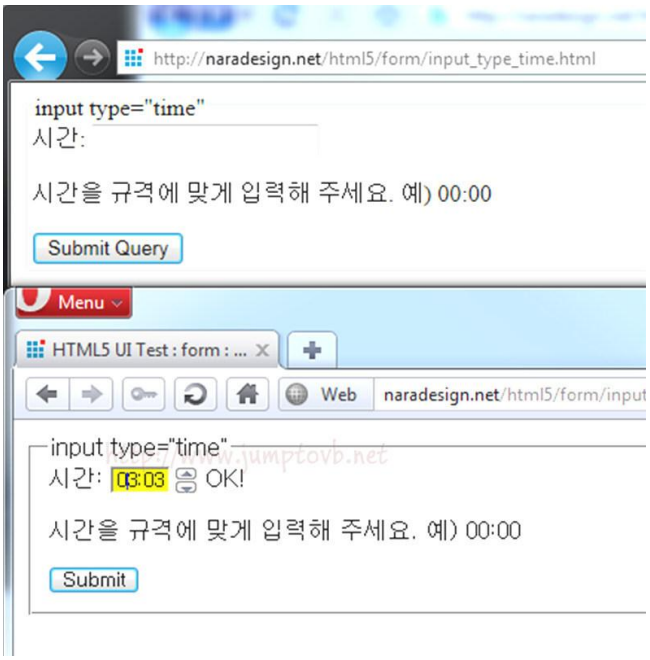
month Type 은 date Type 과 동일하게 나타난다..

다음은 Week Type 이다.



날짜 선택 Control 좌측에 Week 가 표시되어 있는 것을 보실 수 있다.

다음은 time type.

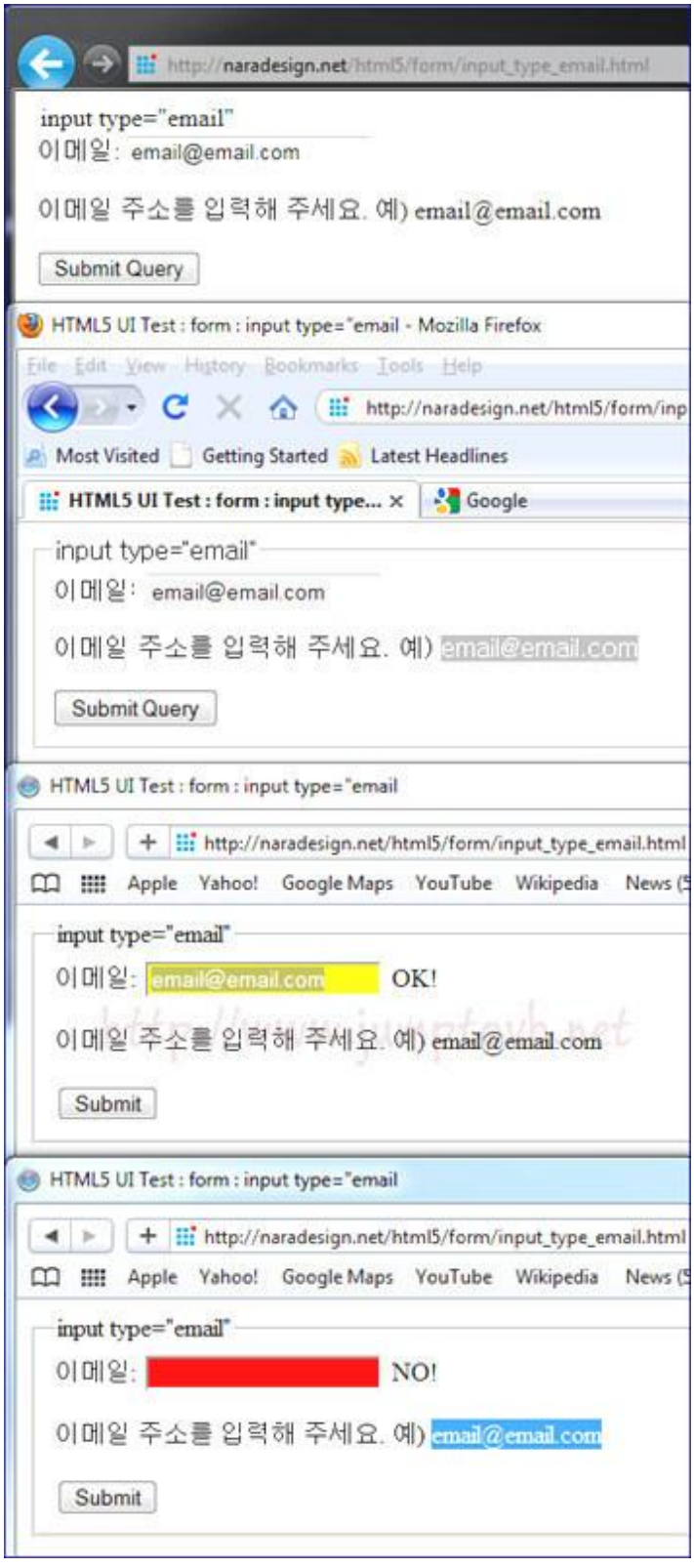


시간을 선택할 수 있도록 Spin Control 을 rendering 한다.

1-2. email

email 을 받아들이는 input type 이다.

email Type 을 지원하지 않는 Web Browser 는 text type 으로 인식을 할 것이고 Web Browser 에 따라서는 Validation Check 까지 할 수 있도록 지원해 줄 것이다.



(참고 : naradesign.net)

또한, iphone 의 경우에는 input field 에 커서를 가져갔을 때 "@" 와 "." 이 포함된 Screen Keyboard 를 띄워 줄것입니다.

(참고 : naradesign.net)

```
<input type="email" />
```

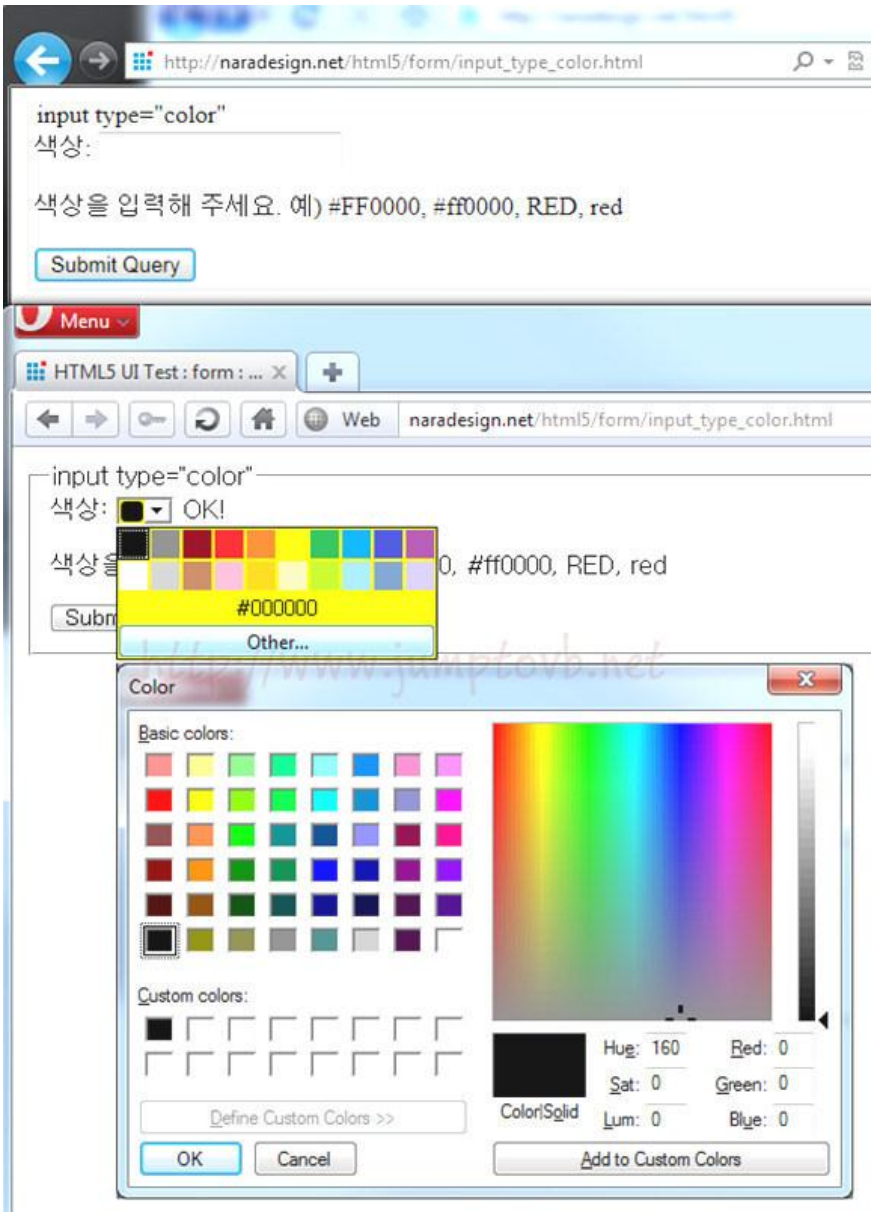
1-3. url

이번엔 URL type 이다.

Web Browser 에서는 text type 의 외관과 별 차이가 없으므로 iphone 에서 어떻게 보이는지만 확인한다.

이미지 하단을 보시면 "/" 와 ".com" 버튼이 보인다. 그리고 URL 에 맞게 Screen Keyboard 가 배치되어 있다.

```
<input type="url" />
```



Color type 은 색상을 선택할 수 있는 창을 띄워 주는데 현재 Opera 에서는 구현이 되어있다.
먼저 단순한 Color 선택 Control 이 나타나고 "Other..." 버튼을 Click 하게 되면 Color 대화상자가 뜬다.

```
<input type="color" />
```

1-5. search

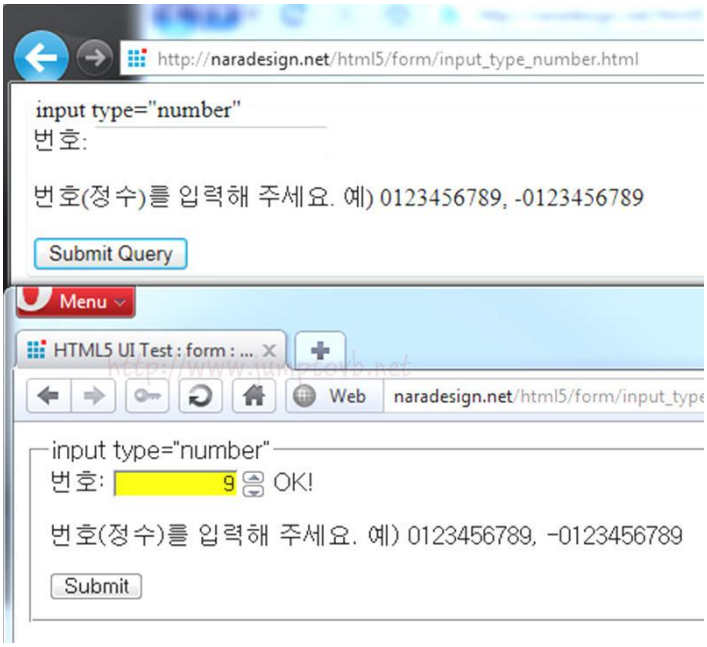
Search Type 또한 외관은 Text Type 과 다르지 않습니다. 그래서 iphone 의 모양을 살펴보도록 하겠다.

보시는 바와 같이 이미지 하단 우측의 버튼이 "Search" 버튼으로 바뀌어 있는 것을 보실 수 있다.

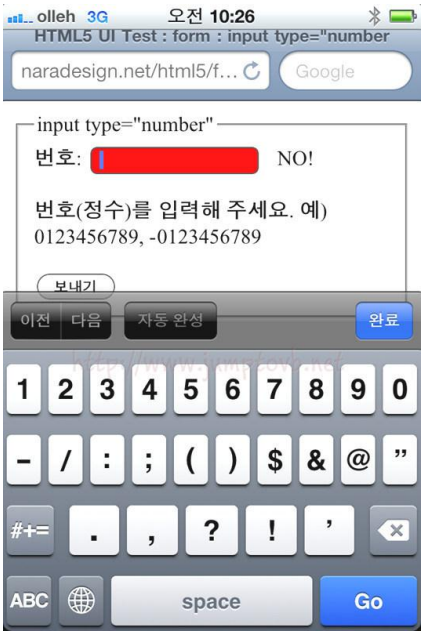
```
<input type="search" />
```

1-6. number

이번엔 number Type 이다.



일단 숫자를 마우스로도 입력할 수 있도록 Spin Control 로 rendering 하고 있다.

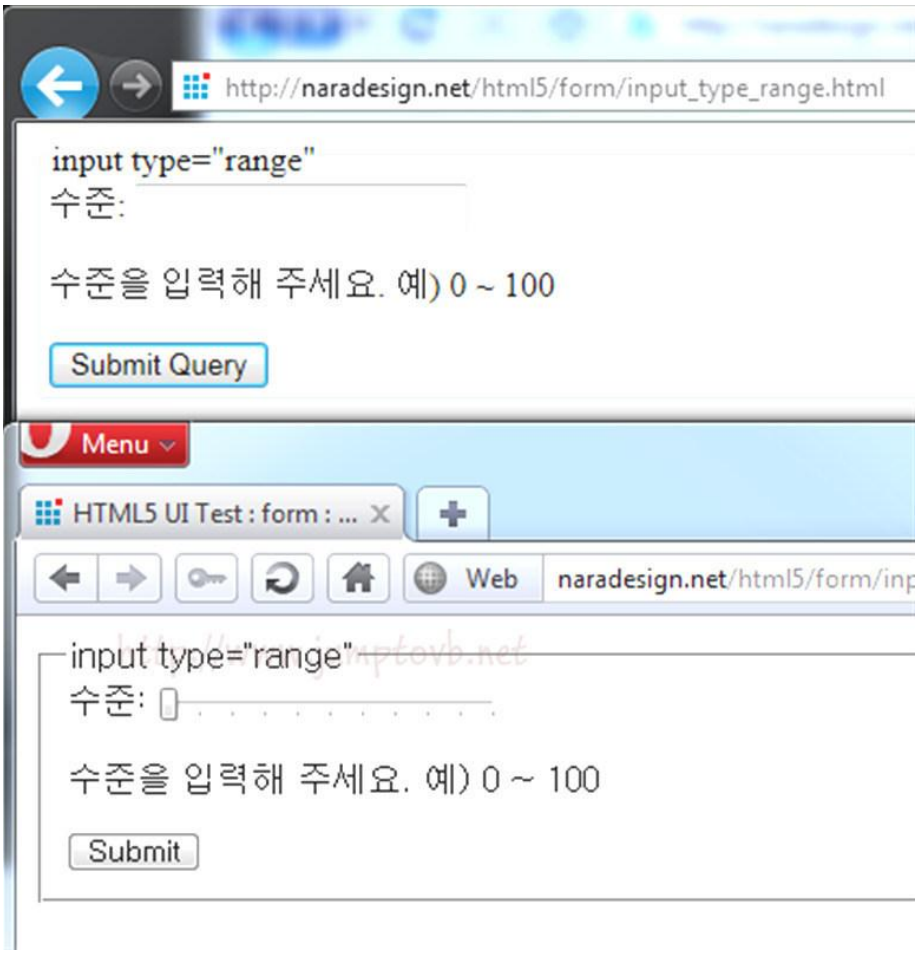


iphone 에서는 Screen Key Board 의 상단에 숫자 Key 가 배열되어 있는 것을 보실 수 있다.

```
<input type="number" min="1" max="100" />
```

1-7. range

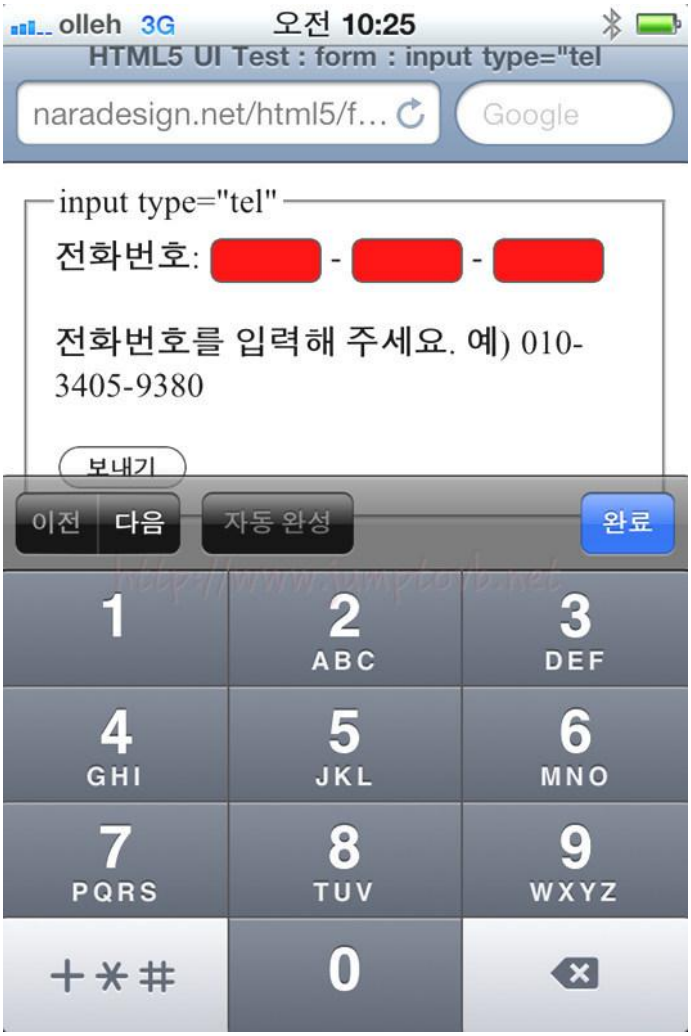
range Type 은 입력형태를 number Type 에서 제공할 수 있는 Spin Control 과는 다르게 Slide Control 로 rendering 한다.




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

1-8. tel

tel Type 의 경우는 완전히 숫자만 입력한다. iphone 에서는 아래와같이 숫자 입력 Screen Keyboard 가 나타난다.



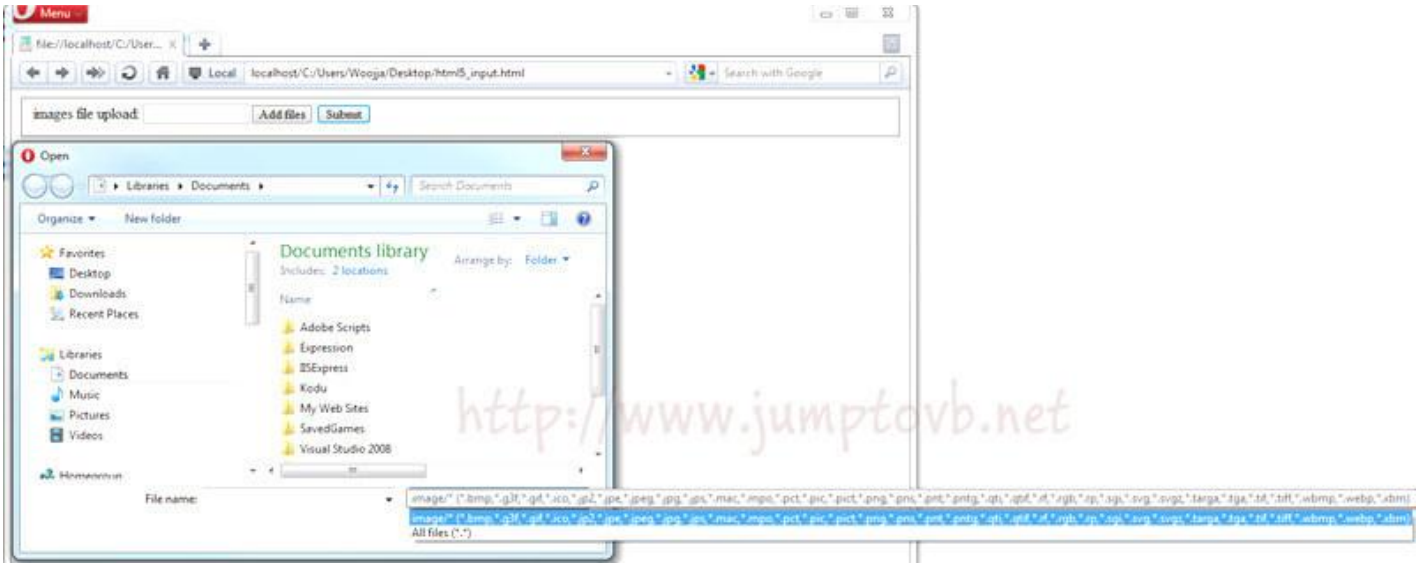

```
<input type="tel" />
```

1-9. file

file Type 의 경우 예전에는 하나의 input Tag 에 하나의 파일만 선택할 수 있었다. 하지만 HTML 5 에서는 multiple 속성을 주어 여러 파일들을 선택할 수 있고 accept 속성을 사용하여 MIME type 을 선택하여 파일들을 filtering 할 수 있다.

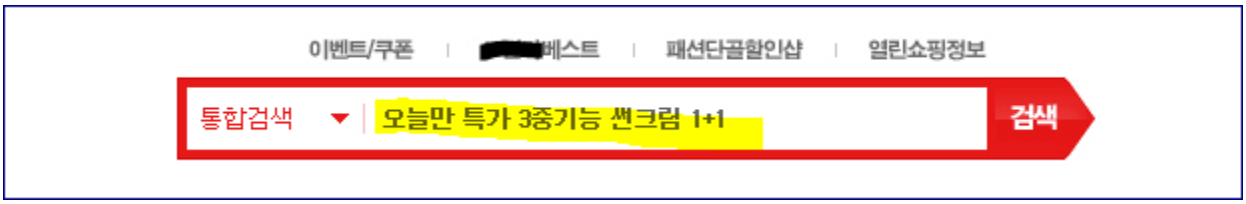
```
<input type="file" accept="image/*" multiple />
```

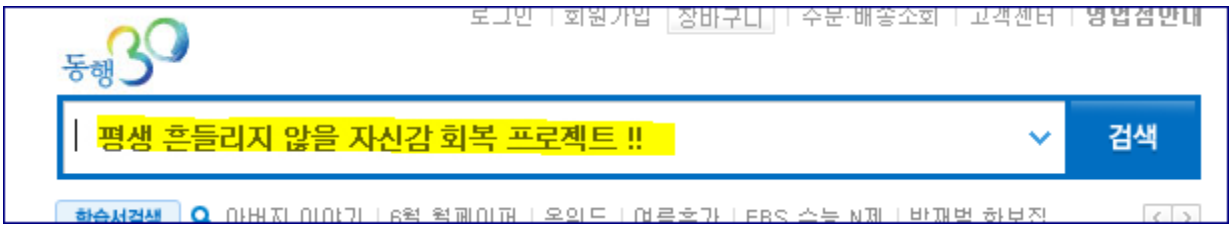
위 Tag 를 Markup 하면 아래 이미지와 같은 파일 선택 대화상자가 나타나며 image 를 Filtering 할 수 있다.



2. input Element 의 새로운 Attribute

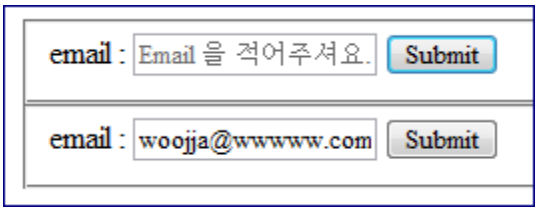
2-1. placeholder





위 이미지를 처럼 비어있는 TextBox 에 미리 Text 을 넣어 두는 것으로 사용자가 Click 을 하거나 Control 에 Focus 가 가게되면 Placeholder Text 는 사라지게 된다. 이젠 이 기능을 javascript 의 도움없이도 사용할 수 있게 되었다.

```
<input name="w" placeholder="Email 을 적어주셔요.">
```



2-2. autofocus

기존의 페이지들은 Autofocus 기능을 Javascript 를 이용해서 구현을 했다. 하지만, 구현하는 것이 그리 만만하지는 않았다. 구현보다도 구현후에 페이지 Loading 과 사용자의 엑션 사이에서 문제가 좀 있었다. Page Loading 중 사용자가 입력을 하고 있는 상황이라면 사용자가 원하지 않는 곳(Page Loading 후에 autofocus 가 위치한 곳) 에 입력을 하고 있을 수 있다.

사이트 로그인을 하겠다고 id 란에 id 를 입력하고 password 란에 password 를 입력하고 있는데 그 순간 Focus 가 javascript 에 의해서 다른 입력 Field 에 가 버려서 입력한 Password 가 훤히 다 보인다면? 사용자는 조금 황당한 상황이 될 수 있다. HTML5 에서는 autoscript 속성을 설정하면 page 가 Load 되자마자 Focus 가 설정된 곳으로 이동할 것이다.

또 Script 가 아닌 Markup 으로 동작하는 것이기 때문에 모든 Web Browser 에서 똑같이 동작할 것이다..

그리고 autofocus 는 다음과 같이 설정하면 된다.

```
<input name="w" autofocus>
```

2-3. formaction

formaction 속성을 사용하면 Page Form 의 action 이 되는 대상 Page 에 가기전에 input Element 단에서 대상 Page 를 바꾸어 버릴수 있다.

```
<form action="woolla.aspx" method="post">
<input type="submit" formaction="woojja.aspx" value="Submit" />
</form>
```

위 Markup 의 내용을 보면 **"woolla.aspx"**페이지로 이동할 것이 **"woojja.aspx"** 페이지로 이동하게 된다.

3. 새로 추가된 Element

3-1. Keygen Element

Keygen Element 는 form 을 전송할 때 키를 생성하는 Control 입니다. form 이 전송되면 비밀키와 공개키를 생성하여 비밀키는 Client 측에 저장하고 공개키는 서버에 전송한다. HTML 5 에서 규정하는 Key 의 Type 은 "RSA" 뿐이므로 keytype 속성에는 "rsa" 만을 사용할 수 있습니다. 지정하지 않아도 기본값으로 지정한다.

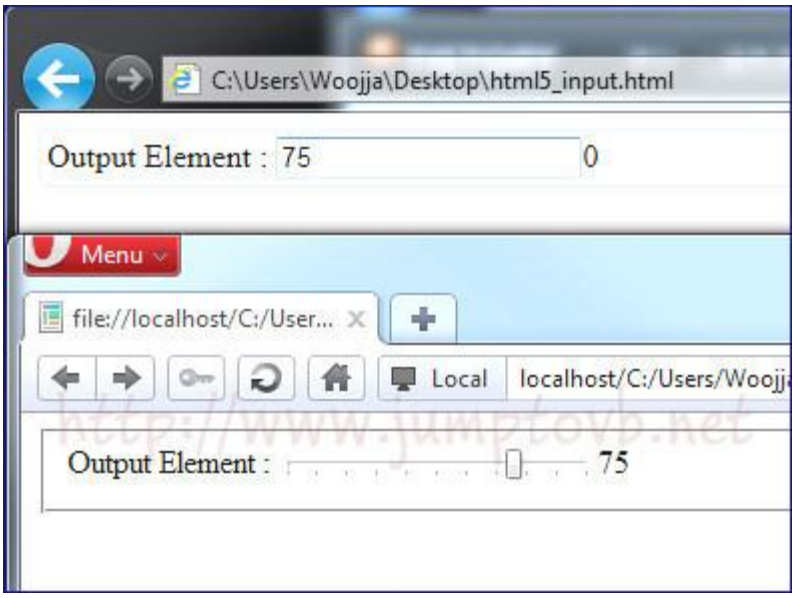
```
<keygen name="key" type="rsa">
```

3-2. Output Element

Output Element 는 무언가의 계산 결과를 나타 낸다.

range Element 와 output Element 를 함께 사용하는 예를 살펴본다.

```
<input type="range" id="rng" value="0"><output onformchange="value=form.elements.rng.value" for="rng">0</output>
```

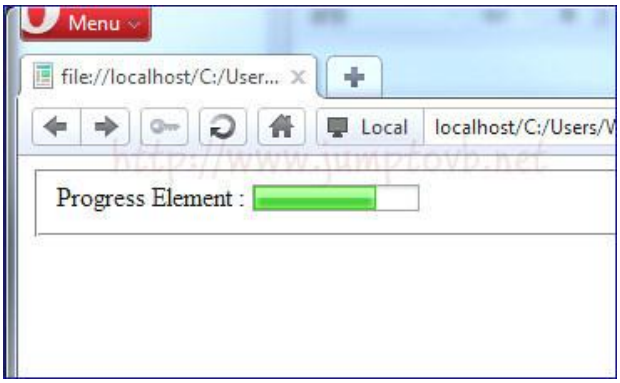


output Element value 속성을 넣을 수 없으므로 주의가 필요하다.

3-3. progress Element

Progress Element 는 작업의 진행정도를 나타낸다.

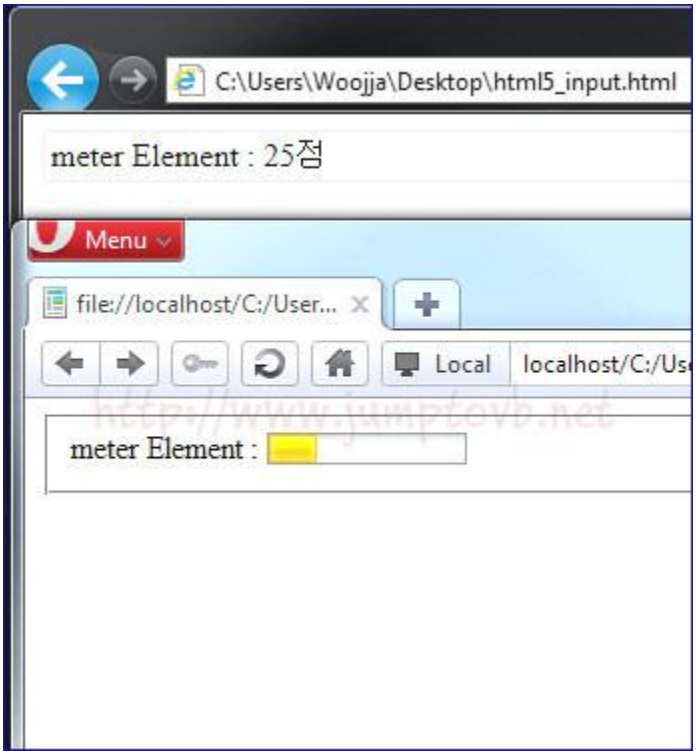
```
<progress value="0.75" max="1">0</progress>
```



3-4. meter

meter Element 는 한정된 범위에서의 값을 나타내고 싶을때 사용하는 것으로 최솟값과 최댓값을 지정할 수 없는 곳에서는 meter Element 를 사용할 수 없다.

```
<meter value="25" min="0" low="30" high="70" max="100" optimum="100">25 점</meter>
```



지금까지 새로 추가된 Element 들 중 중요하다고 생각한 것들을 중심으로 간략하게 살펴보았다.
마지막으로 사용하는 Web Browser 를 이용하여 아래 Page 를 가셔서 input type 의 속성 지원여부를 확인할 수 있다.
http://miketaylr.com/code/input-type-attr.html

http://www.w3.org/TR/2011/WD-html5-20110525/

웹혁명을 꿈꾸다 HTML5 & API 입문
웹표준 가이드 HTML5 CSS3

앞서가는 디자이너와 퍼블리셔를 위한 HTML5 & CSS
철저해설 HTML 5
구글개발자가 들려주는 HTML5 활용

[HTML5 강좌] 7. Rich Text Edit API

이번엔 사용자들이 쉽게 웹페이지를 편집할 수 있게하는 Editing API 에 대해서 살펴보도록 하겠다.

