

<jQuery 시리즈 강좌 리스트>

Contents

[jQuery강좌] 1. 웹 개발자를 위한 jQuery 기본이해	3
[jQuery강좌] 2. jQuery를 이용한 HTML DOM 접근 - 기본 선택터 (1)	8
[jQuery강좌] 3. jQuery를 이용한 HTML DOM 접근 - 기본 선택터 (2)	14
[jQuery강좌] 4. jQuery Selector - 속성(Attribute)	20
[jQuery강좌] 5. jQuery Selector - DOM 계층(Hierarchy)을 이용한 요소 접근 (1)	29
[jQuery강좌] 6. jQuery Selector - DOM 계층(Hierarchy)을 이용한 요소 접근 (2)	32
[jQuery강좌] 7. jQuery Filter - 기본필터(Basic Filter)	41
[jQuery강좌] 8. jQuery Filter - 폼 필터(Form Filter)	49
[jQuery강좌] 9. jQuery Filter - 자식필터(Child Filter)	52
[jQuery강좌] 10. jQuery Traverse - Filtering	57
[jQuery강좌] 11. jQuery Traverse - Miscellaneous Traversing	72
[jQuery강좌] 12. jQuery Traverse - Tree Traversal	79
[jQuery강좌] 13. jQuery Core	86
[jQuery강좌] 14. jQuery CSS - 스타일 관련 메서드에 대하여	90
[jQuery강좌] 15. jQuery Attribute - 요소의 속성 관련 메서드에 대하여	101
[jQuery강좌] 16. jQuery Form API - 폼 지원 메서드에 대하여	107
[jQuery강좌] 17. jQuery Event - 이벤트 지원 메서드	112
[jQuery강좌] 18. jQuery Event - bind() 메서드	120
[jQuery강좌] 19. jQuery Event - 이벤트에 생명을~	127
[jQuery강좌] 20. jQuery Performance	136

[jQuery 시리즈 강좌 및 동영상 리스트]

[\[jQuery 동영상 강좌\] 1. 웹 개발자를 위한 jQuery 기본이해](#)

[\[jQuery 동영상 강좌\] 2. jQuery 를 이용한 HTML DOM 접근 - 기본 선택터 \(1\)](#)

[\[jQuery 동영상 강좌\] 3. jQuery 를 이용한 HTML DOM 접근 - 기본 선택터 \(2\)](#)

[\[jQuery 동영상 강좌\] 4. jQuery Selector - 속성\(Attribute\)](#)

[\[jQuery 동영상 강좌\] 5. jQuery Selector - DOM 계층\(Hierarchy\)을 이용한 요소 접근 \(1\)](#)

[\[jQuery 동영상 강좌\] 6. jQuery Selector - DOM 계층\(Hierarchy\)을 이용한 요소 접근 \(2\)](#)

[\[jQuery 동영상 강좌\] 7. jQuery Filter - 기본필터\(Basic Filter\)](#)

[\[jQuery 동영상 강좌\] 8. jQuery Filter - 폼 필터\(Form Filter\)](#)

[\[jQuery 동영상 강좌\] 9. jQuery Filter - 자식필터\(Child Filter\)](#)

[\[jQuery 동영상 강좌\] 10. jQuery Traverse - Filtering](#)

[\[jQuery 동영상 강좌\] 11. jQuery Traverse - Miscellaneous Traversing](#)

[\[jQuery 동영상 강좌\] 12. jQuery Traverse - Tree Traversal](#)

[\[jQuery 동영상 강좌\] 13. jQuery Core](#)

[\[jQuery 동영상 강좌\] 14. jQuery CSS - 스타일 관련 메서드에 대하여](#)

[\[jQuery 동영상 강좌\] 15. jQuery Attribute - 요소의 속성 관련 메서드에 대하여](#)

[\[jQuery 동영상 강좌\] 16. jQuery Form API - 폼 지원 메서드에 대하여](#)

[\[jQuery 동영상 강좌\] 17. jQuery Event - 이벤트 지원 메서드](#)

[\[jQuery 동영상 강좌\] 18. jQuery Event - bind\(\) 메서드](#)

[\[jQuery 동영상 강좌\] 19. jQuery Event - 이벤트에 생명을~](#)

[\[jQuery 동영상 강좌\] 20. jQuery Performance](#)

[jQuery 강좌] 1. 웹 개발자를 위한 jQuery 기본이해

웹 프런티어와 함께하는 jQuery 기초강좌

1st - 웹 개발자를 위한 jQuery 기본이해

웹 사이트와 사용자간의 원활한 소통을 위해 사용되었던 자바스크립트는 최근 몇 년간 Web2.0 과 Ajax 기술을 활용한 RIA(Rich Internet Application)의 등장으로 인하여 이전보다 복잡하고 늘어난 코드와 다양한 웹 브라우저의 등장으로 크로스브라우징이라는 장벽이 웹 개발자에게 큰 스트레스를 안겨주고 있다. 이러한 문제에서 쉽게 벗어날 수 있는 해법과 웹 개발에 새로운 패러다임(역동적인 인터페이스, 쉬운 프로그래밍)을 제시하고 있는 jQuery에 대해 자세히 알아보는 보도록 하겠다.

jQuery는 2006년 "John Resig"에 의해 디자인된 자바스크립트 라이브러리이다.

자바스크립트의 코드를 단순하고 간결한 상태로 개발이 가능하며, 동일한 코드의 반복과 복잡하고 많은 코드로 개발되던 기존의 작업에 비해 여러 가지 효과나 이벤트를 간단한 함수의 호출만으로 쉽고 빠르게 개발이 가능하도록 도와준다.

The screenshot shows the jQuery website homepage. At the top, there is a navigation bar with links: jQuery, Plugins, UI, Meetups, Forum, Blog, About, and Donate. Below this is the jQuery logo with the tagline "write less, do more.". To the right of the logo is another navigation bar with links: Download, Documentation, Tutorials, Bug Tracker, and Discussion. The main content area features a large heading: "jQuery is a new kind of JavaScript Library." followed by a paragraph describing jQuery as a fast and concise JavaScript library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. Below this paragraph are three checkmarks with labels: "✓ Lightweight Footprint", "✓ CSS3 Compliant", and "✓ Cross-browser". On the right side of the main content area, there is a section titled "GRAB THE LATEST VERSION!" with a sub-heading "CHOOSE YOUR COMPRESSION LEVEL:". Under this, there are two radio buttons: "PRODUCTION (31KB, Minified and Gzipped)" which is selected, and "DEVELOPMENT (227KB, Uncompressed Code)". Below the radio buttons is a large button with a download icon and the text "Download(jQuery);". At the bottom of this section, it says "Current Release: v1.6". At the very bottom of the page, there is a section titled "WHO'S USING JQUERY?" followed by logos of various companies and organizations including Google, Dell, IBM, NBC, CBS, Netflix, Technorati, mozilla.org, and others.

< jQuery 의 특징 >

jQuery 의 가장 큰 특징으로는 다음과 같이 4 가지를 이야기 할 수 있다.

1. CSS 셀렉터

html 문서의 구조를 명료 하면서도 읽기 쉬운 형태로 표현 및 사용 가능 한다.

2. 플러그인 아키텍처

중복되는 기능과 코드가 엉키는 등의 Feature Creep 을 피하고 창의적인 산출물의 공유가 가능하며, 이미 개발된 많은 플러그인을 쉽고 빠르게 이용할 수 있다.

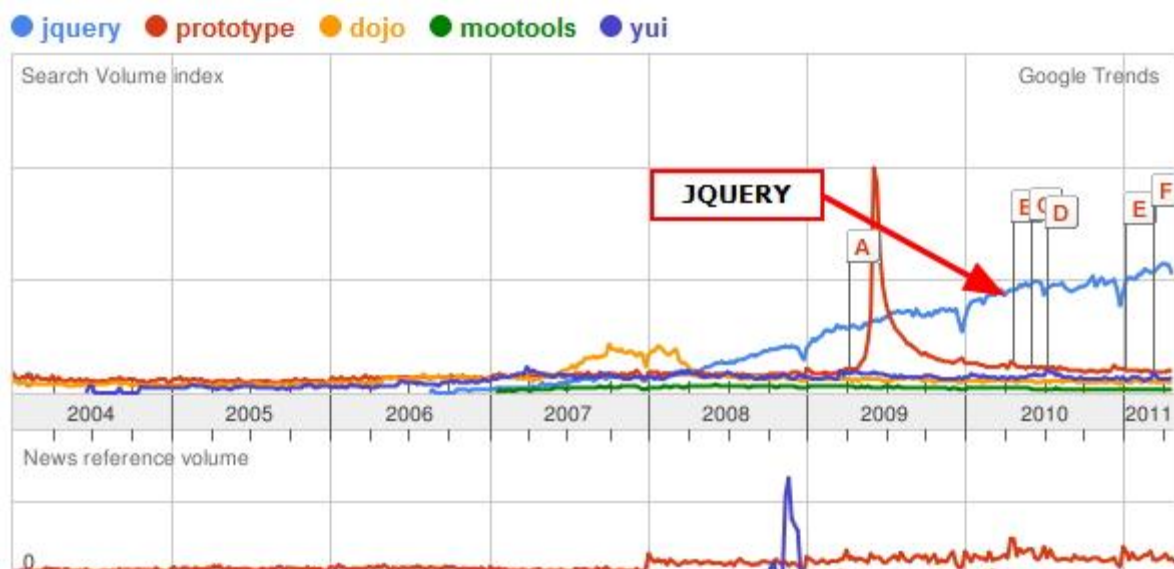
3. 메서드 체인

여러 개의 동작(기능)을 한 줄에 나열하여 임시 변수의 사용을 최소화 하여 불필요한 코드의 반복을 피할 수 있다.

4. 크로스브라우저

각각의 브라우저에 발생하는 이벤트, 객체 처리방법에 따라 여러 개의 함수 또는 여러 번의 분기가 없이 jQuery 에서 제공하는 함수 또는 문장으로 간단히 해결 할 수 있다.

이러한 특징으로 인하여 자바스크립트 라이브러리 중 jQuery 는 Prototype, Script.aculo.us, Dojo, Mootools 와 같은 자바스크립트 라이브러리에 비해 상대적으로 늦게 등장했음에도 불구하고 많은 웹 개발자 사이에서 상당히 많은 지지를 받고 있다.



< jQuery 를 사용할 때 미리 알아두면 좋아요 >

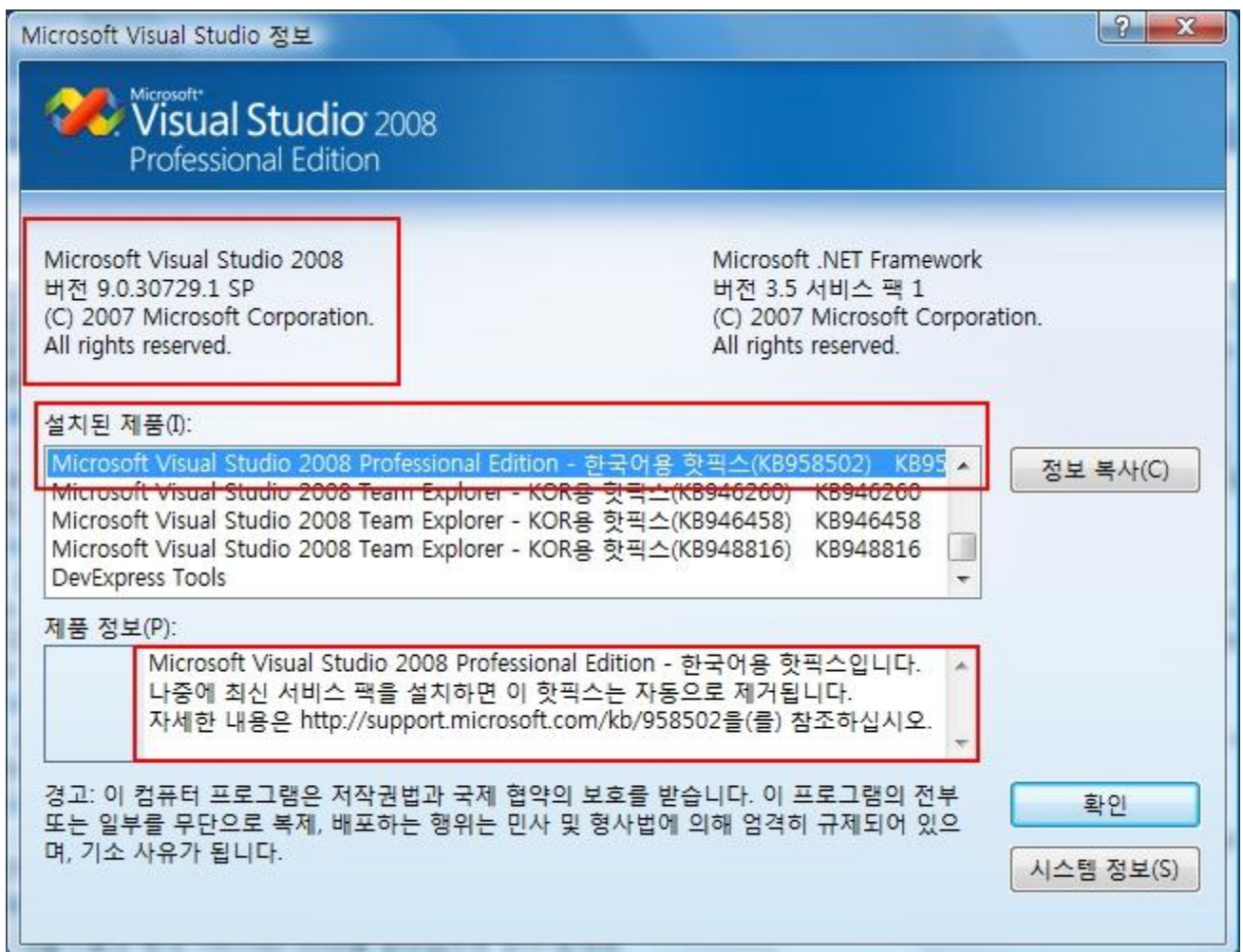
jQuery 를 사용할 때 알아두면 좋은 2 가지 팁을 정리해 보았다.

1. Visual Studio Intellisense ? 자바스크립트 도움말

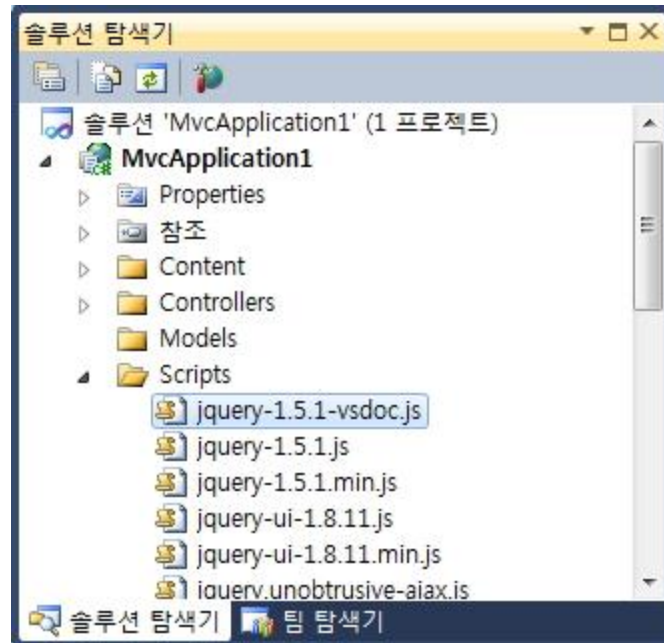
비주얼스튜디오 2010 버전의 경우 기본적으로 자바스크립트 인텔리센스를 지원하고 있다.

Visual Studio 2008 SP1 이상의 버전에서는 “vsdoc.js” 파일과 간단한 설정만을 통해 자바스크립트의 인텔리센스 사용이 가능하다.

인텔리센스를 사용하기 위해서는 우선 vsdoc.js 관련 패치 파일인 [VS90SP1-KB958502-x86.exe](#) 을 다운받아 설치 해야 하며, 만약 자신이 설치한 VS2008 에 서비스 팩 1 이 설치가 되어 있지 않았다면, 관련 패치는 설치 되지 않으니 VS2008 의 서비스 팩의 설치 유무를 먼저 확인 하시길 바란다.



핫픽스 KB958502 가 설치가 완료 되었다면 <http://jquery.com> 의 Download 메뉴에서 제공하고 있는 "jquery.js"와 "?vsdoc.js"를 다운 받은 후에 동일한 디렉터리에 위치 시키면 된다.
(2010, 2008 동일하다.)



```
<head>
  <title>jQuery Selector</title>
  <script src="../../Scripts/jquery-1.5.1.min.js" type="text/javascript"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      //jQuery Script...
      //TODO
    });
  </script>
</head>
```

<script>태그를 이용하여, 해당 jQuery 를 참조시키면 바로 인텔리센스가 동작하는 걸 확인 할 수 있으며, 사용자가 새로 작성하는 독립된 자바스크립트 문서에서는 상단에 <reference path="-vsdoc.js" />를 사용하시면 된다.

Scripts/custom.js*
jquery/MultiSelector.aspx
jquery/ClassSelector.aspx
jquery/ElementSelector.aspx
jquery/Selector.aspx

```

/// <reference path="jquery-1.5.1-vsdoc.js" />
$(function() {
  // jQuery $(String selector, jQuery context)
  1: $(expression, context) - This function accepts a string containing a CSS selector which is then used to match a set of elements.
  2: $(html) - Create DOM elements on-the-fly from the provided String of raw HTML.
  3: $(elements) - Wrap jQuery functionality around a single or multiple DOM Element(s).
  4: $(callback) - A shorthand for $(document).ready().
  5: $() - As of jQuery 1.4, if you pass no arguments in to the jQuery() method, an empty jQuery set will be returned.
  selector: 1: expression - An expression to search with.
             2: html - A string of HTML to create on the fly.
             3: elements - DOM element(s) to be encapsulated by a jQuery object.
             4: callback - The function to execute when the DOM is ready.

```


이제 jQuery API 사이트를 한쪽에 열어놓지 않고 jQuery의 기능을 이용하여 쉽고 빠르게 클라이언트 개발을 할 수 있다.

2. CDN (Content Delivery Network)

CDN HOSTED JQUERY

A number of large enterprises provide hosted copies of jQuery on existing [CDN](#) networks that are available for public use. Below are links to the CDN-hosted copies of jQuery that you may hotlink to.

■ Google Ajax API CDN

- <http://ajax.googleapis.com/ajax/libs/jquery/1.5.2/jquery.min.js>
- [Google Ajax CDN Documentation](#)

■ Microsoft CDN

- <http://ajax.aspnetcdn.com/ajax/jquery/jquery-1.6.min.js>
- [Ajax CDN Announcement](#), [Microsoft Ajax CDN Documentation](#)

■ jQuery CDN (via Media Temple)

- <http://code.jquery.com/jquery-1.6.min.js> Minified version
- <http://code.jquery.com/jquery-1.6.js> Source version

Google, Microsoft, jQuery에서는 각각 CDN을 지원하고 있다. 웹 어플리케이션의 속도를 높이기 위해 전송헤더에 Gzip 압축과 캐싱을 지원하고 있다. jquery의 용량과 다운로드 속도가 우려되는 곳에서는 해당 링크를 참조하시기 바란다.

이제까지 jQuery에 대해 간단히 알아 보았다.

다음 강좌부터는 jQuery의 셀렉터에 대해 알아 보기로 하겠다.

[jQuery 강좌] 2. jQuery 를 이용한 HTML DOM 접근 - 기본 셀렉터 (1)

웹 프런티어와 함께하는 jQuery 기초강좌

2nd - jQuery 를 이용한 HTML DOM 접근(기본 셀렉터 첫 번째 이야기)

<원하는 개체를 쉽고 편하게 선택하자 - 셀렉터>

이번 시간에는 jQuery 의 가장 강력한 기능인 HTML DOM 을 탐색하는 기능에 대해 알아보도록 하겠다.

jQuery 의 DOM 탐색은 CSS SELECTER 를 사용하고 있어, CSS 에 사용한 표현식을 알고 있다면 보다 쉽게 셀렉터를 이해 할 수 있다.

jQuery 에서는 원하는 HTML 의 DOM 요소를 찾기 위해 \$(Selector), jQuery(Selector)와 같은 표현식을 사용하다.

\$는 jQuery 의 축약어로 같은 역할을 하며, 다음과 같은 형태로 사용하여 원하는 DOM 요소를 선택 할 수 있다.

| 셀렉터의 종류 | 셀렉터 표현 방법 |
|-------------------|--|
| All Selector | \$("*") |
| ID Selector | \$("#id") |
| Element Selector | \$("elementName") |
| Class Selector | \$(".className") |
| Multiple Selector | \$("selector1, selector2, selector3, selectorN") |

\$(Selector), jQuery(Selector)를 사용하여 선택한 DOM 의 요소는 "document.getElementById"를 이용한 것과는 다르게 해당 객체를 jQuery 객체로 래핑해서 반환해 주기 때문에 jQuery 에서 지원하는 기능을 쉽게 적용 할 수 있다는 장점이 있다.

1. All Selector : \$("*")

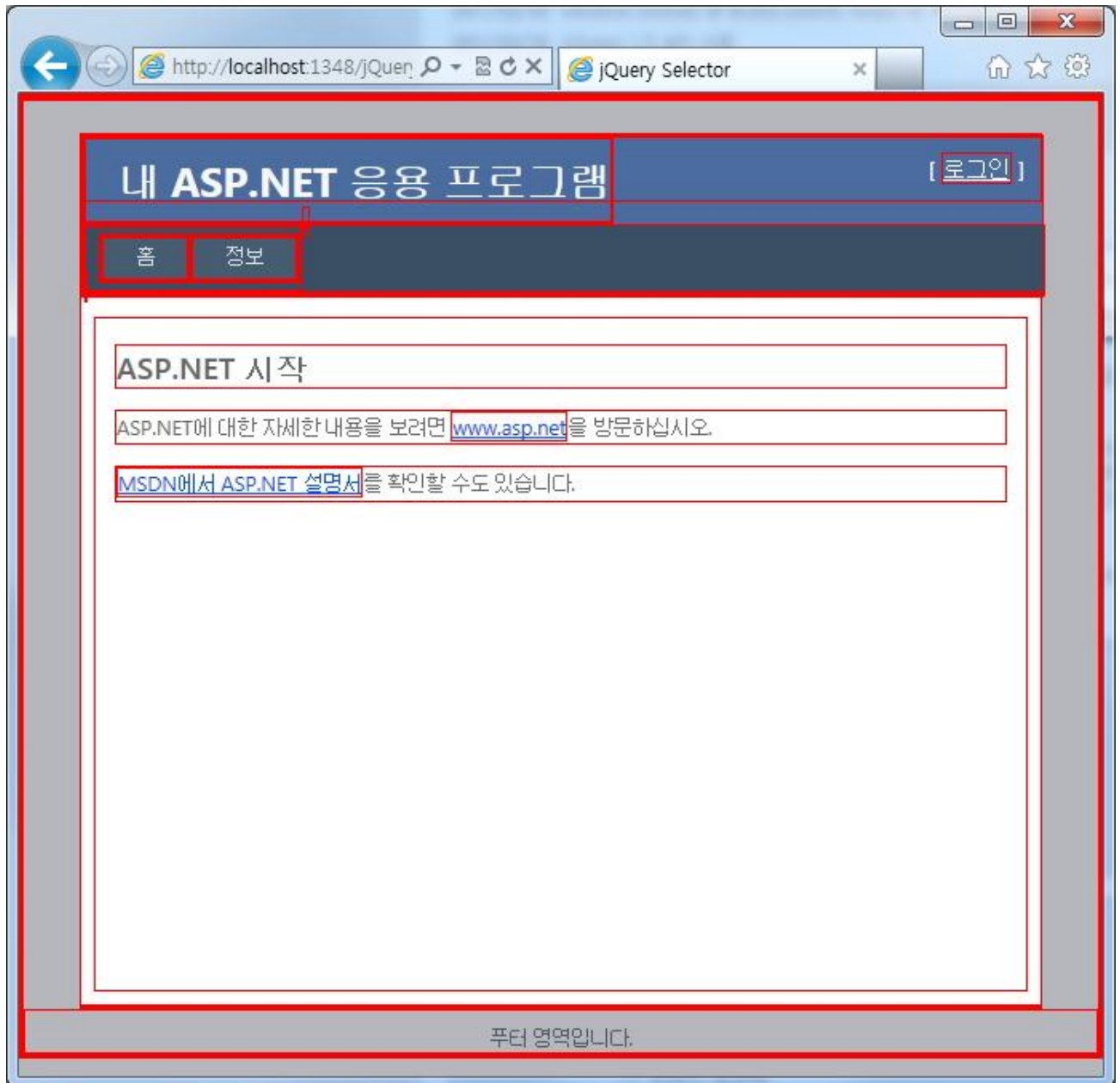
HTML DOM 을 탐색하여 모든 요소를 배열형식의 jQuery 개체로 반환한다.



```
<script src="http://code.jquery.com/jquery-latest.js"></script>
<script type="text/javascript">
    $(document).ready(function () {
        //HTML 문서내의 모든 엘리먼트를 찾아 붉은 선으로 표시합니다.
        $("*").css("border", "1px solid #ff0000");
    });
</script>
```

jQuery 를 페이지에 참조 시키고 \$(document).ready() 함수를 통해 모든 객체를 선택한 후 선택된 jQuery 객체를 눈에 잘 보이게 붉은색의 1 픽셀 테두리를 설정 하도록 해 보겠다.

\$(document).ready() 또는 \$(function() {})는 페이지의 HTML DOM 이 모두 로드가 되면 실행이 되는 함수로 자바스크립트의 onload 메서드의 확장된 역할을 하고 있다. jQuery 를 사용하실 경우 onload 이벤트의 사용보다 \$(document).ready() 또는 \$(function() {})를 사용하시길 적극 추천 드리며 관련 내용에 대해서는 마지막 부분에 다시 한번 정리를 하도록 하겠다.



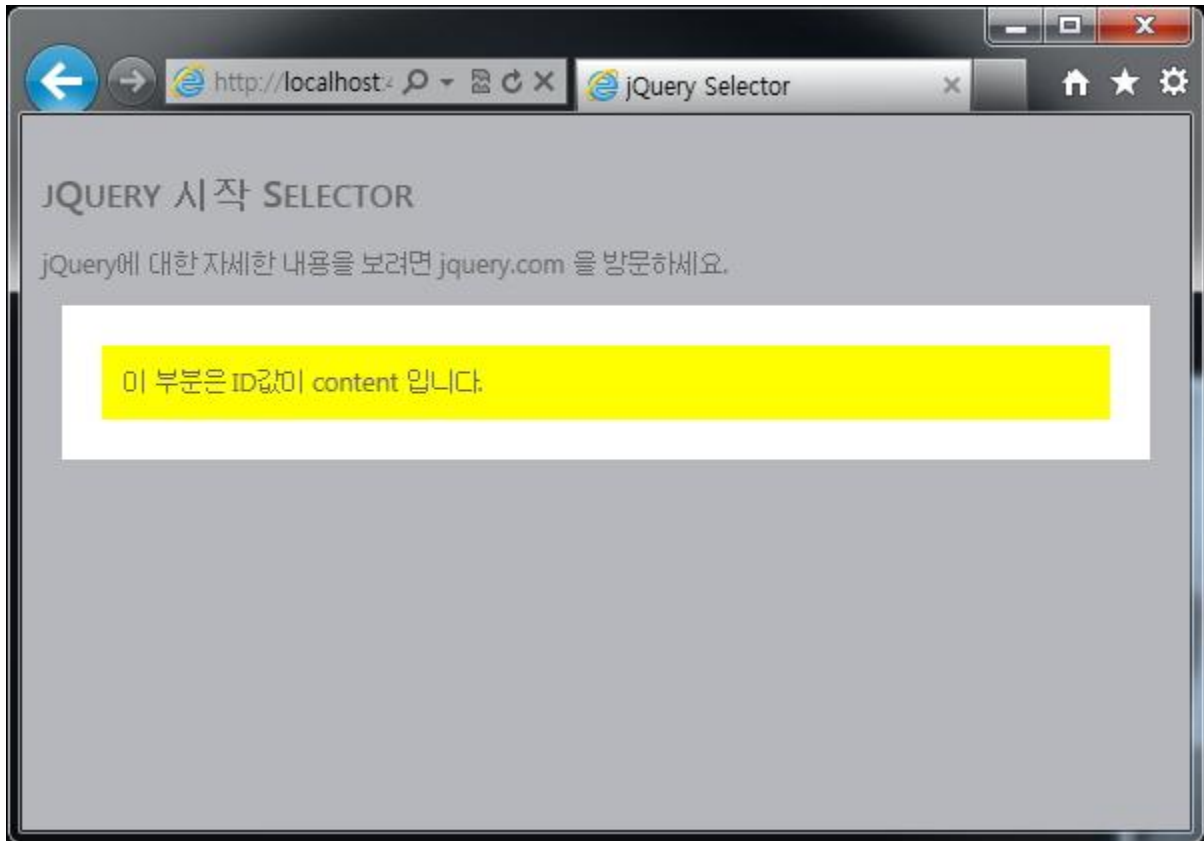
화면 2 와 같이 jQuery 를 통해 선택된 모든 요소에 붉은색 테두리가 설정된 것을 확인 할 수 있다. 이러한 기능이 가능한 것은 일반적인 DOM 개체가 아닌 jQuery 개체로 반환이 되었기 때문이다.

2. ID Selector : \$("#ID")

문서 안에 있는 여러 엘리먼트중 ID 값이 동일한 엘리먼트를 찾아 반환한다.

동일한 값(동일 ID)의 엘리먼트가 여러 개일 경우에는 최상위에 있는 엘리먼트를 선택 반환하며, 한 문서(HTML)에는 한 ID 만 존재하는 것이 원칙으로 동일한 값을 통해 접근을 하고 싶을 경우에는 class 또는 attribute 의 동일한 값을 설정 하여 사용하길 권장한다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div { background : #FFF; padding:10px; margin:10px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      //ID 가 Content 인 요소를 찾아 배경을 "yellow"로 변경한다.
      $("#content").css("background", "yellow");
    });
  </script>
</head>
<body>
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <div id="content">
      이 부분은 ID 값이 content 이다.
    </div>
  </div>
</body>
</html>
```



3. Element Selector : \$("element")

자바스크립트의 `getElementByTagName("tagName")`과 비슷한 역할을 하며 DOM 개체를 구성하는 태그와 동일한 개체를 찾아 다수의 jQuery 개체를 반환한다.

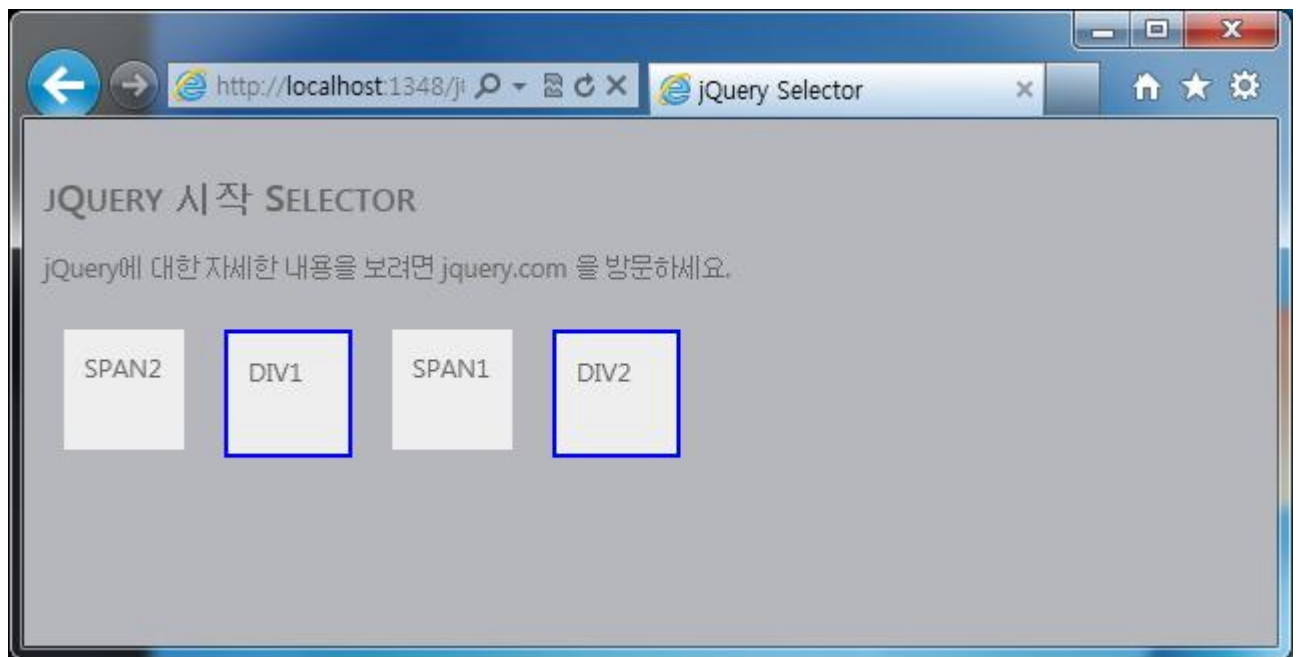
아래의 소스는 jQuery 를 통해 선택된 "DIV" 개체의 테두리는 파란색으로 변경한다.

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,span {
      width:40px; height:40px; float:left; padding:10px; margin:10px;
      background-color:#EEEEEE;
    }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js" ></script>
  <script type="text/javascript">
```

```

$(document).ready(function () {
    //div 개체를 찾아 테두리를 "blue"로 변경한다.
    $("div").css("border", "2px solid blue");
});
</script>
</head>
<body style="padding:10px;">
    <h2>jQuery 시작 Selector</h2>
    <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
    <span>SPAN2</span>
    <div>DIV1</div>
    <span>SPAN1</span>
    <div>DIV2</div>
</body>
</html>

```



이번 시간에는 jQuery 가장 기본이 되는 셀렉터의 기능중 가장 많이 사용하는 id, element 를 이용한 방법에 대해 알아 보았다.

다음 시간에는 요소의 클래스(class)와 id, element, class 를 조합하여 개체에 접근 하는 방법에 대해 알아 보기로 하겠다.

[jQuery 강좌] 3. jQuery 를 이용한 HTML DOM 접근 - 기본 셀렉터 (2)

웹 프런티어와 함께하는 jQuery 기초강좌

3rd - jQuery 를 이용한 HTML DOM 접근(기본 셀렉터 두 번째 이야기)

<원하는 개체를 쉽고 편하게 선택하자 - 셀렉터>

셀렉터의 종류	셀렉터 표현 방법
All Selector	<code>\$("*")</code>
ID Selector	<code>\$("#id")</code>
Element Selector	<code>\$("elementName")</code>
Class Selector	<code>\$(".className")</code>
Multiple Selector	<code>\$("selector1, selector2, selector3, selectorN")</code>

지난 시간에 이어 기본이 되는 셀렉터의 나머지 부분인 클래스와 여러가지 셀렉터를 조합하여 요소에 접근하고, 개체를 탐색 선택하는 방법에 대한 이야기를 진행 하도록 하겠다.

1. Class Selector : `$(".class")`

자바스크립트의 `getElementByClassName()`과 동일한 역할을 하고 있는 셀렉터 이다.

`getElementByClassName()` 메서드의 경우 몇몇 브라우저에서 지원을 하고 있지 않기 때문에 크로스브라우저를 지원해야 하는 개발이라면 올바르게 동작하지 않는 문제가 발생할 수 있다.

IE8 이하 버전에서는 지원을 하고 있지 않다고 하며, IE9 과 HTML 의 차세대 버전인 HTML5에서는 기본적으로 지원을 한다.

프론트 부분을 개발하다 보면 위와 같은 문제로 인하여, 많은 개발자가 고민하고 해결책을 찾다가 많은 시간을 허비하는 경우가 상당하다. 문제를 찾기위해 자바스크립트 디버깅을 해 보지만 상당한 노가다를 필요로 하는 경우도 많을뿐더러 브라우저마다 에러를 내는 형식이 다르기 때문에 상당한 골치 덩어리 이다.

여기서 다시 한번 jQuery 의 강점인 크로스브라우징에 대한 강조를,,, jQuery 의 ClassSelector 의 경우 현존하는 대부분의 브라우저에서 사용이 가능하오니, 위와 같은 문제를 미연에 방지 할 수 있으며 이러한 문제에 대해 고민을 할 필요조차 없게 된다.

다시 본론으로 들어가서 Class Selector 에 대해 알아 보기로 하겠다.

다음은 문서(HTML)내의 클래스명이 myClass 인 요소를 모두 찾아 테두리의 색상을 변경하는 예제이다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,span {
      width:100px; height:40px; float:left; padding:10px; margin:10px;
      background-color:#EEEEEE;
    }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      //Class 가 myClass 인 요소를 찾아 테두리를 "blue"로 변경한다.
      $(".myClass").css("border", "2px solid blue");
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <span class="myClass">SPAN class="myClass"</span>
  <div>DIV</div>
  <span>SPAN</span>
  <div class="myClass">DIV class="myClass"</div>
</body>
```


</html>



2. Multiple, Complex Selector :

`$("selector1, selector2, selectorN")`

`$("#id div.class")`

앞서 설명한 셀렉터의 나열이나 조합을 통하여 개발자가 원하는 개체를 보다 쉽고 정확하게 빠르게 탐색 할 수 있다.

셀렉터를 "," 통하여 나열할 경우 각각의 셀렉터를 통해 탐색된 개체의 집합을 반환하며, 셀렉터의 조합을 통하여 탐색을 했을 경우 각 셀렉터의 교집합 조건의 개체가 탐색되어 반환된다.

`$("#content", "div", "a", ".myclass")` 경우 ID 의 값이 "content"인 개체, "div", "a" 태그를 가지는 개체, 클래스 명이 "myClass"인 개체를 탐색하여 반환을 하게 되는 반면 `$("div.myClass")`의 경우 "div" 태그로 구성되었으며 동시에 클래스 명이 "myClass"인 개체를 반환한다.

다음 두가지 예제를 통해 두 가지 셀렉터의 동작과 차이점을 알아 보도록 하겠다.

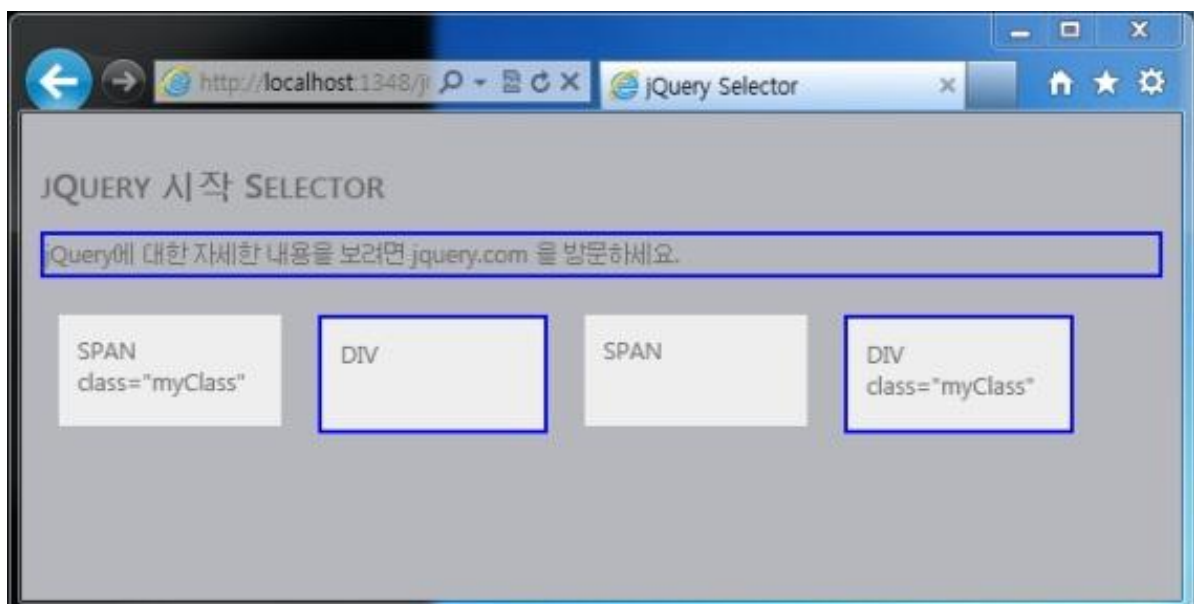
- 1) 첫 번째 예제 : ID 값이 Content 이며 DIV 요소를 찾아 테두리를 파랑색으로 변경한다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
```

```

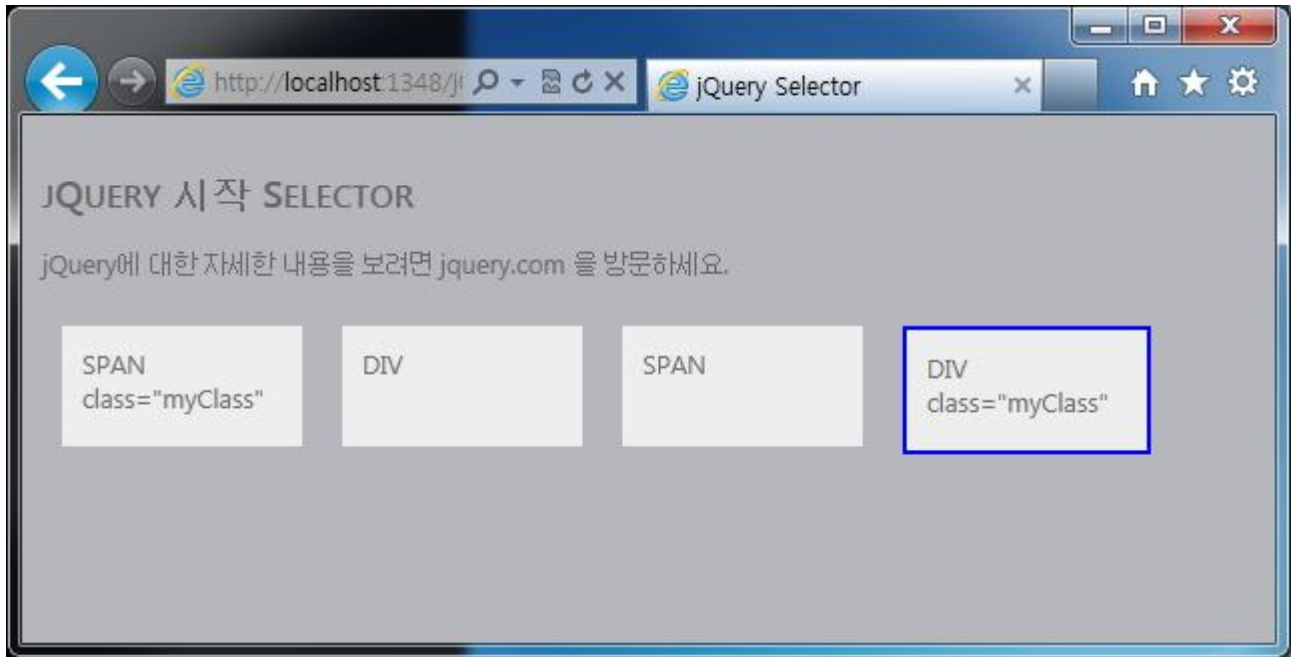
<style>
    div,span
    {
        width:100px;
        height:40px;
        float:left;
        padding:10px;
        margin:10px;
        background-color:#EEEEEE;
    }
</style>
<script src="http://code.jquery.com/jquery-latest.js"></script>
<script type="text/javascript">
    $(document).ready(function () {
        //id=content, div 요소를 찾아 테두리를 "blue"로 변경한다.
        $("#content, div").css("border", "2px solid blue");
    });
</script>
</head>
<body style="padding:10px;">
    <h2>jQuery 시작 Selector</h2>
    <p id="content">jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
    <span class="myClass">SPAN class="myClass"</span>
    <div>DIV</div>
    <span>SPAN</span>
    <div class="myClass">DIV class="myClass"</div>
</body>
</html>

```



- 2) 두 번째 예제 : DIV 요소중에 클래스명이 myClass 인 요소를 찾아 테두리를 파랑색으로 변경한다.