기존엔 웹페이지에 Editor 를 구현하기위해서 각 업체별로, 개발자별로 Javascript 를 사용하여 각자의 방법으로 개발하였다. 하지만, HTML 5 에서부터는 Rich Text 를 편집하기위한 API 가 표준으로 규정되어 있다. 그리고 추가적으로 DesignMode thrtjd 과 Global 속성인 Contenteditable 속성이 포함되어 있다.

5-1. Contenteditable 과 designMode

contenteditable 속성과 designMode 속성은 모두 내용편집이 불가능한 요소를 편집 가능하도록 하기 위한 API 로, 차이점은 contenteditable 속성은 특정 Element 의 내용을, designMode 속성은 문서 전체를 편집할 수 있게 한다.

```
<div id="editor" contenteditable="true"></div>
또는
<div id="editor" contenteditable></div> <!--true 값은 생략가능-->
```

contenteditable 속성의 값은 세가지를 가질 수 있다.

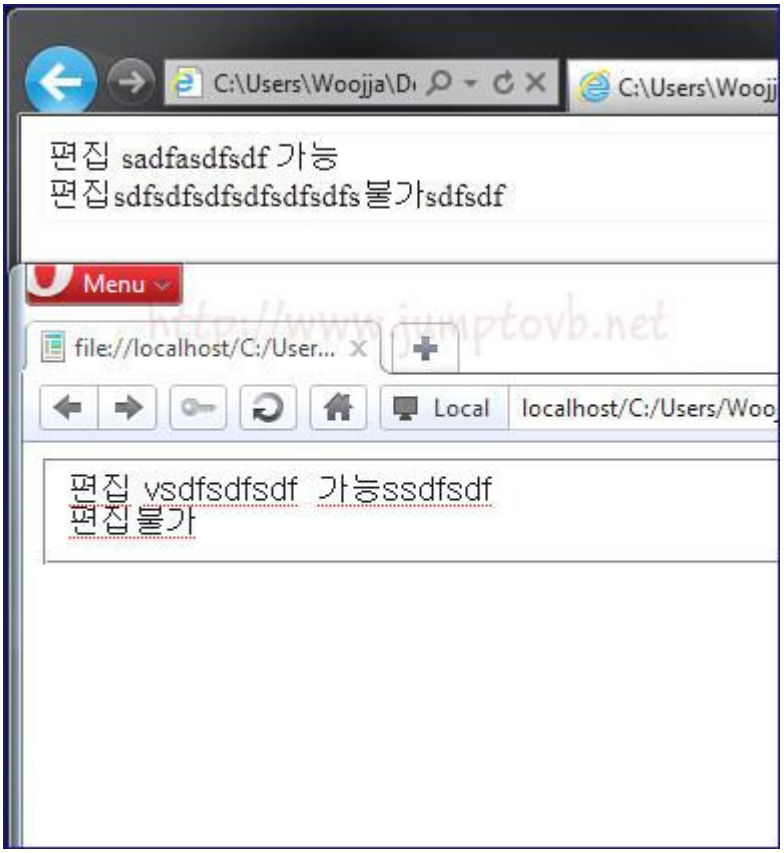
값	의미
true	편집가능
false	편집불가
inherit	부모요소로부터 상속

contenteditable 속성값이 true 인 요소의 자식 요소들은 기본적으로 true 값을 가진다.
하지만 아래와 같이 명시적으로 contenteditable 값을 false 로 지정한다면 편집할 수 없다.

(하지만 부모 div Element 의 contenteditable 의 속성이 true 이므로 그 자식 element 인 두번째 div 를 삭제할 수는 있을 것이다.)

```
<div id="editor" contenteditable="true">
<div>편집 가능</div>
<div contenteditable="false">편집불가</div>
</div>
```

(참고 : 웹혁명을 꿈꾸다 HTML 5 & API 입문)



위 이미지를 보면 상단 Web Browser 가 IE9 이고 아래는 Opera 이다. IE9 은 편집불가인 부분도 편집이 되는 문제가 있다. Opera 는 편집불가인 부분은 편집되지 않았다. 그리고 이렇게 지정한 contenteditable

의 속성값은 아래와 같이 javascirpt 를 이용하여 해당 Element 의 isContentEditable 속성에 접근할 수 있다.

```
var editor = document.getElementById("editor");
var isEditable = editor.isContentEditable
```

designMode 속성을 사용하면 문서 전체를 편집할 수 있다. designMode 속성은 document 객체가 갖는 속성으로 "on", "off" 값을 갖 sms 다. 문서 뿐만아니라 iframe 의 contentDocument 속성에 사용하게되면 iframe 내의 내용 또한 편집할 수 있다.

```
<iframe src="about:blank" width="300" height="300"
onload="this.contentDocument.designMode='on'"></iframe>
```

5-2. Rich Text Edit API

위에서 언급한 contenteditable 속성 이나 designMode 속성을 편집모드로 설정한 상태에서 editing API 를 사용하면 편집내용을 Rich Text 로 다룰 수 있다. 이때 사용하는 것이 execCommand API 이다.

이제 execCommand 를 살펴보도록 하겠다. document 객체에서 사용할 수있는 함수로 commandId, showUI, value 세가지의 Parameter 를 갖는다. 이 중 첫번째 Parameter 이외에는 생략할 수 있지만 Command Id 에 따라 값을 지정하여 호출한다.

```
document . execCommand(commandId [, showUI [, value ] ] )
```

이 때, Command ID 로 사용되는 명령어들은 다음과 같은 것들이 있다. 이 들을 간단히 살펴 본다.

CommandID	설명
bold	선택한 문자열을 굵은 글씨로 전환합니다.
createLink	선택한 문자열을 하이퍼링크로 전환합니다.
delete	선택한 문자열이나 커서 앞의 문자를 삭제합니다.
formatBlock	선택한 문자열을 지정한 Tag 로 감쌉니다.
forwardDelete	선택한 문자열이나 커서 뒤의 문자를 삭제합니다.
insertImage	커서가 위치한 곳에 이미지를 삽입합니다.
insertHTML	문자를 커서가 위치한 곳에 HTML 형태로 삽입합니다.
insertLineBreak	커서의 위치에 줄바꿈을 삽입합니다.
insertOrderedList	커서의 위치에 순서있는 리스트(ol tag)를 삽입합니다.
insertUnorderedList	커서의 위치에 순서없는 리스트(ul tag)를 삽입합니다.
insertParagraph	커서의 위치에 Paragraph (p tag 나 div tag : Browser 마다 구현은 각각 다름) 를 삽입합니다.
insertText	커서의 위치에 문자를 삽입합니다.
italic	선택한 문자열을 italic 체로 전환합니다.
redo	undo 를 재실행합니다.
selectAll	편집가능한 컨텐츠를 모두 선택합니다.
subscript	선택한 문자열을 아랫첨자로 전환합니다.
superscript	선택한 문자열을 윗첨자로 전환합니다.
undo	실행을 취소합니다.
unlink	선택한 문자열에 적용된 하이퍼링크를 제거합니다.
unselect	선택을 취소합니다.
VendorID-customCommandID	Browser 제작사들 마다의 명령어를 지정하여 호출합니다.

(참고 : http://www.w3.org/TR/html5/dnd.html#editing-apis)

execCommand 함수 이외에도 다섯가지의 함수가 더 존재하는데 이 함수들을 이용하면 WYSIWYG Editor 를 만들 수 있을 것이다. 그럼, execCommand 함수 이외의 다섯가지 함수에 대해서 살펴보도록 하겠다.

함수명	**설명**
document.queryCommandEnabled(commandId)	지정한 명령어(commandId)가 실행가능한지 여부를 반환합니다. 현재 커서의 위치에서 실행가능 여부를 알 수 있습니다.
document.queryCommandIndeterm(commandId)	지정한 명령어(commandId)의 효과가 일정한가 여부를 반환합니다.
document.queryCommandState(commandId)	지정한 명령어(commandId)의 상태를 반환합니다. 현재 커서의 위치에서 실행가능 여부를 알 수 있습니다.
document.queryCommandSupported(commandId)	지정한 명령어(commandId)가 지원되는지 여부를 반환합니다.
document.queryCommandValue(commandId)	지정한 명령어(commandId)의 현재값을 반환합니다.

위 함수들과 commandid 들을 사용하면 쉽게 Editor 를 구현할 수 있다. HTML 5 로 넘어오면서 달라진 점이라고 한다면 표준화하여 API 를 작성하고 통일된 사양아래서 개발할 수 있도록 하였다는데 주목해야 한다.

Editor 를 제작하면 더 좋았을 듯 하지만 그 내용은 추후에 추가해 보도록 하겠다.

[HTML5 강좌] 8. Video Element

Video Element 는 HTML 5 에서 관심이 주목되고 있는 새롭게 추가된 element 들 중 하나이다.
지금까지 웹페이지에 동영상을 삽입하고 싶을때는 object Element 나 Embed Element 를 사용하여 Flash, media player 와 같은 플러그인을 이용하여야 했다. 그러나, 이제 HTML 5 에서는 플러그인없이도 동영상을 웹페이지에 쉽게 삽입할 수 있다. 그리고, 다음과 같이 object, embed Element 와는 다른 점이 있다.

- 재생, 일시중지 등 Web Browser 자체 Control 이 있다.
- Source 에 파일을 여러개를 지정할 수 있어서 Web Browser 의 지원 Format 에 따라 표시 할 수 있다.

```
<!DOCTYPE HTML>
<html lang="ko">
  <head>
    <meta charset="UTF-8">
    <title>HTML 5 Video 예제</title>
  </head>
  <body>
    <video width="320" height="240" controls="controls">
      <source src="http://www.w3schools.com/html5/movie.mp4" type="video/mp4" />
      <source src="http://www.w3schools.com/html5/movie.webm" type="video/webm" />
      <source src="http://www.w3schools.com/html5/movie.ogg" type="video/ogg" />
      <p>Your browser does not support the video tag.</p>
    </video>
  </body>
</html>
```

(참고 : http://www.w3schools.com)

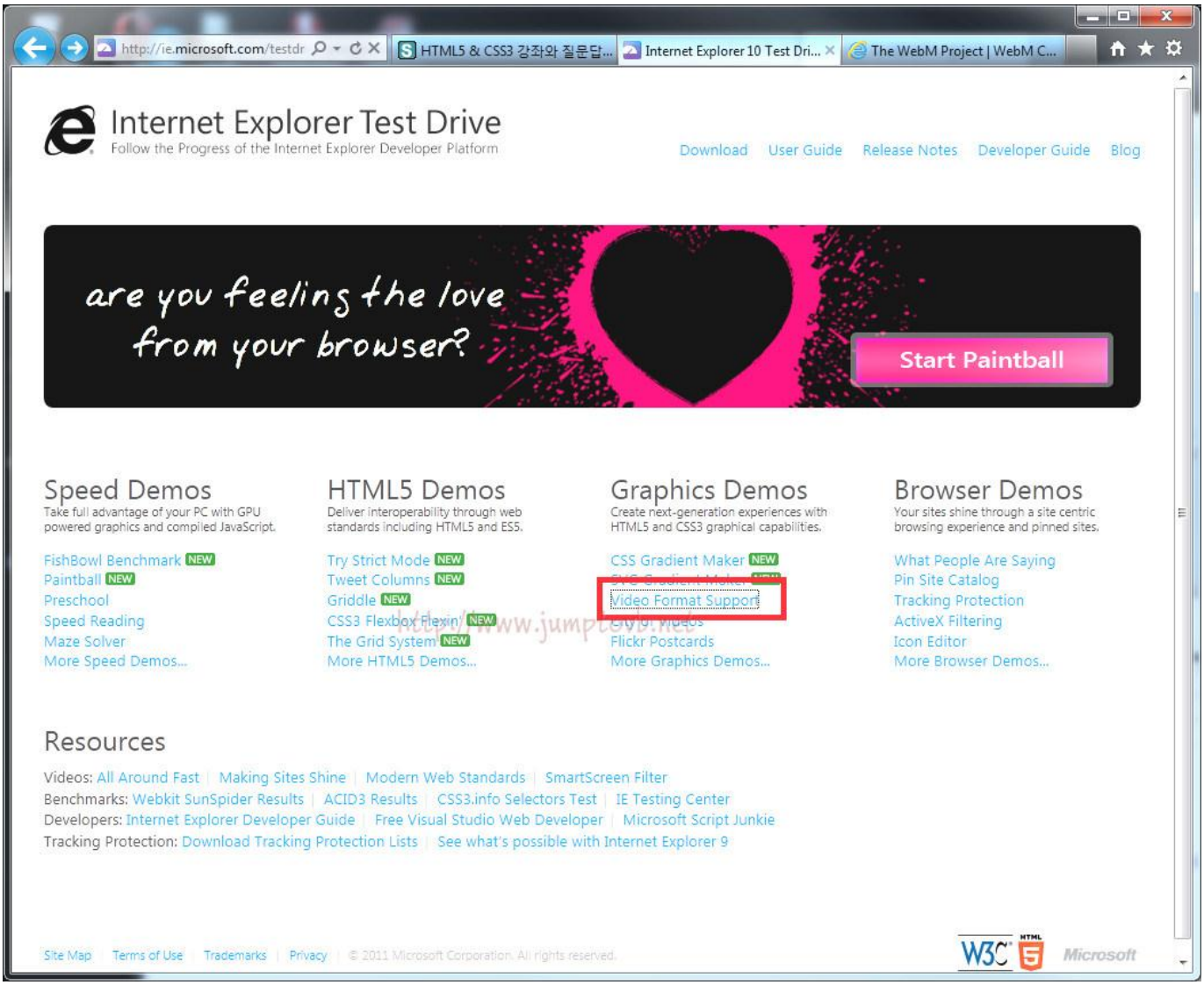
rowser 별 지원 비디오 코덱을 위 Markup 을 바탕으로 아래 표에 간략히 정리해 보았다.

	IE 9	FireFox 3.6.17	Opera 11.11	Chrome 12.0.742	Safari 5.0.5
Ogg	X	O	O	O	X
MPEG 4	O	X	X	O	O
WebM	△	X	O	O	X

IE 9 에다가 WebM 란에는 "△" 표시를 했는데 기본적으로 지원을 하는 것이 아니고,

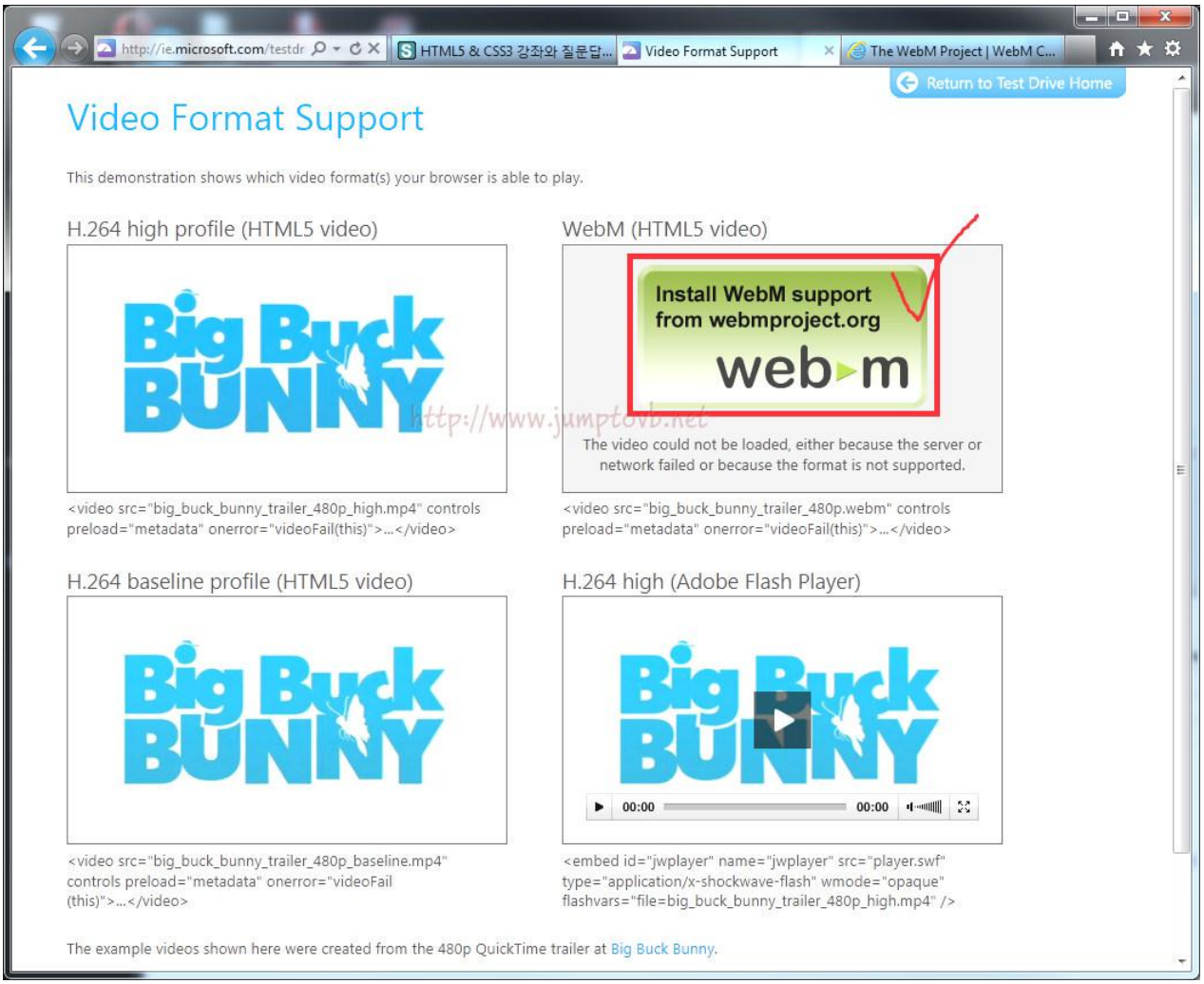
http://blogs.msdn.com/b/ie/archive/2011/03/16/html5-video-update-webm-for-ie9.aspx

webM Componenent 를 설치하시고 Refresh 를 하셔야 볼 수 있다.

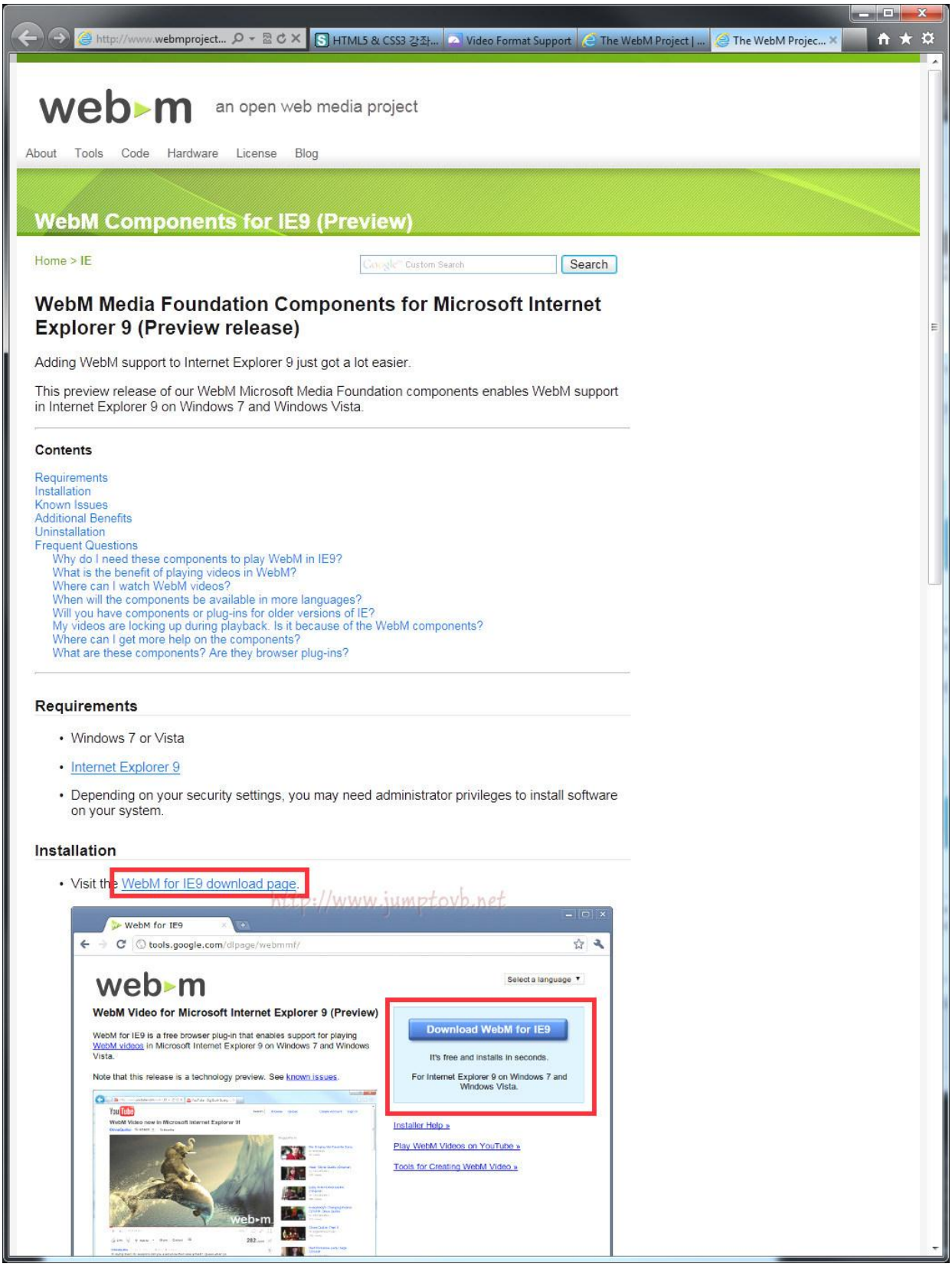


그 해당싸이트는 위와 같이 접근 할 수도 있다.

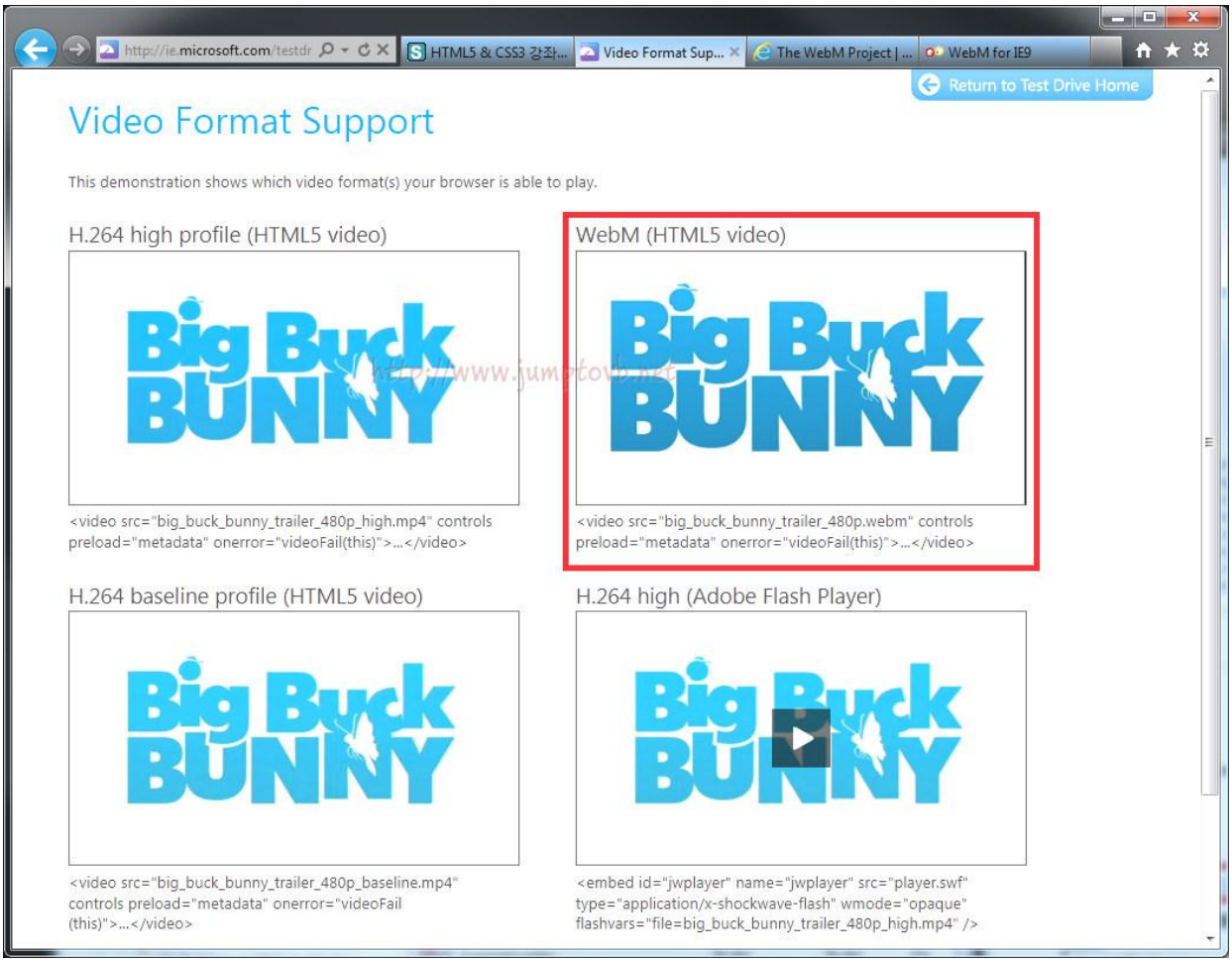
http://ie.microsoft.com/testdrive/



위 이미지 버튼을 클릭한다.

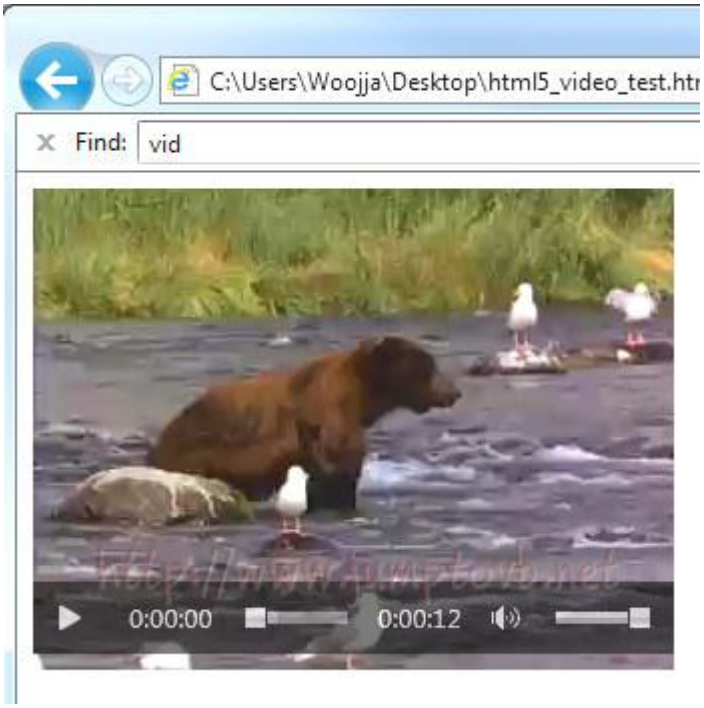

http://www.webmproject...
HTML5 & CSS3 강좌...
Video Format Support
The WebM Project | ...
The WebM Projec...
web▸m
an open web media project
About
Tools
Code
Hardware
License
Blog
WebM Components for IE9 (Preview)
Home > IE
Custom Search
Search
WebM Media Foundation Components for Microsoft Internet Explorer 9 (Preview release)
Adding WebM support to Internet Explorer 9 just got a lot easier.
This preview release of our WebM Microsoft Media Foundation components enables WebM support in Internet Explorer 9 on Windows 7 and Windows Vista.
Contents
Requirements
Installation
Known Issues
Additional Benefits
Uninstallation
Frequent Questions
Why do I need these components to play WebM in IE9?
What is the benefit of playing videos in WebM?
Where can I watch WebM videos?
When will the components be available in more languages?
Will you have components or plug-ins for older versions of IE?
My videos are locking up during playback. Is it because of the WebM components?
Where can I get more help on the components?
What are these components? Are they browser plug-ins?
Requirements
Windows 7 or Vista
Internet Explorer 9
Depending on your security settings, you may need administrator privileges to install software on your system.
Installation
Visit the WebM for IE9 download page.
http://www.jumptovb.net
WebM for IE9
tools.google.com/dlpage/webmmf/
web▸m
Select a language
WebM Video for Microsoft Internet Explorer 9 (Preview)
WebM for IE9 is a free browser plug-in that enables support for playing WebM videos in Microsoft Internet Explorer 9 on Windows 7 and Windows Vista.
Note that this release is a technology preview. See known issues.
Download WebM for IE9
It's free and installs in seconds.
For Internet Explorer 9 on Windows 7 and Windows Vista.
Installer Help »
Play WebM Videos on YouTube »
Tools for Creating WebM Video »

다시 다운 받을수 있는 곳으로 이동을 한다.

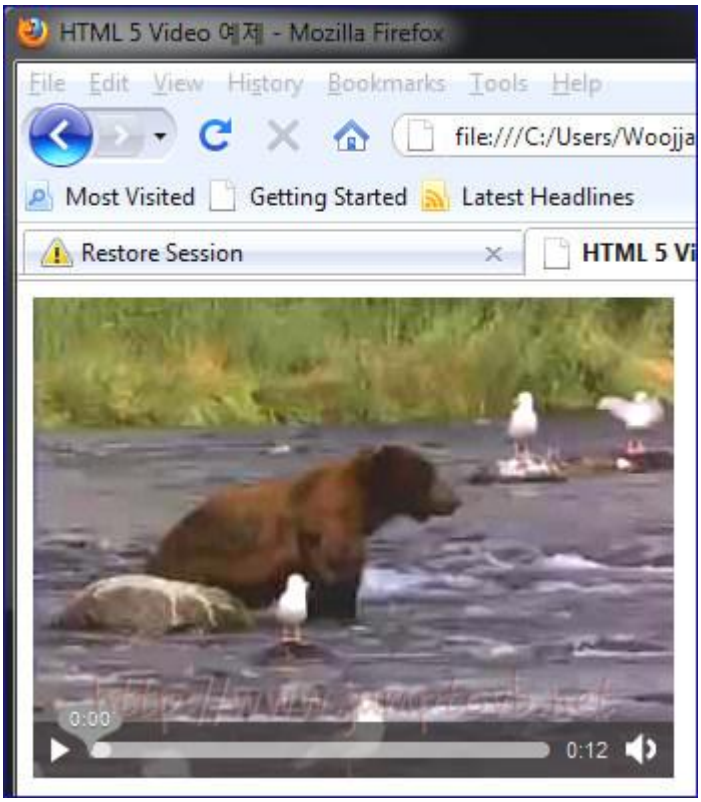


그럼 이번에는 각 Web Browser 마다 제공하는 기본 Control 의 모양을 살펴본다.

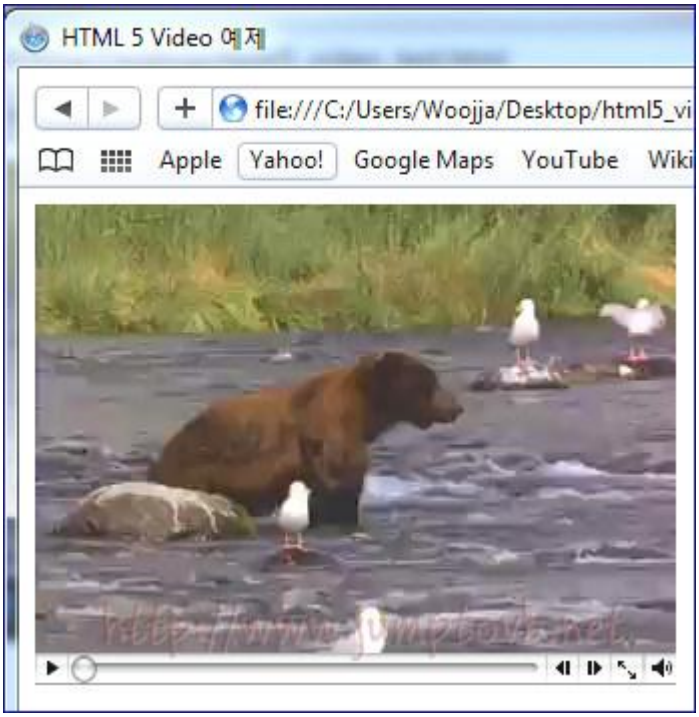
IE 의 출력 결과는 아래와 같다.



FireFox 의 출력결과.



Safari 의 결과.



각 Web Browser 마다 특색이 나타나는 것을 확인해 볼 수 있다. 이제 Video Element 를 사용하는 방법을 알아보도록 하겠다.

```
<video src="sqler.mp4" loop="loop">
        <p>이 문장이 보인다면 video Tag 를 지원하지 않는 Web Browser 를 사용하고 계시는 겁니다.</p>
</video>
```

속성	**설명**
controls	재생 Control 표시를 제어합니다.
autoplay	동영상이 Loading 되면 곧바로 재생을 시작합니다.
autobuffer	이 속성을 사용하게 되면 사용하기 전부터 다운로드가 진행됩니다. 사용자가 재생할 때쯤이면 동영상이 어느 정도 다운로드가 된 상태일 것입니다.
poster	동영상이 Download 중이거나 Buffering 중에 나타낼 이미지를 지정합니다.
loop	동영상을 반복 재생합니다.
width	동영상의 너비를 지정합니다.
height	동영상의 높이를 지정합니다.

하지만, Web Browser 마다 지원하는 동영상이 다르므로 아래 Markup 처럼 둘 이상의 여러 포맷을 준비해 두어 Source Element 를 이용하여 모든 Web Browser 에서 재생가능하도록 해야 한다.

```
<video controls=controls poster="image/poster/sqler.png">
        <source src="media/video/sqler.ogg" type="video/ogg">
        <source src="media/video/sqler.mp4" type="video/mp4">
        <object data="media/video/sqler.mov">
                <p>이 문장이 보인다면 HTML 5 의 video Tag 를 지원하지 않는 Web Browser 를 사용하고 계시는 겁니다.</p>
        </object>
</video>
```

이런 Video Element 는 javascript 를 사용하여 자유롭게 동영상을 조작할 수 있다. 너무 간단하긴 하지만 아래와 같은 arkup 으로 Web Browser 가 제공하는 Control 이 아닌 사용자가 직접 제작할 수 있는 Control 을 제작해 보았다.

```
<video id="vdo" src="media/video/sqler.ogg" poster="image/poster/sqler.png">
        <p>이 문장이 보인다면 HTML 5 의 video Tag 를 지원하지 않는 Web Browser 를 사용하고 계시는 겁니다.</p>
</video>
<p>
        <button onclick="document.getElementById('vdo').pause()">Pause</button>
        <button onclick="document.getElementById('vdo').play()">Play</button>
        <button onclick="document.getElementById('vdo').currentTime = 0">|<</button>
        <button onclick="document.getElementById('vdo').currentTime += 5">>></button>
</p>
<script>
        var vdo = document.getElementById('vdo');
        if(vdo != null && vdo.addEventListener)
        {
                vdo.addEventListener(
                'ended',
                function() {
                        alert('The video has ended.');
                        },
                false);
        }
</script>
```

아래와 같이 외부 Control 을 확인하실수 있다.

이 상으로 간단하게 Video Element 를 사용하는 방법에 대해서 간단하게 살펴보았다. 동영상 조작을 위한 외부 Contorl 들을 제작하기에도 그리 어렵지 않을 것이다. 다음은 Audio Element 에 대해서 살펴보기로 하겠다.

[HTML5 강좌] 9. Audio Element

Audio Element 도 HTML 5 에서 관심이 주목되고 있는 새롭게 추가된 element 들 중 하나이다. Poster 속성이 없다는 점을 빼고는 큰 차이점이 없다.

```
<audio src="http://www.w3schools.com/html5/horse.ogg"></audio>
```

그럼 여기에 audio file 을 자동적으로 재생시키려면

```
<audio src="http://www.w3schools.com/html5/horse.ogg" autoplay></audio>
또는
<audio src="http://www.w3schools.com/html5/horse.ogg" autoplay="autoplay"></audio>
```

이렇게 하면 되겠다.

Audio Element 도 Video Element 처럼 Web Browser 가 Audio Element 를 지원하느냐에 따라 실행여부를 알 수 있는데 Web Browser 가 Audio Element 를 지원하지 않을 때 특정 문구를 표시할 수 있도록 지원 시킬수 있다.

```
<audio src="http://www.w3schools.com/html5/horse.ogg" autoplay>
        <p>Your browser does not support audio tag</p>
</audio>
```

Audio Element 도 Vedio Element 와 같이 재생 control 을 표시할 수 있다.

```
<audio src="http://www.w3schools.com/html5/horse.ogg" controls></audio>
또는
<audio src="http://www.w3schools.com/html5/horse.ogg" controls="controls"></audio>
```

다음은 Audio Element 가 사용할 수 있는 속성들은 아래와 같다.

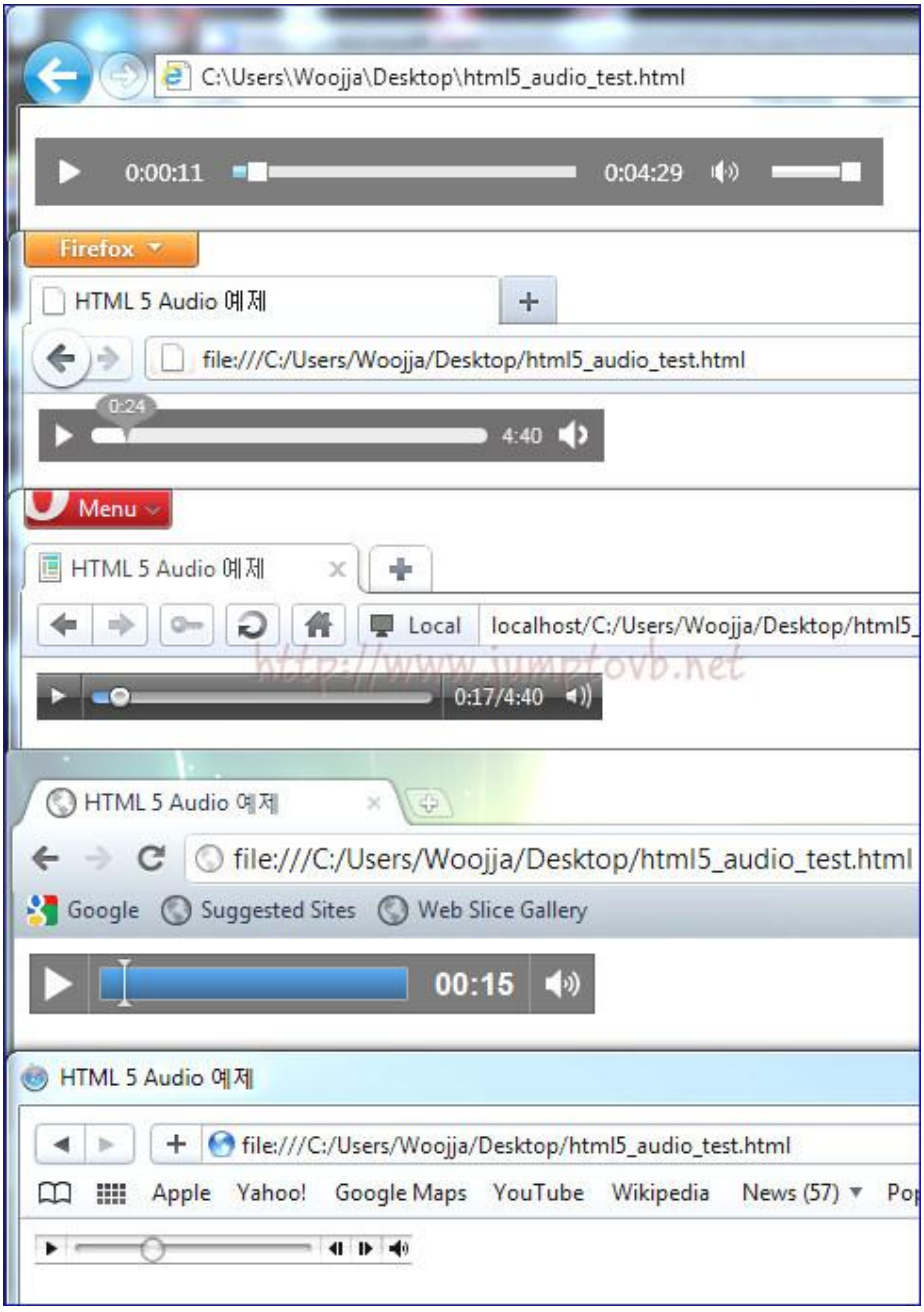
속성명	**값**	**기능**
src	파일 경로	Source
autoplay	autoplay	페이지가 Load 되자마자 사운드를 재생할지를 지정합니다.
controls	controls	플레이어를 표시합니다.
loop	재생횟수	audio 를 반복 재생할 횟수를 지정합니다.
preload	none, auto, meta	Page 가 열리면 audio 를 미리 Load 합니다.

이번엔 audio Element 가 지원하는 Audio 코덱들을 살펴보도록 하겠다.

	IE	**FireFox**	**Opera**	**Chrome**	**Safari**
Vorbis(oga, ogg)	X	O	O	O	X
wav, wma	O	O	O	X	O
mp3	O	X	X	O	O
AAC	O	X	X	O	O

마지막으로 Control 의 모양은 아래와 같다.

```
<audio controls="controls" id="ado">
    <source src="http://test2.momac.net/html5/demo_audio/youaremylife_instrumental.mp3"/>
    <source src="http://test2.momac.net/html5/demo_audio/youaremylife_instrumental.ogg"/>
    <source src="http://test2.momac.net/html5/demo_audio/youaremylife_instrumental.wav"/>
    <p>Your browser does not support the audio tag.</p>
</audio>
```



Video 의 Control 모양과 다르지 않다. 오늘은 간단히 audio Element 에 대해서 살펴보았다.
다음은 가장 관심이 많으실 Canvas Element 에 대해서 알아보도록 하겠다.

[HTML5 강좌] 10. Canvas Element

Canvas Element 는 HTML 5 에서 가장 많이 주목받고 있는 새롭게 추가된 element 들 중 하나이다.

10-1. 사용법

```
<canvas id="vas" width="200" height="300"></canvas>
```

위 Markup 을 보면 width, height 속성을 부여하고 있습니다만, 만약 이들을 지정해주지 않는다면 기본적으로 background Color 는 "Transparent" 로 width 는 300 pixel, height 는 150 pixel 로 지정한다. 그리고 사용하는 Web Browser 가 Canvas Element 를 지원하지 않는 경우 다음과 같이 대체해서 보여줄 수있는 이미지를 지정할 수 있다.

```
<canvas id="vas" width="200" height="300">
        <img src="eximg.jpg" width="200" height="300" alt="대체이미지" />
</canvas>
```

그리고 다음과 같이 Javascript 를 이용해서 Canvas Element 에 접근할 수 있다.

```
var vas=document.getElementById('vas');


var cntxt = vas.getContext('2d') // 요로케하면 2d 그림을 그릴 수 있다.
```

혹시나 Canvas Element 를 지원하지 않는 IE 를 사용하시는 분들을 위해서 아래 링크에 가셔서 Javascript 라이브러리를 다운받으셔서 사용해 보시기 바란다.

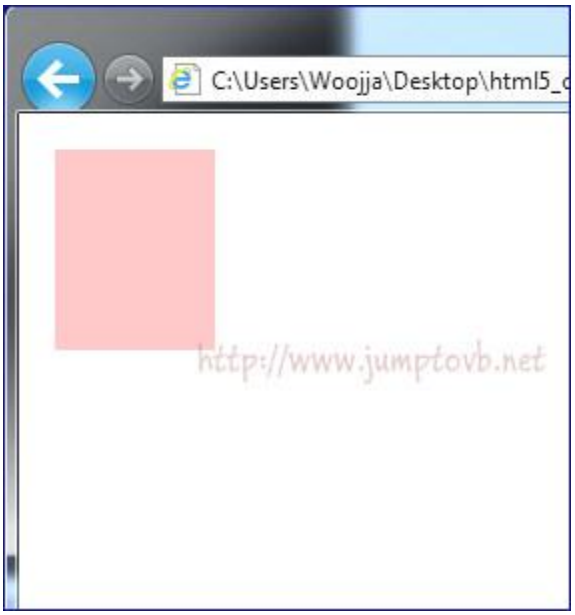
http://excanvas.sourceforge.net
또는
http://code.google.com/p/explorercanvas/에 가셔서 excnavas.zip

10-2. 2D 도형그리기

```
<!DOCTYPE HTML>
<html lang="ko">
  <head>
    <meta charset="UTF-8">
    <title>HTML 5 Canvas 예제</title>
  </head>
  <body>
    <canvas id="woojjaCanvas" width="200" height="300"></canvas>
    <script type="text/javascript">
      var canvas=document.getElementById('woojjaCanvas');
      var cvas=canvas.getContext('2d');
      cvas.fillStyle='#FF0000';
      //cvas.fillStyle="rgba(255,0,0,0.3)";
      cvas.fillRect(10,10,80,100);
    </script>
  </body>
</html>
```

fillStyle 은 16 진수로 적어도 된다.

cvas.fillStyle="rgba(255,0,0,0.3)"; 와 같이 rgba 값을 이용해도 되는데 a 값은 alpha 값으로 투명도를 나타낸다. 값은 0~1 사이값을 넣어주시면 된다. 아래 이미지는 rgba 값을 적용한 것이다.



함수	
fillRect	속을 채운 사각형을 그립니다.
strokeRect	사각형 테두리만 그립니다.
clearRect	지정한 사각영역을 투명하게 지웁니다.

```
<!DOCTYPE HTML>
<html lang="ko">
  <head>
    <meta charset="UTF-8">
    <title>HTML 5 Canvas 예제</title>
     <style type="text/css">
        #woojjaCanvas { border:1px solid #000000;}
     </style>
  </head>
  <body>
    <canvas id="woojjaCanvas" width="300" height="200"></canvas>
    <script type="text/javascript">
      var canvas=document.getElementById('woojjaCanvas');
      var cvas=canvas.getContext('2d');
      cvas.fillStyle='#FF0000';
      //cvas.fillStyle="rgba(255,0,0,0.3)";
      cvas.fillRect(10,10,150,180);
      cvas.clearRect(20,20,70,90);
      cvas.strokeRect(30,30,200,100);
    </script>
  </body>
</html>
```

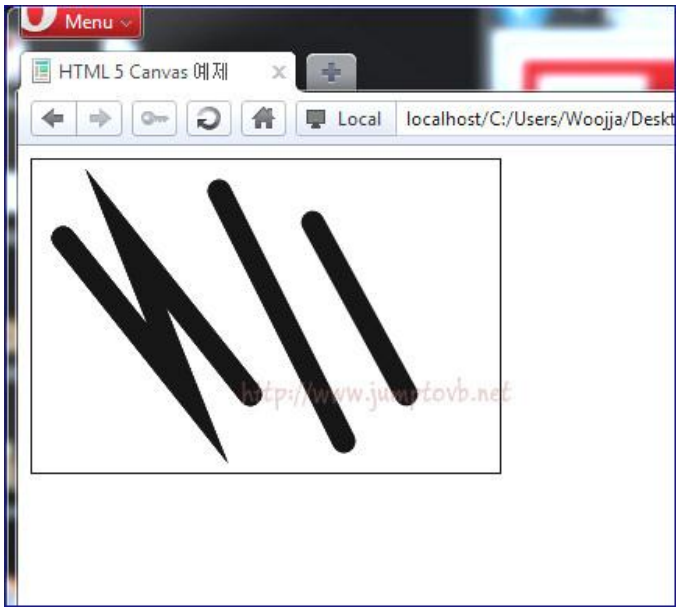
위 Markup 을 Web Browser 로 봤다

사각형만 살펴보았다. arc 를 이용하여 원을 그릴 수 있다.

10-3. 선긋기

이번엔 선을 그려 보겠다.

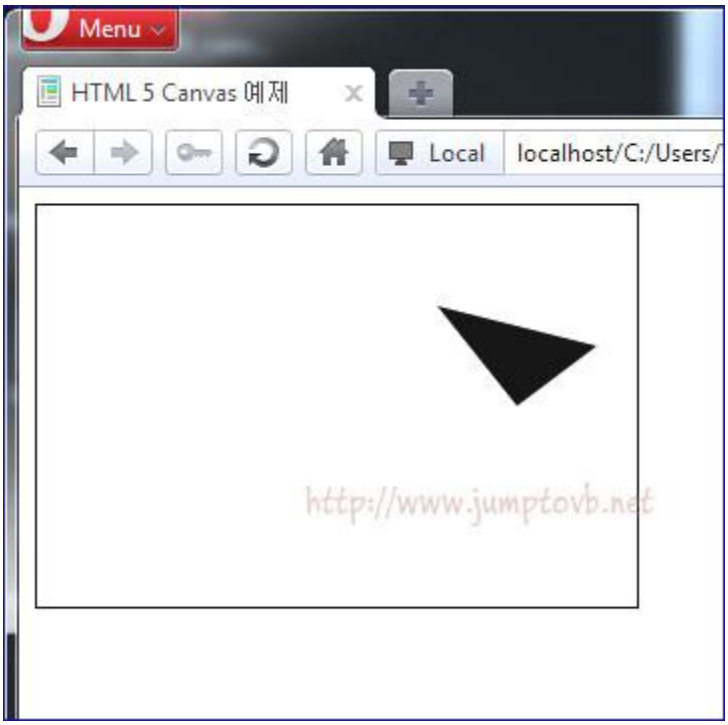
```
<!DOCTYPE HTML>
<html lang="ko">
   <head>
      <meta charset="UTF-8">
      <title>HTML 5 Canvas 예제</title>
    <style type="text/css">
     #woojjaCanvas { border:1px solid #000000;}
    </style>
   </head>
   <body>
      <canvas id="woojjaCanvas" width="300" height="200"></canvas>
      <script type="text/javascript">

         var canvas=document.getElementById('woojjaCanvas');
         var cvas=canvas.getContext('2d');

         cvas.beginPath();

         cvas.lineWidth=15;
         cvas.lineCap="round";

         cvas.lineTo(20, 50);
         cvas.lineTo(100, 150);
         cvas.stroke();

         cvas.lineTo(60, 50);
         cvas.lineTo(140, 150);
         cvas.stroke();


         cvas.moveTo(120, 20);
         cvas.lineTo(200, 180);
         cvas.stroke();

         cvas.moveTo(180, 40);
         cvas.lineTo(240, 150);
         cvas.stroke();

         cvas.clothPath();

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


Menu
HTML 5 Canvas 예제
Local
localhost/C:/Users/Woojja/Deskt

직선말고도 quardratiCurve(2 차 베지어곡선), bezierCurveTo(3 차 베지어곡선) 등의 함수를 이용해서

```
<!DOCTYPE HTML>
<html lang="ko">
   <head>
      <meta charset="UTF-8">
      <title>HTML 5 Canvas 예제</title>
       <style type="text/css">
       #woojjaCanvas { border:1px solid #000000;}
       </style>
   </head>
   <body>
      <canvas id="woojjaCanvas" width="300" height="200"></canvas>
      <script type="text/javascript">
         var canvas=document.getElementById('woojjaCanvas');
         var cvas=canvas.getContext('2d');
         cvas.beginPath();
         cvas.lineWidth=15;
         cvas.lineCap="round";
         cvas.lineTo(200, 50);
         cvas.lineTo(240, 100);
         cvas.lineTo(280, 70);
         cvas.lineTo(200, 50);
         cvas.clothPath();
         cvas.fill();
      </script>
   </body>
</html>
```

곡선을 그릴 수 있다. 다음은 선긋기를 이용한 도형 그리기이다.



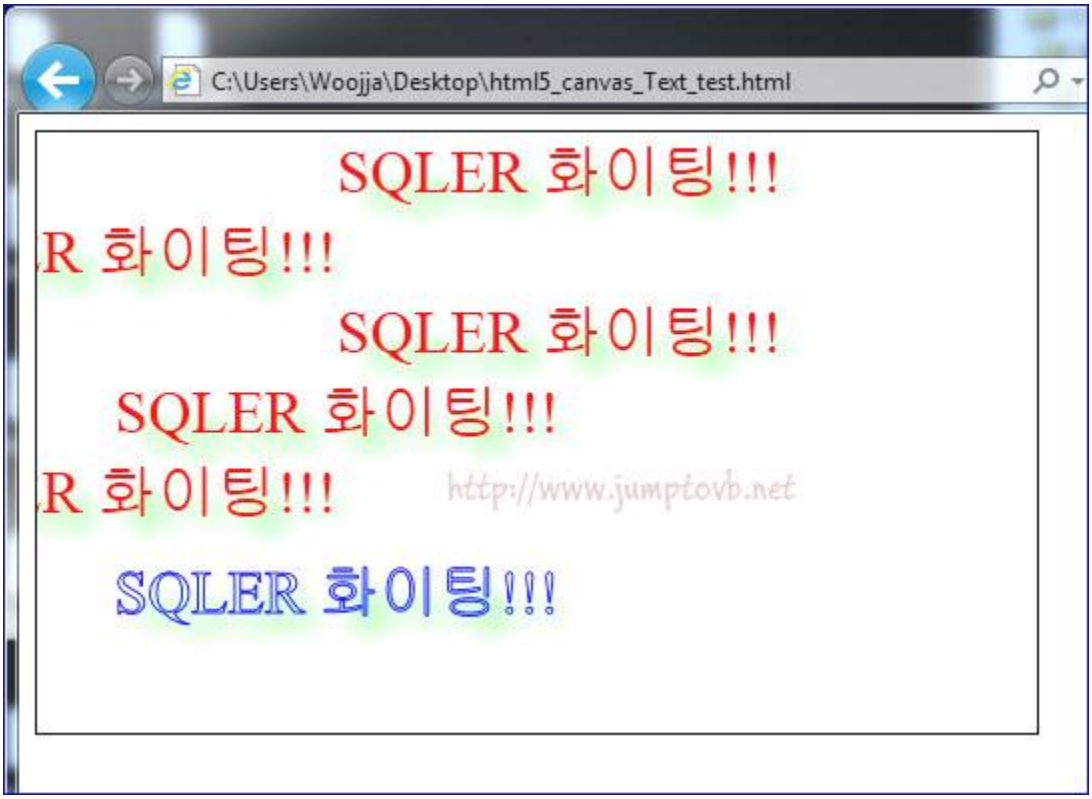
위와 같이 clothPath 함수를 이용하여 선긋기를 닫으면 속이 찬 도형을 그릴 수 있다.

10-4. 문자 넣기

문자를 Cavas 에 넣는 방법도 어렵지 않다. 아래에는 여러가지 textAlign 이 적용되어 있다. shadow 효과도 적용되어 있다.

```
<!DOCTYPE HTML>
<html lang="ko">
  <head>
    <meta charset="UTF-8">
    <title>HTML 5 Canvas 예제</title>
     <style type="text/css">
     #woojjaCanvas { border:1px solid #000000;}
     </style>
  </head>
  <body>
    <canvas id="woojjaCanvas" width="500" height="300"></canvas>
    <script type="text/javascript">
      var canvas=document.getElementById('woojjaCanvas');
      var cvas=canvas.getContext('2d');

      cvas.font = "30px Gothic";
      cvas.fillStyle = "rgba(255,0,0,1)";

      cvas.shadowColor = "rgba(0,255,0,0.9)";
      cvas.shadowOffsetX="10";
      cvas.shadowOffsetY= "10";
      cvas.shadowBlur="15";

      cvas.textAlign = "start";
      cvas.fillText("SQLER 화이팅!!!", 150, 30);

      cvas.textAlign = "end";
      cvas.fillText("SQLER 화이팅!!!", 150, 70);

      cvas.textAlign = "left";
      cvas.fillText("SQLER 화이팅!!!", 150, 110);

      cvas.textAlign = "center";
      cvas.fillText("SQLER 화이팅!!!", 150, 150);

      cvas.textAlign = "right";
      cvas.fillText("SQLER 화이팅!!!", 150, 190);

      cvas.textAlign = "center";
      cvas.strokeStyle = "#0000FF";
      cvas.strokeText("SQLER 화이팅!!!", 150, 240);

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

fillText 함수와 strokeText 함수에 따라서 Text 의 모양을 결정할 수 있다. shadow 관련 함수에 따라서 아래와 같이 Text 에 음영을 적용할 수 있다.



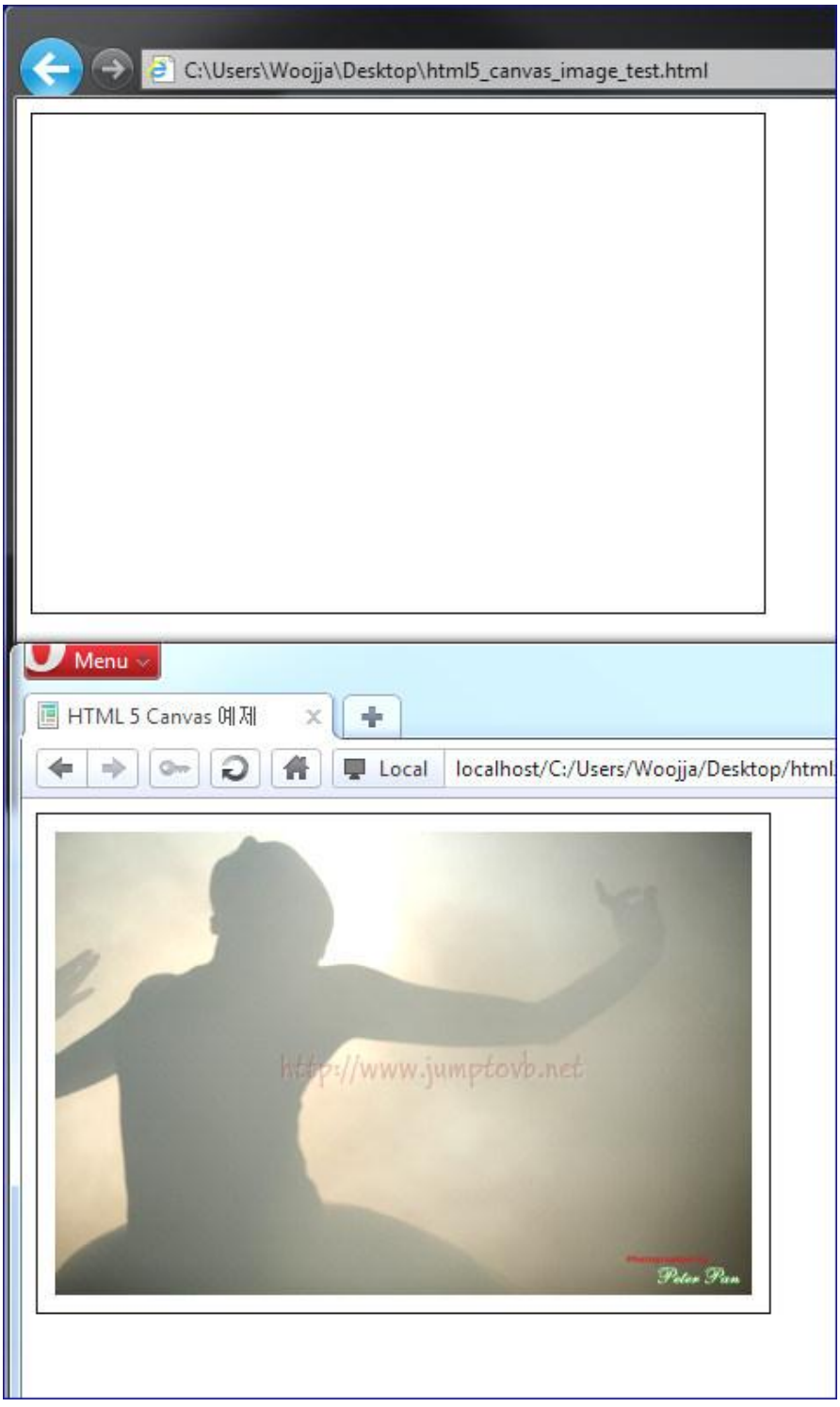
위에 보시는 바와 같이 textAlign 값 변경에 따라서 배치가 달라지는 것을 보실 수 있다.

10-5. 이미지 삽입

이미지를 삽입하는 방법은 아래와 같다.

```
<!DOCTYPE HTML>
<html lang="ko">
   <head>
      <meta charset="UTF-8">
      <title>HTML 5 Canvas 예제</title>
      <style type="text/css">
         #woojjaCanvas { border:1px solid #000000;}
      </style>
      <script type="text/javascript">
         function drawCanvas() {
            var canvas=document.getElementById('woojjaCanvas');
            var cvas=canvas.getContext('2d');
            var ballet = new Image();
            ballet.src='IMG_4687.jpg';
            cvas.drawImage(ballet, 10, 10);
         };
      </script>
   </head>
   <body onload="drawCanvas();">
      <canvas id="woojjaCanvas" width="420" height="286"></canvas>
   </body>
</html>
```

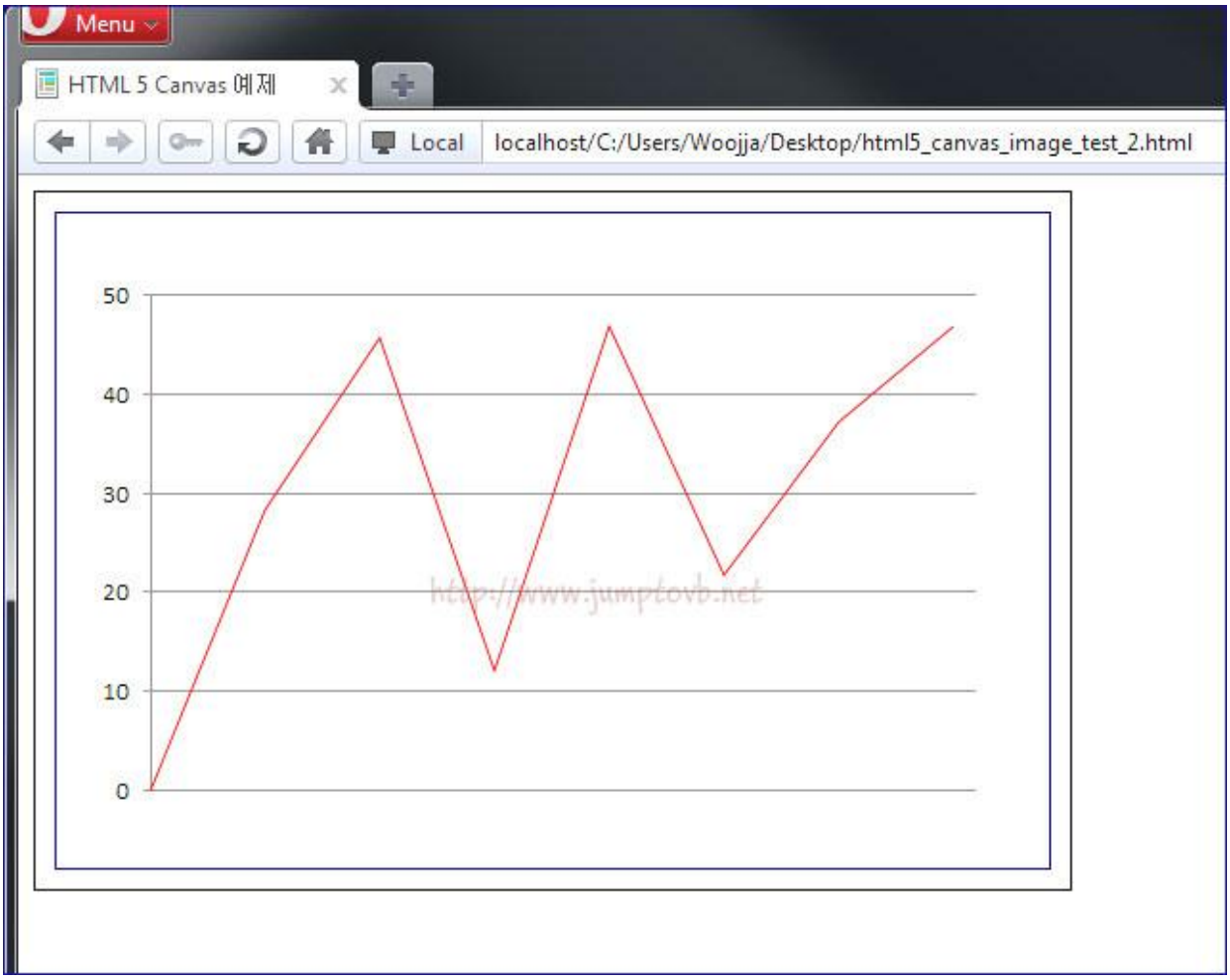
drawImage 함수를 사용하여 image 를 Canvas 에 그려 넣는다.