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
  <title>jQuery Selector</title>
  <link href="../../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,span
    {
      width:100px;
      height:40px;
      float:left;
      padding:10px;
      margin:10px;
      background-color:#EEEEEE;
    }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      //div 요소 중 class 가 myclass 인 개체를 찾아 테두리를 "blue"로 변경한다.
      $(".div.myClass").css("border", "2px solid blue");
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p id="content">jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <span class="myClass">SPAN class="myClass"</span>
  <div>DIV</div>
  <span>SPAN</span>
  <div class="myClass">DIV class="myClass"</div>
</body>
</html>
```



이번 시간에는 jQuery의 셀렉터의 기본적인 내용을 알아 보았다.

셀렉터는 jQuery에서 가장 많이 사용하는 부분이며, 동적인 웹 개발에 필수적인 항목이므로 반듯이 이해해야 하는 부분이다.

다음 강좌에서는 요소의 속성(Attribute)을 통해 요소(개체)에 접근하는 방법에 대해 알아 보기로 하겠다.

[jQuery 강좌] 4. jQuery Selector - 속성(Attribute)

웹 프런티어와 함께하는 jQuery 기초강좌

4th 이야기 - 셀렉터 Attribute 를 활용한 요소 선택

<개체의 속성값을 통해 요소를 선택하자 - Attribute Selector>

각각의 요소는 속성(attribute)을 가질 수 있다. 각 요소의 속성은 미리 정해진 이름이거나, 사용자의 필요에 의해서 만들어질 수 있으며 jQuery 의 CSS 셀렉터와 필터의 조합으로 통해 관련된 요소에 접근 할 수 있다.

다음은 jQuery 에서 지원하고 있는 속성관련 셀렉터 이다. (더 많은 종류의 속성관련 셀렉터를 지원하고 있으니 자세한 내용은 jQuery.com 에서 확인을 하시기 바란다.)

형식(셀렉터)	설명
\$(Selector[attr])	attr 속성(attribute)값을 가지는 Selector 요소와 일치
\$(Selector[attr="value"])	attr 속성의 값이 value 와 동일한 값인 Selector 요소와 일치
\$(Selector[attr!="value"])	attr 속성의 값이 value 와 같지 않은 값인 Selector 요소와 일치
\$(Selector[attr^="value"])	attr 속성의 값이 value 값으로 시작하는 Selector 요소와 일치
\$(Selector[attr\$="value"])	attr 속성의 값이 value 값으로 끝나는 Selector 요소와 일치
\$(Selector[attr*="value"])	attr 속성의 값이 value 값을 포함하는 Selector 요소와 일치
\$(Selector[attr~="value"])	attr 속성의 값이 공백과 함께 value 값을 포함하는 Selector 요소와 일치

1. Has Attribute Selector : \$(Selector[attr])

attr 이라는 속성(Attribute)값을 가지는 셀렉터의 요소와 일치하는 요소를 반환한다.

input, checkbox, src, href 등 "<", ">" 사이에 정의된(사용자가 임의로 정의한 값 포함) 값에 대해 일치하는 항목이 존재 한다면 해당 요소를 반환한다.

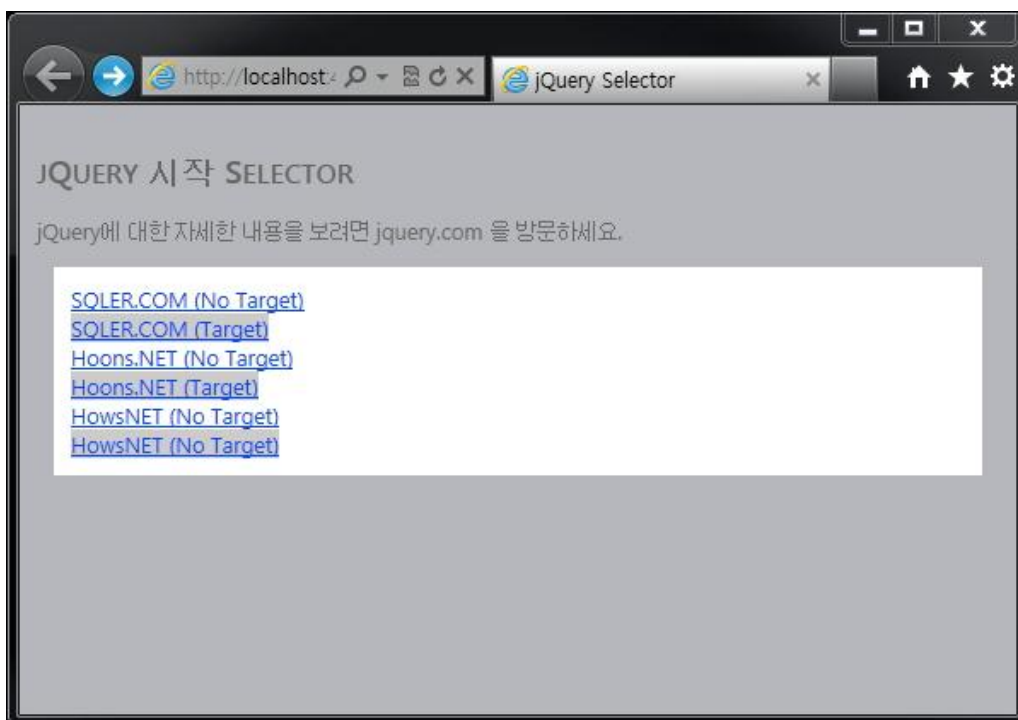
다음은 "a" 태그의 속성 "target"이 존재하는 요소를 선택하는 예제이다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
```

```

    div { background : #FFF; padding:10px; margin:10px; }
</style>
<script src="http://code.jquery.com/jquery-latest.js"></script>
<script type="text/javascript">
    $(document).ready(function () {
        $("div > a[target]").css("background", "#CCC");
    });
</script>
</head>
<body style="padding:10px;">
    <h2>jQuery 시작 Selector</h2>
    <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
    <div>
        <a href="http://www.slqer.com">SQLER.COM (No Target)</a><br />
        <a href="http://www.slqer.com" target="_blank">SQLER.COM (Target)</a><br />
        <a href="http://www.hoons.kr">Hoons.NET (No Target)</a><br />
        <a href="http://www.hoons.kr" target="_blank">Hoons.NET (Target)</a><br />
        <a href="http://www.hows.kr">HowsNET (No Target)</a><br />
        <a href="http://www.hows.kr" target="_blank">HowsNET (No Target)</a><br />
    </div>
</body>
</html>

```

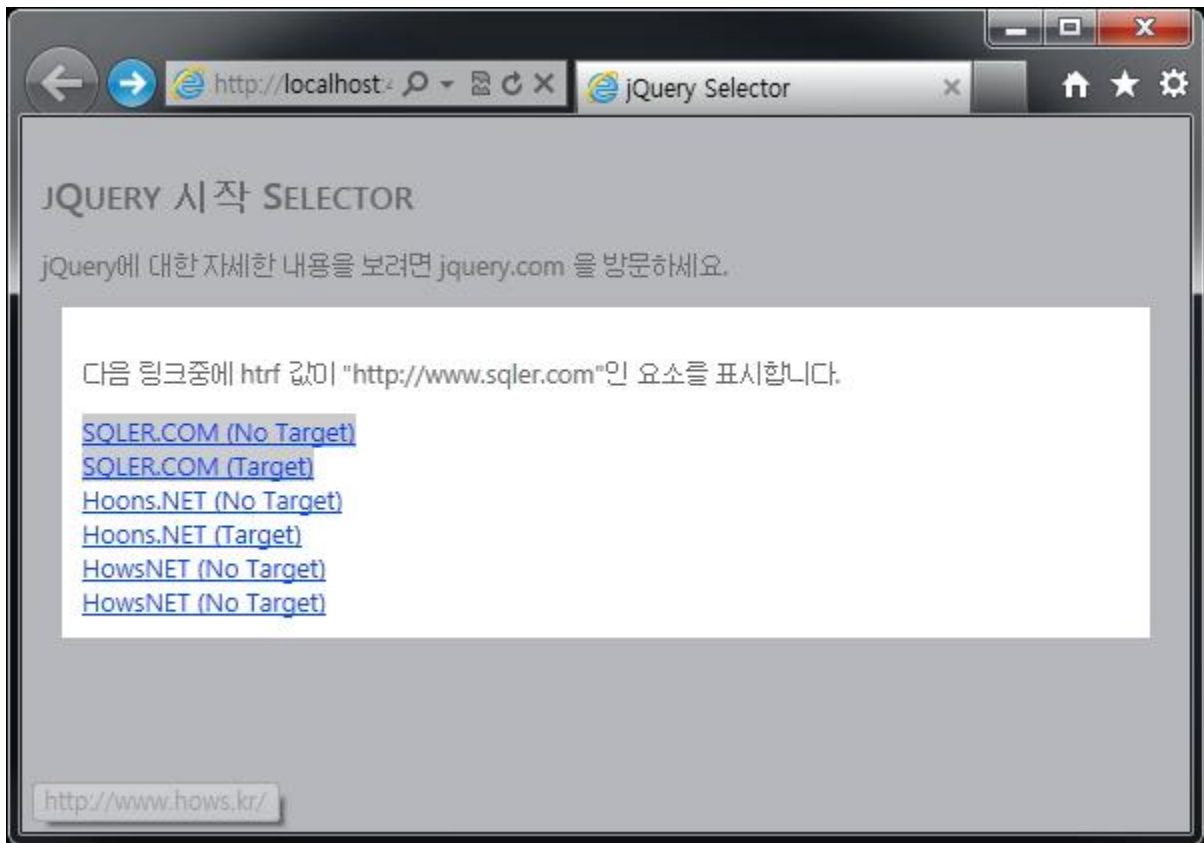


2. Attribute Equals Selector : \$(Selector[attr="value"])

셀렉터의 요소중 attr 과 value 의 값이 동일한 요소를 찾아 반환한다.

다음 예제는 HTML 문서에 존재하는 링크중에 <http://www.sqler.com> 인 요소를 찾아 표시한다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div { background : #FFF; padding:10px; margin:10px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("div > a[href='http://www.sqler.com']").css("background", "#CCC");
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <p>
      다음 링크중에 href 값이 "http://www.sqler.com"인 요소를 표시한다.
    </p>
    <a href="http://www.sqler.com">SQLER.COM (No Target)</a><br />
    <a href="http://www.sqler.com" target="_blank">SQLER.COM (Target)</a><br />
    <a href="http://www.hoons.kr">Hoons.NET (No Target)</a><br />
    <a href="http://www.hoons.kr" target="_blank">Hoons.NET (Target)</a><br />
    <a href="http://www.hows.kr">HowsNET (No Target)</a><br />
    <a href="http://www.hows.kr" target="_blank">HowsNET (No Target)</a><br />
  </div>
</body>
</html>
```

3. Attribute Not Equals Selector : \$(Selector[attr!="value"])

바로 앞에서 설명한 셀렉터와는 반대로 이번 셀렉터는 요소중 attr 의 값이 value 가 아닌 요소를 찾아 반환한다. 앞의 예제를 활용하여 간단히 알아 보도록 하겠다.

앞에서 설명한 예제 코드에서 다음 부분만 변경을 해 보겠다.

```
<script type="text/javascript">
    $(document).ready(function () {
        $("div > a[href='http://www.sqler.com']").css("background", "#DDD");
    });
</script>
[변경전]
<script type="text/javascript">
    $(document).ready(function () {
        $("div > a[href!='http://www.sqler.com']").css("background", "#CCC");
    });
</script>
[변경후]
```



4. Attribute Starts With Selector : \$(Selector[attr^="value"])

지정된 속성의 값으로 시작되는 요소와 일치하는 요소를 찾아 반환한다.

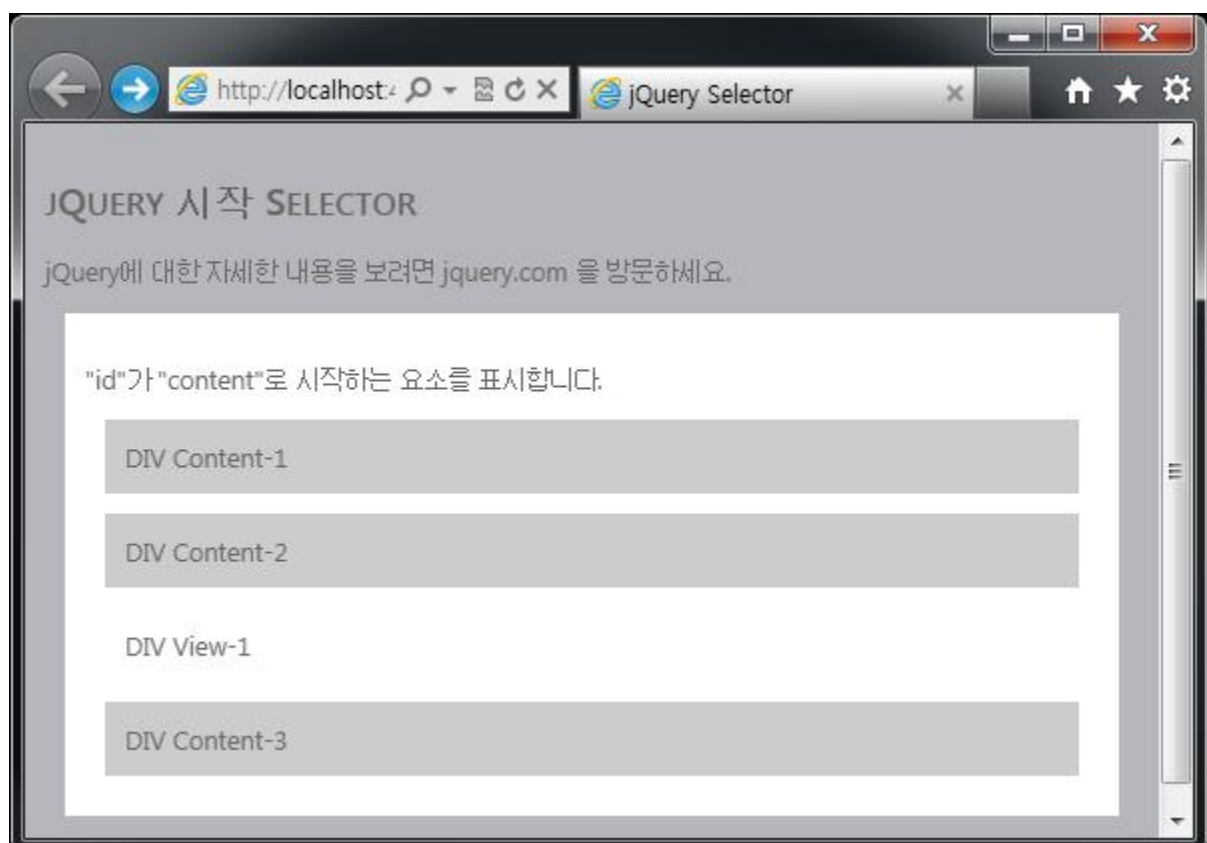
일정한 패턴으로 정의된 속성이 있을 경우 해당 셀렉터를 사용하면 매우 유용하다. 다음 예제를 통해 간단히 알아 보도록 하겠다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div { background : #FFF; padding:10px; margin:10px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("div[id^='content-']").css("background", "#CCC");
    });
  </script>
</head>
```

```

<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <p>
      "id"가 "content"로 시작하는 요소를 표시한다.
    </p>
    <div id="content-1">DIV Content-1</div>
    <div id="content-2">DIV Content-2</div>
    <div id="view-1">DIV View-1</div>
    <div id="content-3">DIV Content-3</div>
  </div>
</body>
</html>

```

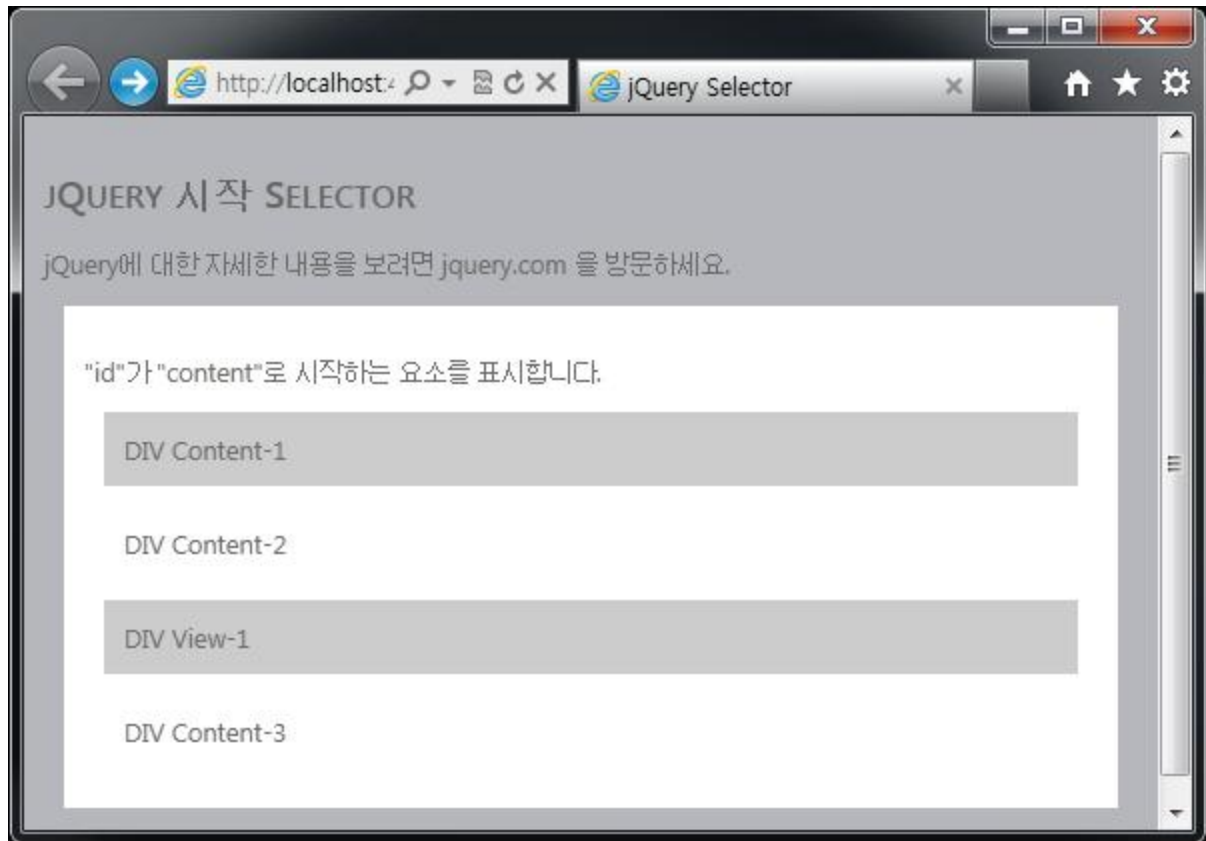


5. Attribute Ends With Selector : \$(Selector[attr\$="value"])

지정된 속성의 값으로 끝나는 요소와 일치하는 요소를 찾아 반환한다.