C:\Users\Woojja\Desktop\html5_canvas_image_test.html
Menu
HTML 5 Canvas 예제
Local
localhost/C:/Users/Woojja/Desktop/html
http://www.jumptovb.net

10-6. 그래프

그럼 이미지와 선들을 함께 그려서 그래프를 그려보도록 하겠다.

```
<!DOCTYPE HTML>
<html lang="ko">
   <head>
      <meta charset="UTF-8">
      <title>HTML 5 Canvas 예제</title>
      <style type="text/css">
         #woojjaCanvas { border:1px solid #000000;}
      </style>
      <script type="text/javascript">
         function drawCanvas() {
            var canvas=document.getElementById('woojjaCanvas');
            var cvas=canvas.getContext('2d');
            var ballet = new Image();
            ballet.src='canvas_back.png;

            cvas.drawImage(ballet, 10, 10);
            cvas.beginPath();
            cvas.strokeStyle='#FF0000';

            cvas.lineTo(60,313);
            cvas.lineTo(120,166);
            cvas.lineTo(180,76);
            cvas.lineTo(240,250);
            cvas.lineTo(300,70);
            cvas.lineTo(360,200);
            cvas.lineTo(420,120);
            cvas.lineTo(480,70);

            cvas.stroke();

         };
      </script>
   </head>
   <body onload="drawCanvas();">
      <canvas id="woojjaCanvas" width="541" height="364"></canvas>
   </body>
</html>
```



배경이미지를 그리고나서 그 위에 선을 긋는 방식으로 그래프를 완성 했다.

위 Markup 의 값과 그래프의 값이 맞지는 않지만 이런 방식으로 그린다는 것을 볼 수 있다.

10-7. 여러 예제 사이트

지금까지는 Canvas 를 단순하게 다루는 것을 설명했었지만, 캔버스는 정말 많은 것을 만들어 낼 수 있다.

1) Test Drive Site Map

http://ie.microsoft.com/testdrive/Views/SiteMap/Default.html

내의 Canvas 예제

Canvas Pad

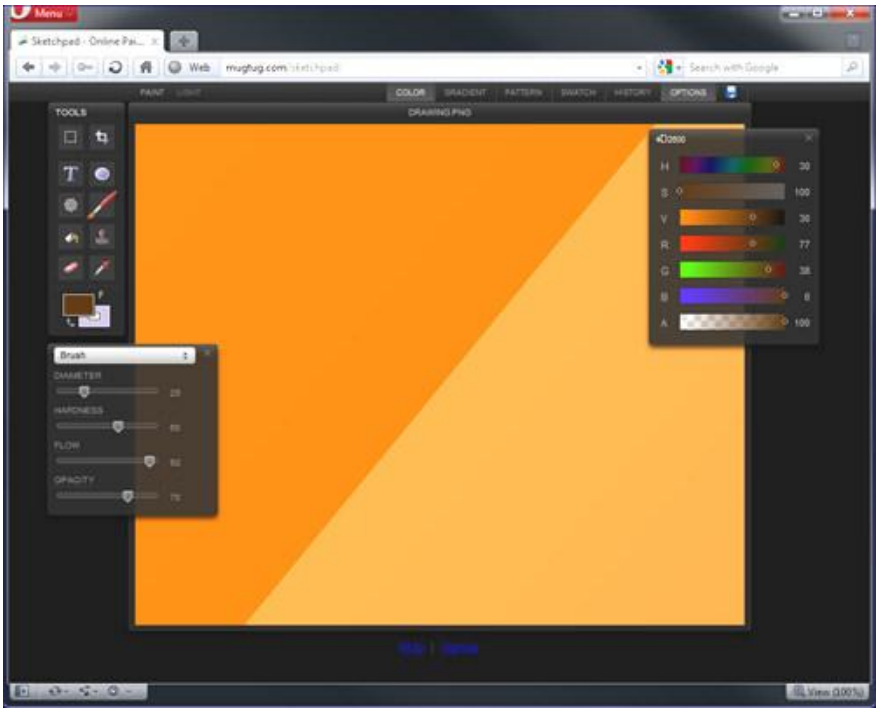
Canvas Pinball

Canvas Zoom

2) Tron

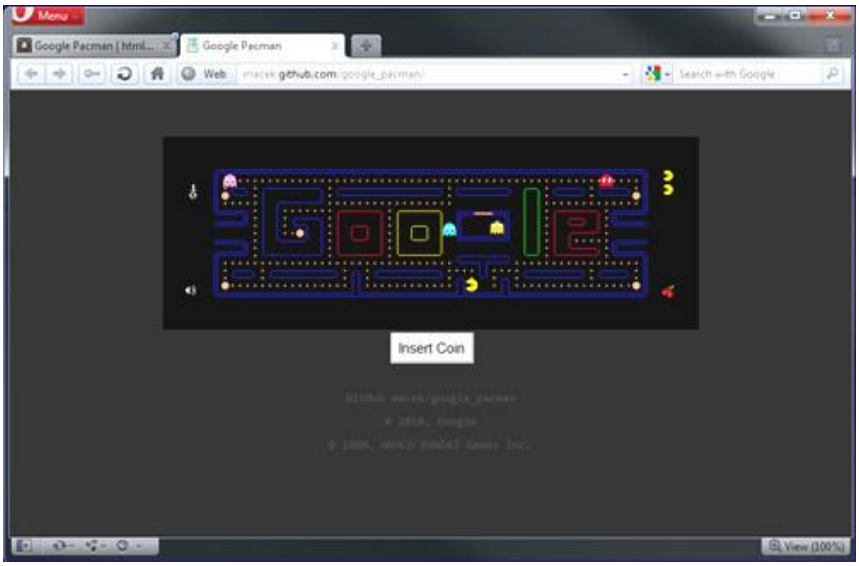
http://disneydigitalbooks.go.com/tron/

3) MugTug



http://mygtug.com/sketchpad/

4) Pacman



http://html5games.net/game/google-pacman

[HTML5 강좌] 11. Drag & Drop API

Drag & Drop API 에 대해서 알아보려 한다. 가끔 우리는 사진, 파일 같은 항목들을 Drag & Drop 할 수 있도록 만든 사이트들을 보았을 것이다. 이전에 구현되어있던 Gmail 의 Drag & Drop 은 mouseDown, mouseOver 같은 Mouse 이벤트를 이용하여 Javascirpt 로 구현을 한 것인데 Javascript 를 사용하지 않았다면 ActiveX 를 사용 했을 것이다. 이런 Javascript 나 ActiveX 는 별도의 설치가 필요하거나 Javascript 를 사용함으로써 별도의 부하가 생기고 하지만, 이제 HTML 5 에서는 Drag & Drop API 를 제공하여 좀 더 손쉽게, 간단하게 다른 플러그인들이 도움없이도 구현할 수 있게 되다.

11-1. draggable 속성

HTML 5 에 새로이 추가된 속성중에 draggable 속성이 있습니다. true, false 값을 가질 수 있다. draggable 속성이 true 이면 drag 할 수 있다.

```
<img draggable="true" src="IMG_4687.jpg"></img>
```

사실 HTML 5 에서는 img Element 와 href 속성이 지정된 A Element 만 기본값 상태에서 Drag 할 수 있도록 되어 있다. 하지만 Draggable 값이 True 라고 해서 무조건 Drag 할수 있는 것은 아니지만 API 를 이용한 Javascript 를 사용하셔야 움직이실 수 있습니다.

11-2. Drag & Drop 관련 Event

그럼, HTML 5 에 새롭게 추가된 Drag & Drop API 의 관련 Event 들을 살펴 보겠다.

Event 이름	Event 가 발생하는 곳	Action
dragstart	Drag 를 할 Element	Drag 가 시작됨
drag	Drag 를 할 Element	Drag 중
dragenter	Drag 중 Mouse 가 Element 와 겹치는 순간의 Element	Element 에 진입했음
dragleave	Drag 중 Mouse 가 Element 에서 떠나는 순간의 Element	Element 를 떠남
dragover	Drag 중 Mouse 가 현재 위치한 Element	Element 위를 지나고 있음
drop	Drop 할 Element	Drop 됨
dragend	Drag 하고 있는 Element	Drag 가 끝남

특히나 dragenter, dragover, drop 이 세 event 는 Drop 과 관련하여 적절히 사용할 필요가 있습니다.

11-3. DataTransfer

DataTransfer 객체는 Drag & Drop API 를 사용하는데 있어서 반드시 있어야 하는 객체이다. 이것은 Drag & Drop 되는 것의 Data 를 담는 역할을 한다. 그럼, DataTransfer 의 속성과 Method 를 살펴 보겠다.

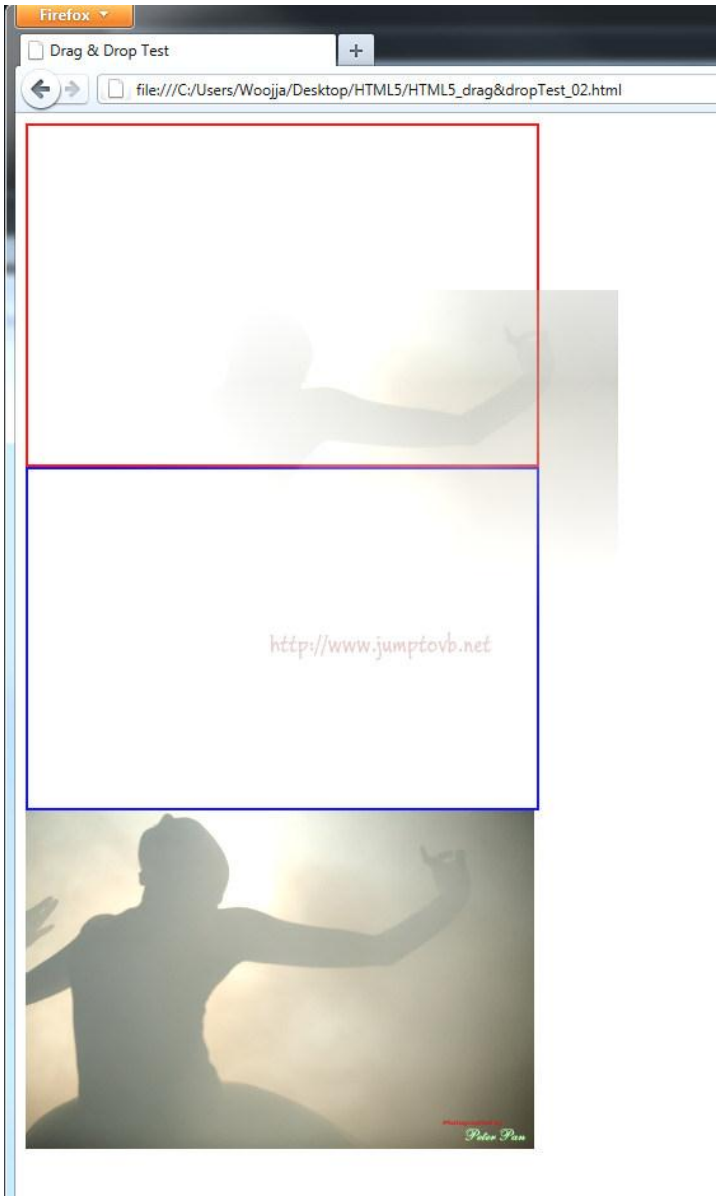
속성/Method	설명
dropEffect	Drag & Drop 동작의 종류를 나타내는 문자열 copy, move, link, none(초기값) 중 하나 effectAllowed 속성에 의해 허용되지 않은 종류의 동작은 처리할 수 없음.
effectAllowed	허용할 dropEffect 를 지정하는 문자열 copy, move, link 와 같은 동작을 허용하며 copyLink, copyMove, linkMove, all 과 같은 둘 이상의 동작을 동시에 허용할 수도 있음. none(drop 을 허용하지 않음) 을 지정할 수 있음. dragstart 이벤트에 값을 지정하는 것이 일반적임
Types	dragstart 이벤트 발생시 DOM 목록에 있는 data format 을 설정하며 setData 함수를 호출할때 지정되는 format 문자열을 배열형식으로 얻을 수 있음.
clearData(type)	Drag & Drop 에 사용할 데이터를 초기화하며 Drag 중인 데이터를 삭제함.
setData(type, data)	Drag & Draop 하기위해 새로운 요소를 정의하며 Drag & Drop 할 데이터를 저장함.
getData(type)	setData 에서 정의한 Element 와 Format, 데이터를 가져옴.
Files	다른 application 으로부터 Drag & Drop 할 파일을 가져옴. types 속성에 "Files" 라는 문자열이 저장됨.
setDragImage(image, x, y)	Drag 중 img Element 를 이용하여 피드백을 지정함.
addElement(element)	Drag 중 피드백 image 에 추가할 Element 를 추가함.

11-4. 예제...

이제 간단한 Drag & Drop 예제를 만들어 보겠다.

```
<!DOCTYPE html>
< html>
<head>
<title>Drag & Drop Test</title>
<script type="text/javascript">
function drag(target, e) {
e.dataTransfer.setData('Text', target.id);
};
function drop(target, e) {
var id = e.dataTransfer.getData('Text');
target.appendChild(document.getElementById(id));
```

```
e.preventDefault();
};
</script>
<style>
div#Red {border:2Px solid #F00;}
div#Blue {border:2Px solid #00F;}
div {width:400px;height:266px;}
a {margin:50px;display:block;}
</style>
</head>
<body>
<div id="Red" ondrop="drop(this, event);" ondragenter="return false;" ondragover="return
false;"></div>
<div id="Blue" ondrop="drop(this, event);" ondragenter="return false;" ondragover="return
false;"></div>
<img draggable="true" id="textlink" ondragstart="drag(this, event);" src="IMG_4687.jpg"></img>
</body>
< /html>
```



위와 같이 firefox, safari, chrome 는 이미지를 이동시킬때 기본적으로 이미지를 표시하지만 ie 의 경우는 마우스커서를 파일 복사할 때 나타내는 커서로 변경시킨다.

[HTML5 강좌] 12. Offline Web Application

Offline Web Application 이란 말그대로 Offline 상태에서도 동작하는 Application 이다.

12-1. Manifest 작성

Offline Web Applicaiton 은 Offline 상태에서 Web Application 을 동작시키기 위해서 Cache 를 사용한다. Cache 를 사용하기 위해서는 어떤 파일들을 Cache 해야하는지도 알려주어야 하는데 이때 사용하는 것이 manifest 파일이다.

먼저 manifest 파일은 Browser 에 의해서 Download 되는 파일이므로 **웹서버에 manifest 파일에 대해서 "text/cache-manifest" 라는 MIME Type 으로 등록**해 주어야 한다. html5news.html 파일을 작성했다고 생각해 보겠습니다. 이 페이지는 Offline 에도 작동하려면 html5news.manifest 를 작성해야 한다.

CACHE MANIFEST 부분은 반듯이 맨 윗에 포함되어야하는 필수 사항이다.

```
CACHE MANIFEST
#Cache Section
html5news.html
html5news.js
html5news.css
IMG_4687.jpg
#Network Section : 무조건 Network 상태에서만 사용할 수 있습니다. 아래 파일과 Directory 에 있는 파일들은 Offline 에서는 사용할 수 없습니다.
NETWORK:
search.aspx
/core/
#Fallback Section : html5news.html 파일을 찾을 수 없는 경우 notice.html 을 cache 하도록 합니다.
FALLBACK:
html5news.html notice.html
```

그리고 이 Manifest 파일은 사이트에 접속을 하여 해당 페이지에 접근을 하게 되면 다운로드되도록 해야 하는데 아래와 같은 코드를 그렇게 하기위해 **<HTML> Tag 에 추가**해 준다.

```
<!DOCTYPE html>
<html manifest="html5news.manifest">
.
.
.
</html>
```

12-2. Offline Web Application API 사용

그럼 이제 Web Browser 단에서 Offline Web Application API 를 지원하는지 Check 하고, online, offline 시 각각 어떻게 처리해야하는지를 정의해야 한다.

아래와 같은 Code 를 사용하여 현재 Browser 가 Offline Web Application API 를 지원하는지 여부를 check 할 수 있다.

```
if(window.applicationCache) {
…
}
```

아래와 같은 Code 로 Network 이 Online 인지 Offline 인지를 Check 한다.

```
function init() {
    if (navigator.onLine) {
        …
    } else {
        …
    }
}
```

그리고 online , offline 으로 상태가 바뀔때 어떻게 처리할 지에 대한 event Handler 를 추가해 준다.

```
window.addEventListener("online", function(e) {
    …
}, true);

window.addEventListener("offline", function(e) {
    …
}, true);
```

12-3. applicationCache 의 Status

applicationCache 는 다음과 같은 상태값을 갖는다.

상태	정수값	설명
UNCACHE	0	Cache 하지 않음.
IDLE	1	최신 Cache 를 이용 중.
CHECKING	2	Update Check 중.
DOWNLOADING	3	Update Download 중.
UPDATEREADY	4	최신 Cache 를 이용할 수 있음.
OBSOLETE	5	Error 에 의해 cache 무효화.

12-4. applicationCache 의 event

applicationCache 로 부터 발생하는 Event 는 다음과 같다.

상수	설명
checking	Update Check 중.
error	Update 가 Error 로 종료.
noupdate	Manifest 가 Update 되지 않음.
downloading	Update Download 중.
progress	Update Download 중.
updateready	최신 cache 얻기 완료. swapCache() 함수를 호출할 수 있음.
cached	cache 완료.
obsolete	manifest 얻기에 실패하여 cache 를 무효로 함.

12-5. Sample

html5news.manifest 파일의 내용입니다.

```
CACHE MANIFEST
# JavaScript
./html5newsOffline.js
./html5newsLog.js

# stylesheets
./html5news.css
```

html5news.html 파일의 내용입니다.

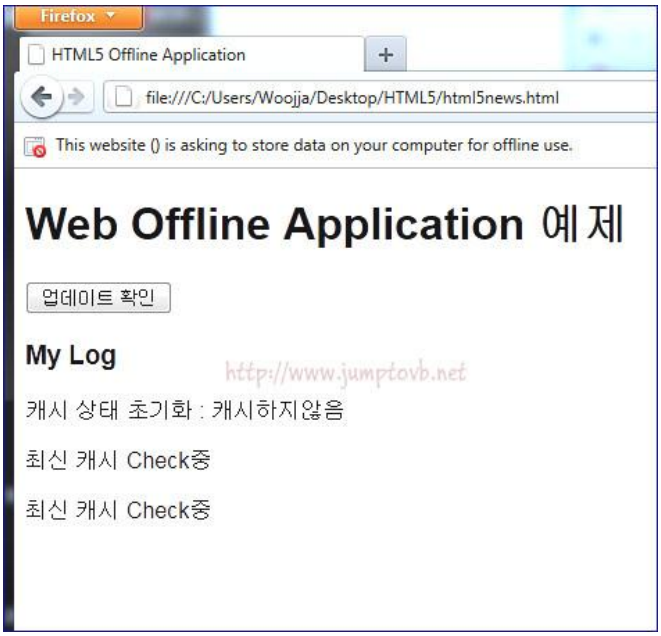
```
<!DOCTYPE html>
<html lang="ko" manifest="html5news.manifest">
   <head>
   <title>HTML5 Offline Application</title>
      <script src="html5newsLog.js"></script>
      <script src="html5newsOffline.js"></script>
      <link rel="stylesheet" href="html5news.css">
   </head>
   <script type="text/javascript">
   </script>
   <body onload="init();">
      <header>
        <h1>Web Offline Application 예제</h1>
      </header>

      <section>
      <article>
         <button id="btnCheckUpdate">업데이트 확인</button>
         <h3>My Log</h3>
         <div id="info">
         </div>
       </article>
      </section>
   </body>
</html>
```

html5newsOffline.js 파일의 내용입니다.

```
window.applicationCache.onchecking = function(e) {
	log("Application update 사항을 Check 하고 있습니다.");
}

window.applicationCache.onnoupdate = function(e) {
   log("application update 할 사항이 없습니다.");
}

window.applicationCache.onupdateready = function(e) {
   log("Application 을 update 할 준비가 되었습니다.");
}

window.applicationCache.onobsolete = function(e) {
	log("Cache 를 삭제합니다.");
}

window.applicationCache.ondownloading = function(e) {
	log("application update 사항을 Download 하고 있습니다.");
}

window.applicationCache.oncached = function(e) {
   log("Application cached 되었습니다.");
}

window.applicationCache.onerror = function(e) {
	log("Application cache error 입니다.");
}

window.addEventListener("online", function(e) {
	log("Online 입니다.");
}, true);

window.addEventListener("offline", function(e) {
	log("Offline 입니다.");
}, true);


showCacheStatus = function(n) {
	 statusMessages = ["캐시하지않음","최신 캐시이용중","최신 캐시 Check중","Download 중","최신 캐시 사용가능","캐시 무효
화"];
return statusMessages[n];
}
```

```
install = function() {
        log("최신 캐시 Check중");
        try {
            window.applicationCache.update();
        } catch (e) {
            applicationCache.onerror();
        }
}

init = function(e) {
    if (!window.applicationCache) {
            log("이 브라우져는 HTML5 Web Offline Application API 를 지원하지 않습니다.");
            return;
    }

        log("캐시 상태 초기화 : " + showCacheStatus(window.applicationCache.status));
    document.getElementById("btnCheckUpdate").onclick = install;
}
```

(Pro HTML5 Programming 참고)



이번엔 Web Offline Application API 에 대해서 살펴보았다.

[HTML5 강좌] 13. Communication API

이젠 Communication API 를 살펴보려 한다. 다음의 내용과 소스는 웹혁명을 꿈꾸다 HTML5 & API 입문을 참고 하였다. Communication API 는 MessageEvent 라고 하는 javaScript 객체나 문자열을 비동기 방식으로 주고 받음으로써 여러프로그램간의 데이터를 공유하게 한다. 이러한 방식은 다음에 살펴보게될 Web Worker 나 Server-Sent Event 등에서도 사용하는 방식입니다. 공통의 API 를 사용한다는 점이 장점이라고 할 수 있다.

13-1. MessageEvent

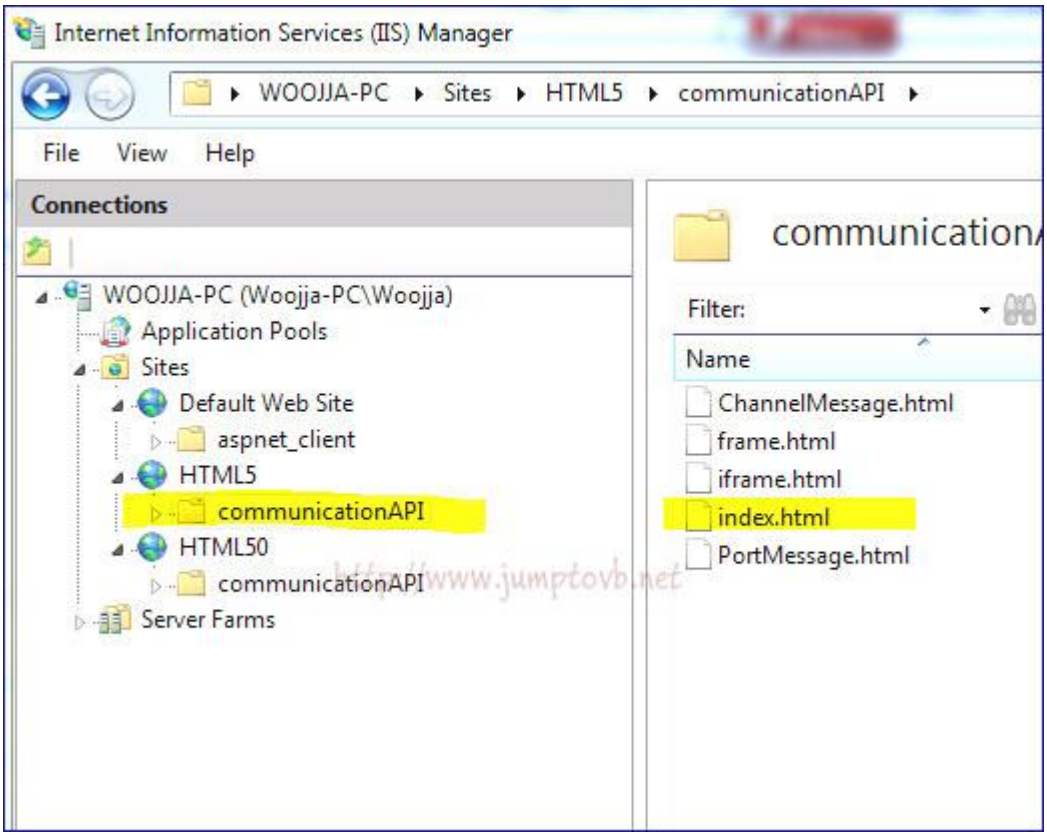
그럼 Communication API 의 중심인 MessageEvent 부터 살펴보도록 하겠다. 앞에서도 말한바와 같이 MessageEvent 는 두 지점간에 주고 받는 메세지를 말하는데요. Javascript 객체이다.

그리고 다음과 같은 속성을 가지고 있다.

속성	설명
data	송신되는 Message 의 내용이 되는 데이터
origin	Message 송신처의 Domain (Cross Document Messaging 과 Server Sent Event 에서만 사용)
lastEventId	마지막 Event ID (Server Sent Event 에서만 사용)
source	Message 를 보내는 Windows 객체(Cross Document Messaging 에서만 사용)
ports	Message 송신시 지정한 포트의 복사본

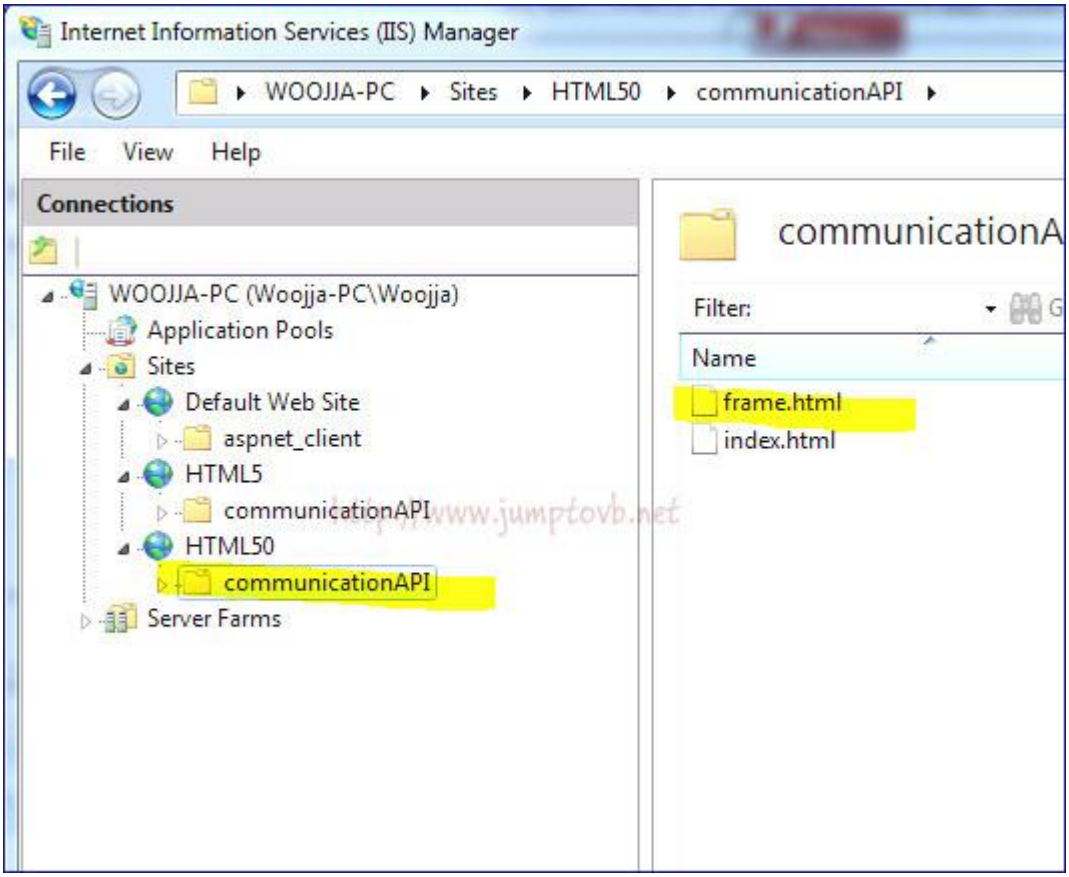
13-2. Cross Document Messaging

Cross Document Messaging 을 사용하면 둘 이상의 웹페이지가 서로 데이터를 주고 받을 수 있다. 가장 큰 특징은 다른 도메인간의 통신이 가능해 졌다는 것이다. 구현도 간단해서 수신측 Window 의 postMessage() Method 를 호출하고 수신하는 곳에서는 자신의 window 에 대해 onmessage Event Handler 를 지정하면 된다. 일단 이것을 구현하기 위해서 웹서버에 Site 를 두 개 만들었다.
구성은 아래 그림과 같다.



비슷해 보이지만 위의 사이트에는 iframe Tag 를 포함한 index.html 페이지가 있다. 위 사이트는 8880 번 포트를 사용하고 있다

아래에는 iframe 의 Source 가 되는 페이지인 frame.html 파일이 있다. 아래 사이트는 8881 번 포트를 사용하고 있다.



index.html 페이지에서 타이머를 돌려서 매 초당 한번씩 계속 iframe 의 Source 인 frame.html 페이지에 데이터를 전송 된다.

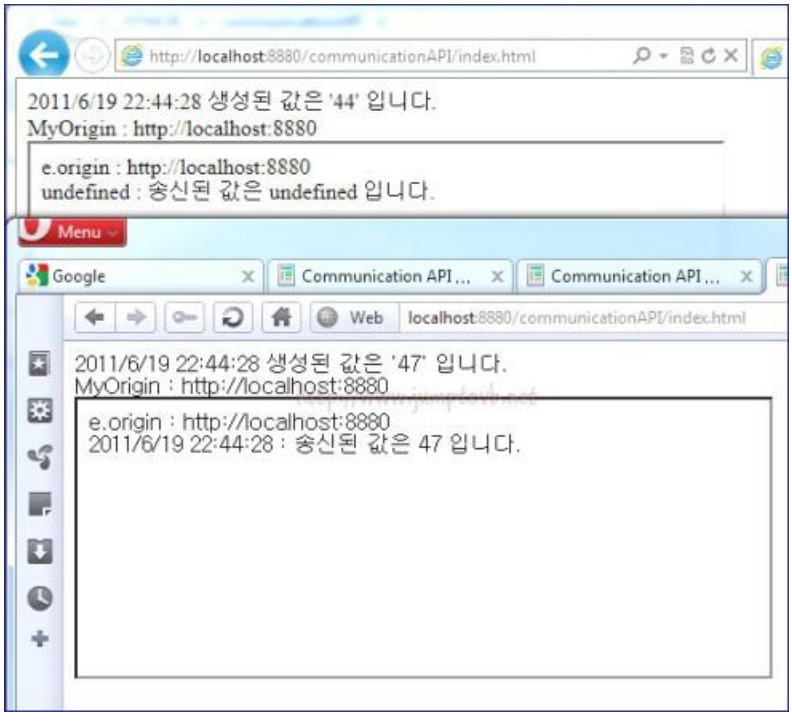
index.html 페이지의 내용이다.

```
<!DOCTYPE html>
< html lang="ko">
  <head>
      <meta charset="utf-8" />
      <title>Communication API 예제</title>

      <script type="text/javascript">
        function init() {
          //1초마다 IFrame 에 메시지를 보냄
          setInterval(Send, 1000);
        }

        function Send(){
          var ifrm = document.getElementById("ifrm");
          var MyOrigin = location.protocol + "//" + location.host;
          var date = new Date();
          var dateStr = date.getFullYear() + "/" + (date.getMonth() + 1) + "/" + date.getDate() + " " + date.getHours() + ":" +
date.getMinutes() + ":" + date.getSeconds();
          var number = Math.floor(Math.random() * 100);
          ifrm.contentWindow.postMessage({date:dateStr, number:number}, "http://localhost:8881");
          document.getElementById("msg").innerHTML = dateStr + " 생성된 값은 '" + number + "' 입니다. <br/> MyOrigin : " +
MyOrigin;
      }
      </script>
  </head>
  <body onload="init();">
    <div id = "msg">8880<br/>MyOrigin</div>
    <iframe id="ifrm" src="http://localhost:8881/communicationAPI/frame.html" width=500 height=200></iframe>
  </body>
< /html>
```

데이터를 받는 쪽의 내용을 살펴보면 아래와 같이 구현된다.

```
<!DOCTYPE html>
< html lang="ko">
  <head>
   <meta charset="utf-8" />
   <title>Communication API 예제</title>
        <script type="text/javascript">

        var MyOrigin = http://localhost:8880;
        window.addEventListener("message", function(e){
                if(e.origin==MyOrigin) {
                   document.body.innerHTML = "<div>e.origin : " + e.origin + " <br/> " + e.data.date + " : 송신된 값은 " +
e.data.number + " 입니다.</div>";
                }
              }, false);

        </script>
  </head>
  <body>
   <div id = "msg"></div>
 8881
  </body>
< /html>
```

위 Markup 에서 가장 중요한 부분은 addEventHandler 도 있지만 바로 "e.origin==MyOrigin" 이 부분이다. 이렇게 보낸측의 Domain 을 확인하지 않는다면 위와 같이 대상 사이트를 iframe 에 담고 postMessage() 로 계속해서 메세지를 보내는 것도 가능하다.
하지만 보안적으로다가 취약한 점이 있으므로 **반드시 받는 쪽에서는 보낸곳의 Domain 을 확인**해야 한다.



13-3. Channel Messaging

이번엔 Channel Messaging 에 관한 내용이다. Channel Messaging 은 다대다 메세지 통신을 실현하기위한 API 이다.

MessageChannel 이라는 Channel 을 통해 Message 를 송수신하게 되는데 Message 의 출입구가 되는 Channel 에는 두개의 Port 가 있어서 Port1 에 수신된 Message 는 Port2 로 Port2 에 수신된 Message 는 Port1 으로 전달된다.

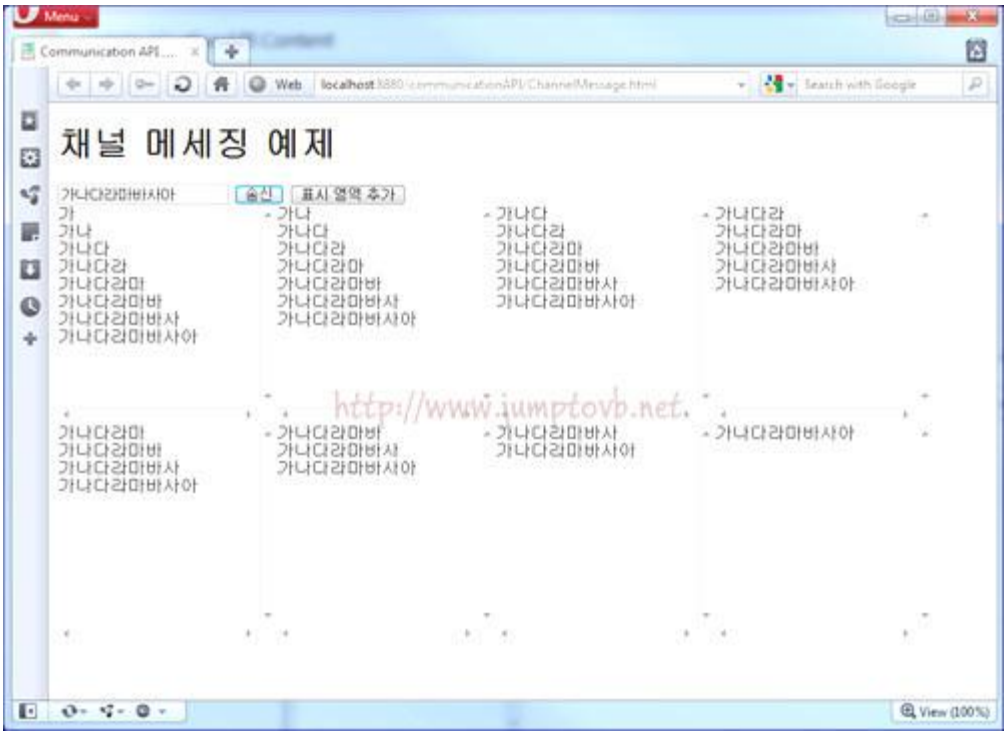
이런 각각의 Port 는 MessagePort 라는 Type 의 Object 로 다음과 같은 Method 와 속성을 가진다.

Method 명	설명
start()	Port 를 사용할 수 있게 합니다. 두개의 Port 모두 Start 해주어야 합니다.
close()	Port 를 사용할 수 없게 합니다.
PostMessage	Port 에 Message 를 보냅니다. 보낸 Message 는 상대편 Port 로 부터 읽어들여집니다.
onmessage	Port 에 Message 가 도착하면 이 속성에 지정된 Event Handler 가 호출됩니다.

그럼 예제를 만나보도록 하겠습니다.

```
<!DOCTYPE html>
< html lang="ko">
<head>
  <meta charset="utf-8" />
   <title>Channel Messaging 예제</title>
       <style>
         .messageLog{
            width:200px;height:200px;
            overflow:scroll;float:left;
         }
       </style>
       <script type="text/javascript">

           var channel = new MessageChannel();

           channel.port1.start();
           channel.port2.start();

         function addDisplay() {
            var div = document.createElement("div");
            div.className = "messageLog";
            document.getElementById("messageLogs").appendChild(div);

            channel.port2.addEventListener("message", function(e){
              div.innerHTML += "<div>" + e.data + "</div>";
              }, false);

         }
         function showMsg() {
            var msg = document.getElementById("msg").value;
            channel.port1.postMessage(msg);
         }
       </script>
  </head>
  <body>
       <h1>채널 메세징 예제</h1>
   <input id="msg" type="text">
       <button onclick="showMsg()">송신</button>
       <button onclick="addDisplay()">표시 영역 추가</button>
       <div id="messageLogs"></div>
  </body>
< /html>
```

Channel 을 생성한 후에 두 개의 Port 를 시작하고, port 1 의 postMessage 를 통해 Message 를 보내고, port2 에서 Message 를 처리하는 부분이 중요한 부분이다. 아래 그림은 입력란에 "가나다라마바사아" 를 한글자씩 늘려가면서, 표시영역을 하나씩 추가하여 송신을 늘렸다.



그리고 참고로 Communication API 에는 포트 공개라고 하는 것이 하나 더 있다..

[HTML5 강좌] 14. Web Storage

이젠 Web Storage 를 살펴보려 한다.
다음의 내용과 소스는 혁명을 꿈꾸다 HTML5 & API 입문 앞서가는 디자이너와 퍼블리셔를 위한 HTML5 & CSS 을 참고 했다. Offline Web Application 과 비슷하게 갑자기 전원이 꺼진다거나 다시 PC 를 켰을때 이전에 작업한 데이터를 보존할수도 있고, 웹메일을 Web Storage 에 저장해 두었다가 읽는다거나, 서버의 많은 정보를 Client 에 저장해 둘 수 있다.

Web Stroage 는 Client 의 disk 에 소량의 데이터를 저장하기 위한 Storage 로 이전까지는 Cookie 를 사용했습니다만 Cookie 와는 다음과 같은 몇가지 다른 점이 있다.

크기에 제한이 없음.	Cookie 는 4KB 로 제한이 있지만 Web Storage 는 제한이 없습니다.
서버로 보내지지 않음.	Cookie 는 HTTP Request 에 의해서 자동으로 서버에 전송이 되었지만 Web Storage 는 서버로 전송되지 않습니다.
유효기간의 제한이 없음.	Cookie 처럼 특정기간이 지나면 자동으로 삭제되지 않습니다.
JavaScript 객체를 저장할 수 있음.	JavaScript 객체를 저장할 수 있습니다.

Web storage 는 Web Browser 마다 별도로 가지고 있는 저장 공간에 Data 를 Key-Value 형식으로 저장한다.

Web Storage 는 용도에 따라서 Local Storage 와 Session Storage 로 나뉘는데요. 두 Storage 의 차이는 저장기간이나 유효범위만 다를 뿐 거의 같은 API 를 사용한다.

그럼 Local Storage 부터 살펴보도록 하겠습니다.

14-1. Local Storage

Local Storage 는 저장기간에 제약이 없고 Web Site 의 Domain 에 따라서 생성되는 Storage 이다. 저장기간도 사용자가 삭제하기 전까지 영구적으로 저장할 수 있다. 그렇기 때문에 기존의 Cookie 를 이용한 저장작업을 Local Storage 가 대신하기에 충분하다. 거기다 Server 측과의 통신작업도 없기 때문에 Client 들이 더욱 선호하게 될 것으로 보인다.
먼저 Web Browser 가 Local Storage 를 지원하는지 Check 해 봐야 할 것이다.

```
<script type="text/javascript">
if(!window.localStorage) {
        document.write('이 Browser 는 Local Storage 를 지원하지 않습니다.');
} else {
        .....
};
</script>
```

저장하는 방법은 아래와 같다.

```
<script type="text/javascript">
if(!window.localStorage) {
        document.write('이 Browser 는 Local Storage 를 지원하지 않습니다.');
} else {
        localStorage.setItem("NickName", "woojja");
};
</script>
```

저장후 저장한 결과를 조회하는 코드이다.

```
<script type="text/javascript">
if(!window.localStorage) {
        document.write('이 Browser 는 Local Storage 를 지원하지 않습니다.');
} else {
        localStorage.setItem("NickName", "woojja");
        alert(localStorage.getItem("NickName"));
};
</script>
```

지울 때는 다음과 같이 할 수 있다.

```
<script type="text/javascript">
if(!window.localStorage) {
        document.write('이 Browser 는 Local Storage 를 지원하지 않습니다.');
} else {
        localStorage.setItem("NickName", "woojja");
        alert(localStorage.getItem("NickName"));
        localStorage.removeItem("NickName");
};
</script>
```

한꺼번에 지우기 위해서는 다음과 같이 하면 된다.

```
<script type="text/javascript">
if(!window.localStorage) {
        document.write('이 Browser 는 Local Storage 를 지원하지 않습니다.');
} else {
        localStorage.setItem("NickName", "woojja");
        alert(localStorage.getItem("NickName"));
        localStorage.removeItem("NickName");

        localStorage.clear();
};
</script>
```

이번엔 조금 다른 방법을 살펴보도록 하겠다.

저장할 때는

```
localStorage.setItem("NickName", "woojja");
localStorage.NickName = "woojja";
localStorage["NickName"] = "woojja";
```

가져올 때는

```
var NickName = localStorage.getItem("NickName");
var NickName = localStorage.NickName;
var NickName = localStorage["NickName"];
```

지울 때는

```
localStorage.removeItem("NickName");
delete localStorage.NickName;
delete localStorage["NickName"];
```

와 같은 방법으로 사용한다.

전체 Storage Data 에 대해서

```
for(var i=0; i<localStorage.length; i++) {
        var key = localStorage.key(i);
        var value = localStorage[key];
        ...
}
```

또는

```
for(var key in localStorage) {
        var value = localStorage[key];
        ...
}
```

이렇게 접근해서 사용할 수 있다. Session Storage 도 사용 방법은 다르지 않은데 Session Storage 도 쉽게 이해 할 수 있을 것으로 보인다.

14-2. Session Storage

Session Storage 는 Local Storage 와 같이 도메인마다 생성이 되지만 Browser Window 의 유효범위와 생존기간을 갖는다. 그래서 Local Storage 와는 다르게 Browser Window 가 닫히면 Session Storage 도 같이 삭제 된다. 그럼 Browser Window 를 복제하면 어떻게 될까? 예상은 하고 계시겠지만, 복제된 순간까지는 Session Storage 의 내용은 같겠지만 그 이후의 내용은 서로에게 아무런 영향을 주지도 않고 별개의 Storage 로 작동하게 된다.

이런 특징을 갖는 Session Storage 는 사용자의 동작을 추적하여 기록할 때 사용할 수 있다.

Session Storage 의 사용은 Local Storage 와 같다고 했는데 실제로도 단지 localStorage 를 **sessionStorage** 객체로 바꿔 사용하시기만 하면 된다.

14-3. Storage Event Handling

Storage 는 Storage 의 변경에 대해서 window 객체가 Storage Event 를 발생시킨다. 다음 표는 그 Event 가 가지고 있는 속성들 이다.

속성	설명
Key	변경된 Key, clear() Method 가 호출되면 null 을 반환한다.
oldValue	변경되기 전의 값(복사본). 새로운 키로 값을 등록하였다면 null.

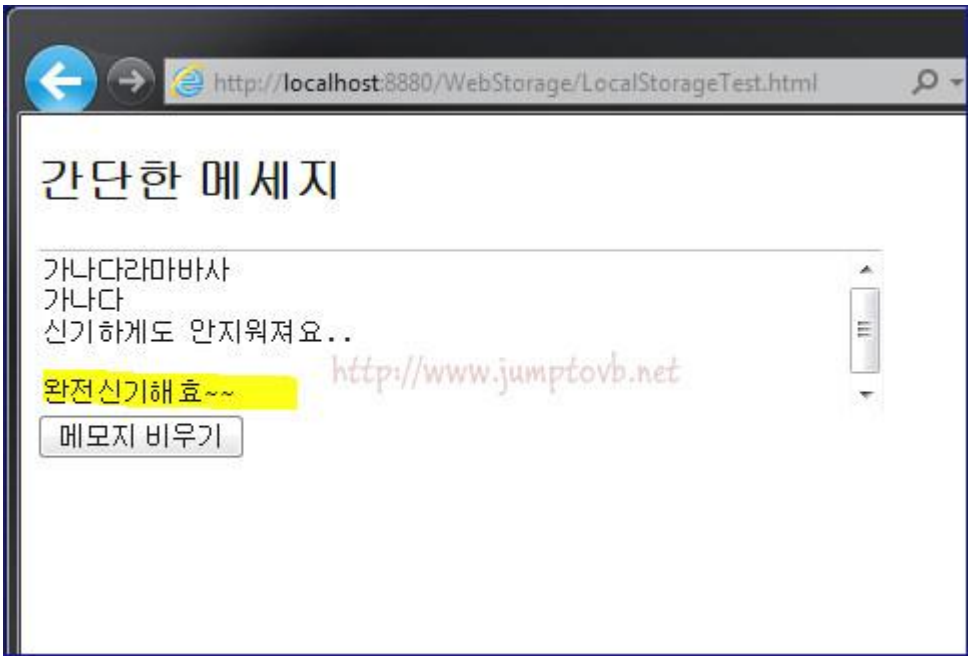
newValue	변경된 후의 값(복사본). 값이 삭제되었을 때는 null.
url	Event 가 발생한 문서의 URL.
storageArea	변경된 Storage 참조.

14-4. 예제

아래 예제는 LocalStorage 에 대한 내용입니다.

```
<!DOCTYPE html>
< html lang="ko">
    <head>
            <meta charset="utf-8" />
     <title>간단 메모지</title>
          <script type="text/javascript">
             function saveText() {
                info = document.getElementById("info");
                info.style.display = "block";
                localStorage.setItem("memo", msg.value);
             };
             function pageload() {
                msg = document.getElementById("txtBox");
                msg.value = localStorage.getItem("memo");
             };
             function clr() {
                msg.value = "";
                localStorage.clear();
                info.style.display = "none";
             };
           </script>
    </head>
    <body onload="pageload()">
          <h2>간단한 메세지</h2>
          <textarea id="txtBox" onKeyDown="saveText();" onKeyUp="saveText();" cols="50" rows="5"></textarea><br/>
          <input type="button" value="메모지 비우기" onClick="clr();" onKeyUp="clr();" />
          <p id="info" style="display:none;">메모내용이 자동 저장되었습니다.</P>
    </body>
< /html>
```

(발췌 : 앞서가는 디자이너와 퍼블리셔를 위한 HTML5 & CSS)

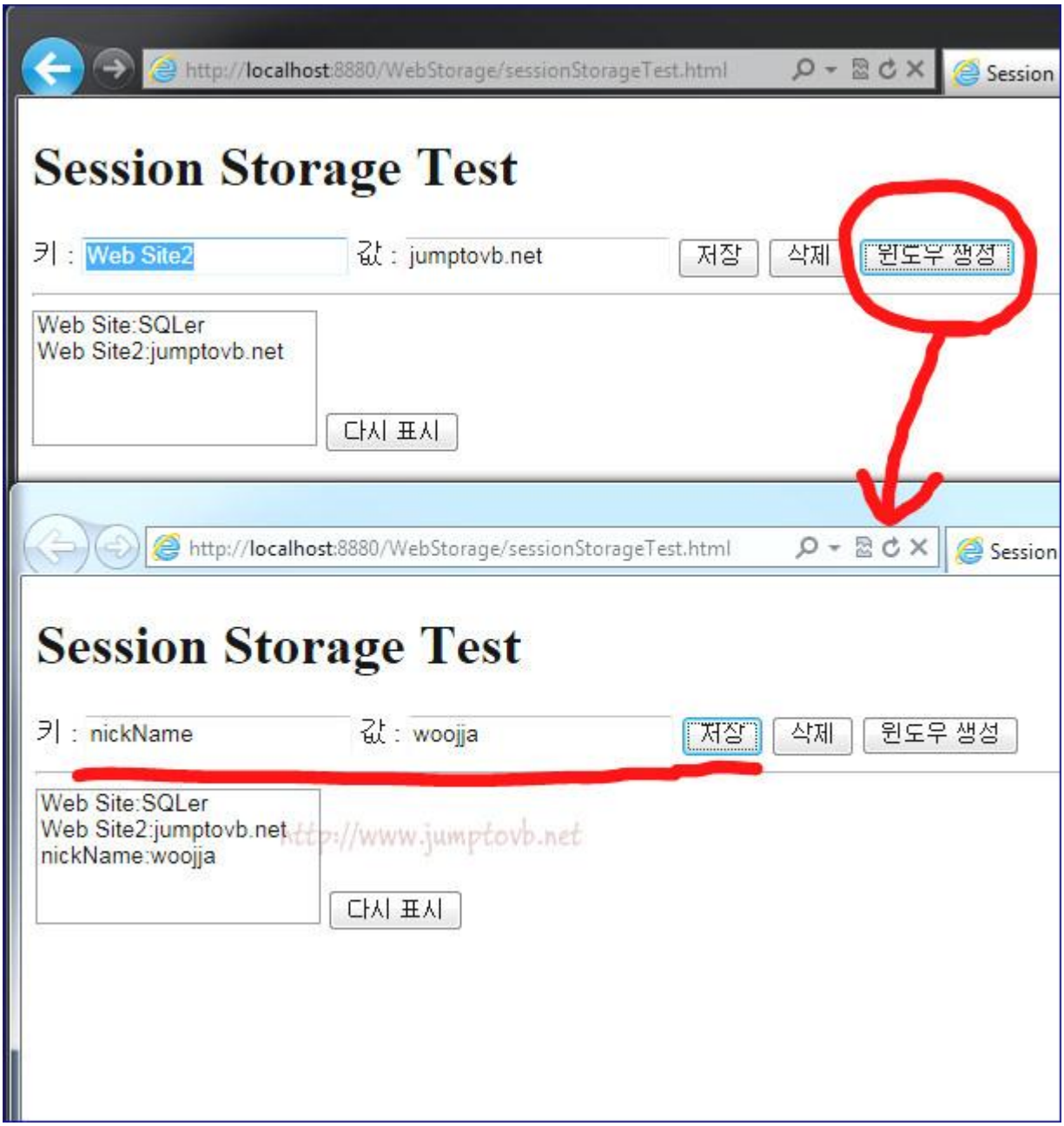


다음 예제는 session Storage 에 대한 내용이다. 값을 두 개 넣고 난 후 윈도우 창을 복제하였다.
그리고 두 번째 생긴 Window 에서 값을 넣어 본 것이다. 두 개의 storage 가 별개라는 것을 볼 수 있다.

```
<!DOCTYPE html>
< html lang="ko">
   <head>
        <meta charset="utf-8" />
    <title>Session Storage Viewer</title>
   </head>
   <body onload="ShowAll()">
        <h1>Session Storage Test</h1>
        키 : <input id="k" type="text"> 값 : <input id="v" type="text">
        <button onclick="Save()">저장</button>
        <button onclick="Remove()">삭제 </button>
        <button onclick="window.open(location.href);">윈도우 생성 </button>
        <hr>
        <select id="entries" size=5 onchange="onEntrySelected()">
        </select>
        <button onclick="ShowAll()">다시 표시 </button>

        <script type="text/javascript">

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

```
            function ShowAll() {
                entries.innerHTML = "";
                for (var i=0; i < sessionStorage.length; i++ )
                {
                    var k = sessionStorage.key(i);
                    entries.options[entries.options.length] = new Option(k + ":" + sessionStorage[k], k);
                }
            };

            function Save() {
                sessionStorage[key.value] = value.value;
                ShowAll();
            };

            function Remove() {
                delete sessionStorage[key.value];
                ShowAll();
            };

            function onEntrySelected() {
                var selectedOption = entries.options[entries.selectedIndex];
                key.value = selectedOption.value;
                value.value = sessionStorage[selectedOption.value];
            };
        </script>

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

(발췌 : 혁명을 꿈꾸다 HTML5 & API 입문)


http://localhost:8880/WebStorage/sessionStorageTest.html
Session
Session Storage Test
키 : Web Site2
값 : jumptovb.net
저장
삭제
윈도우 생성
Web Site:SQLer
Web Site2:jumptovb.net
다시 표시
http://localhost:8880/WebStorage/sessionStorageTest.html
Session
Session Storage Test
키 : nickName
값 : woojja
저장
삭제
윈도우 생성
Web Site:SQLer
Web Site2:jumptovb.net
nickName:woojja
다시 표시

[HTML5 강좌] 15. Web SQL Database

다음의 내용과 소스는 혁명을 꿈꾸다 HTML5 & API 입문 앞서가는 디자이너와 퍼블리셔를 위한 HTML5 & CSS 을 참고 하였습니다.

이번엔 Web SQL Database 를 알아볼 차례이다. Web SQL Database 는 지난 아티클에서 소개한 Storage 와 함께 기본적인 Web Storage 중 하나이다. 말씀 안드려도 알고 계시겠지만 Client 에 데이터를 저장한다는 공통점을 가지고 있다. 개발자분들께서는 SQL 에 익숙하시니 금방 접근 하실수 있으리라 생각된다. 그럼 Database 를 생성하는 작업부터 시작해 보겠다.

15-1. Create Database

생성하기 전에 먼저 사용하는 Web Browser 가 Web SQL Database 를 지원하는지 확인해 볼 필요가 있다.

```
if (!window.openDatabase) {
        document.write("이 Browser 는 Web SQL Database 를 지원하지 않습니다.");
        return false;
} else {
        ...
}
```

다음과 같은 Database 를 생성하는 Method 를 제공합니다.

```
if (!window.openDatabase) {
        document.write("이 Browser 는 Web SQL Database 를 지원하지 않습니다.");
        return false;
} else {
        var db = openDatabase("MyFirstDatabase", "1.0", "첫번째 테스트 데이터베이스", "2*1024*1024");
}
```

openDatabase 함수를 사용하여 Database 를 생성한다. 그리고 함수에 사용되는 parameter 들은 아래와 같다.

```
openDatabase("Database 이름", "Database Version", "Database 설명", "Database Size");
```

함수명 이름만 봐서는 Open 이라는 말에 이미 존재하는 Database 를 연다고 생각하기 쉬운데 Database 가 존재하지 않으면 Database 를 생성한다.

15-2. transaction

Database 가 생성도 되었으니 이젠 슬슬 접근해 보겠다. 접근하는 Method 는 transaction()이다.

```
if (!window.openDatabase) {
        document.write("이 Browser 는 Web SQL Database 를 지원하지 않습니다.");
        return false;
} else {
        var db = openDatabase("MyFirstDatabase", "1.0", "첫번째 테스트 데이터베이스", "2*1024*1024");
        db.transaction(function (tx) { //SQL 을 실행하고 Control 합니다.
                ...
        },
        function(error) { //transaction 에 Error 가 발생한 경우
                ...
        },
        function() { //transaction 을 성공했을 때
                ...
        });
}
```

Error 발생시 그리고 성공완료시에 진행할 코드는 생략가능한 부분이다. 요기까지가 Database 를 생성하고 접근하는 것까지의 순서다.

15-3. executeSql

이젠 테이블을 생성하고 데이터를 입력하는 작업을 해보겠다. 이런 작업을 할때 사용하는 Method 는 executeSql() 이다.

```
if (!window.openDatabase) {
        document.write("이 Browser 는 Web SQL Database 를 지원하지 않습니다.");
        return false;
} else {
        var db = openDatabase("MyFirstDatabase", "1.0", "첫번째 테스트 데이터베이스", "2*1024*1024");
        db.transaction(function (tx) { //SQL 을 실행하고 Control 합니다.
                tx.executeSql('CREATE TABLE worklist(idx integer primary key, subject, memo)');

                tx.executeSql('INSERT INTO worklist(idx, subject, memo) VALUES (1, "업무", "WCF Server 작성")');
                tx.executeSql('INSERT INTO worklist(idx, subject, memo) VALUES (2, "약속", "SQLer Article작성")');
                tx.executeSql('INSERT INTO worklist(idx, subject, memo) VALUES (3, "가정", "일주일식량 장보기")');
        });
}
```

Table 을 생성하고 세건의 Data 를 입력하는 코드이다.

삭제하는 구문은 아래와 같다.

```
tx.executeSql('DELETE FROM worklist WHERE idx=2');
```

테이블 삭제는 아래와 같다.

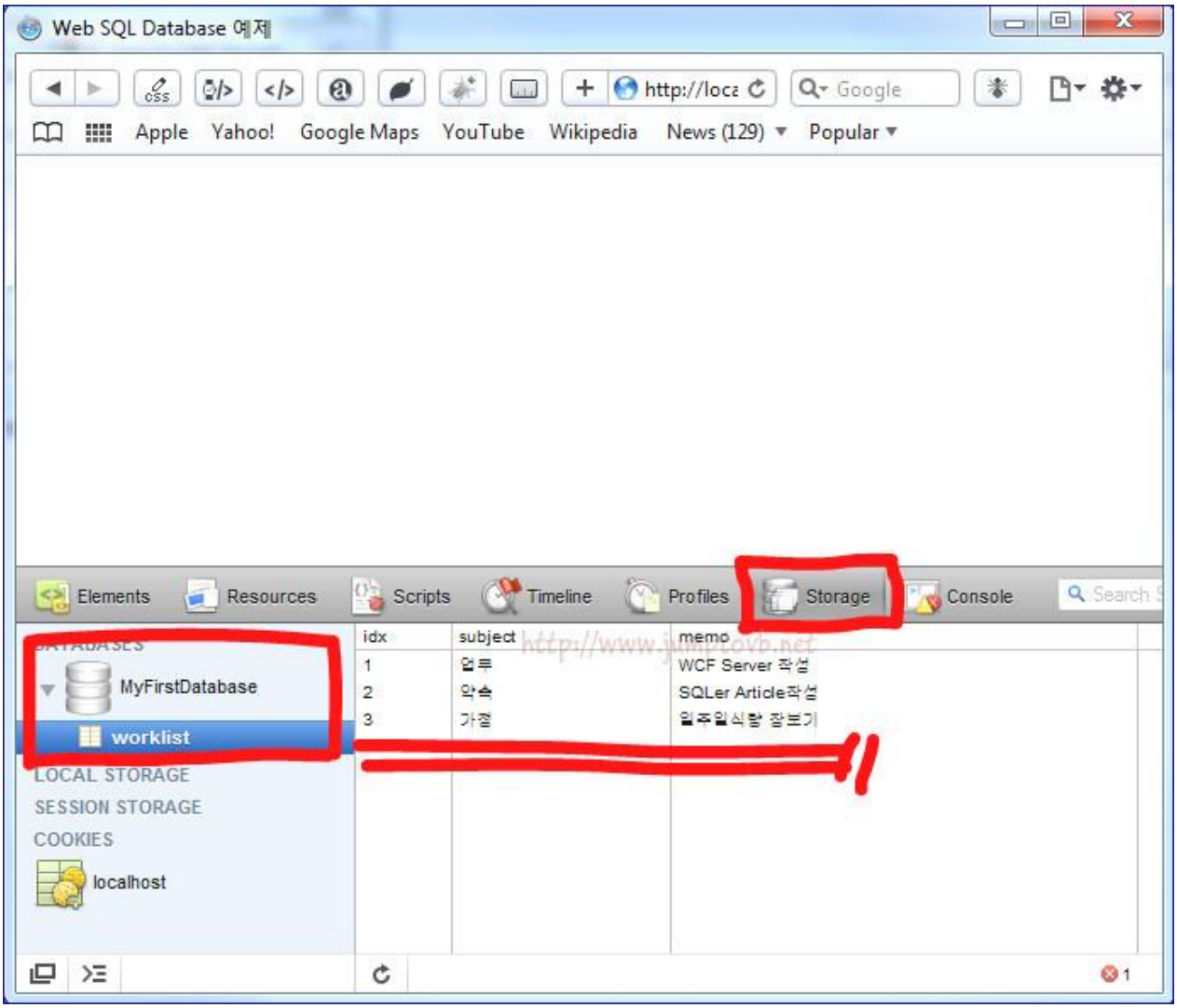
```
tx.executeSql('DROP TABLE worklist');
```

15-4. 예제

그럼 위 사항들을 모두 적용한 예제를 만들어 보겠다

```
<!DOCTYPE html>
< html lang="ko">
 <head>
      <meta charset="utf-8" />
  <title>Web SQL Database 예제</title>
      <script type="text/javascript">

          if (!window.openDatabase) {
             document.write("이 Browser 는 Web SQL Database 를 지원하지 않습니다.");
             return false;
          } else {

             var db = openDatabase("MyFirstDatabase", "1.0", "첫번째 테스트 데이터베이스", "2*1024*1024");

             db.transaction(function (tx) { //SQL 을 실행하고 Control 합니다.
                tx.executeSql('SELECT Count(idx) FROM worklist');
             },
             function(error) { //transaction 에 Error 가 발생한 경우
                alert("Select Error : " + error.message);
                tx.executeSql('CREATE TABLE worklist(idx integer primary key, subject, memo)');
             },
             function() { //transaction 을 성공했을 때
             });

             db.transaction(function (tx) { //SQL 을 실행하고 Control 합니다.
                tx.executeSql('INSERT INTO worklist(idx, subject, memo) values (1, "업무", "WCF Server 작성")');
                tx.executeSql('INSERT INTO worklist(idx, subject, memo) values (2, "약속", "SQLer Article작성")');
                tx.executeSql('INSERT INTO worklist(idx, subject, memo) values (3, "가정", "일주일식량 장보기")');
             },
             function(error) { //transaction 에 Error 가 발생한 경우
                alert("insert Error : " + error.message);
             },
             function() { //transaction 을 성공했을 때
                alert("insert Success");
             });

          };
      </script>
 </head>
 <body>
 </body>
< /html>
```

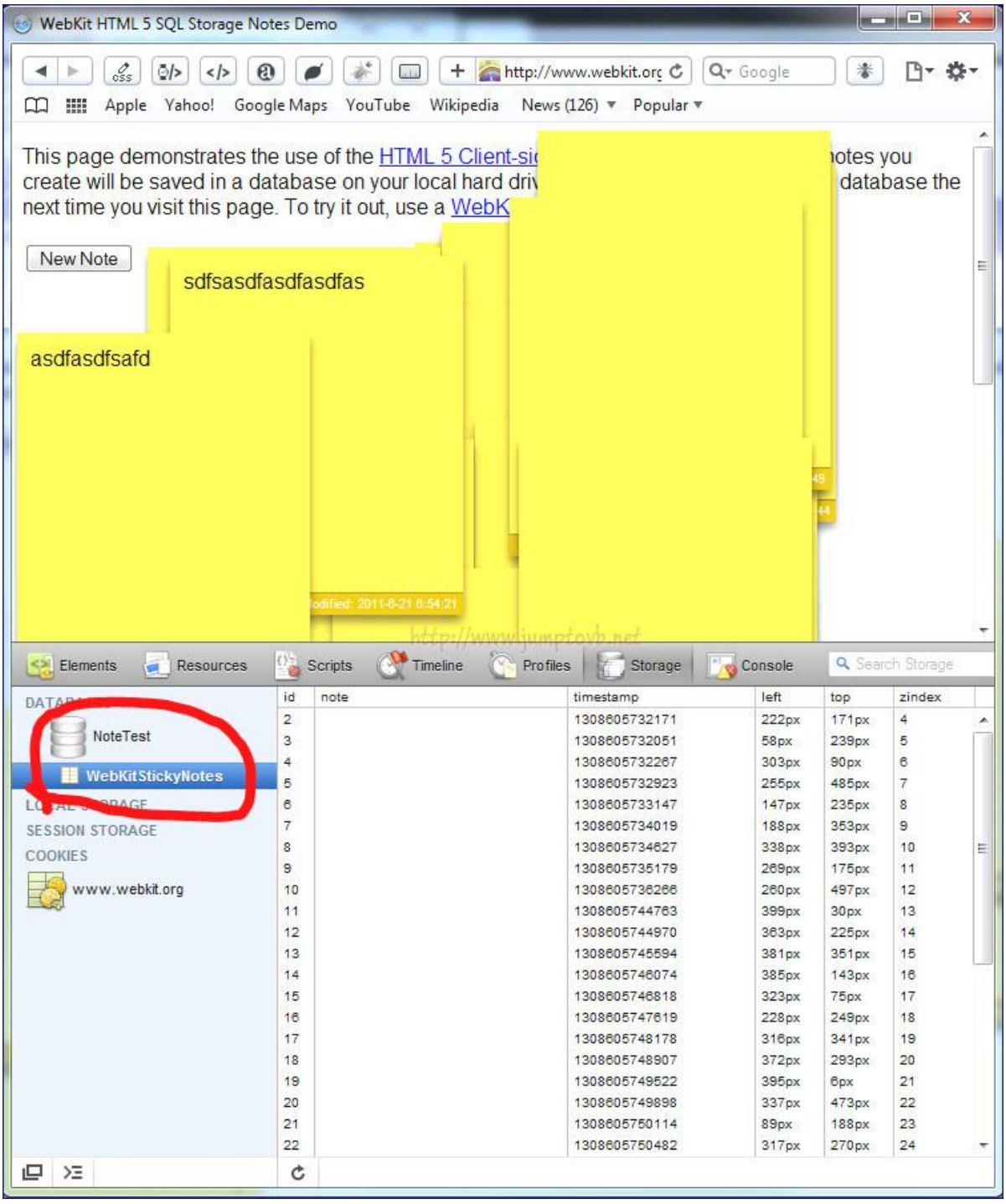
(발췌/수정 : 앞서가는 디자이너와 퍼블리셔를 위한 HTML5 & CSS)



Safari 의 Development Tool 인 Web Inspector 를 통해서 본 모습이다. Database 의 모습을 확인해 볼 수 있으며 데이터도 확인해 볼 수 있다.