앞에서 설명한 예제 코드에서 다음 부분만 변경을 해 보겠다.

```
$("#div[id$='1']").css("background", "#CCC");
```



6. Attribute Contains Selector : \$(Selector[attr*="value"])

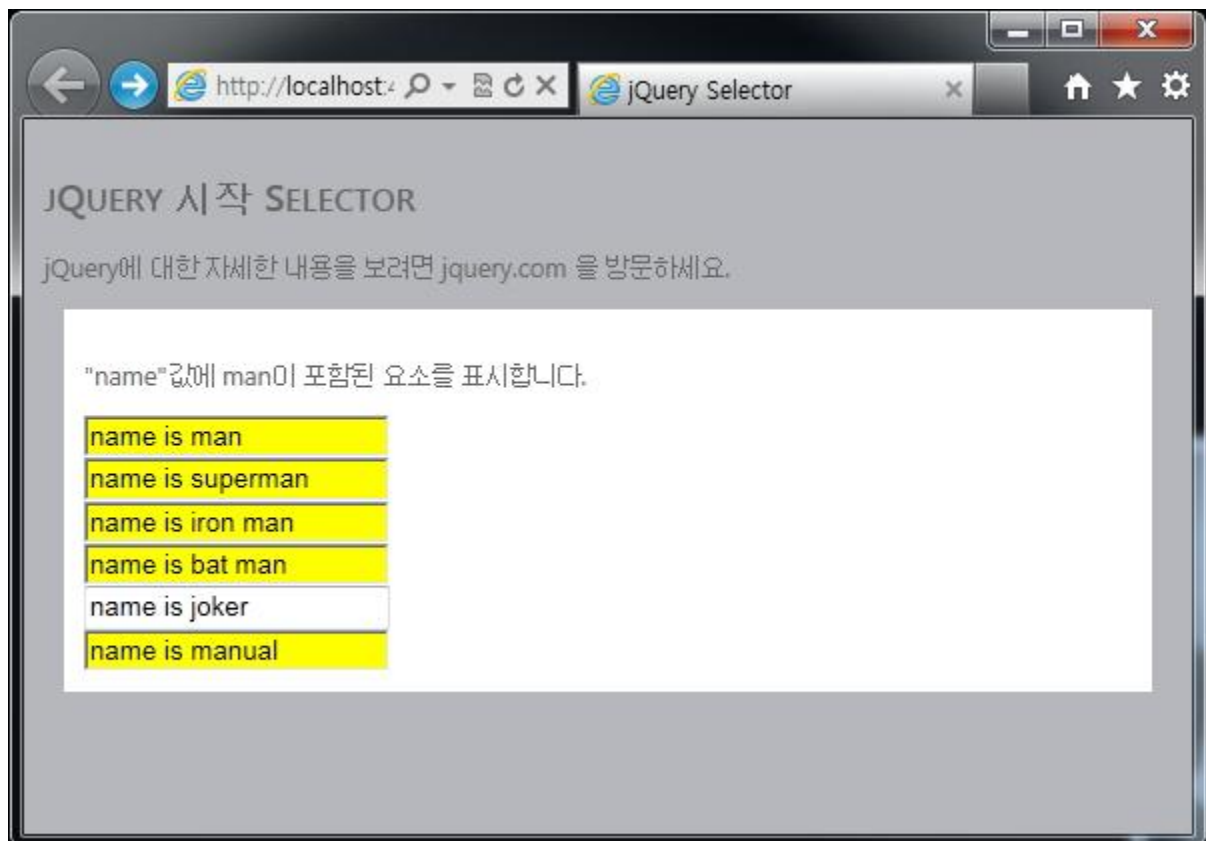
지정된 속성의 값이 포함된 요소와 일치하는 요소를 찾아 반환한다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div { background : #FFF; padding:10px; margin:10px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("input[name*='man']").css("background", "yellow");
    });
  </script>
</head>
<body>
  <div id="div1">
    <input type="text" value="man" />
  </div>
  <div id="div2">
    <input type="text" value="woman" />
  </div>
  <div id="div3">
    <input type="text" value="man" />
  </div>
  <div id="div4">
    <input type="text" value="woman" />
  </div>
</body>
</html>
```

```

});
</script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <p>"name"값에 man 이 포함된 요소를 표시한다.</p>
    <input name="man" value="name is man" /><br />
    <input name="superman" value="name is superman" /><br />
    <input name="iron man" value="name is iron man" /><br />
    <input name="bat man" value="name is bat man" /><br />
    <input name="joker" value="name is joker"/><br />
    <input name="manual" value="name is manual"/><br />
  </div>
</body>
</html>

```



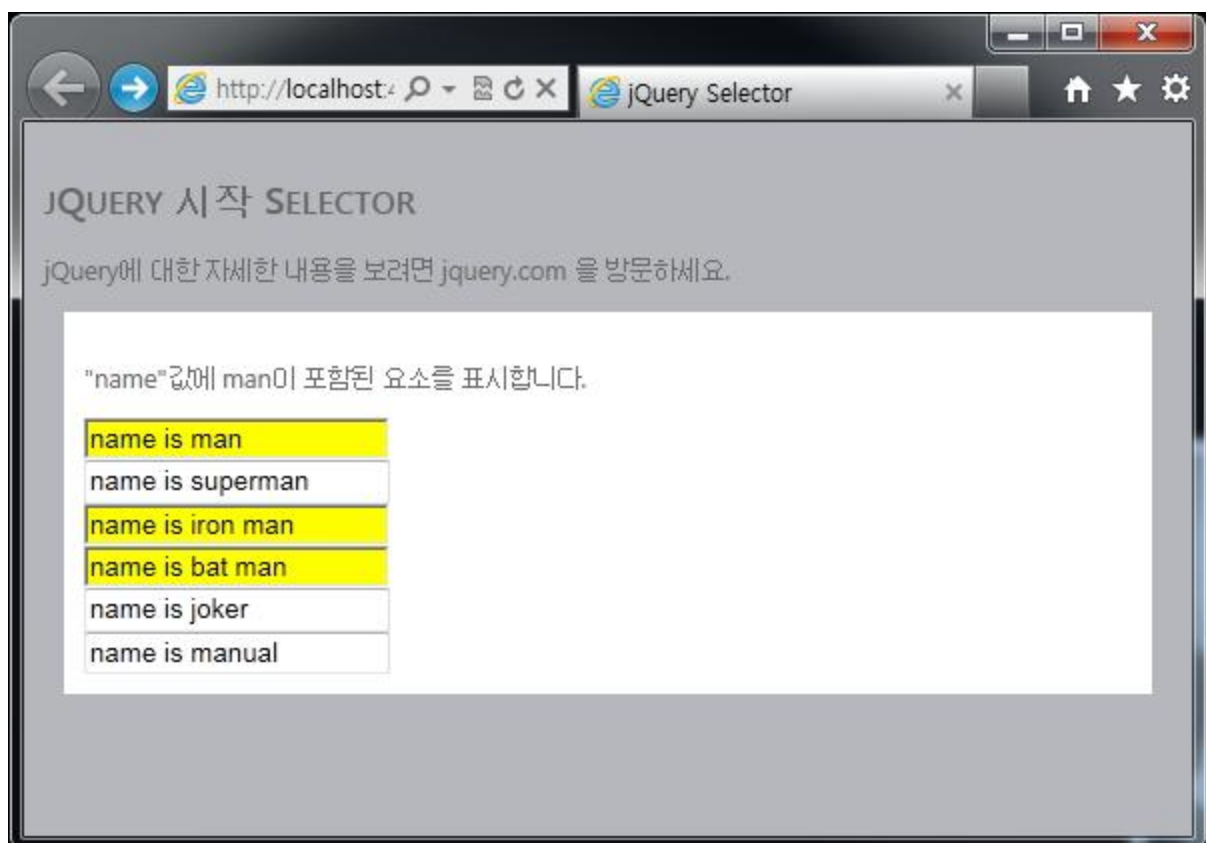
Name 값의 문자열에 "man"과 일치하는 항목이 있으면 무조건 해당 요소를 선택하며, 마지막 항목 "manual"도(man 이 포함된 문자열도 일치하는 항목으로 선택) 함께 선택된 모습을 확인 할 수 있다.

7. Attribute Contains Word Selector : \$(Selector[attr~="value"])

지정된 속성의 값이 공백과 함께 일치하는 요소를 찾아 반환한다.

\$(input[name*="man"])의 경우 "man, superman, iron man, manual"... 에 대한 모든 항목에 대해 일치하지만 \$(input[name~="man"])의 경우 "man"과 정확히 일치하는 요소만 반환한다. 앞의 셀렉터는 문자에 대해서 일치하는 항목을 찾았다면, 두번째의 경우는 단어의 단위로 일치하는 요소를 찾아 반환을 한다.

앞에서 설명한 예제 코드에서 \$(input[name*="man"]) 부분을 \$(input[name~="man"])로 변경 후의 결과를 보면 다음과 같다.



앞의 결과와는 전혀 다르게 요소가 선택된 결과를 보이며, "man"이란 단어 단위로 정확하게 일치하는 요소에 대해서만 선택된 모습을 확인 할 수 있다. 성값을 이용한 요소의 선택은 기본 셀렉터를 이용하는 것 보다는 느리나, 보다 정확하게 원하는 요소에 접근 할 수 있다는 장점이 있다.

속성값을 이용하여 요소를 선택하기 앞서 클래스나 DOM 을 이용한 요소의 선택 방법이 있다면 해당 방법을 사용 하길 권해 드린다. 다음 강좌에서는 HTML DOM 의 구조를 셀렉터를 통해 접근하는 방법에 대해 알아 보기로 하겠다.

[jQuery 강좌] 5. jQuery Selector - DOM 계층(Hierarchy)을 이용한 요소 접근 (1)

웹 프런티어와 함께하는 jQuery 기초강좌

5th - DOM 계층(Hierarchy)을 이용한 요소 접근 - 첫 번째 이야기 DOM 의 이해

이번시간에는 jQuery 에서 제공하는 HTML DOM 의 계층을 이용한 요소의 접근에 대해 알아보도록 하겠다.

우선 본론으로 들어가기 전에 HTML DOM 에 대한 이야기를 먼저 하려고 한다.

HTML DOM 에 대해 정확하게 아시는 것도 중요 하지만, 기초를 다지기 위한 강좌이니 다음 강좌에서 꼭 알아야 하는 내용을 간단하게 정리해 보았다.

<DOM 이란?>

HTML 트리 구조를 통해 클라이언트 영역에서 재조합 기능을 제공하여 사용자와 상호 작용을 하는 구조적인 모델로 HTML 트리 구조 내에서 개별 객체에 접근이 제공되는 방법을 말한다.

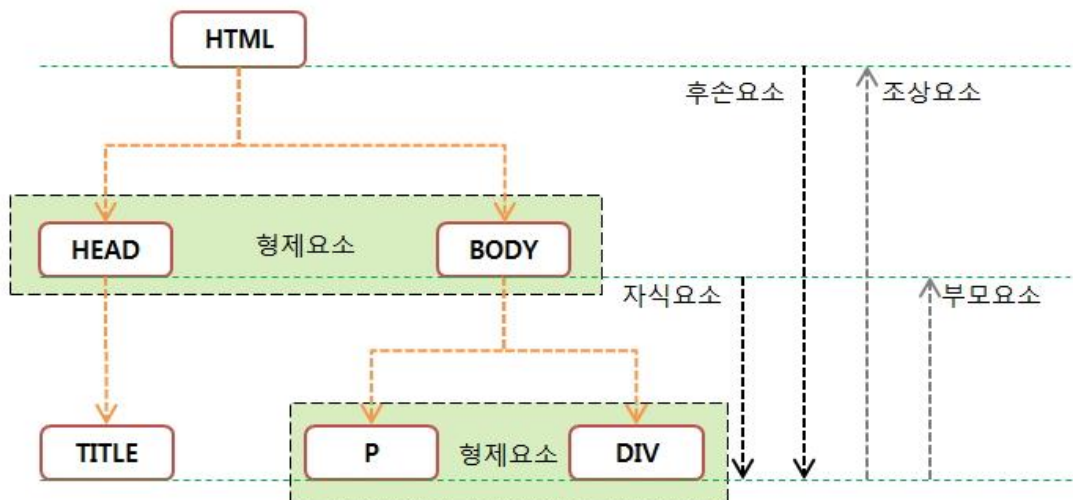
HTML 은 트리 구조로 이루어져 있어 자신의 부모, 조상, 자식, 후손에 대해 쉽게 접근이 가능하다.

<HTML DOM 의 구조>

```
<html>
  <head>
    <title>DOM Tree</title>
  </head>
  <body>
    <div>Hello jQuery</div>
    <p>안녕할 수 있다!<em>승연아빠</em>이다.</p>
  </body>
</html>
```


위와 같은 코드를 트리 형태로 표현 하자면 다음과 같다.

그림에서 중요하게 보실 부분은 "자신의 부모, 조상, 자식, 후손, 형제"에 대한 부분이다.



HTML DOM 은 그림 1 과 같은 관계를 형성하고 있다.

HEAD, BODY 는 HTML 의 자식 요소이면서 서로 HTML 부모에 묶여있는 형제 요소이며, HEAD 는 TITLE 의 부모요소가 BODY 는 P, DIV 의 부모요소가 된다. 또한 BODY 를 부모로 두고 있는 P, DIV 는 형제요소이면서 HTML 의 후손 요소가 되며, 반대로 HTML 은 P, DIV, TITLE 에 대해 조상 요소가 된다.

각각 요소에 대한 상하관계와 평행관계를 정확히 이해를 하셔야 다음에 진행될 강좌를 이해 할 수 있다.

마치 가족의 족보를 옮겨 놓았다고 생각하면 이해가 쉽다.

DOM 에 대한 설명은 여기서 간단히 마치며, 다음 강좌에서는 jQuery 에서 지원하는 DOM 계층에 접근하는 방법에 대해 자세히 알아 보도록 하겠다.

쉬어가는 페이지.....

<HTML DOM Tutorial>

HTML DOM 에 대해 자세히 알고 싶으시다면 이곳을 방문할 수 있다.

[W3School](http://www.w3schools.com/html/html_dom.asp)

« W3Schools Home

Next Chapter »

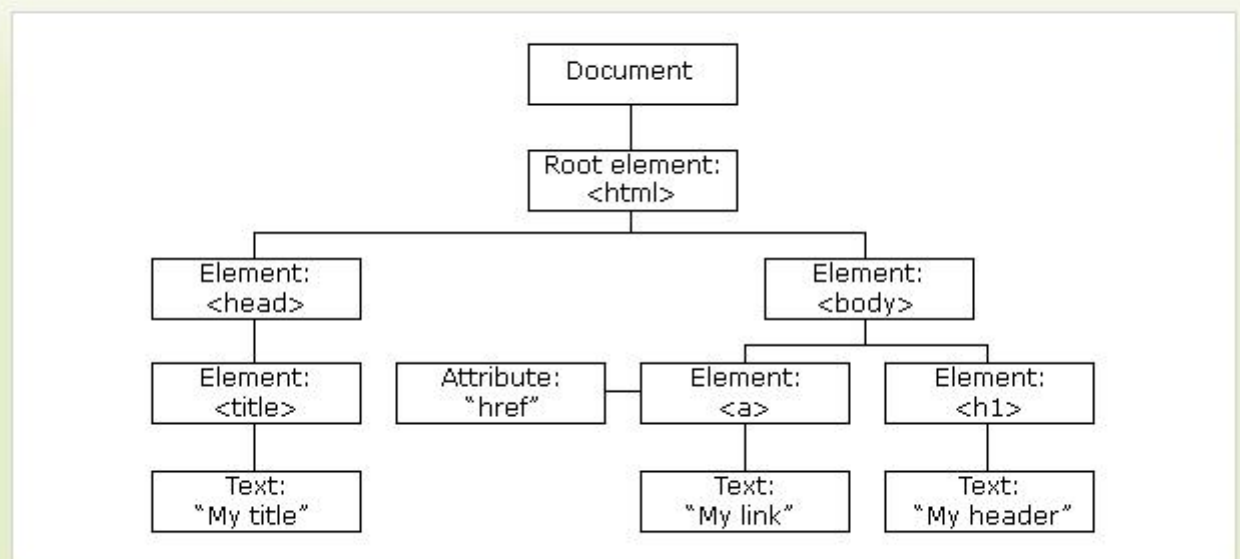


The HTML DOM defines a standard way for accessing and manipulating HTML documents.

The DOM presents an HTML document as a tree-structure.

[Start learning the HTML DOM now!](#)

HTML DOM Tree Example



[jQuery 강좌] 6. jQuery Selector - DOM 계층(Hierarchy)을 이용한 요소 접근 (2)

웹 프런티어와 함께하는 jQuery 기초강좌

6th - DOM 계층(Hierarchy)을 이용한 요소 접근(2)

이전 시간에 알아본 HTML DOM 을 jQuery 에서 지원하는 DOM 접근 방법에 대해 알아 보도록 하겠다.

<jQuery 계층 접근 선택터의 종류>

jQuery 에서는 다음과 같이 4 개의 계층 접근용 선택터를 지원하고 있으며, 각각의 사용법에 대해서는 간단한 예제와 함께 알아 보도록 하겠다.

형식(선택터)	선택터 표현식
Child Selector	\$("parent > child")
Descendant Selector	\$("ancestor descendant")
Next Adjacent Selector	\$("prev + next")
Next Siblings Selector	\$("prev ~ siblings")
표 1. [jQuery 계층(Hierarchy) 선택터의 종류]	

1. Child Selector : \$("Parent > Child")

부모(Parent) 요소 바로 아래 자식(Child)인 요소를 반환 한다.

최근에 출시된 대부분의 브라우저를 지원하고 있으나, IE6 를 포함한 이전의 브라우저에서는 지원을 하고 있지 않으니 사용에 각별한 주의가 필요하다. IE6 를 포함한 이전의 브라우저의 지원을 위해서는 3 번째 강좌에서 설명한 기본 선택터를 사용 하시기 바란다. 기본 선택터의 경우 모든 브라우저를 지원하고 있으므로 브라우저의 버전이나, 종류에 신경 쓸 필요가 없다.

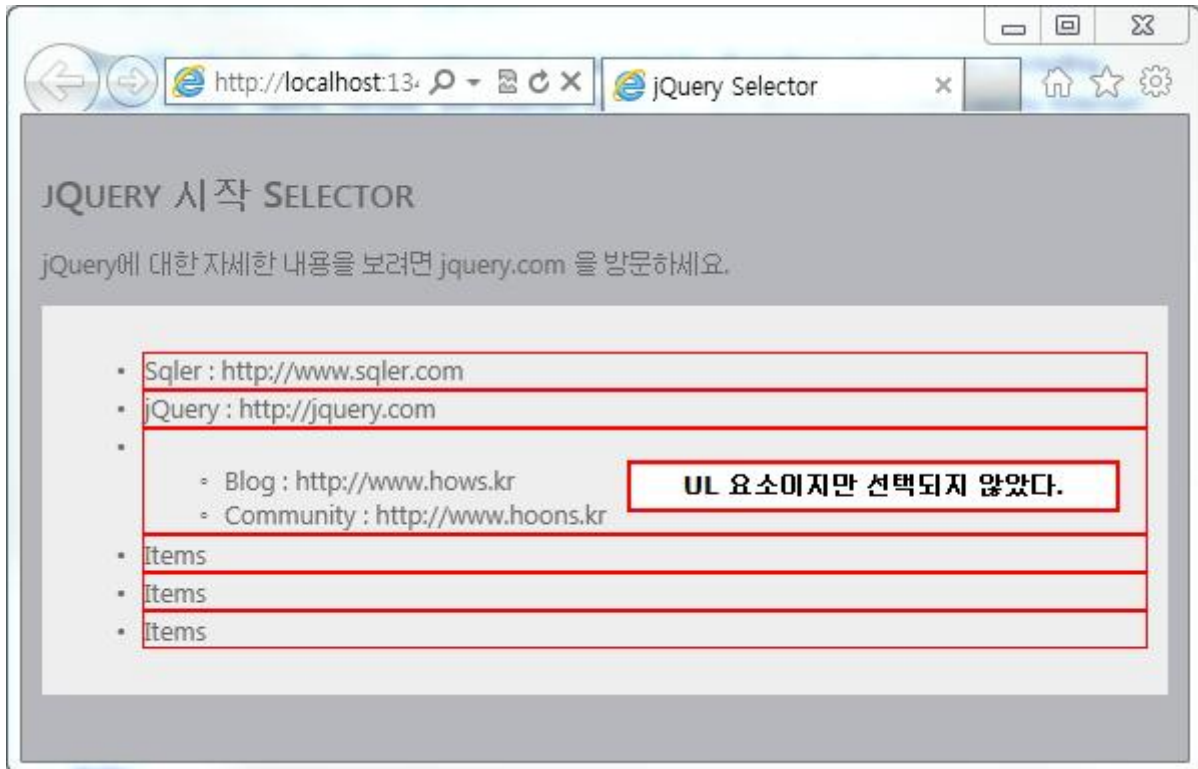
```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div { background-color:#EEEEEE; padding:10px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
```

```

<script type="text/javascript">
    $(document).ready(function () {
        $("ul.siteUrl > li").css("border", "1px solid #ff0000");
    });
</script>
</head>
<body style="padding:10px;">
    <h2>jQuery 시작 Selector</h2>
    <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
    <div>
        <ul class="siteUrl">
            <li>Sqler : http://www.sqler.com</li>
            <li>jQuery : http://jquery.com</li>
            <li>
                <ul>
                    <li>Blog : http://www.hows.kr</li>
                    <li>Community : http://www.hoons.kr</li>
                </ul>
            </li>
            <li>Items</li>
            <li>Items</li>
            <li>Items</li>
        </ul>
    </div>
</body>
</html>

```

[부모요소(ul)에서 자식요소(li)과 일치하는 항목 선택]



[부모요소(ul)에서 자식요소(li)과 일치하는 항목 선택된 모습]

`$("ul.siteUrl > li").css("border", "1px solid #ff0000");` 코드설명

1. HTML 문서에서 ul 요소 중 "siteUrl"이란 클래스 명을 가진 요소를 찾는다.
2. 1 번 항목에서 찾은 요소의 자식 중에 "li" 항목과 일치하는 요소를 찾는다.
3. 2 번 항목에서 찾은 요소를 jQuery의 "css" 메소드를 이용해 테두리 값을 빨강색으로 변경한다.

2. Descendant Selector : \$("ancestor descendant")

조상(Ancestor)과 일치하는 요소에 포함된 모든 후손(Descendant)중에 후손의 요소와 일치하는 모든 항목을 반환한다.

부모(Parent)와 자식(Child)간의 관계가 조상(Ancestor)과 후손(Descendant)로 변경이 되었을 뿐 Child 셀렉터와 동일한 기능을 제공 한다.

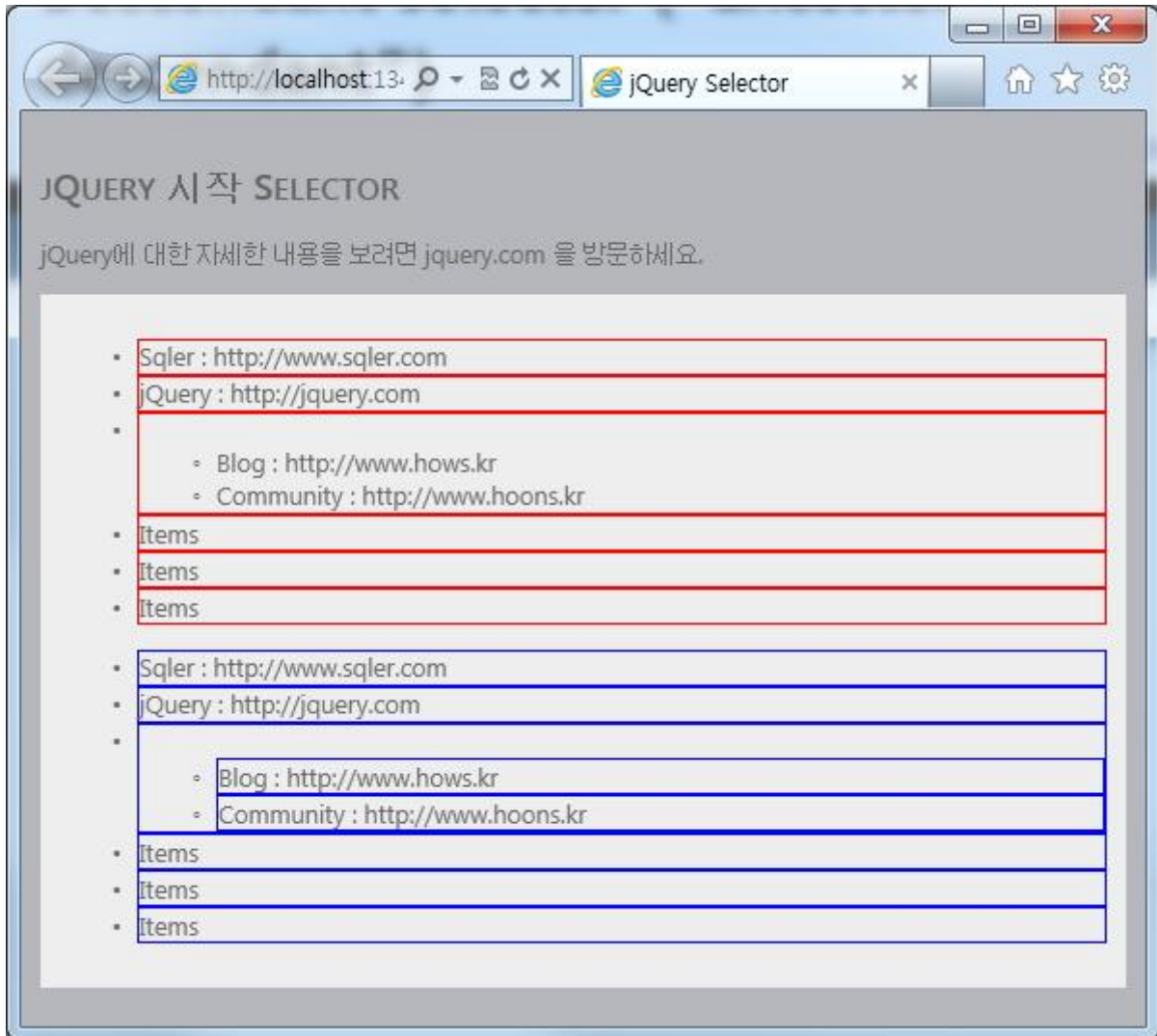
```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>div { background-color:#EEEEEE; padding:10px; }</style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("#siteUrl1 > li").css("border", "1px solid #ff0000");
    });
  </script>
</head>
<body>
  <div id="siteUrl1">
    <ul>
      <li>Sqler : http://www.sqler.com</li>
      <li>jQuery : http://jquery.com</li>
      <li>
        <ul>
          <li>Blog : http://www.hows.kr</li>
          <li>Community : http://www.hoons.kr</li>
        </ul>
      </li>
      <li>Items</li>
      <li>Items</li>
      <li>Items</li>
    </ul>
  </div>
</body>
</html>
```

```

        $("#siteUrl2 li").css("border", "1px solid blue");
    });
</script>
</head>
<body style="padding:10px;">
    <h2>jQuery 시작 Selector</h2>
    <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
    <div>
        <ul id="siteUrl1">
            <li>Sqler : http://www.sqler.com</li>
            <li>jQuery : http://jquery.com</li>
            <li>
                <ul>
                    <li>Blog : http://www.hows.kr</li>
                    <li>Community : http://www.hoons.kr</li>
                </ul>
            </li>
            <li>Items</li>
            <li>Items</li>
            <li>Items</li>
        </ul>
        <ul id="siteUrl2">
            <li>Sqler : http://www.sqler.com</li>
            <li>jQuery : http://jquery.com</li>
            <li>
                <ul>
                    <li>Blog : http://www.hows.kr</li>
                    <li>Community : http://www.hoons.kr</li>
                </ul>
            </li>
            <li>Items</li>
            <li>Items</li>
            <li>Items</li>
        </ul>
    </div>
</body>
</html>

```

[자식(Child) 셀렉터와 후손(Descendant) 셀렉터]



[자식(Child) 셀렉터와 후손(Descendant) 셀렉터에 의해 선택 된 모습]

빨강색으로 표시된 부분은 자식(Child) 셀렉터를 이용하여 개체를 선택한 모습이며, 파랑 색으로 표시된 부분은 후손(Descendant) 셀렉터에 의해 개체를 선택한 모습이다.

자식(Child) 셀렉터를 이용했을 경우 ul -> li -> ul -> li 에 해당되는 요소는 선택이 되지 않고 바로 아래에 있는 자식(Child) 요소만 선택이 된 반면에 후손(Descendant) 셀렉터를 이용했을 경우 li 에 해당하는 모든 요소를 선택 한 것을 확인 할 수 있다.

간단한 문법의 차이로 인해서 반환하는 요소의 결과의 차이가 많으니 두 가지 셀렉터의 차이점을 확실히 이해를 하시는 것은 매우 중요하다.

3. Next Adjacent Selector : \$("prev + next")

이전에는 요소와 요소 사이의 상, 하 관계를 통해 일치하는 항목을 찾았다면 이번 셀렉터와 다음에 설명드릴 셀렉터는 평행관계를 통해 일치하는 요소를 반환한다.

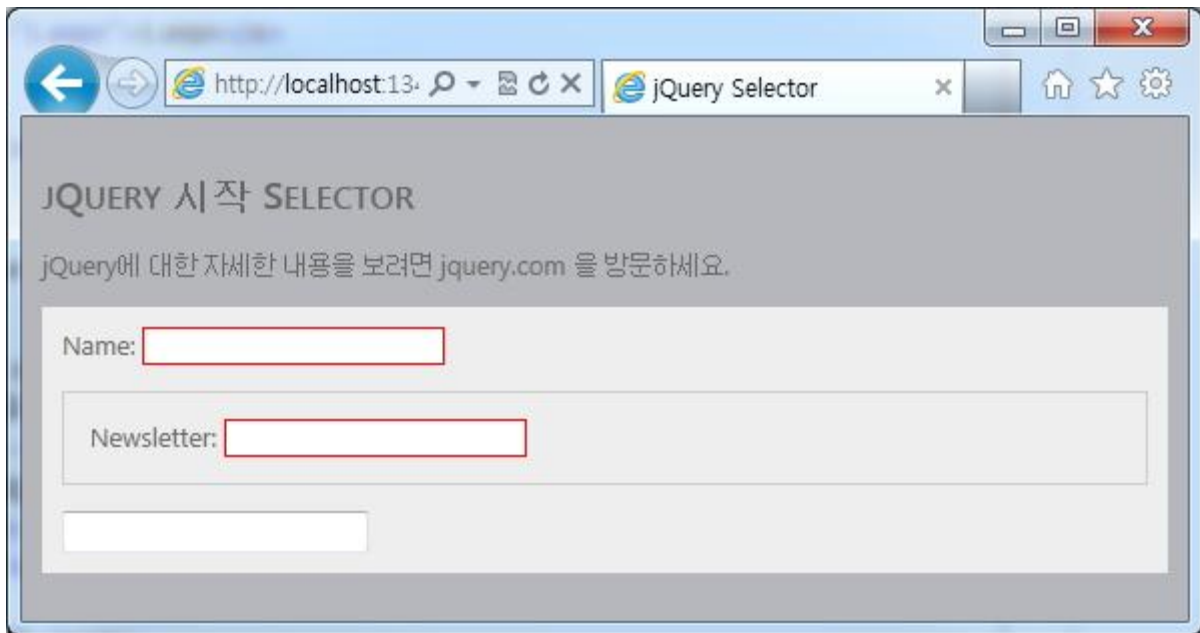
Prev 요소 바로 다음에 나오는 형제(Adjacent) 수준의 next 요소와 일치하는 항목을 반환한다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div { background-color:#EEEEEE; padding:10px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("label + input").css("border", "1px solid #ff0000");
    });
  </script>
</head>
<body>
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <form>
      <label>Name:</label>
      <input name="name" />

      <fieldset>
        <label>Newsletter:</label>
        <input name="newsletter" />
      </fieldset>

    </form>
    <input name="none" />
  </div>
</body>
</html>
```

[label 다음의 input 의 요소와 형제 수준이 동일한 요소를 찾는다.]



[지정한 요소와 형제 수준이 동일한 요소의 모습]

```
$("#label + input").css("border", "1px solid #ff0000");
```

<label> 요소 바로 다음(형제요소)에 있는 <input> 요소를 찾는다. 동일한 패턴으로 된 모든 요소를 찾아 반환한다.

\$("#label > input") 부분과 혼동하시는 분들이 많아 두 선택어의 차이를 다시 설명하자면 \$("#label > input")의 경우 <label>을 부모로 가지고 있는 <input> 요소를 찾는 것이고, \$("#label + input")의 경우는 <label>과 평등한 관계에 있는 <input> 요소를 찾는 것이다.

4. Next Siblings Selector : \$("#prev ~ siblings")

Prev 요소 이후에 형제 요소 중 siblings 와 동일한 요소를 반환한다.

\$("##prev ~ div")의 경우 id의 값이 prev 인 요소를 찾고 해당 요소를 제외한 다음 형제 요소 중에 div 와 동일한 요소를 찾아 반환을 한다.

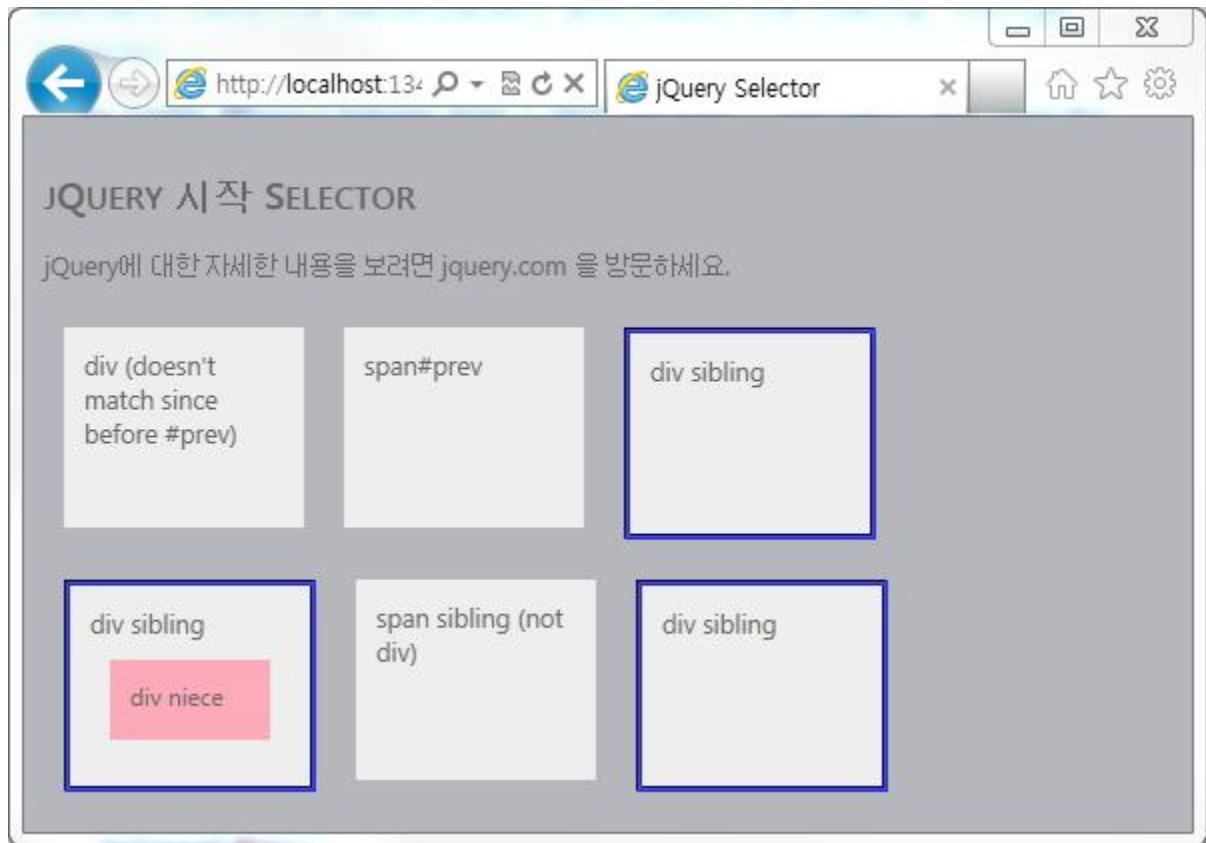
```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,span
    {
      width:100px;
      height:80px;
      float:left;
```

```

padding:10px;
margin:10px;
background-color:#EEEEEE;
}
div#small
{
width:60px;
height:20px;
font-size:12px;
background:#fab;
}
</style>
<script src="http://code.jquery.com/jquery-latest.js"></script>
<script type="text/javascript">
$(document).ready(function () {
    $("#prev ~ div").css("border", "3px groove blue");
});
</script>
</head>
<body style="padding:10px;">
<h2>jQuery 시작 Selector</h2>
<p id="content">jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
<div>div (doesn't match since before #prev)</div>
<span id="prev">span#prev</span>
<div>div sibling</div>
<div>div sibling
    <div id="small">div niece</div>
</div>
<span>span sibling (not div)</span>
<div>div sibling</div>
</body>
</html>

```

[자신을 제외한 이후 형제 요소 선택]



[자신을 제외한 동일한 형제 요소에 속하는 div 요소를 선택한 모습]

이번 시간에는 DOM 계층을 이용한 접근 방법에 대해 알아 보았다. 동적인 화면의 구성과 복잡한 UI 에서 원하는 요소(개체)에 접근하기 위해서는 DOM 의 구조에 대해 이해하고 접근방법에 익숙해 있어야 한다.

[jQuery 강좌] 7. jQuery Filter - 기본필터(Basic Filter)

웹 프런티어와 함께하는 jQuery 기초강좌

7th - jQuery Filter 의 기본필터의 사용

이번 시간에는 jQuery 에서 제공하는 필터에 대해서 알아 보겠다. 부제를 붙이자면 "셀렉터와 필터의 만남"이 될까 같다.

< 셀렉터에 날개를 달자 ? Filter 를 이용한 개체 접근 >

이번 강좌에서는 jQuery 에서 지원하고 있는 여러가지 필터중에 기본이 되는 부분과 jQuery 의 DOM 필터 메서드,(filter())를 사용하여 사용자가 원하는 요소를 선택하는 방법에 대해 알아보도록 하겠다.

형식(표현식)	설명
:animated	애니메이션이 동작중인 모든 요소와 일치하는 요소를 반환한다.
:eq(index)	Index 에 해당하는 요소를 반환한다.(단일요소)
:even	짝수의 요소를 반환한다. (0 부터 시작)
:odd	홀수의 요소를 반환한다. (0 부터 시작)
:first	첫번째 요소를 반환한다.
:last	마지막 요소를 반환한다.
:gt(index)	Index 보다 높은 Index 에 해당되는 요소를 모두 반환한다.
:lt(index)	Index 보다 낮은 Index 에 해당되는 요소를 모두 반환한다.
:header	모든 헤더 요소들을 반환한다.(h1,h2,h3....)
:not(selector)	Selector 와 일치되는 요소를 제외한 나머지 요소를 반환한다.
:focus	현재 포커스가 위치한 요소를 반환한다. (1.6 이상에서 지원)
표 1. [jQuery 필터의 종류]	

jQuery 의 필터는 말 그대로 거르다, 여과하다의 뜻으로 원하는 요소를 다양한 방식으로 걸러내는 역할을 한다.

일반적으로 셀렉터와 함께 사용하는 경우가 대부분으로 셀렉터와 함께 써야 하는하는 것 알고 있는 분이 많이 있다.

하지만 필터는 단독으로도 사용이 가능하며, 필터와 필터를 연결해서도 사용이 가능하다.

테이블의 컬럼인 td 항목을 예로 들어 설명을 하자면 다음과 같다.

<code>\$("td:eq(0)")</code>	td 요소중에 첫번째 항목만을 선택한다. (eq 의 index 는 0 부터 시작이다.) eq 의 index 는 0 부터 시작을 하며 0 이면 요소중 첫번째, 1 이면 두번째 요소가 된다.
<code>\$("td:even")</code> <code>\$("td:odd")</code>	td 요소중에 짝/홀수번째 요소를 선택한다. 짝/홀수의 구분은 eq 의 index 와 동일하게 구분이 된다.
<code>\$("td:first")</code> <code>\$("td:last")</code>	td 요소중에 첫번째 요소, 마지막 요소를 선택한다.
<code>\$("td:gt(2)")</code> <code>\$("td:lt(2)")</code>	td 요소중에 2 번째 요소 이후의 모든 요소를 선택하거나 2 번째 요소 이전의 모든 요소를 선택한다.

각각의 설명만으로도 각 필터가 어떠한 역할을 하고 있는지 쉽게 이해를 하실 수 있을것이다.
간단한 예제를 통해 위에 다시 한번 살펴보도록 하겠다.

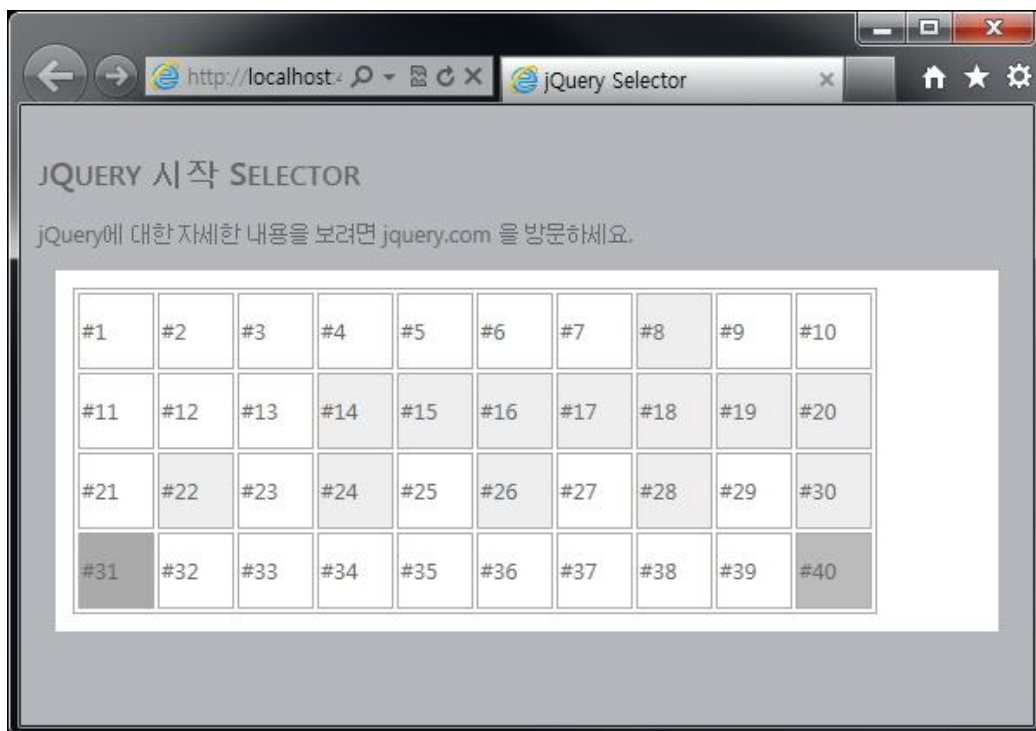
```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:40px; height:40px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("tr:eq(0) > td:eq(7)").css("background", "#EEE");
      $("tr:eq(1) > td:gt(2)").css("background", "#EEE");
      $("tr:eq(2) > td:odd").css("background", "#EEE");
      $("tr:eq(3) > td:first").css("background", "#AAA");
      $("tr:eq(3) > td:last").css("background", "#BBB");
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <table>
      <tr>
```

```

        <td>#1</td><td>#2</td><td>#3</td><td>#4</td><td>#5</td>
        <td>#6</td><td>#7</td><td>#8</td><td>#9</td><td>#10</td>
    </tr>
    <tr>
        <td>#11</td><td>#12</td><td>#13</td><td>#14</td><td>#15</td>
        <td>#16</td><td>#17</td><td>#18</td><td>#19</td><td>#20</td>
    </tr>
    <tr>
        <td>#21</td><td>#22</td><td>#23</td><td>#24</td><td>#25</td>
        <td>#26</td><td>#27</td><td>#28</td><td>#29</td><td>#30</td>
    </tr>
    <tr>
        <td>#31</td><td>#32</td><td>#33</td><td>#34</td><td>#35</td>
        <td>#36</td><td>#37</td><td>#38</td><td>#39</td><td>#40</td>
    </tr>
</table>
</div>
</body>
</html>