아래 이미지는 Webkit 에서 제공하는 예제인 Sticky Note 이다. (안타깝지만 internet explore 9.0 과 firefox 4.0.1 에서는실행되지 않는다.)

http://www.webkit.org/demos/sticky-notes/index.html


WebKit HTML 5 SQL Storage Notes Demo
http://www.webkit.org
Google
Apple
Yahoo!
Google Maps
YouTube
Wikipedia
News (126)
Popular
This page demonstrates the use of the HTML 5 Client-si
otes you
create will be saved in a database on your local hard driv
database the
next time you visit this page. To try it out, use a WebK
New Note
sdfsasdfasdfasdfas
asdfasdfsafd
http://www.jumptovb.net
Elements
Resources
Scripts
Timeline
Profiles
Storage
Console
Search Storage
NoteTest
WebKitStickyNotes
SESSION STORAGE
COOKIES
www.webkit.org
id note timestamp left top zindex
2 1308605732171 222px 171px 4
3 1308605732051 58px 239px 5
4 1308605732267 303px 90px 6
5 1308605732923 255px 485px 7
6 1308605733147 147px 235px 8
7 1308605734019 188px 353px 9
8 1308605734627 338px 393px 10
9 1308605735179 269px 175px 11
10 1308605736266 260px 497px 12
11 1308605744763 399px 30px 13
12 1308605744970 363px 225px 14
13 1308605745594 381px 351px 15
14 1308605746074 385px 143px 16
15 1308605746818 323px 75px 17
16 1308605747619 228px 249px 18
17 1308605748178 316px 341px 19
18 1308605748907 372px 293px 20
19 1308605749522 395px 6px 21
20 1308605749898 337px 473px 22
21 1308605750114 89px 188px 23
22 1308605750482 317px 270px 24

[HTML5 강좌] 16. Web Worker

다음의 내용과 소스는 혁명을 꿈꾸다 HTML5 & API 입문 앞서가는 디자이너와 퍼블리셔를 위한 HTML5 & CSS 을 참고 하였다.

이번엔 Web Worker 에 대해서 설명하려 한다. HTML5 를 사용하면 Web Browser 에서도 Desktop Application 에서 처럼 Multitasking 을 할 수 있게 되는데 완전히 Desktop Application 에서 지원하는 Multitasking 과 완전히 같지는 않지만 Web Browser 의 UI Thread 와 완전히 분리된 Background 에서 동작한다.

기존에는 Javascript 가 오랜 계산을 해야 할 경우 그 계산을 마칠 때까지 사용자는 기다려야 했다면 Web Worker 를 사용하면 Javascript 의 오랜 계산에도 사용자는 아무런 버벅거림도, 기다림도 없이 Web Application 을 이용할 수 있다.

16-1. 사용법

Web Worker 를 사용하기 위해서는 먼저 Web Worker 가 호출하여 사용하게될 Javascript 파일이 외부에 필요하게 된다.

```
var worker = new Worker("외부 Javascript 파일명")
```

그리고, 중요한 함수인 postMessage() 와 onmessage . 두 가지를 기억해야 한다. 이 두 가지는 UI Thread 와 BackGround 사이에서 통신을 하게 된다. postMessage 로 Message 를 던지면 먼저 등록해 놓은 onmessage Event handler 로 Message 를 받게 된다. 그리고 다시 posetMessage 로 결과를 되돌려 주게 되고요. onmessage EventHandler 로 결과를 받는다.

아래와 같이 Javascript 파일을 작성한다.(worker.js)

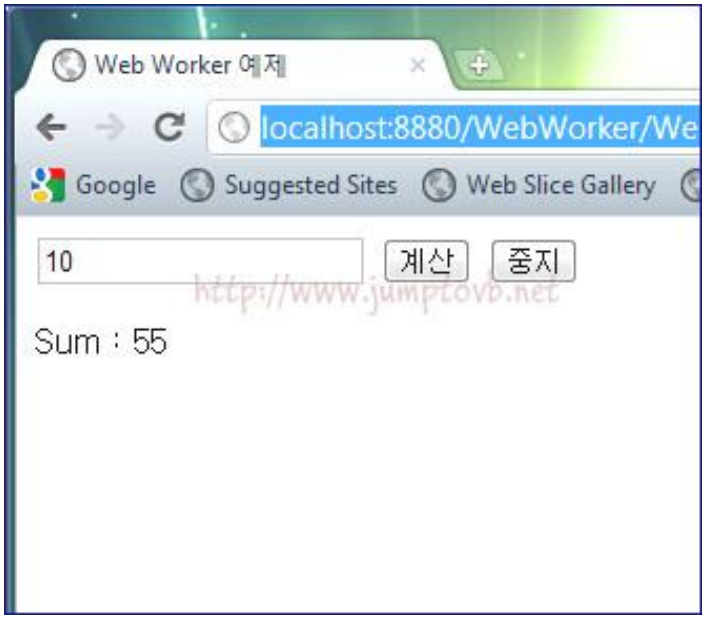
```
onmessage = function(evt) {
        var num = evt.data;
        var result = 0;
        for(var i = 1; i <= num; i++)
        {
           result += i;
        }
     postMessage(result);
};
```

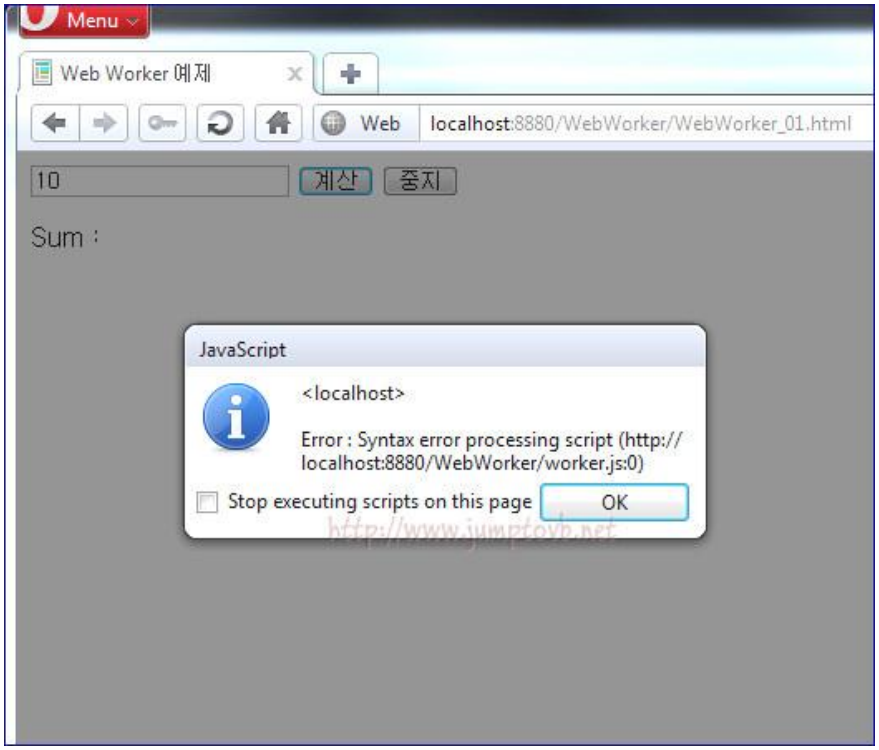
(발췌/수정 : 혁명을 꿈꾸다 HTML5 & API 입문)

(WebWorker.html) (발췌/수정 : 혁명을 꿈꾸다 HTML5 & API 입문)

```
<!DOCTYPE html>
	<html lang="ko">
	 <head>
		<meta charset="utf-8" />
	  <title>Web Worker 예제</title>
	 </head>
	 <body>
		<input type="text" id="num">
		<button onclick="calculate()">계산</button>
		<button onclick="stopCalculate()">중지</button><br/>
		<p id="sum">Sum :</p>
	 </body>
		<script type="text/javascript">

			var worker;

			function calculate() {

			   if(worker) {
			      worker.terminate();
			   };

			   var num = document.getElementById("num").value;

			   worker = new Worker("worker.js");

			   worker.onmessage = function(evt) {
			       document.getElementById("sum").innerHTML = "Sum : " + evt.data;
			   };

			   worker.onerror=function(evt) {
			      alert("Error : " + evt.message + " (" + evt.filename + ":" + evt.lineno + ")");
			   };

			   worker.postMessage(num);

			}

			function stopCalculate() {
			   if(worker) {
			         worker.terminate();
			   };
			   alert("Worker is Stopped");
			}
	</script>
</html>
```

실행하면 아래와 같은 결과를 볼 수 있다.





아래 예제는 page 와 worker 간에 계속 계산값을 던지게 된다.

(cals.js)

```
onmessage = function(evt) {
        for (var i = evt.data; i < 1000000; i++)
        {
            postMessage(i);
        };
};
```

(발췌/수정 : 앞서가는 디자이너와 퍼블리셔를 위한 HTML5 & CSS)

(webWorker.html)

```
<!DOCTYPE html>
 <html lang="ko">
   <head>
      <meta charset="utf-8" />
      <title>Web Worker 예제</title>
   </head>
   <body>
      <p>
         <input id="worker_work" type="button" value="Work!!!">
         <input id="worker_work_stop" type="button" value="Stop!!!">
      </p>
      <p id="worker_say">Say</p>
      <textarea rows="4" cols="60">딴짓할 수 있어요.</textarea>
   </body>
      <script type="text/javascript">

            var worker;

            function makeWorker() {
               worker = new Worker("cal.js");
            };

            document.getElementById("worker_work").onclick = function() {
                  makeWorker();

                  worker.postMessage(0);

                 worker.onmessage = function(evt) {
                          document.getElementById("worker_say").innerHTML=evt.data;
                  };

                  document.getElementById("worker_work_stop").onclick = function() {
                          if(worker) {
                                  worker.terminate();
                                  alert("Worker. Stop!!! ");
                          };
                 };
                  worker.onerror=function(evt) {
                         document.getElementById("worker_say").innerHTML = "Error : " + evt.message + " (" + evt.filename + ":" +
evt.lineno + ")";
                 };
             };

 </script>
</html>
```

(발췌/수정 : 앞서가는 디자이너와 퍼블리셔를 위한 HTML5 & CSS)

[HTML5 강좌] 17. Web Socket

Web Socket 을 이야기 할 차례이다. 웹의 단점을 보완하고자 Ajax 가 등장을 했지만 언제나 DisConnected 인 상태인 웹으로써는 완벽한 실시간 웹을 구현하기에는 부족했다. 기존에 조금 부족한 Ajax 이외에 Flash, Flex, Silverlight 의 경우에는 Socket 을 지원하고 있으므로 실시간 웹을 구현할 수 있었다. 하지만 HTML5 에서는 Web Socket 이 그 역할을 대신 할 수 있을 것으로 보인다.

17-1. Web Socket 환경 구성

Web Socket 구성은 여러분들도 짐작하시겠지만 Socket Server 를 구성해야 된다. Socket Server 구성하면 된다. Nugget 으로 된 서버를 구성할 수 있는데 Codeplex 에서 받을 수 있다.
(**http://nugget.codeplex.com/**)
Socket 이 어떻게 돌아가는지 한번 살펴 보실 수 있다. Web Socket Server 에서 접속한 Client 들에게 Message 를 Push 하는 예제도 함께 포함되어 있는데 Client 를 조금 수정해서 Server 로 전송할 수있게 해 보았다.

17-2. 사용법

사용법도 간단하다. Web Socket 을 생성하는 구문은 다음과 같다.

```
var wsSocker = new WebSocket("ws://Web Socket Server URL");
```

WebSocket 생성시 Web Socket Server 의 URL 을 Parameter 로 사용하며 Web Socket 프로토콜 "**ws://**" 을 사용한다. 생성한 WebSocket 으로 send Method 를 사용하여 Message 를 전송합니다.

```
wsSocket.send("전송할 Message");
```

wsSocket interface 에는 다음과 같은 Event Handler 가 있다.

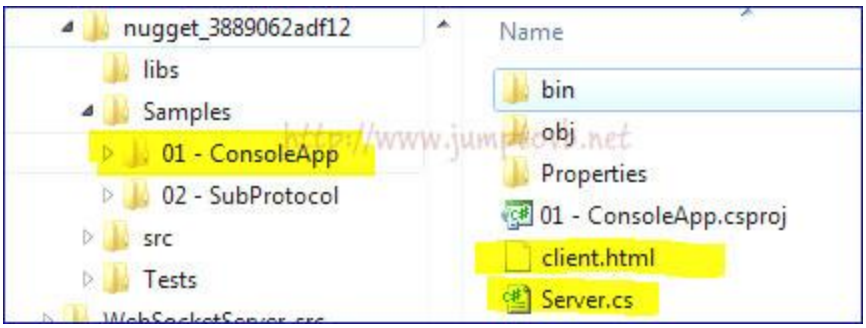
Event Handler	설명
onmessage	Server 로 Message 를 전달 받을때
onopen	Web Socket Server 가 열릴 때
onclose	Web Socket Server 가 닫힐 때

다음과 같이 사용한다.

```
wsSocket.onmessage = function() {

…

};

wsSocket.onopen = function() {

…

};

wsSocket.onclose = function() {

…

};
```

17-3. 예제

위 내용을 바탕으로한 예제를 살펴보기로 하겠다.



nugget Sample 예제 압축을 푸시면 위와 같은 파일들이 있다. 두 예제 중 첫 번째 예제이다.

오른쪽의 Server.cs, client.html 파일을 살펴볼 예정이다.

Server.cs

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using Nugget;
6
7 namespace ConsoleApp
8 {
9
10        class ConsoleAppSocket : WebSocket
11        {
12
13            public override void Incoming(string data)
14            {
15                Console.WriteLine(data);
16            }
17
18            public override void Disconnected()
19            {
20                Console.WriteLine("--- disconnected ---");
21            }
22
23            public override void Connected(ClientHandshake handshake)
24            {
25                Console.WriteLine("--- connected --- ");
26            }
27    }
28
```

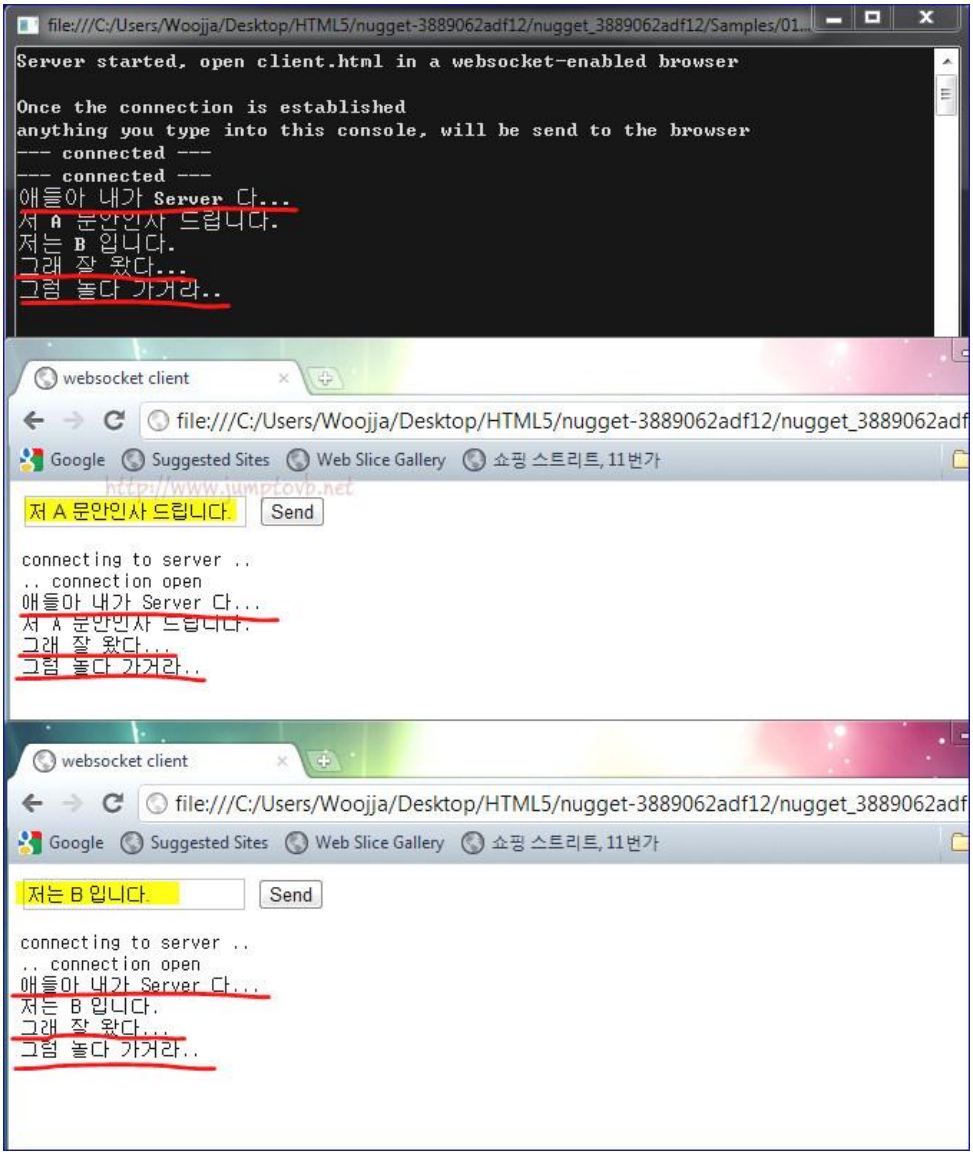
```
29
30      class Server
31      {
32          static void Main(string[] args)
33          {
34
35              var nugget = new WebSocketServer(8181, "null", "ws://localhost:8181");
36
37              nugget.RegisterHandler<ConsoleAppSocket>("/consoleappsample");
38
39              Nugget.Log.Level = LogLevel.None;
40
41              nugget.Start();
42
43              Console.WriteLine("Server started, open client.html in a websocket-enabled browser\n");
44              Console.WriteLine("Once the connection is established\nanything you type into this console, will be send to the browser");
45
46              var input = Console.ReadLine();
47              while (input != "exit")
48              {
49                  nugget.SendToAll(input);
50                  input = Console.ReadLine();
51              }
52
53          }
54      }
55 }
```

Line 35, 37 의 내용을 보면 Web Socket Server 에서 8181 Port 에 consoleappsample 이라는 곳을 통해서 Web Socket 을 Start 했다. 아래는 Client Source 이다.

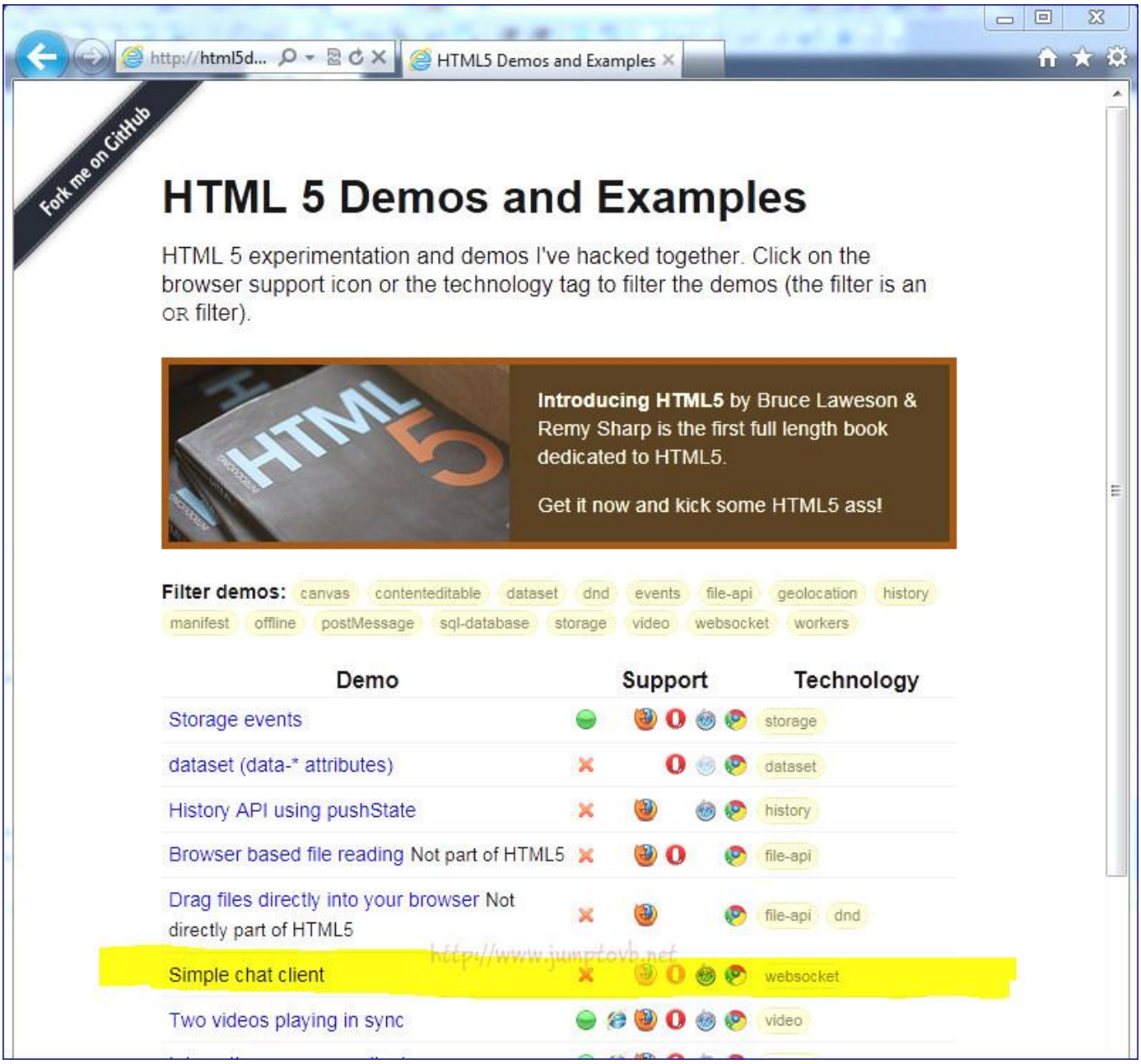
client.html

```
<!DOCTYPE lang="ko">
<html>
  <head>
      <title>websocket client</title>
  </head>
    <body>
      <input type="text" name="txtinput" id="txtinput">
      <button onclick="Send()" name="btnSend" id="btnSend">Send</button><br />
      <pre id="incomming"></pre>

       <script type="text/javascript">

                var inc = document.getElementById('incomming');
                inc.innerHTML += "connecting to server ..<br/>";

                var ws = new WebSocket('ws://localhost:8181/consoleappsample');

                ws.onmessage = function (evt) {
                   inc.innerHTML += evt.data + '<br/>';
                };

                ws.onopen = function () {
                   inc.innerHTML += '.. connection open<br/>';
                };

                ws.onclose = function () {
                   inc.innerHTML += '.. connection closed<br/>';
                };

                function Send() {
                   var strMessage = document.getElementById("txtinput").value;

                   var isSend = ws.send(strMessage);

                   if (isSend)
                      inc.innerHTML += strMessage + '<br/>';
                   else
                      inc.innerHTML += 'Send Error<br/>';

                };
       </script>
    </body>
</html>
```

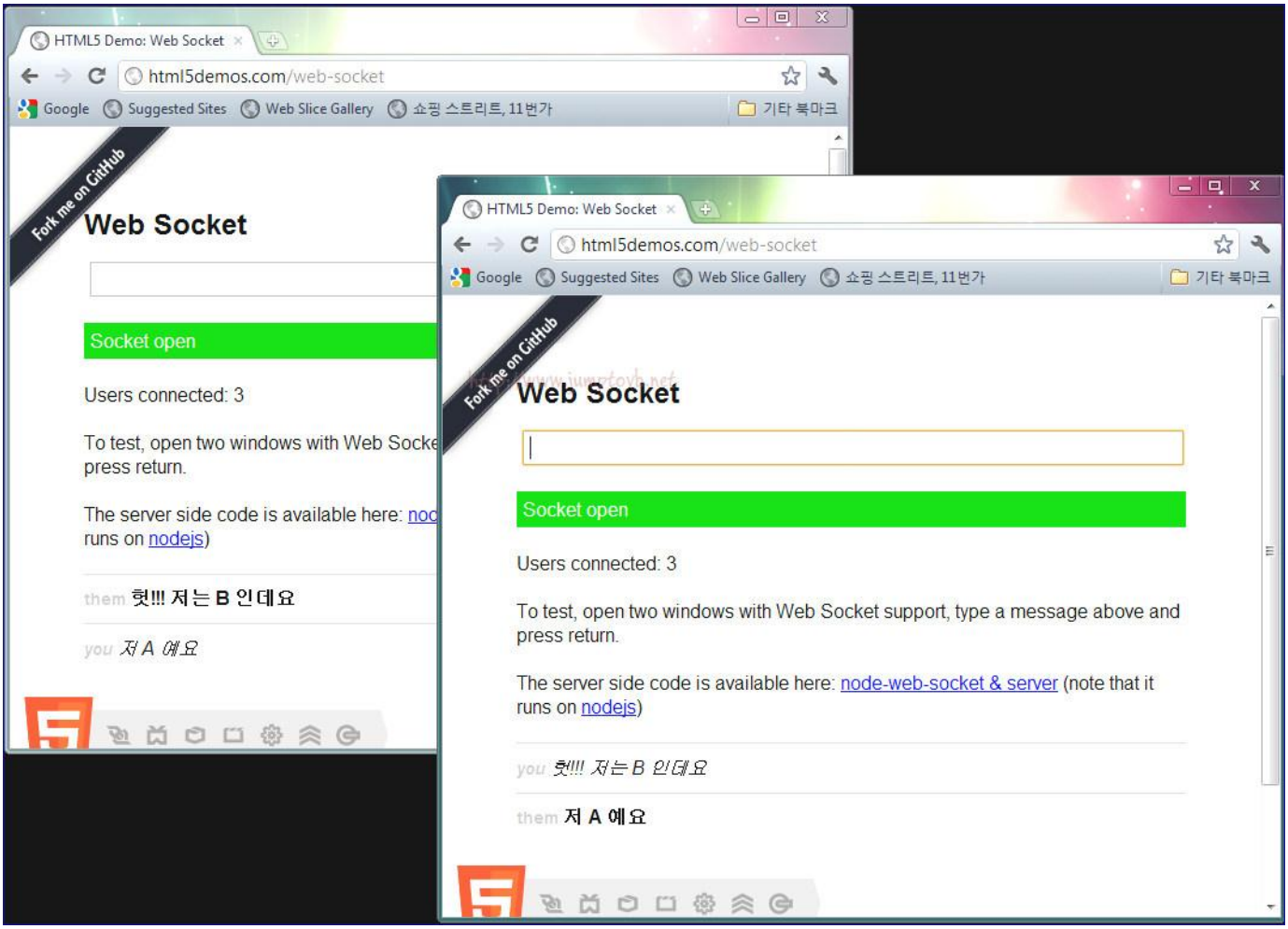
16 Line 을 보면 WebSocket 을 생성하고 18, 22, 26 Line 에서 onmessage, onopen, onclose 이벤트가 도착했을 때 작업을 작성했다. 위 코드만으로는 Server 가 Client 는 무조건 Sever 에서 보낸 데이터를 처리하도록만 되어 있는데 거꾸로 보내는 동작을 구현 했다. 아래는 동작시키는 모습을 Capture 한 것이다. 붉은 색으로 줄을 친 부분은 곧바로 명령창에서 입력한 것이다. 입력하고 Enter 를 치면 내용이 전송된다. 그럼 그 아래 열린 두개의 Browser 에서 그 내용을 받게 된다. 각 Client 에서 보내는 내용은 Server 에만 보내도록 했다.



다른 Web Browser 에서는 제대로 동작을 하지 못한다. Test 를 해보시려면 Chrome 에서 Test 해보시기 바란다. **html5Demos** Site 에서 보시면 Safari 도 지원을 한다고 나오는데 Test 결과 제대로 작동을 안했다.


http://html5d...
HTML5 Demos and Examples
Fork me on GitHub
HTML 5 Demos and Examples
HTML 5 experimentation and demos I've hacked together. Click on the browser support icon or the technology tag to filter the demos (the filter is an OR filter).
Introducing HTML5 by Bruce Lawson & Remy Sharp is the first full length book dedicated to HTML5.
Get it now and kick some HTML5 ass!
Filter demos: canvas contenteditable dataset dnd events file-api geolocation history manifest offline postMessage sql-database storage video websocket workers
Demo
Support
Technology
Storage events
storage
dataset (data-* attributes)
dataset
History API using pushState
history
Browser based file reading Not part of HTML5
file-api
Drag files directly into your browser Not directly part of HTML5
file-api dnd
http://www.jumptovb.net
Simple chat client
websocket
Two videos playing in sync
video

아래의 Capture 이미지는 위 **html5Demos** Site 에 Link 되어있는 예제로 단순한 Chatting Page 이다.



http://bohuco.net/labs/ 에도 Web Socket 에 관한 PHP 예제가 있다.

위의 내용들은
혁명을 꿈꾸다 HTML5 & API 입문 앞서가는 디자이너와 퍼블리셔를 위한 HTML5 & CSS
을 참고 하였다.

[HTML5 강좌] 18. Geolocation API

이번엔 Geolocation API 를 살펴보겠다. Geolocation API 는 HTML 5 에 새롭게 추가된 사용자의 위치정보를 얻기 위한 JavaScript API 이다. 위치정보를 기반한 서비스가 우리 생활 곳곳에 퍼져있으며 많은 서비스들이 위치정보와 연동하여 사용자들에게 UX 의 편리함을 줄 것이다.

Geolocation API 는 세 개의 Method 로 이루어진 API 이다.

그 전 먼저 알아야 할 점은 현재의 위치정보가 Network 정보로부터 추측한 것인지 GPS 로 부터 얻은 것인지에 관한 자세한 내용은 알 수 없다는 것이다. 단지 GPS 가 내장된 Smartphone 과 같은 Device 에서는 GPS 기능을 활용할 수있고 일반 PC 에서는 WiFi 같은 정보를 이용해서 현재 정보를 알아 낼 수 있다.
Geolocation 은 위치정보에 대한 정보를 GPS, WiFi IP Address, GSM/CDMA 망을 사용하는 휴대전화의 IDs 등에서 얻어온다. 다만 PC 에서는 한정된 정보만 제공하여 이용이 불가능한 경우가 있고 특히 Mobile Browser 에서 유용한 API 라고 할 수 있다.

18-1. 사용법
Geolocation API 와 관련된 함수는 모두 window.navigator 객체에 정의되어 있다.
다음 Method 를 사용하여 위치정보를 얻어 올수 있다.

현재 위치를 한번만 얻기 위한 함수이다.