```

[예제 1]



[예제 1 을 적용한 결과 화면]

예제에서는 \$("td:even")으로 사용했지만 \$("tr:even").css("background", "#DDD")와 같이 사용하면 각 tr 부분에 스타일을 작성하지 않고 간단하게 "zebra" 스타일의 목록을 만들 때 매우 유용하다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; width:100% }
    td { border:1px solid #AAA; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("tr:even").css("background", "#AAA");
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <table>
      <tr>
        <td>#1</td>
      </tr>
      <tr>
        <td>#2</td>
      </tr>
      <tr>
        <td>#3</td>
      </tr>
      <tr>
        <td>#4</td>
      </tr>
      <tr>
        <td>#5</td>
      </tr>
    </table>
  </div>
</body>
</html>
```

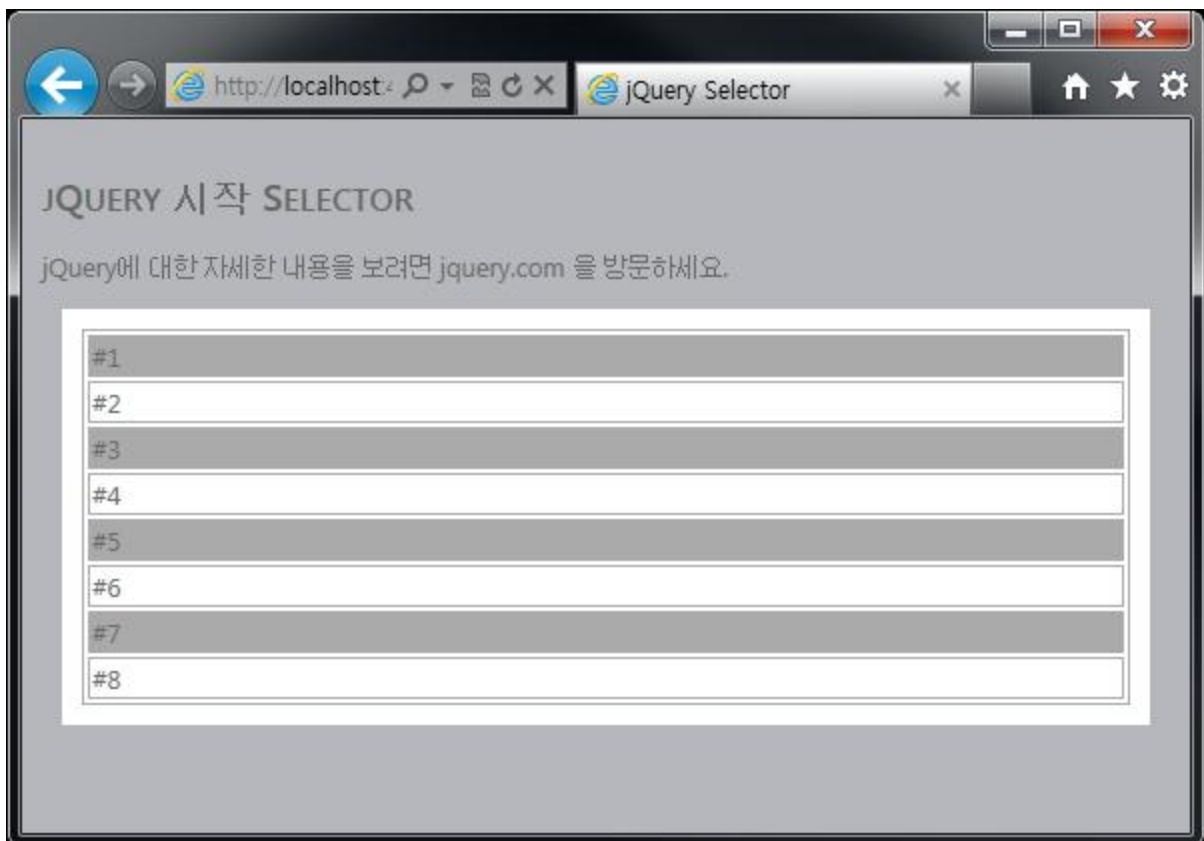


```

        </tr>
        <tr>
            <td>#6</td>
        </tr>
        <tr>
            <td>#7</td>
        </tr>
        <tr>
            <td>#8</td>
        </tr>
    </table>
</div>
</body>
</html>

```

[예제 2. :even 필터를 이용한 Zebra 스타일 목록 만들기]



[예제 2 를 이용한 Zebra 스타일 목록 결과]

예제중에서 ":even, :odd" 경우 "zebra" 줄무늬 스타일의 목록을 만들 때 자주 사용이 된다.

1. :not(selector)

`$("td:not('.noselect')")`

td 요소중에 클래스명이 "notSelect"인 항목을 제외한 요소의 집합을 선택한다.

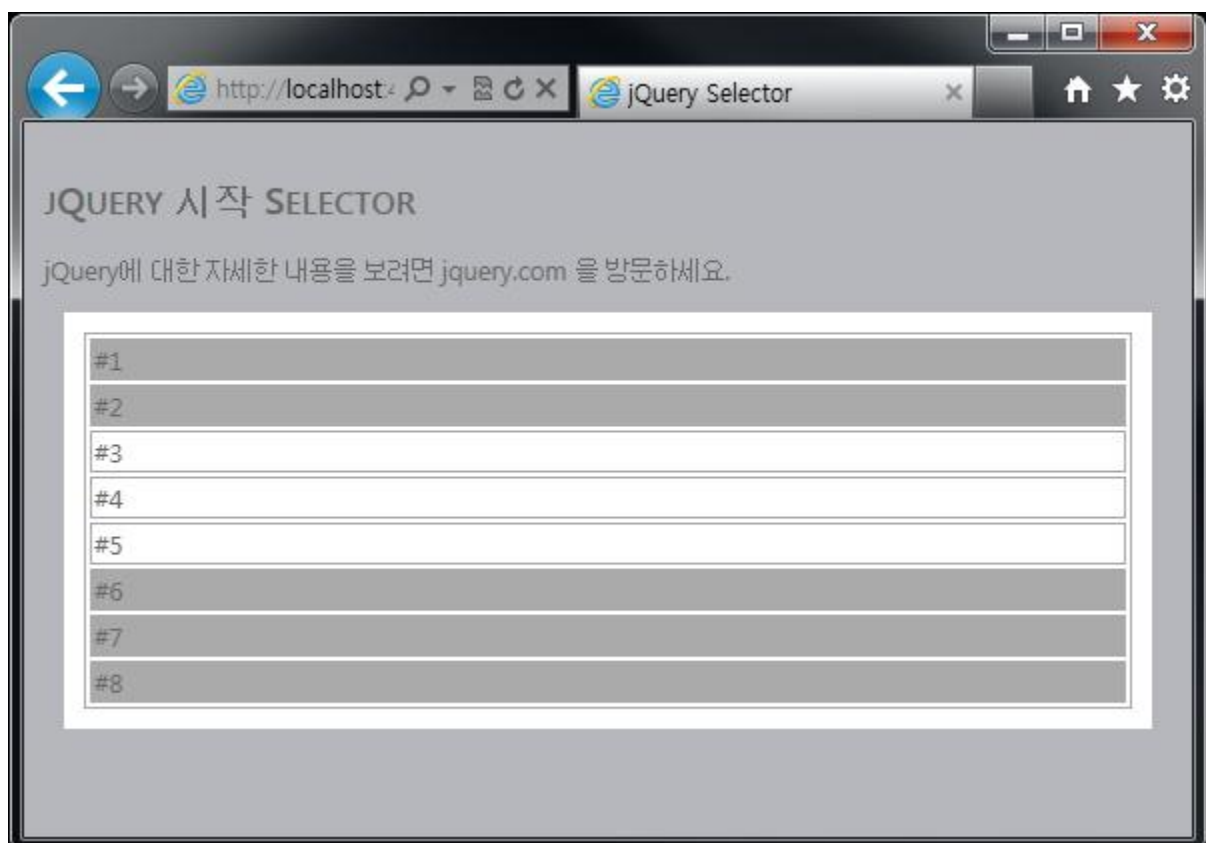
```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; width:100% }
    td { border:1px solid #AAA; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("td:not(.notSelect)").css("background", "#AAA");
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <table>
      <tr>
        <td>#1</td>
      </tr>
      <tr>
        <td>#2</td>
      </tr>
      <tr>
        <td class="notSelect">#3</td>
      </tr>
      <tr>
        <td class="notSelect">#4</td>
      </tr>
      <tr>
```

```

        <td class="notSelect">#5</td>
    </tr>
    <tr>
        <td>#6</td>
    </tr>
    <tr>
        <td>#7</td>
    </tr>
    <tr>
        <td>#8</td>
    </tr>
</table>
</div>
</body>
</html>

```

[예제 3. :not 필터의 사용]



[예제 3 :not 필터를 사용한 결과 화면]

2. :focus

jQuery 1.6 버전에서 새롭게 추가된 필터로 현재 포커스가 위치한 요소를 선택 하며, 유저의 포커스가 위치한 지점의 위치를 표시하거나, 입력하고 있는 폼 요소를 강조하고 싶을 때 사용할 수 있다.

이번 시간에는 jQuery 의 필터중에서 첫번째인 Basic Filter 에 대해 알아 보았으며, 다음 시간에는 jQuery 의 필터 두번째인 폼필터(FormFilter)에 대해 알아 보도록 하겠다.

[jQuery 강좌] 8. jQuery Filter - 폼 필터(Form Filter)

웹 프런티어와 함께하는 jQuery 기초강좌

8th - jQuery Filter 의 폼(Form) 필터의 사용

지난 시간에 설명드린 기본 필터의 사용법에 이어 폼(Form) 필터에 대해 알아 보기로 하겠다.

<폼 필터>

폼 필터는 형식(text, checkbox, password, radio, file)을 기반으로 하는 폼 요소를 선택할 때 사용하는 필터로 사용법은 필터와 동일하여, 자세한 설명보다는 아래의 표로 대신 한다.

폼 필터 종류	선택 폼
:button	<input type="button" />
:checkbox	<input type="checkbox" />
:checked	<input type="checkbox" checked="checked" />
:disabled	<input type="text" disabled="disabled" />
:enabled	<input type="text" enabled="enabled" />
:file	<input type="file" />
:focus	(1.6 이상에서 지원)
:image	<input type="image" />
:input	<input> 모든 input 요소
:password	<input type="password" />
:radio	<input type="radio" />
:reset	<input type="reset" />
:selected	<select><option selected="selected"></option></select>
:submit	<input type="submit" />
:text	<input type="text" />
:hidden	<input type="hidden" />
표 1. [jQuery 폼 필터의 종류]	

표에서 보듯이 폼 필터 사용은 매우 간단하여 자세한 설명을 생략하고 필터 중 상태를 체크하는 필터에 대해 알아 보도록 하겠다.

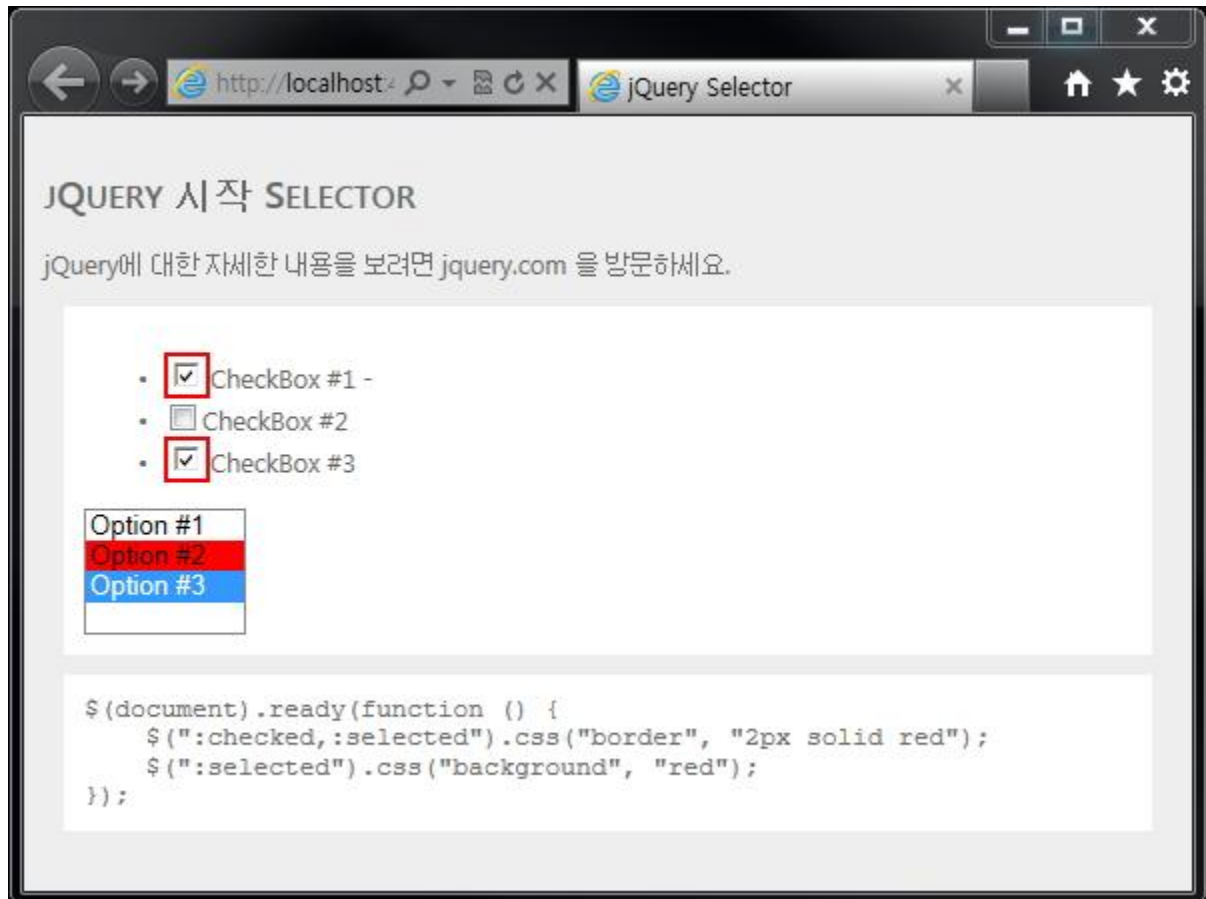
1. :checked , :selected

<input type="checkbox"/> 체크박스, 라디오버튼 <input type="radio" />, 셀렉트 <select>의 상태를 확인 하여 선택된 상태 요소들의 집합을 선택 반환한다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $(".:checked,:selected").css("border", "2px solid red");
      $(".:selected").css("background", "red");
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <ul>
      <li><input type="checkbox" checked="checked" />CheckBox #1 - </li>
      <li><input type="checkbox" />CheckBox #2</li>
      <li><input type="checkbox" checked="checked" />CheckBox #3</li>
    </ul>
    <select multiple="multiple">
      <option>Option #1</option>
      <option selected="selected">Option #2</option>
      <option>Option #3</option>
    </select>
  </div>
</body>
```

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

[예제 1. 폼필터의 사용예제]



[예제 1 Selected, Checked 를 폼 필터를 이용한 결과 모습]

2. :enabled, :disabled

사용이 가능하거나, 사용이 불가능한 요소를 선택하여 해당 요소의 집합을 반환한다.

`<input type="text" disabled="disabled" />`의 경우 `:disabled` 로, 일반적인 모습일 경우 `:enabled` 로 관련 요소의 집합을 선택 할 수 있다.

이번 시간에는 jQuery 의 필터중에서 두 번째인 Form Filter 에 대해 알아 보았으며, 다음 시간에는 jQuery 의 필터 세 번째인 자식필터(Child Filter)에 대해 알아 보도록 하겠다.

[jQuery 강좌] 9. jQuery Filter - 자식필터(Child Filter)

웹 프런티어와 함께하는 jQuery 기초강좌

9th - jQuery Filter 의 자식(Child) 필터의 사용

지난 시간에 설명드린 필터의 사용법에 이어 자식(Child) 필터에 대해 알아 보기로 하겠다.

자식(Child)에 대한 내용을 모르고 계신다면 이전에 설명드린 HTML DOM 에 대한 강좌를 한번 확인 하시기 바란다.

1. 자식필터<Child Filter>

아래 목록을 통해 나열된 내용을 보면 이번강의 초반에 나왔던 필터와 비슷하게 보이실 겁니다. 사용하는 구문도 많은 부분이 비슷하지만 결과는 정말 다릅니다. 우선 자식필터(Child Filter)에 대해 간단히 정리하고 앞서 말한 결과의 차이에 대해 알아 보기로 하겠다.

필터 종류 및 형식	필터 설명
:first-child	자식 요소 중 첫번째에 해당하는 요소를 모두 반환한다.
:last-child	자식 요소 중 마지막에 해당하는 요소를 모두 반환한다.
:nth-child(index/odd/even/equation)	자식 요소 중 지정된 값에 해당하는 요소를 모두 반환한다.
:only-child	자신이 부모 요소와 유일한 자식인 모든 요소를 반환한다.

표 1. jQuery 자식(Child)필터의 종류]

앞서 다룬 기본 필터와 유사한 모습을 하고 있어 많은 분들이 비슷한 동작을 하지 않을까 하고 생각 하시는 경우가 많으나, 기본 필터와는 많은 부분에서 차이가 존재한다.

기본필터의 :first, :last 의 경우 해당되는 단일 요소만 선택하지만 자식필터(Child Filter)의 경우 단일요소가 아닌 집합을 선택 하게 된다.

테이블의 "td" 요소를 기준으로 두개의 차이점을 비교해 보았다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
```



```

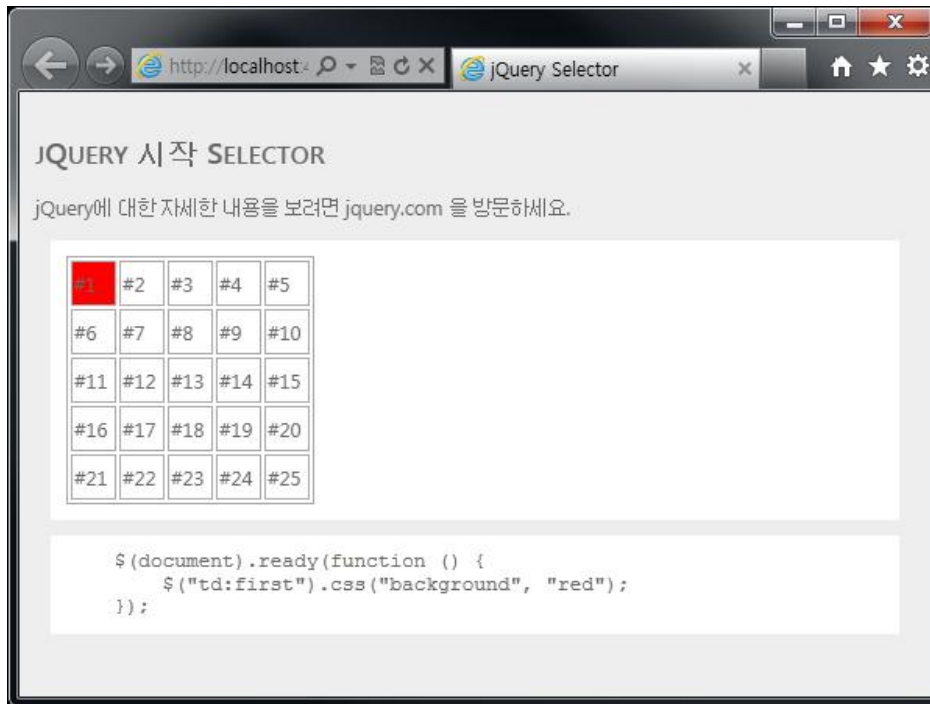
</style>
<script src="http://code.jquery.com/jquery-latest.js"></script>
<script type="text/javascript">
    $(document).ready(function () {
        //Script Here
    });
</script>
</head>
<body style="padding:10px;">
    <h2>jQuery 시작 Selector</h2>
    <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
    <div>
        <table>
            <tr>
                <td>#1</td><td>#2</td><td>#3</td><td>#4</td><td>#5</td>
            </tr>
            <tr>
                <td>#6</td><td>#7</td><td>#8</td><td>#9</td><td>#10</td>
            </tr>
            <tr>
                <td>#11</td><td>#12</td><td>#13</td><td>#14</td><td>#15</td>
            </tr>
            <tr>
                <td>#16</td><td>#17</td><td>#18</td><td>#19</td><td>#20</td>
            </tr>
            <tr>
                <td>#21</td><td>#22</td><td>#23</td><td>#24</td><td>#25</td>
            </tr>
        </table>
    </div>
</body>
</html>

```

```

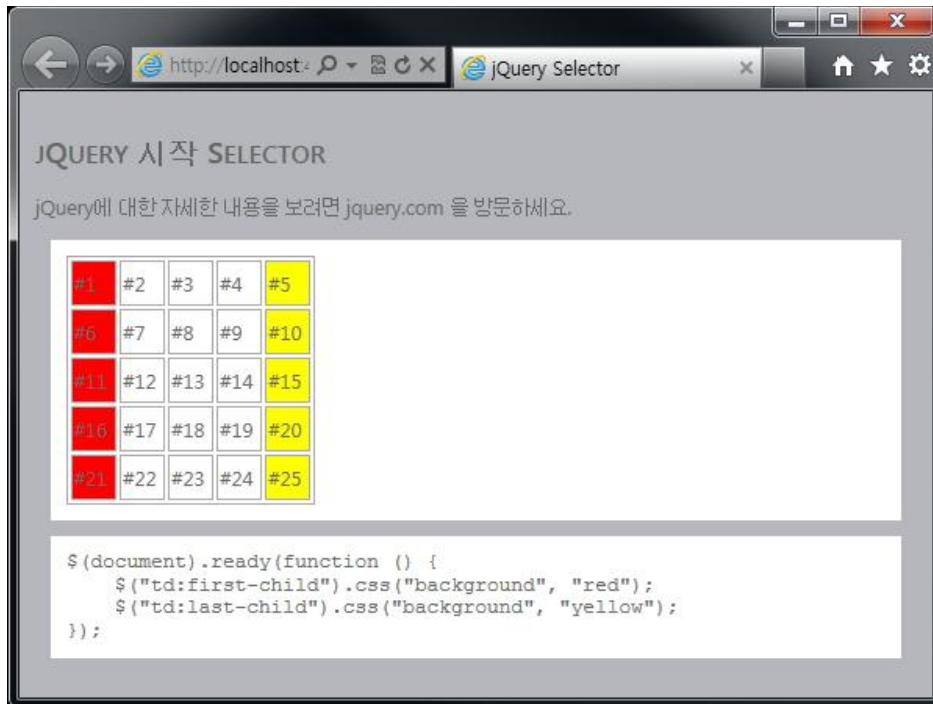
<script type="text/javascript">
    $(document).ready(function () {
        $("td:first").css("background", "red");
    });
</script>

```



[기본 필터 :first 를 사용한 결과]

```
<script type="text/javascript">
    $(document).ready(function () {
        $("td:first-child").css("background", "red");
        $("td:last-child").css("background", "yellow");
    });
</script>
```



[자식필터인 :first-child, :last-child 를 사용한 결과]

두개의 차이점이 보이시나요? 기본 필터인 :first 의 경우 "td"요소 하나만 선택하는 반면 :first-child, :last-child 는 첫번째 마지막 수준의 "td"요소를 모두 선택한다.

1) :nth-child

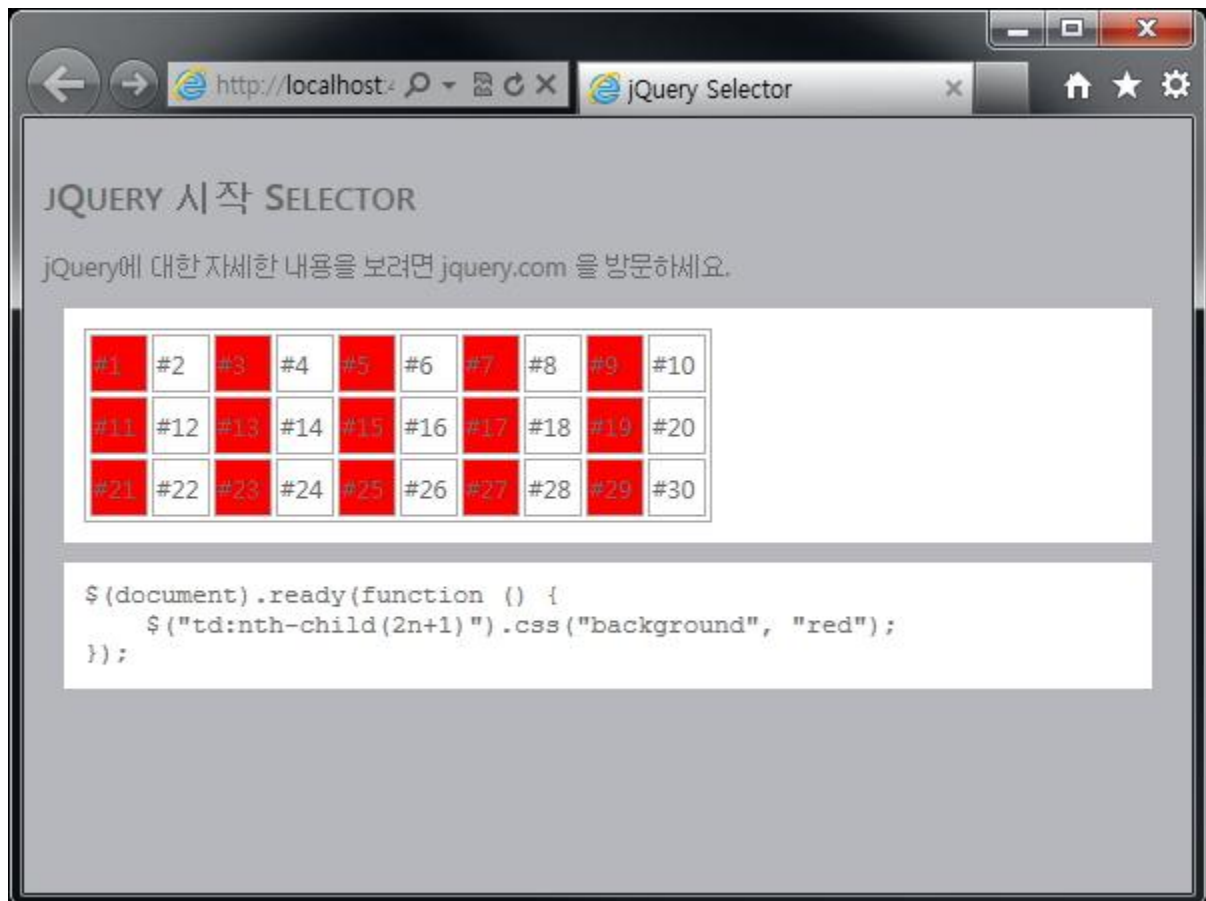
마지막으로 nth-child 라는 필터에 대해 알아 보겠다.

기본적인 동작은 다른 자식필터와 동일하다. 특이한 점은 index, even, odd 같은 값을 지정하여 사용을 하거나 수식을 이용하여 규칙적인 동작을 하도록 할 수 있다는 점이다.

예를 들면 다음과 같은 형태를 가지고 있다.

`$("td:nth-child(0))`, `$("td:nth-child(even))`, `$("td:nth-child(2n+1))` 앞에 두가지 경우에 대해서는 앞에서 설명한 내용만으로 충분히 이해가 되실꺼라 생각하며 마지막 수식을 이용한 자식필터에 대해 간단히 설명을 하도록 하겠다.

`$("#td:nth-child(2n+1)")`을 풀어보면 2의배수($2n$)에 1을 더한 값에 해당하는 위치에 있는 요소를 선택을 하게 된다. 여기서 가장 중요한 부분은 n 은 1이 아닌 0부터 시작을 한다는 것이다. 1이 아닌 0부터 시작을 하므로 수식을 풀면 "1, 3, 5, 7, 9, 11..."의 형식이 된다. 다음 예제를 통해 정말 위에 풀이한 값이 맞는지 확인을 해 보도록 하겠다.



[`:nth-child` 필터를 사용한 결과 (수식에 일치하는 항목을 찾아 붉은색으로 표시한다.)]

앞에서 계산 값(1, 3, 5, 7, 9, 11...)처럼 해당 요소가 선택 되어진 결과를 볼 수 있다. 1이 아닌 0으로 시작한다는 것만 다시 강조 드리면서 필터에 대한 이야기를 여기서 정리를 한다.

지금까지 jQuery 에서 제공하고 있는 필터에 대해 알아 보았다.


[jQuery 강좌] 10. jQuery Traverse - Filtering

웹 프런티어와 함께하는 jQuery 기초강좌

10th - jQuery Traverse(탐색) - 첫 번째 이야기 필터링

<jQuery의 탐색(Traverse) 메서드를 이용하여 셀렉터에 날개를 달자>

영어사전 1-5 / 47건

traverse 동사 [美trəˈvɜːrs]  명사 [美ˈtrævɜːrs] ★

1. 가로지르다, 횡단하다
2. 횡단; 횡단 지역

“Traverse”는 위에 보시는 바와 같이 가로지르다, 횡단이란 뜻을 가지고 있으며, jQuery에서 지원하는 Traverse 메서드 또한 HTML을 가로지르거나, 횡단하는 것처럼 탐색하는 역할을 한다. 해당 메서드를 간략히 설명하면 셀렉터를 통해 선택한 개체에서 다시 한번 개체를 찾고, 필터링하고, 추가하는 작업을 쉽게 구현할 수 있도록 도와주는 메서드라고 생각하시면 된다.

물론 기존의 셀렉터를 통해서도 충분히 원하는 개체를 탐색하거나, 선택 하는데 큰 지장이 없지만, Traverse 메서드를 이용하면 셀렉터를 통해 가져온 개체를 즉 1차 결과물에 추가적인 작업을 통해 2차, 3차 등의 결과를 쉽게 얻을 수 있다는 큰 장점이 있다.

이러한 Traverse 메서드는 크게 Filtering, Miscellaneous Traverse, Tree Traverse 3가지로 구분하고 있으며, 이번 시간을 통해 자세히 알아보도록 하겠다.

1. Filtering

첫 번째로 알아볼 필터링 관련 메서드 이다.

우선 필터링 메서드의 종류에 대해서 알아보기로 하겠다.

메서드 종류(형식)	메서드 설명
.eq(index)	선택한 요소들 중에서 인덱스와 일치하는 단일 요소를 선택 반환한다.
.filter(expr)	선택한 요소에서 표현식(expr)과 일치하는 요소의 집합을 선택 반환한다. 표현식에는 selector, function, element, jQuery object가 올 수 있다.
.first()	선택한 요소에서 첫 번째 단일 요소를 선택 반환한다.

.has(selector)	선택한 요소에서 selector 항목을 가지고 있는 요소의 집합을 선택 반환한다.
.is(expr)	표현식과 일치하는 조건이 있으면 true 를 반환한다. 표현식에는 selector, function, element, jQuery object 가 올 수 있다.
.last()	.first() 와 반대되는 메서드로 마지막 단일 요소를 선택 반환한다.
.map(callback)	jQuery 개체에 있는 요소의 집합을 다른 집합으로 변경해서 이동 시킵니다.
.not(expr)	표현식과 일치하지 않는 요소의 집합을 선택 반환한다.
.slice(start,[end])	선택한 요소에서 start, end 번 째에 해당하는 집합을 선택 반환한다.

표 1. [jQuery 탐색 필터 메서드 종류]

생각보다 많은 메서드를 제공하고 있으나, 이중에 몇 개는 기존에 설명 드렸던 셀렉터의 필터와 겹치는 내용이 있다.

대표적으로 first(), last(), eq(), not() 메서드를 들 수 있다.

이전 필터와 살짝 비교를 하면 다음과 같다.

```
$("td:eq(0)") == $("td").eq(0)
```

```
$("td:first") == $("td").first()
```

```
$("td:last") == $("td").last()
```

```
$("td:not('.myClass')") == $("td").not(".myClass")
```

양쪽다 모두 동일한 결과를 보인다.

그런데, 결과적으로 동일한 역할을 하는데 왜 여러 개를 만들어 놓았을까요? 단순히 개발자의 맘일까요? 이번강의 앞 부분에 언급한 1 차 결과물에 추가적인 작업과 이후에 설명을 드릴 miscellaneous 항목에 존재하는 .end() 라는 메서드를 보시면 좀더 쉽게 이해가 되실꺼라 생각한다.

이제부터는 앞서 비교한 메서드를 제외한 나머지 메서드에 대해 간단히 알아 보도록 하겠다.

1) .filter(expr)

1 차적으로 셀렉터를 통해 선택한 요소들의 집합에서 또 한번의 필터링을 통해 원하는 개체를 선택하는 메서드이다.

표현식 부분에는 Selector, function, element, jQuery object 가 위치할 수 있다.

이 메서드 또한 앞의 필터와 중복되는 부분이 존재한다. 간단한 예를 들면 다음과 같다.

```
$("td:even") == $("td").filter(":even")
```

동일한 결과를 나타냅니다. 하지만 필터 메서드의 강점은 function 을 이용하여 마치 필터를 확장해서 쓰는 것과 같은 효과를 얻을 수 있는 점이다.

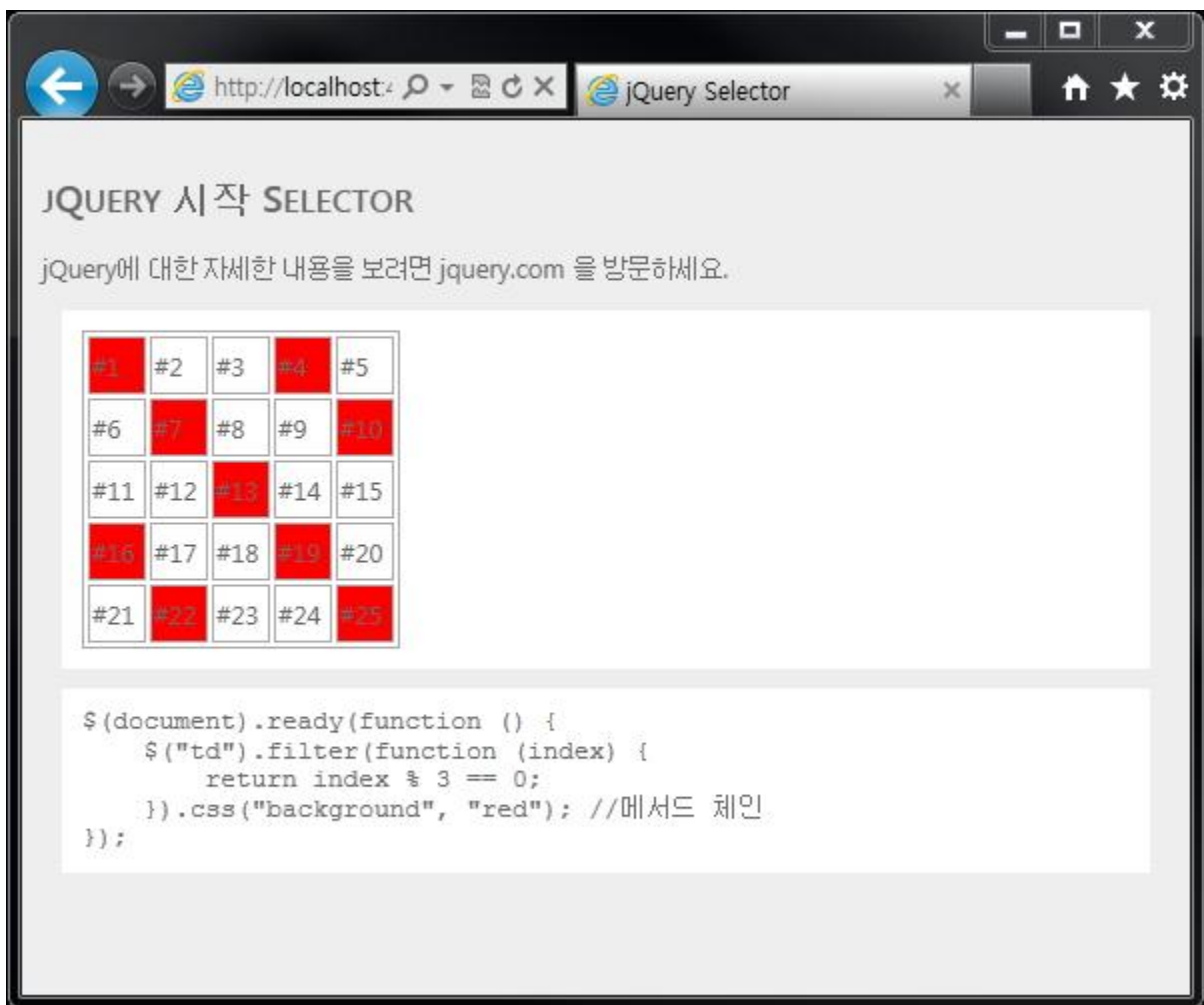
다음은 셀렉터를 통해 선택한 개체의 집합을 필터 메서드를 통해 3 의 배수에 해당하는 요소를 선택하는 예제이며, function(index) index 는 \$("li")를 통해 가져온 개체 집합의 번호이다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("td").filter(function (index) {
        return index % 3 == 0;
      }).css("background", "red");
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <table>
      <tr>
        <td>#1</td><td>#2</td><td>#3</td><td>#4</td><td>#5</td>
      </tr>
      <tr>
        <td>#6</td><td>#7</td><td>#8</td><td>#9</td><td>#10</td>
      </tr>
      <tr>
        <td>#11</td><td>#12</td><td>#13</td><td>#14</td><td>#15</td>
      </tr>
    </table>
  </div>
</body>
</html>
```

```

        <td>#16</td><td>#17</td><td>#18</td><td>#19</td><td>#20</td>
    </tr>
    <tr>
        <td>#21</td><td>#22</td><td>#23</td><td>#24</td><td>#25</td>
    </tr>
</table>
</div>
</body>
</html>

```



[필터 메서드를 사용하여 3의 배수에 해당하는 요소에 스타일을 적용]

물론 위 예제는 자식필터(child filter)를 이용하여 구현이 가능하지만 여기서 눈 여겨 볼 내용은 이렇게 자신이 원하는 기능을 추가로 하여 개체를 선택 할 수 있다는 것이다.

2) .not()

필터 메서드와는 반대로 동작하는 메서드이다. 필터 메서드는 표현식과 일치하는 요소를 선택하는 반면에 이(.not()) 메서드는 표현식과 일치하지 않는 요소를 선택 반환한다.

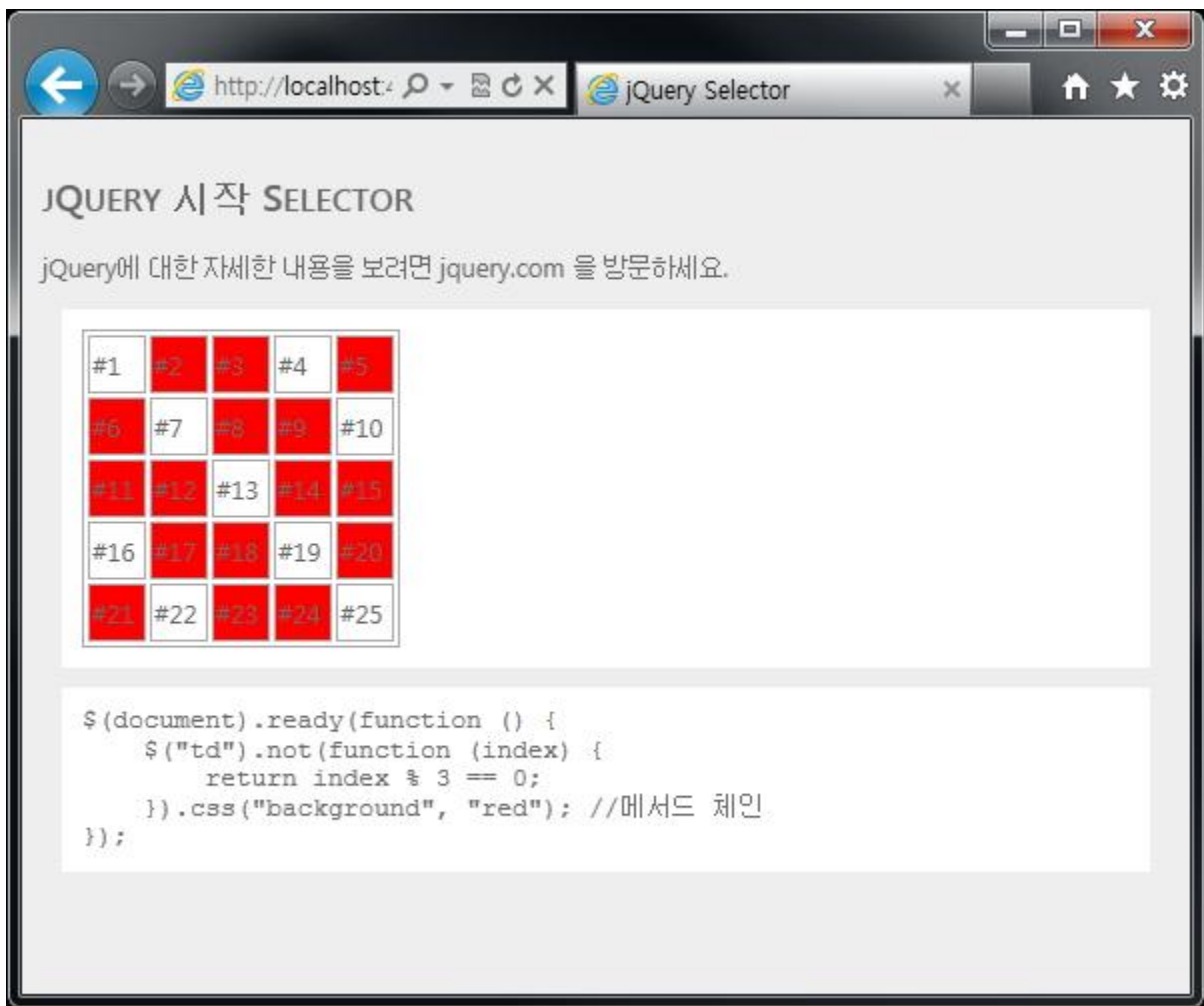
앞의 예제에서 스크립트만 변경을 해 보도록 하겠다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("td").not(function (index) {
        return index % 3 == 0;
      }).css("background", "red");
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <table>
      <tr>
        <td>#1</td><td>#2</td><td>#3</td><td>#4</td><td>#5</td>
      </tr>
      <tr>
        <td>#6</td><td>#7</td><td>#8</td><td>#9</td><td>#10</td>
      </tr>
      <tr>
        <td>#11</td><td>#12</td><td>#13</td><td>#14</td><td>#15</td>
      </tr>
    </table>
  </div>
</body>
</html>
```