```
navigator.geolocation.getCurrentPosition(successCallback, errorCallback, options);
```

다음 함수를 이용하여 위치정보를 계속 확인 한다.

```
var watchId = navigator.geolocation.watchPosition(successCallback, errorCallback, options);
```

성공시 호출되는 successCallback 함수에 전달되는 위치정보에는 다음과 같은 정보들이 포함 된다.

위치정보 속성	coords 속성	설명
coords	latitude	위도
	longitude	경도
	altitude	표고
	accuracy	위도, 경도의 오차 (단위와 오차)
	altitudeAccuracy	표고의 오차 (단위와 오차)
	heading	Device 의 진행 방향. 북쪽을 기준으로 한 시계방향의 각도로 나타냄
	speed	Device 의 진행 속도(미터/초). (이용할 수 없을 때는 null)
timestamp		위치정보를 얻은 시각(1970 년 1 월 1 일 부터의 milisecond)

Error 발생시 호출되는 errorCallback 함수에 전달되는 객체에 담겨져 있는 속성과 상수들 이다.

속성/상수	설명
code	Error Code
UNKNOWN_ERROR	알수 없는 Error (Error code 값 : 0)
PERMISSION_DENIED	권한 없음 Error (Error code 값 : 1)
POSITION_UNAVAILABLE	위치정보를 얻을 수 없음 (Error code 값 : 2)
TIMEOUT	시간제한 초과 (Error code 값 : 3)
message	Error Message

위치정보를 조회시 입력하는 세번째 Parameter 인 option 에 지정할 수 있는 것들은 다음과 같다.

속성/상수	설명
enableHighAccuracy	정확도가 높은 위치 정보를 요청
timeout	위치 정보 확인에 시간제한을 설정. 시간제한을 초과하면 TIMEOUT error 발생
maximumAge	위치정보의 유효기간을 설정. 0 을 지정하면 항상 새로운 위치정보를 요청함

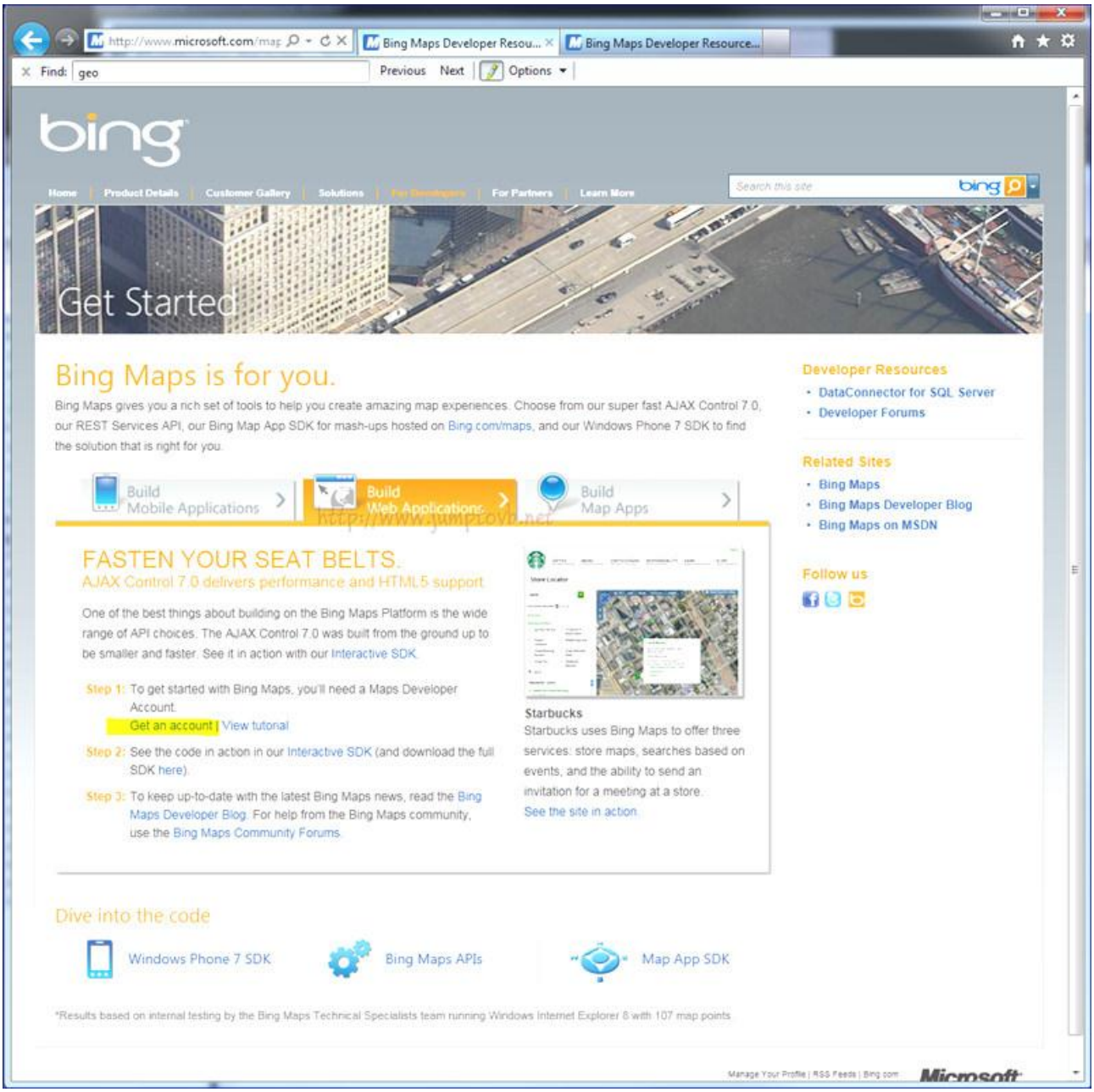
option 의 사용법은 아래와 같다.

내용을 보면 위치정보의 유효기간은 0 으로 항상 새로운 위치정보를 가져오며, 정확도 높은 위치정보를 요청 한다. 그리고 timeout 은 3 초로 설정하고 있다.

```
navigator.geolocation.getCurrentPosition(successCallback, errorCallback,
        {maximumAge: 0, enableHighAccuracy: true, timeout: 3000 }
);
```

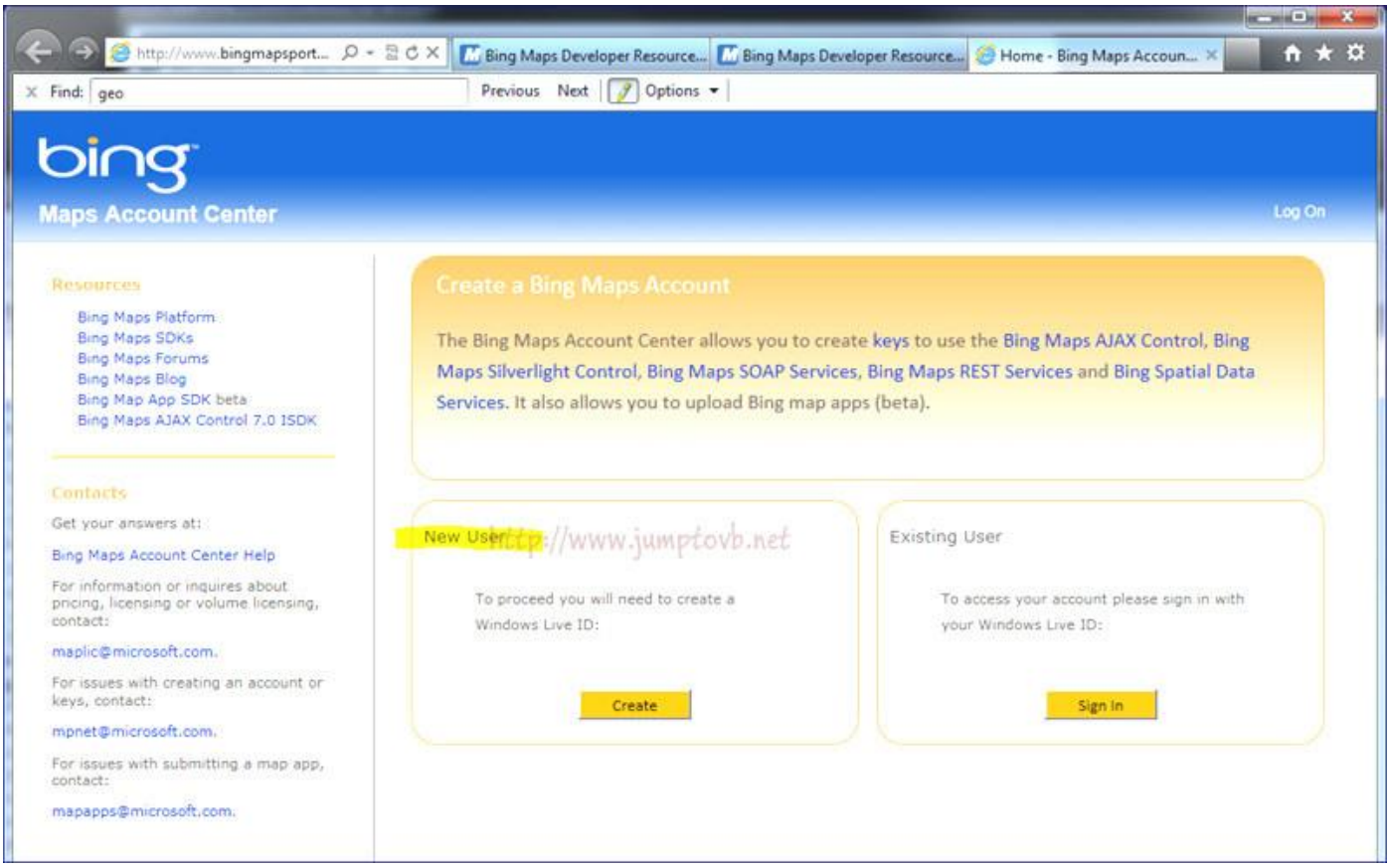
18-2. Bing Map 사용하기

예제로 Bing Map 을 사용해 보려 한다. Bing Map 을 사용하기 위해서 사전에 사용자 등록을 하고 사용 Key 를 받아야 한다. Site 주소는 **http://www.microsoft.com/maps/developers/web.aspx** 입니다.

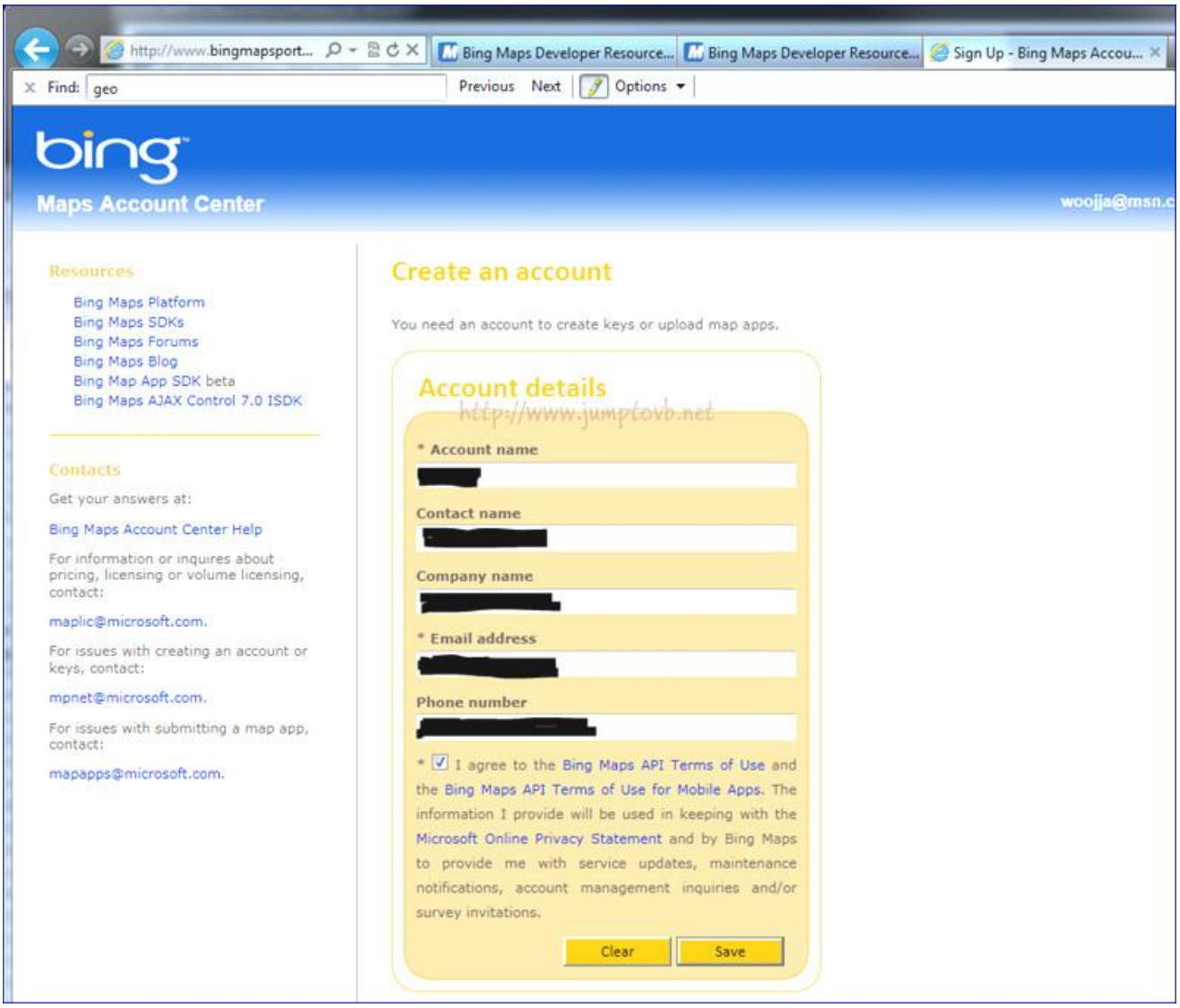


Get Account 를 클릭 한다.

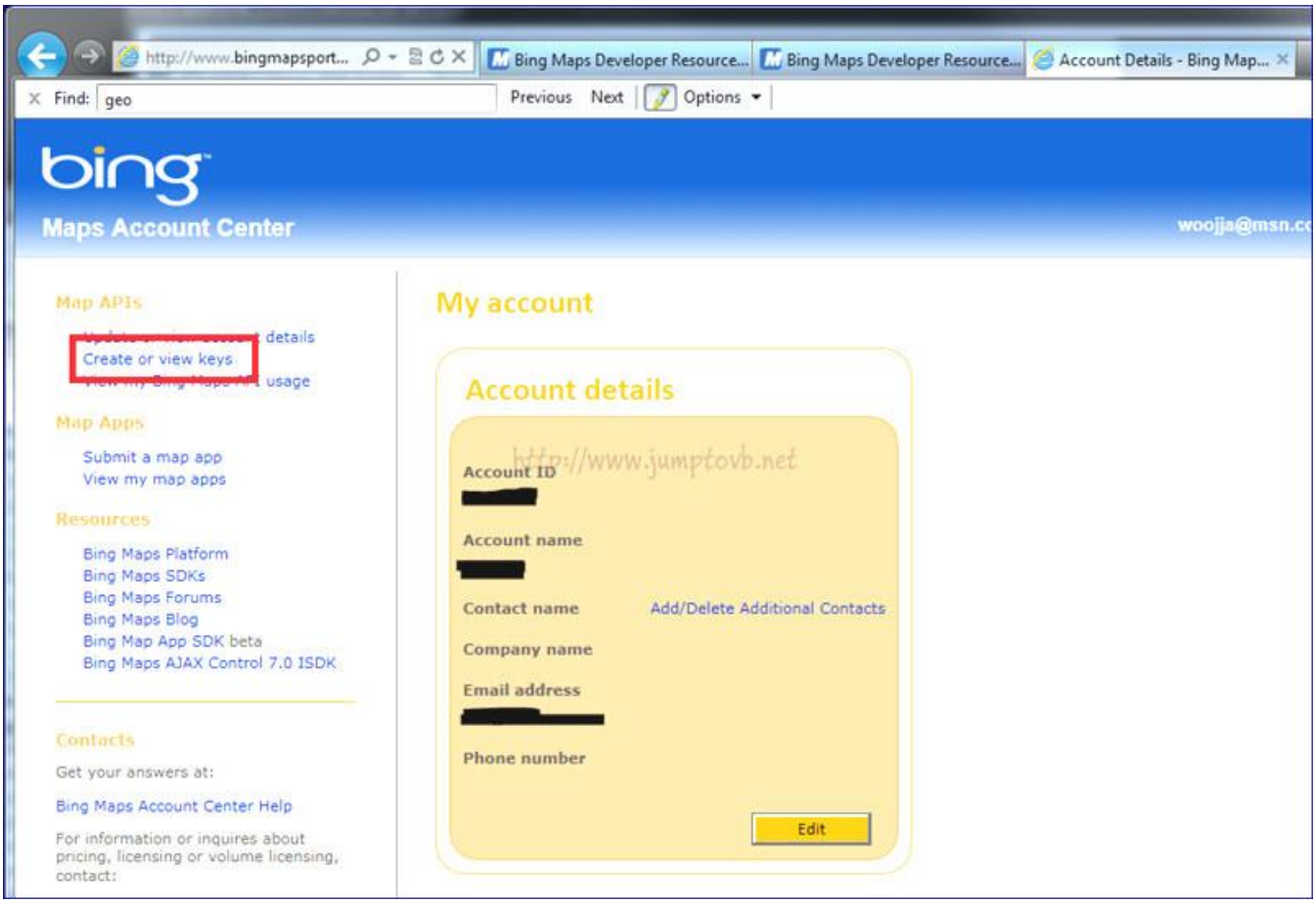
Capture 가 안되었지만 Windows Live ID Passport 인증을 한번 묻는다. Login 이 완료되면 다음 화면으로 넘어갑니다.



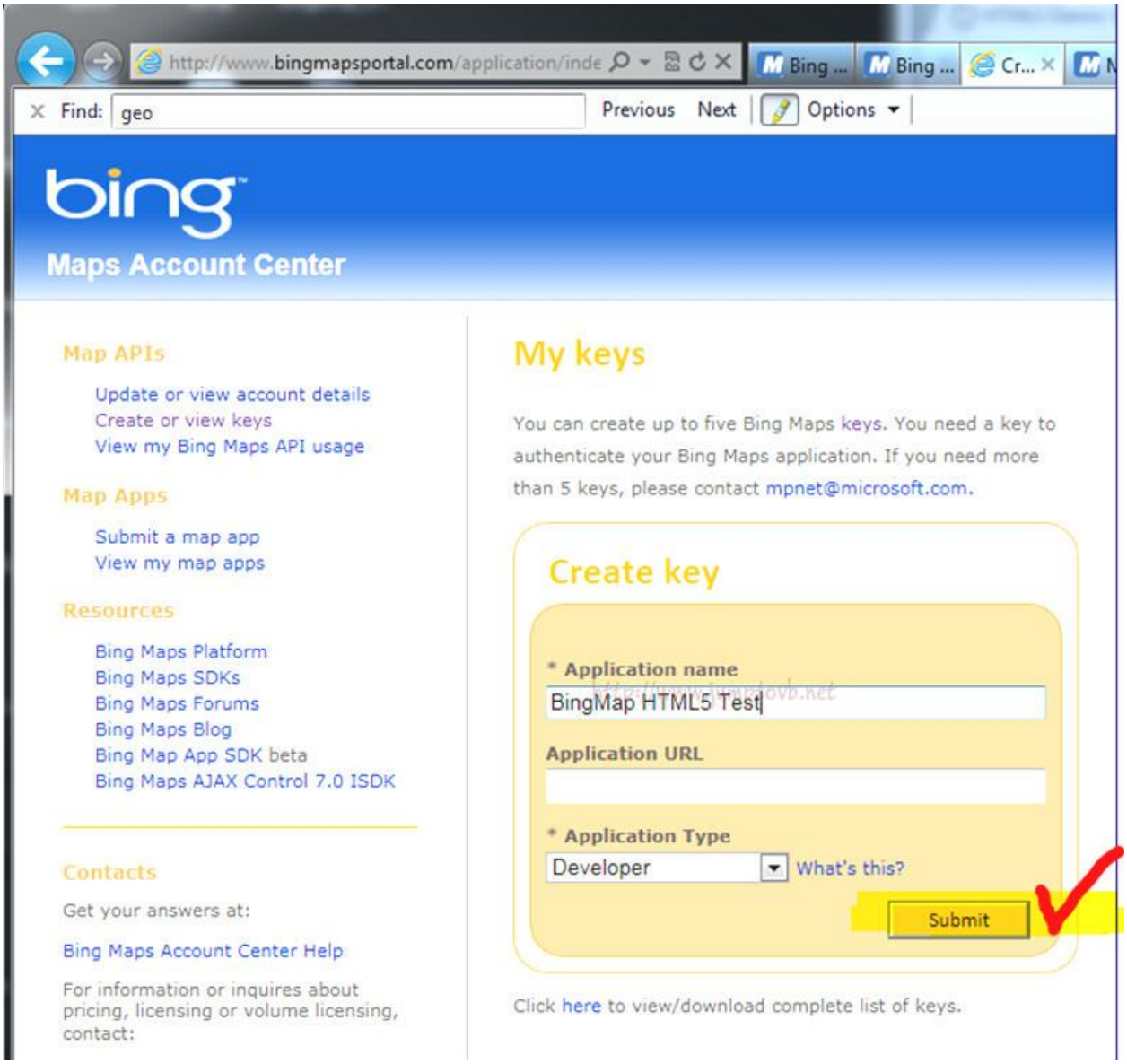
만약 계정이 없다면 New User 쪽의 Create 를 클릭 한다.



* 표시가 되어 있는 란에 간단하게 입력한다.

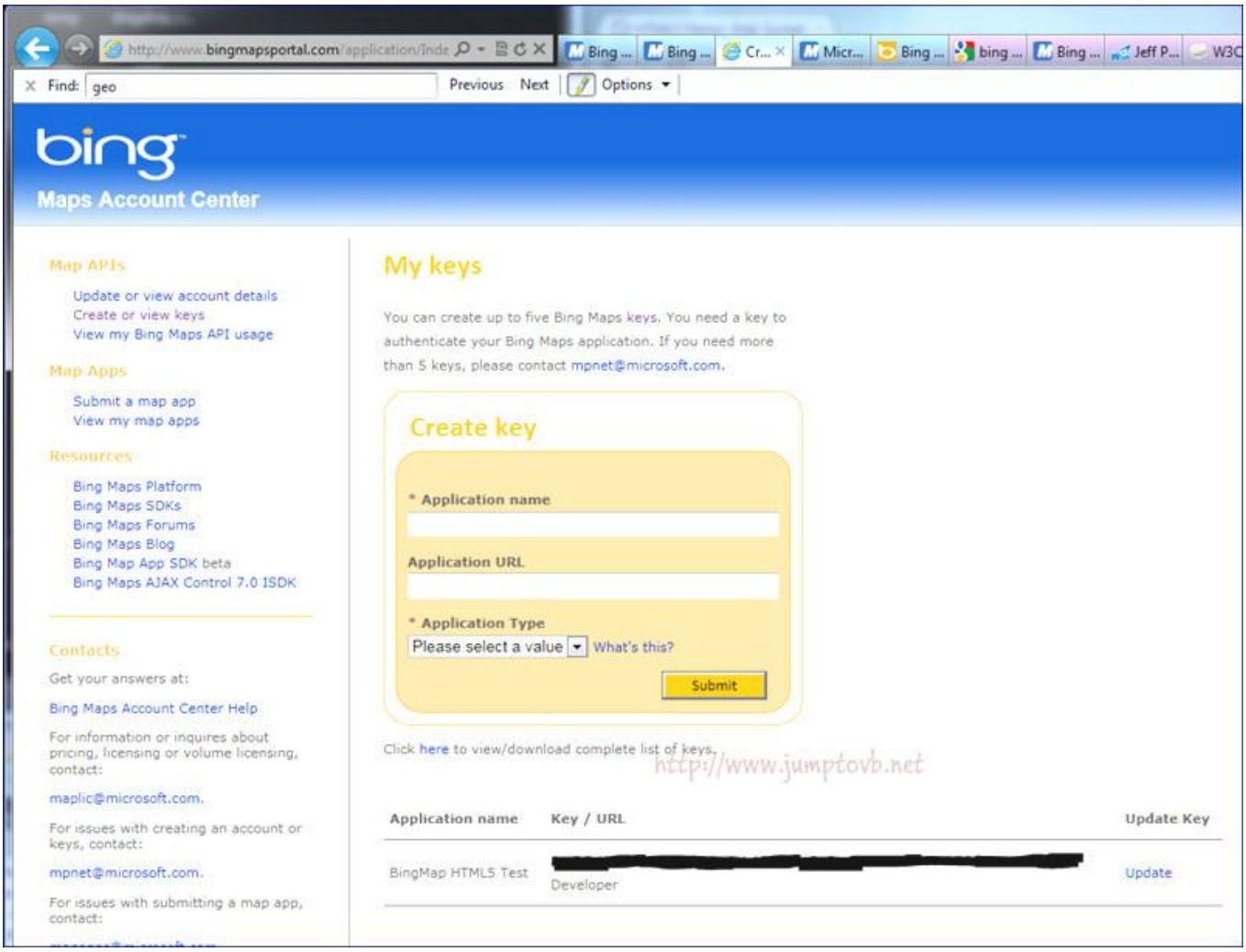


조금 기다리시면 Account ID 가 생성이 된다. 그럼 "Create or view Keys" 를 Click 한다.



* 표시가 되어 있는 란에 간단히 입력한다 "Submit" 버튼을 Click 한다.

또 잠시 기다립니다.



이 생성된 Key 를 다음에 진행될 Geolocation 예제 소스에 Key 를 입력하는 곳에 입력한다.

18-3. 예제.

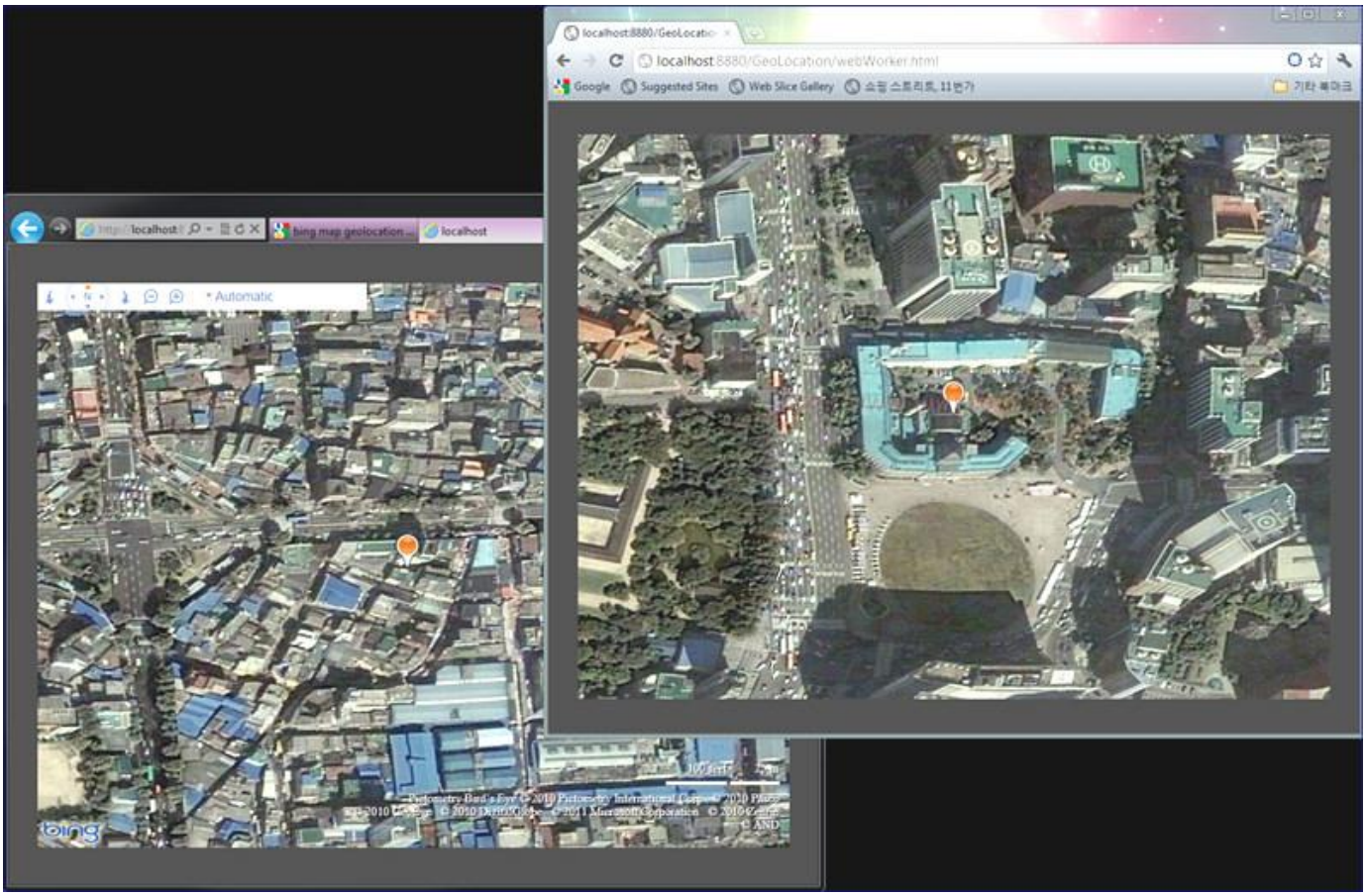
이전에 설명드렸듯이 PC 에서 진행하는 경우 아주 제한적입니다.

```
<!DOCTYPE html>
<html>
<head>
<style type="text/css">
   body {
      margin: 0;
      height: 100%;
   background-color: #404040;
   }
   #map {
       position: absolute;
   top: 50%;
      left: 50%;
      width: 800px;
      height: 600px;
   margin-left: -400px;
      margin-top: -300px;
   }
</style>

<script charset="UTF-8" type="text/javascript" src="http://ecn.dev.virtualearth.net/mapcontrol/mapcontrol.ashx?v=7.0"></script>
<script type="text/javascript">

      var _map;

      function GetMap() {
         // Create a Bing map
   _map = new Microsoft.Maps.Map(document.getElementById("map"), { credentials: "이곳에 Key를 넣습니다." });
         // Get the current position from the browser
         if (!navigator.geolocation)
            alert("This browser doesn't support geolocation");
         else
               navigator.geolocation.getCurrentPosition(onPositionReady, onError, { maximumAge: 0, timeout: 30000,
enableHighAccuracy: true });
      };
```

```
    function onPositionReady(position) {
        // Apply the position to the map
        var location = new Microsoft.Maps.Location(position.coords.latitude, position.coords.longitude);
        _map.setView({ zoom: 18, center: location });
        // Add a pushpin to the map representing the current location
        var pin = new Microsoft.Maps.Pushpin(location);
        _map.entities.push(pin);
    }

    function onError(err) {
        switch (err.code) {
            case 0:
                alert("Unknown error");
                break;
            case 1:
                alert("The user said no!");
                break;
            case 2:
                alert("Location data unavailable");
                break;
            case 3:
                alert("Location request timed out");
                break;
        }

  }
</script>
</head>
<body onload="GetMap();">
        <div id="map" />
</body>
</html>
```

(참조 : http://www.wintellect.com/CS/blogs/jprosise/archive/2011/03/27/making-web-apps-sizzle-with-bing-maps-and-html5-s-geolocation-api.aspx)

위 소스를 각 Browser 에서 실행시키면 먼저 사용자의 위치정보를 공유하겠느냐 라는 메세지 창을 띄워 Geolocation 사용에 대한 동의를 구한 후 화면을 띄우게 된다.



위의 내용들은 [혁명을 꿈꾸다 HTML5 & API 입문 앞서가는 디자이너와 퍼블리셔를 위한 HTML5 & CSS]
을 참고 하였습니다.

[HTML5 강좌] 19. SVG

SVG. Scalable Vector Graphics 의 약자로 XML 을 기반으로 한 Drawing 표준을 이야기 한다. SVG 는 HTML 5 가 논의되기 이전부터 사용되던 기술로 HTML 5 페이지 상에 선, 곡선, Path, 도형 등을 Vector 로 표현할 수 있다는 Canvas 와는 조금 다른 점이 있다. Canvas 와는 다르게 Pixel 단위의 이미지밖에 표현할 수 없는 Web Page 에서 interactive 한 animation 이나 Vector graphic 을 SVG 를 이용하여 표현할 수 있다. SVG 가 표현할 수 있는 효과들을 간단히 살펴보면

- 기본 도형(예: 원, 다각형)
- 기타 image 형식 (예: PNG)
- Bezier Pass, Curve
- Text
- Transparency
- Transformation(회전, 기울이기, 확대/축소)
- Gradient
- Animation

Canvas 에서도 SVG 를 읽어 표현할 수있다고 하는데 직접 관련이 있는 것은 아니다.

그럼 Canvas 와 SVG 와의 차이점을 알아 보면 다음과 같다.

	Canvas	**SVG**
이미지 처리방식	Bitmap	Vector
DOM	존재하지 않음(DOM Control 불가)	존재함(Script 로 Control 가능)
외부 이미지 편집	Bitmap image 편집 용이	Vector image 편집 용이
성능	높은 해상도의 이미지를 사용하면 성능 저하	이미지가 복잡해질수록 Markup 이 복잡해져 성능이 저하
Animation	Animation API 가 없으므로 Script 의 Timer 를 사용	높은 수준의 Animation 을 지원
Cross Browsing	모든 Web Browser 에서 지원하지 않음	모든 Browser 에서 지원되는 Drawing 표준

외부 이미지로 저장	jpg, png 등으로 저장 가능	불가
적합한 서비스	Graph 구현, Game	Graph 구현, 매우 세밀한 해상도를 지원하는 UI 및 Application
적합하지 않은 서비스	Standalone Application UI	Game

이런 SVG 의 장점은 다음과 같은 것들을 들 수 있다.

접근성	image 를 접근 가능하게 만들기 위해 설계되었습니다.
파일크기	bitmap image 보다 작습니다.
크기 조절에도 깨지지 않음	Vector graphic 의 특징.
Script 조작 가능	JavaScript 와 DOM 으로 접근 가능.
Animation	SGV 언어 Core 자체에 animate 기능이 내장되어 있습니다.

19-1. 사용법과 예제

XML 을 기반으로하기 때문에 기본적으로는 XHTML 에서만 사용할 수 있습니다만 HTML 상에서도 표현이 가능하다. SVG 문서는 SVG Element, SVG Attribute, CSS 로 구성된다. SVG 는 XHTML 문서 자체에 인라인으로 들어가거나 별도의 SVG 파일로 작성된 후 object 나 img Element 로 HTML 이나 XHTML 문서와 링크되어 삽입될 수 있는데 어느 방식이든 모든 SVG 의 Root Element 는 SVG 이다.

```
<svg xmlns="http://www.w3.org/2000/svg">...</svg>
```

SVG 를 Web Page 에 넣는 방법은 두가지가 있다.

1. SVG 파일을 img 나 Object Tag 에 삽입하는 방법.

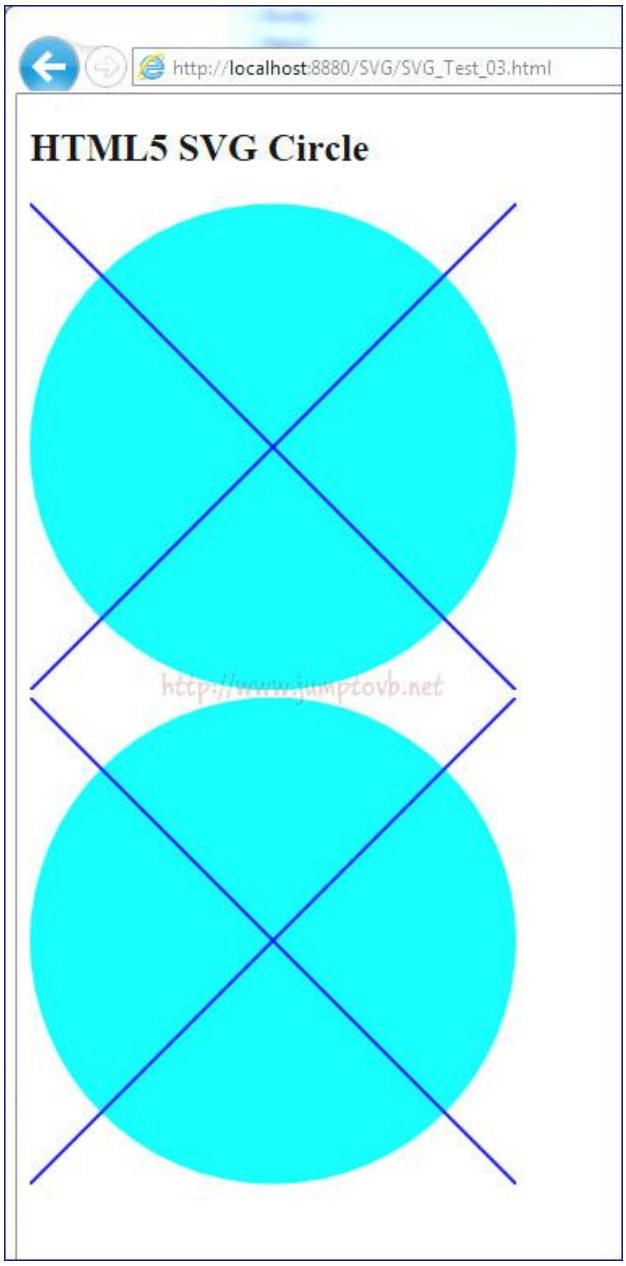
2. Web Page 에 직접 삽입하는 방법.

먼저 Tag 에 SVG 파일을 삽입하는 방법이다.

```
<!DOCTYPE html>
< head>
      <title>SVG Test</title>
      <meta charset="utf-8" />
< /head>
< body>
      <h2>HTML5 SVG Circle</h2>
       <img src="SVG_Test.svg" alt="Test SVG" />
       <object type="image/svg+xml" data="SVG_Test.svg" width="300px" height="300px">
         <p>사용하시는 Browser 는 이 그림을 표시하는 기능이 없습니다.
         <a href="http://www.adobe.com/svg/viewer/install/main.html">SVG 플러그인</a>을 설치해보셔요.</P>
      </object>
< /body>
< /html>
```

SVG_Test.svg

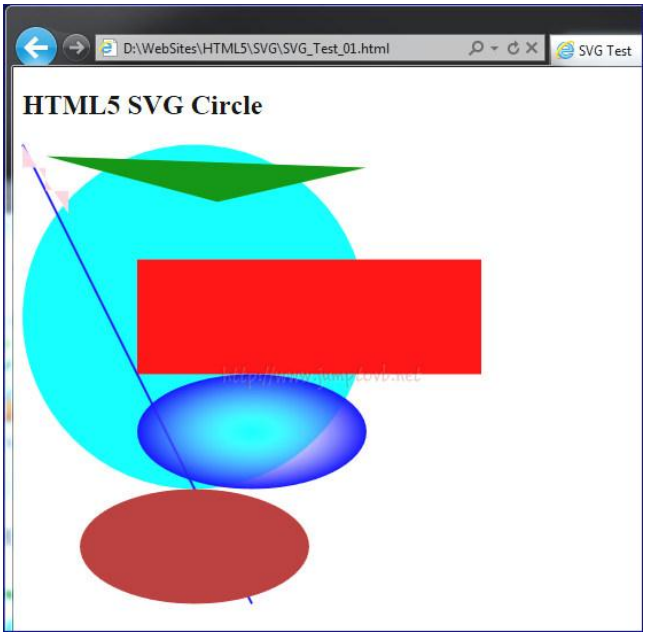
```
<svg id="svgelem" height="300" xmlns="http://www.w3.org/2000/svg">
      <circle id="redcircle" cx="150" cy="150" r="150" fill="cyan" />
      <line x1="0" y1="0" x2="300" y2="300" style="stroke:blue;stroke-width:2"/>
      <line x1="0" y1="300" x2="300" y2="0" style="stroke:blue;stroke-width:2"/>
< /svg>
```



이 방법의 특징은 모든 Web Browser 에서 지원을 하는 방법이라는 것이다.
반면 아래의 직접 SVG 를 입력하는 방법은 ie, chrome 을 제외한 Browser 는 지원하지 못한다.

다음은 Web Page 에 직접 SVG 를 입력하는 방법입니다.

```
<!DOCTYPE html>
< head>
      <title>SVG Test</title>
      <meta charset="utf-8" />
< /head>
< body>
      <h2>HTML5 SVG Circle</h2>
      <svg id="svgelem" height="600" xmlns="http://www.w3.org/2000/svg">
        <circle id="redcircle" cx="150" cy="150" r="150" fill="cyan" />
        <rect id="redrect" x="100" y="100" width="300" height="100" fill="red" />
        <line x1="0" y1="0" x2="200" y2="400" style="stroke:blue;stroke-width:2"/>
        <ellipse cx="150" cy="350" rx="100" ry="50" fill="brown" />
        <polygon points="20,10 300,20, 170,50" fill="green" />
        <polyline points="0,0 0,20 20,20 20,40 40,40 40,60" fill="pink" />
        <defs>
          <radialGradient id="gradient" cx="50%" cy="50%" r="50%" fx="50%" fy="50%">
          <stop offset="0%" style="stop-color:rgb(200,200,200); stop-opacity:0"/>
          <stop offset="100%" style="stop-color:rgb(0,0,255); stop-opacity:1"/>
          </radialGradient>
        </defs>
        <ellipse cx="200" cy="250" rx="100" ry="50" style="fill:url(#gradient) />
      </svg>
< /body>
< /html>
```



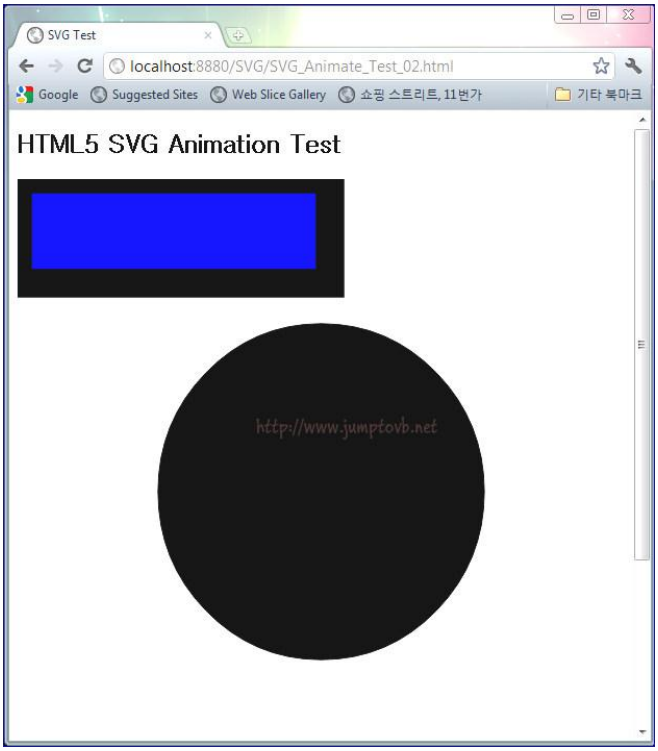
```
<!DOCTYPE html>
< head>
      <title>SVG Test</title>
      <meta charset="utf-8" />
< /head>
< body>
      <h2>HTML5 SVG Animation Test</h2>
      <img src="SVG_Animate_Test.svg" alt="Test SVG" />
      <object type="image/svg+xml" data="SVG_Animate_Test.svg">
         <p>사용하시는 Browser 는 이 그림을 표시하는 기능이 없습니다.
         <a href="http://www.adobe.com/svg/viewer/install/main.html">SVG 플러그인</a>을 설치해보셔요.</P>
      </object>
< /body>
< /html>
```

SVG_Animate_Test.svg

```
<svg id="svgelem" height="600" xmlns="http://www.w3.org/2000/svg">
      <rect width="300" height="100" style="fill:rgb(0,0,255);stroke-width:1px;stroke:rgb(0,0,0)">
         <animate attributeName="stroke-width" values="1px;50px;1px;" dur="2s" repeatCount="indefinite" />
      </rect>
      <ellipse stroke="#000" cx="50%" cy="50%" rx="50%" ry="50%">
         <animate attributeName="rx" values="0%;50%;0%" dur="2s" repeatCount="indefinite" />
         <animate attributeName="ry" values="0%;50%;0%" dur="2s" repeatCount="indefinite" />
      </ellipse>
< /svg>
```

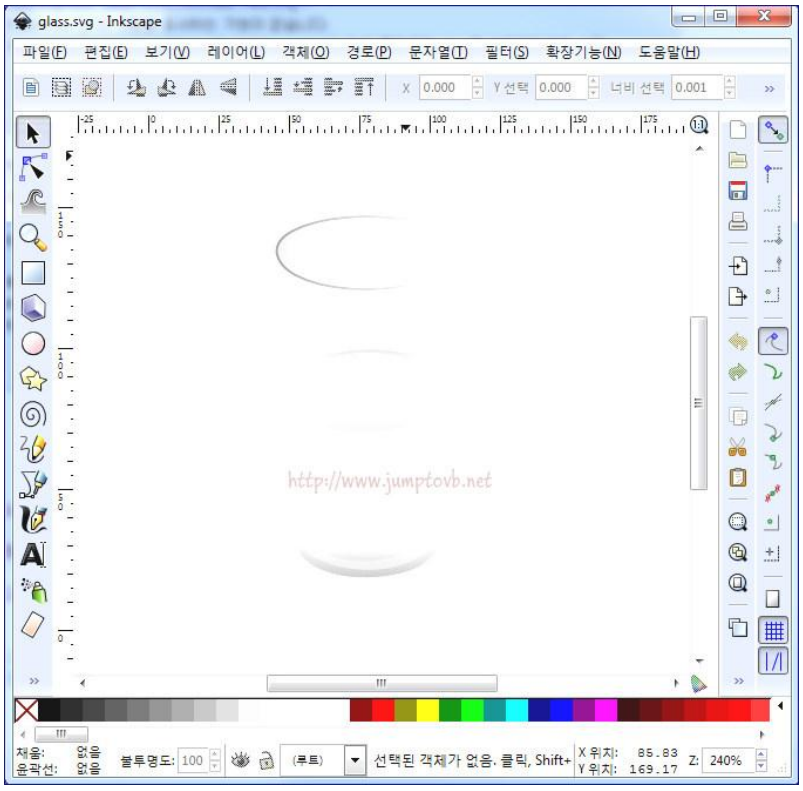
(발췌 : 앞서가는 디자이너와 퍼블리셔를 위한 HTML5 & CSS)

아래 Capture image 는 위 코드를 실행한 결과이다. Animation 기능이 동작한다. (두 도형이 점차로 커졌다가 작아지는 것을 반복하는 예제다.)

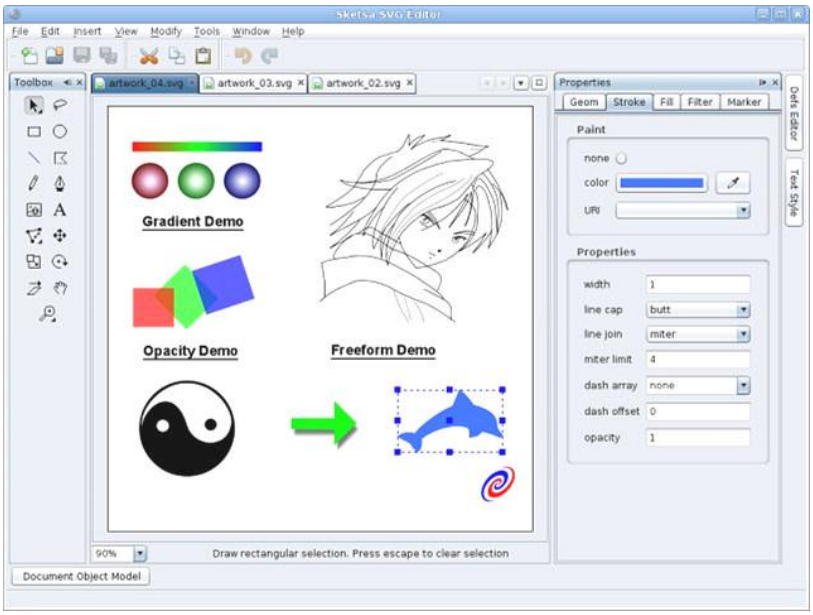


SVG Tool 을 몇 가지 소개하면 다음과 같다.

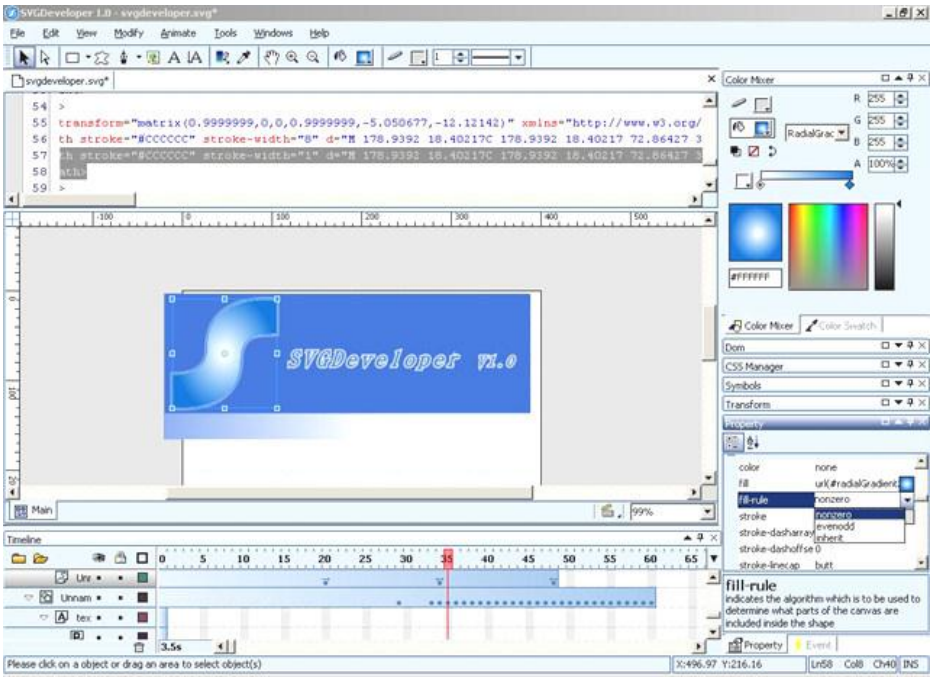
inkscape (http://inkscape.org/)



Sketsa SVG Editor (http://www.kiyut.com/products/sketsa/index.html)



SVGDeveloper (http://www.perfectsvg.com/)



SVG 에 관심이 있으신 분들께서는 위 Editor 를 다운받아서 설치해보시고 테스트 해보셔도 좋을 듯 하다.

그럼 이만 SVG 에 대한 살펴보기를 마치겠습니다.

위의 내용들은 웹표준 가이드 HTML5 CSS3, 앞서가는 디자이너와 퍼블리셔를 위한 HTML5 & CSS 을 참고 하였습니다.

[HTML5 강좌] 20. File API

이번엔 HTML 5 의 File API 를 살펴보려 한다. 이젠 Web Application 으로 사용자 PC 의 파일에 접근 할 수 있다. File API 로 이젠 훨씬 더 많은 일들을 할 수 있게 되었다.

20-1. File API Interface

- File interface
- FileReader interface

File interface 를 사용하면 File 의 File Name 이나 Size 등 기본적인 정보에 접근 할 수 있다.
File Reader interface 는 File 의 내용을 읽을 수 있는 기능을 제공한다.

File interface 는 Browser 가 막 건드릴 수있는 것이 아니라 File 선택 Form 이나 Drag & Drop 을 통해서 사용자가 직접 선택한 File 에 한정하는 것으로 보안 이슈를 피해가고 있다.

20-2. File interface

File interface 가 가지고 있는 속성과 함수는 다음과 같다.

Attribute/Method	설명
name	File 이름
type	File 의 MIME Type(알수 없을때는 null)
urn	File 의 범용 식별자
size	File Size
slice(start, length)	시작위치와 끝 위치를 지정하여 파일의 내용을 잘라내 새로운 Blob 객체를 만드는 함수

위 사항들은 아래 예제에서 사용법을 확인하실 수 있을 것이다.

20-3. FileReader interface

FileReader interface 는 비동기적인 상황에서 사용을 하는데 다음은 FileReader interface 가 가지고 있는 속성과 함수들이다.

Attribute/Method	설명
readAsBinaryString(fileBlob)	File 내용을 읽어 Binary 문자열로 저장
readAsText(fileBlob, encoding)	File 내용을 읽어 들여 문자열로 저장, 두번째 인수는 File 의 문자 encodeing 을 지정할 수 있음(기본값 : UTF-8)
readAsDataURL("file)	File 내용을 읽어 dataURL 형식의 문자열로 저장
Result	읽어들인 File 내용
Error	error 발생시의 error 정보
Onload	읽어들이기에 성공 했을 때 호출하는 event handler. load event 에 대응
Onprogress	읽어들이기의 진행 상황을 얻을 수있는 event handler. progress event 에 대응
Onerror	읽어들이기 error 시에 호출되는 event handler. error event 에 대응.

20-4. Error

다음은 FileReader 가 동작하는 중에 발생하는 Error 에 대한 상수값들이다.

상수	Code	설명
NOT_FOUND_ERR	1	읽을 File 을 찾지 못할때
SECURITY_ERR	2	Web Application 이 Access 하기에 안전하지 못한 File 일 때 File 에 너무 많은 읽기 호출이 있을 때 사용자의 선택한 이후에 File 에 변경이 있을 때
ABORT_ERR	3	예를 들어 abort() 함수 호출과 같은 것으로 인해 읽기가 중지되었을 때
NOT_READABLE_ERR	4	File 접근 권한 문제와 같은 것으로 인해 File 을 읽지 못할때
ENCODING_ERR	5	동기적, 비동기적으로 readAsText() 함수를 사용할 때는 사용할 수없습니다. DataURL FH 로 표현될 수 있는 File 이나 Blob 을 구현한 제한된 곳의 DataURL 에 대한 URL 길이 제한에 걸렸을 때

20-5. 사용법

File interface 는 다음과 같이 사용할 수있습니다. File Name 과 File Size 를 조회 한다.

```
var file = document.getElementById("file").files[0];

document.getElementById("fileName").textContent = file.name;
document.getElementById("fileSize").textContent = "(" + file.size + "byte)";
```

다음은 FileReader 를 생성하여 사용하는 방법이다.

```
var reader = new FileReader();
```

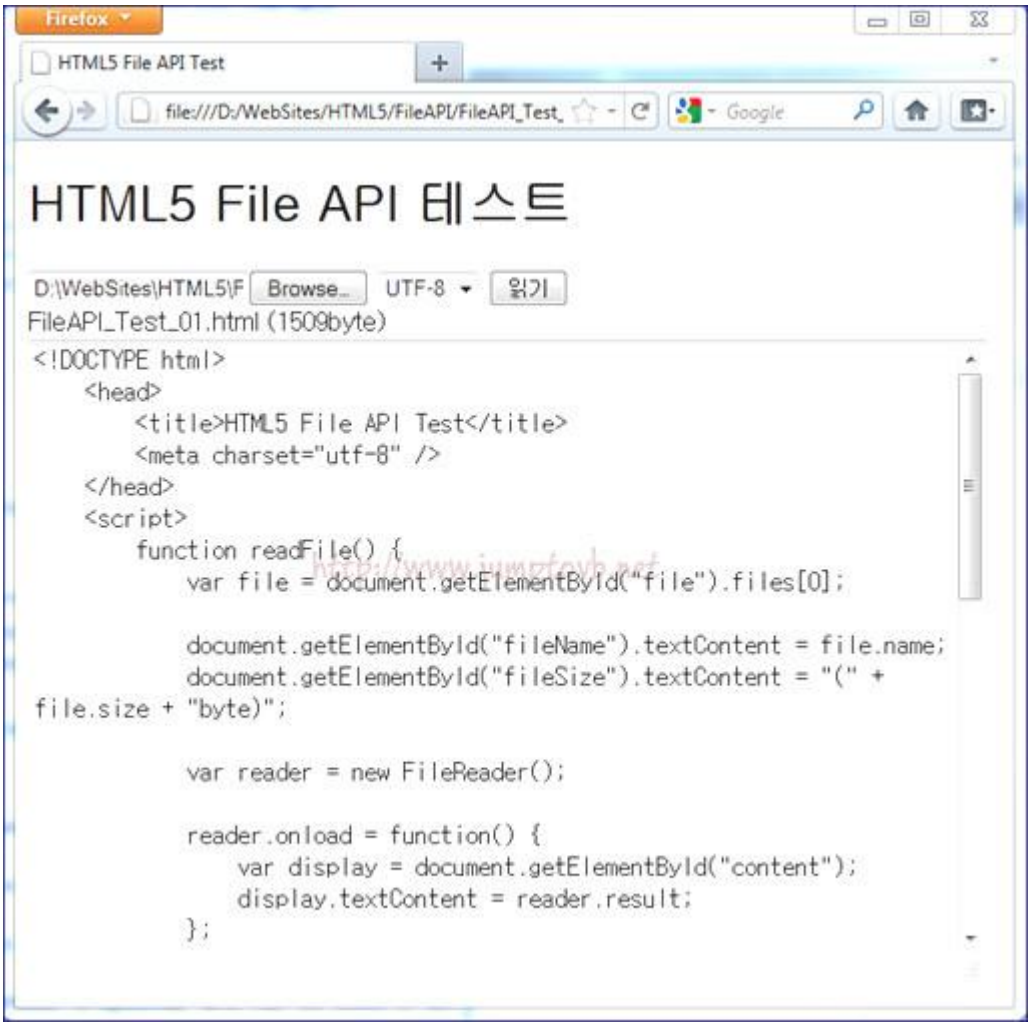
20-6. 예제

아래 예제는 File interface 와 FileReader interface 를 사용하여 File 을 선택하여 File 의 이름과 Size 를 표시하고, File 의 내용을 읽어 TextBox 에 표시한다.

(발췌: 혁명을 꿈꾸다 HTML5 & API 입문)

```
<!DOCTYPE html>
  <head>
    <title>HTML5 File API Test</title>
     <meta charset="utf-8" />
  </head>
  <script>
    function readFile() {
       var file = document.getElementById("file").files[0];

       document.getElementById("fileName").textContent = file.name;
       document.getElementById("fileSize").textContent = "(" + file.size + "byte)";

         var reader = new FileReader();

       reader.onload = function() {
          var display = document.getElementById("content");
          display.textContent = reader.result;
       };

        reader.onerror = function(evt) {
          alert(evt.target.error.code);
        };

       var encodingList = document.getElementById("encoding");
       var encoding = encodingList.options[encodingList.selectedIndex].value;

      reader.readAsText(file, encoding);

    };
  </script>
  <body>
     <h1>HTML5 File API 테스트</h1>
    <input id="file" type="file">
    <select id="encoding">
      <option>UTF-8</option>
     </select>
     <button onclick="readFile()">읽기</button><br />
     <div>
       <span id="fileName">File Name</span>
       <span id="fileSize">File Size</span>
     </div>
     <textarea id="content" readonly style="width:600px; height:400px;"></textarea>
  </body>
</html>
```

위 파일은 저장할 때 **UTF-8 방식으로 저장**해 주어야 한다. 읽을 때 Encoding 을 UTF-8 로 지정했기 때문이다. Web Page 를 실행시키고 위 파일을 다시 읽어보면 다음과 같습니다.



비동기 방식의 FileReaderSync 도 있습니다만 Worker 안에서 사용해야하고 FileReader 보다는 간단하다는 특징이 있다. 다른 사항들은 거의 비슷하므로 접근하기는 비동기방식의 FileReader 보다 간단하다.

이제까지 File API 를 살펴 보았다. 이젠 Web Page 에서 내 PC 의 File 을 읽기위해서 File 을 서버로 Upload 할 필요가 없다. 그 만큼 Server 의 부하도 줄일수 있고, Upload 하고 File 을 읽고 다시 읽은 Streaming 을 내 PC 로 받는 등의 진행되는 동안의 대기 시간이 줄어들 것이다.

위의 내용들은 혁명을 꿈꾸다 HTML5 & API 입문 을 참고 하였습니다.