```

        <td>#16</td><td>#17</td><td>#18</td><td>#19</td><td>#20</td>
    </tr>
    <tr>
        <td>#21</td><td>#22</td><td>#23</td><td>#24</td><td>#25</td>
    </tr>
</table>
</div>
</body>
</html>

```



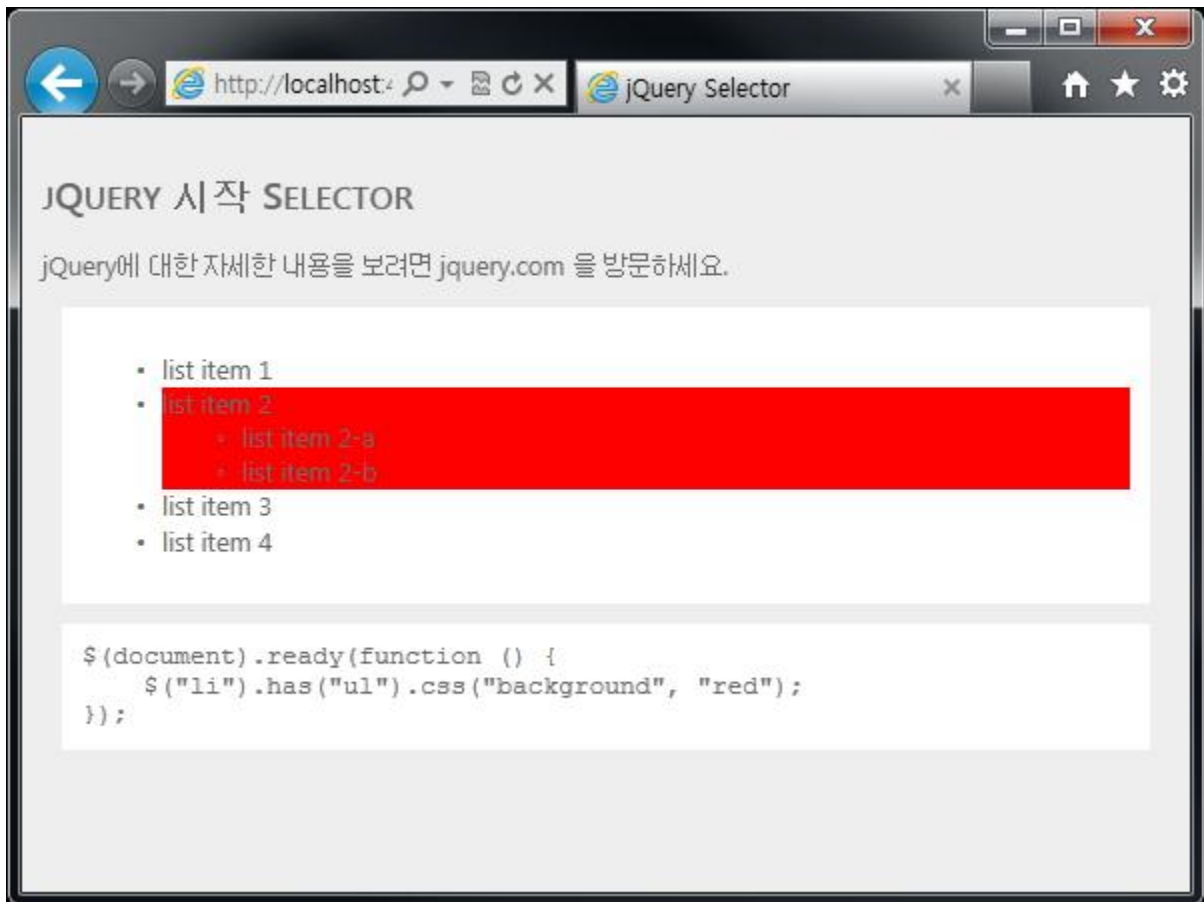
[앞의 결과와는 다르게 3의 배수가 아닌 요소를 표시한다.]

3) .has(selector)

기본 셀렉터를 통해 가져온 개체의 집합에서 selector 에 해당하는 요소를 가진 개체만을 선택 반환한다.

다음 예제를 통해 간단히 알아보고 넘어 가도록 하겠다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("li").has("ul").css("background", "red");
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <ul>
      <li>list item 1</li>
      <li>list item 2
        <ul>
          <li>list item 2-a</li>
          <li>list item 2-b</li>
        </ul>
      </li>
      <li>list item 3</li>
      <li>list item 4</li>
    </ul>
  </div>
</body>
</html>
```



셀렉터를 통해 가져온 li 개체에서 has() 함수를 통해 "ul" 요소를 가지고 있는 개체만을 선택하는걸 확인 할 수 있다.

4) .is(expr)

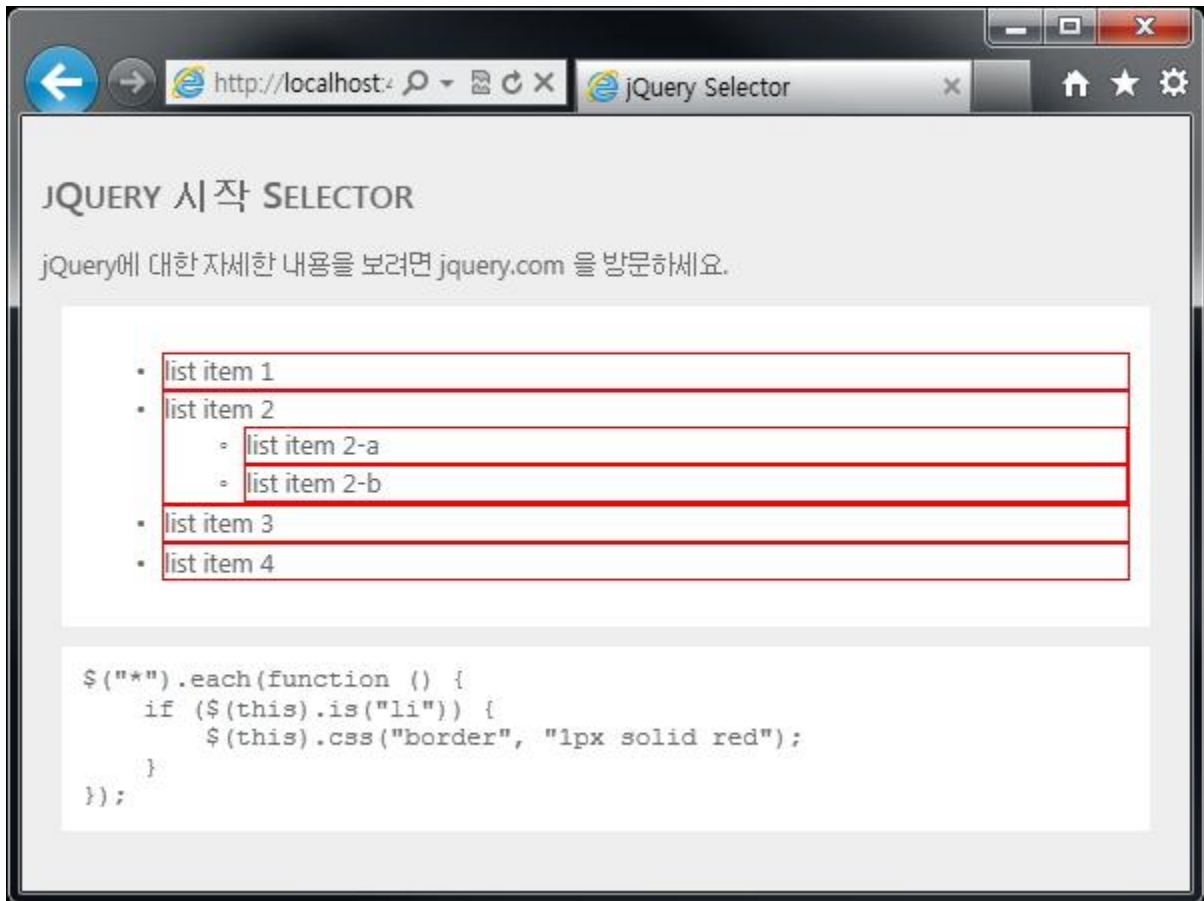
기본 셀렉터에서 가져온 개체의 집합에서 표현 식과 일치하는 부분이 있는지에 대한 여부를 반환한다. 집합에서 단 하나라도 일치하는 항목이 존재한다면 "true"를 반환한다. 물론 일치하는 항목이 존재하지 않다면 "false"를 반환하게 된다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("*").each(function () {
        if ($(this).is("li")) {
          $(this).css("border", "1px solid red");
        }
      });
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <ul>
      <li>list item 1</li>
      <li>list item 2
        <ul>
          <li>list item 2-a</li>
          <li>list item 2-b</li>
        </ul>
      </li>
      <li>list item 3</li>
      <li>list item 4</li>
    </ul>
  </div>
</body>
</html>
```

```

    </ul>
  </div>
</body>
</html>

```



모든 요소를 선택하고 "each()"라는 메서드를 통해서 개체의 집합에 접근 후 "li"와 동일한 요소일 경우 지정한 스타일이 적용된 걸 확인 할 수 있다. "each()" 라는 메서드는 이후에 설명을 다시 드리겠지만, jQuery 를 통해 가져온 개체의 집합을 foreach 와 같이 순환하는 기능을 제공한다. 자세한 설명은 이후에 다시 드리도록 하겠다.

5) .map(callback)

셀렉터를 통해 가져온 개체의 집합을 jQuery 개체의 배열 형식으로 반환한다.

설명을 쓰고보니, 그냥 셀렉터와의 큰 차이점이 없어 보인다, 하지만 다행이도 아직 callback 이라는 함수가 남아 있다.

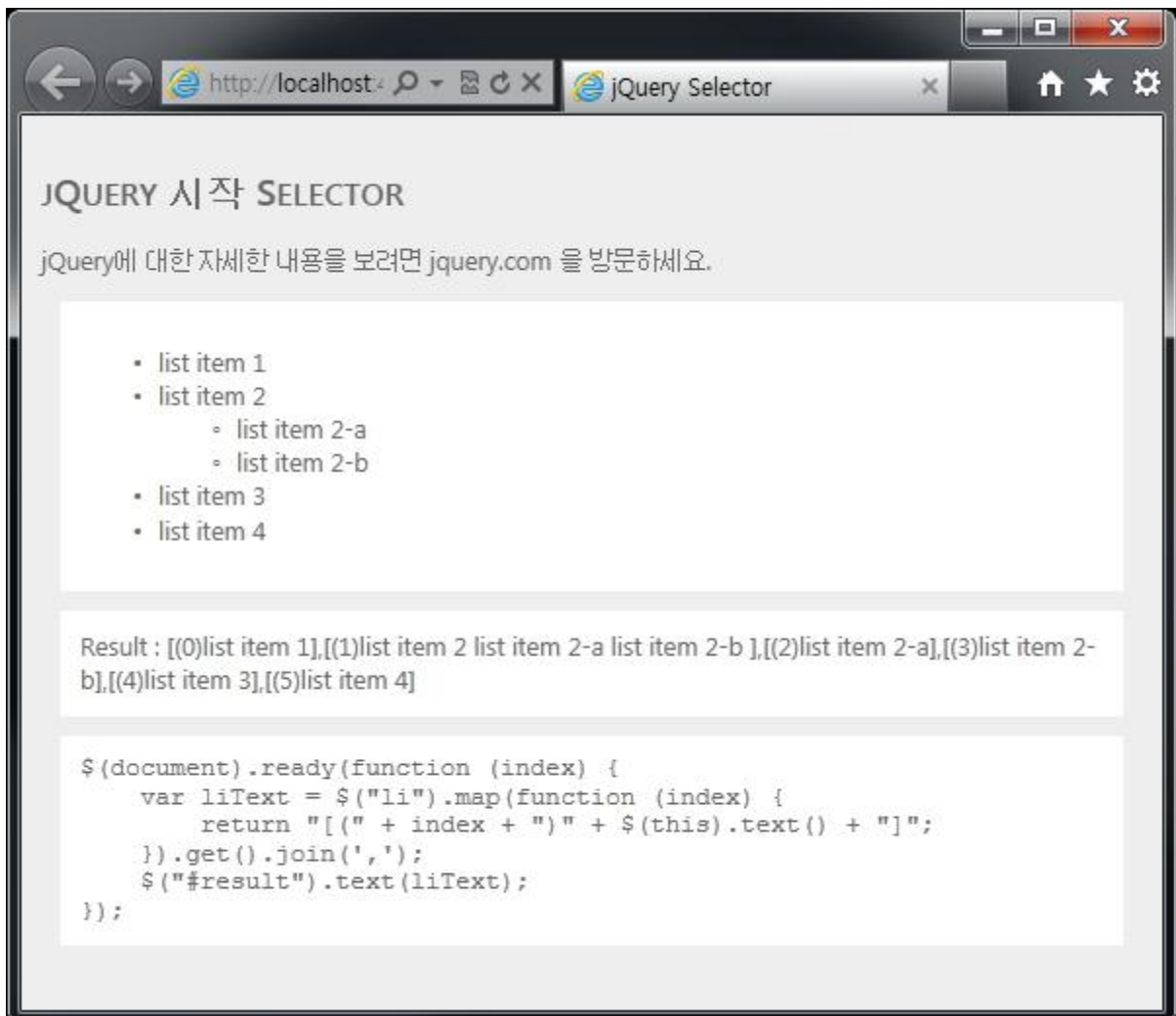
Callback 함수를 이용하면 셀렉터를 이용해 가져온 개체의 집합에서 원하는 작업을 진행하고 해당 작업의 결과를 jQuery 의 배열 형식으로 반환 할 수 있다. 다음 예제를 보시면 쉽게 이해가 될 것으로 생각된다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      var liText = $("li").map(function (index) {
        return "[" + index + "]" + $(this).text() + ";";
      }).get().join(';');
      $("#result").text(liText);
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <ul>
      <li>list item 1</li>
      <li>list item 2
        <ul>
          <li>list item 2-a</li>
          <li>list item 2-b</li>
        </ul>
      </li>
    </ul>
  </div>
</body>
</html>
```

```

        </ul>
    </li>
    <li>list item 3</li>
    <li>list item 4</li>
</ul>
</div>
<div>
    Result : <span id="result"></span>
</div>
</body>
</html>

```



이번에는 get(), join(), text() 라는 새로운 메서드가 보인다.

예제의 설명을 위해 간단히 말씀을 드리면, get(), join() 메서드는 집합(배열) 형식의 개체에 접근하는 방법과 join() 메서드의 인자 값으로 연결된 문자를 반환하게 된다. text()메서드는

셀렉터를 통해 가져온 개체의 text 를 가져오는 역할을 한다. Value 값에 접근 하기 위해서는 val()이라는 메서드를 사용한다.

위 예제는 "li" 요소를 선택 후 해당 개체의 text 값을 map() 메서드를 통해 반환하고 있다. 지금까지 설명 드린 기본 셀렉터, 필터에서도 위와 동일한 기능을 제공하고 있으나 이번 경우에도 개발자가 원하는 작업을 정의, 확장하여 사용할 수 있다는 큰 장점이 있다.

6) .slice(start, [end])

선택한 요소에서 start 번째에서 시작하고 end 번째에서 끝나는 개체의 집합을 반환한다.

기본 셀렉터에서 선택된 요소는 0 부터 시작을 한다. start 는 포함을 하나, end 번째 요소는 포함을 하지 않으니, 이점 유의 하시기 바란다. 간단한 두 개의 예시를 통해 알아보면 다음과 같다.

총 8 개의 요소를 선택했다고 가정한다.

각 요소의 집합에 번호를 주면, 해당 인덱스는 0 부터 시작이므로 0, 1, 2, 3, 4, 5, 6, 7 이 된다.

1) .slice(2) 경우 : 2, 3, 4, 5, 6, 7 에 해당하는 개체의 집합을 반환한다.

2) .slice(2,4) 경우 : 2, 3 에 해당하는 개체의 집합을 반환한다.

1)의 경우 0, 1, 2, 3, 4, 5, 6, 7 앞에서 2 개의 집합을 제거 한다.

2)의 경우 0, 1, 2, 3, 4, 5, 6, 7 앞에서 2 개의 집합을 동일하게 제거 후 4 번째를 포함한 집합을 한번 더 제거 후 반환하게 된다.

두 경우의 차이가 이해가 되시나요 ?

jQuery 사이트에서 제공하는 .slice() 메서드 설명에 있는 예제를 통해 살펴 본다면 아마도 쉽게 이해가 되리라 생각한다.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    div { width:40px; height:40px; margin:10px; float:left;
          border:2px solid blue; }
    span { color:red; font-weight:bold; }
    button { margin:5px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
</head>
<body>
  <p><button>Turn slice yellow</button>
  <span>Click the button!</span></p>
```

```

<div></div>
<div></div>

<div></div>
<div></div>
<div></div>
<div></div>
<div></div>

<div></div>
<div></div>
<script>

function colorEm() {
  var $div = $("div");
  var start = Math.floor(Math.random() *
                        $div.length);
  var end = Math.floor(Math.random() *
                      ($div.length - start)) +
            start + 1;
  if (end == $div.length) end = undefined;
  $div.css("background", "");
  if (end)
    $div.slice(start, end).css("background", "yellow");
  else
    $div.slice(start).css("background", "yellow");

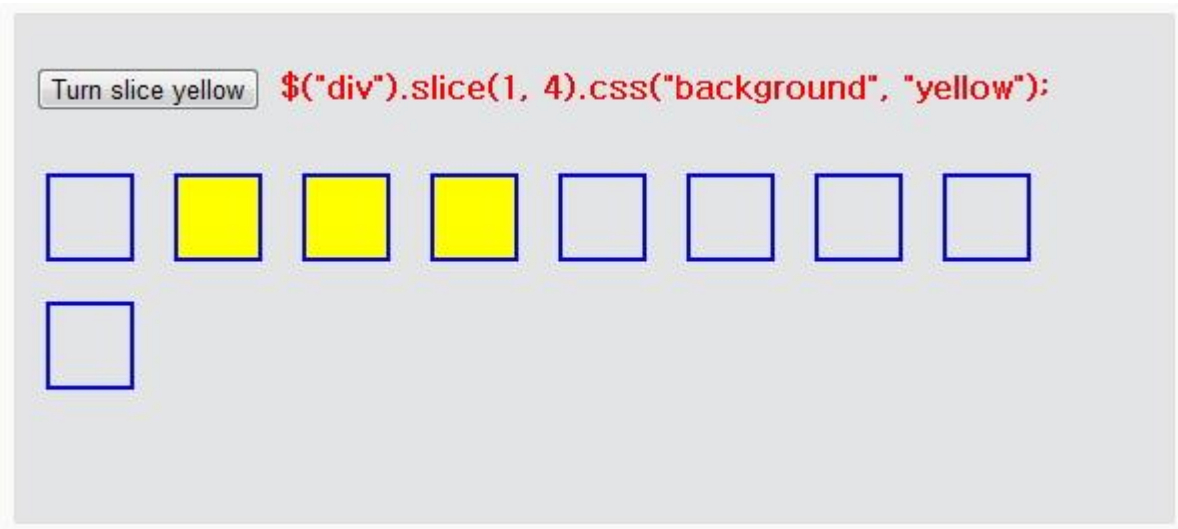
  $("span").text('$("div").slice(' + start +
                (end ? ', ' + end : '') +
                ').css("background", "yellow");');
}
$("button").click(colorEm);

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

```

해당 소스는 버튼을 클릭할 때 마다 랜덤 한 숫자를 slice() 메서드에 대입 후 결과를 보여주는 코드이며, 아직 설명하지 않은 부분이 있어 자세한 설명은 생략을 하도록 하겠다.

실행 화면은 다음과 같다.



버튼을 누를 때마다 어떠한 .slice() 메서드에 어떠한 인자가 들어갔으며, 인자 값에 따른 결과를 보여주고 있다. 아직 이해가 안 되신다면 <http://api.jquery.com/slice/> 페이지에서 예제를 실행해 보시기 바란다.

[jQuery 강좌] 11. jQuery Traverse - Miscellaneous Traversing

웹 프런티어와 함께 하는 jQuery 기초강좌

11th - jQuery Traverse(탐색) - 두 번째 이야기 - Miscellaneous

탐색 메서드 두 번째 시간으로 이번에는 Miscellaneous 메서드에 대해 알아 보도록 하겠다.

우선 Miscellaneous 메서드의 종류를 정리하면 다음과 같다.

메서드	메서드 설명
.add()	일치하는 요소의 집합에 요소를 추가한다.
.andSelf()	현재 설정 스택에 요소의 이전 설정을 추가한다.
.contents()	텍스트 및 주석 노드를 포함 일치하는 요소 집합의 자식 집합을 반환한다.
.end()	이전 상태로 일치하는 집합을 반환한다.

표 1. [jQuery Miscellaneous 메서드 종류]

Miscellaneous 의 뜻을 사전에서 찾아 보면 '여러 가지 종류의, 이것저것 다양한' 이란 뜻을 가지고 있다. 아마도 해당 메서드의 성격이 너무나도 달라서 카테고리의 이름을 이렇게 잡은 듯 하다.

1. add()

이름에서도 쉽게 알 수 있듯이 뭔가를 더하는 역할을 하는 메서드로 다음과 같이 사용을 한다.

```
.add(selector)
.add(elements)
.add(html)
```

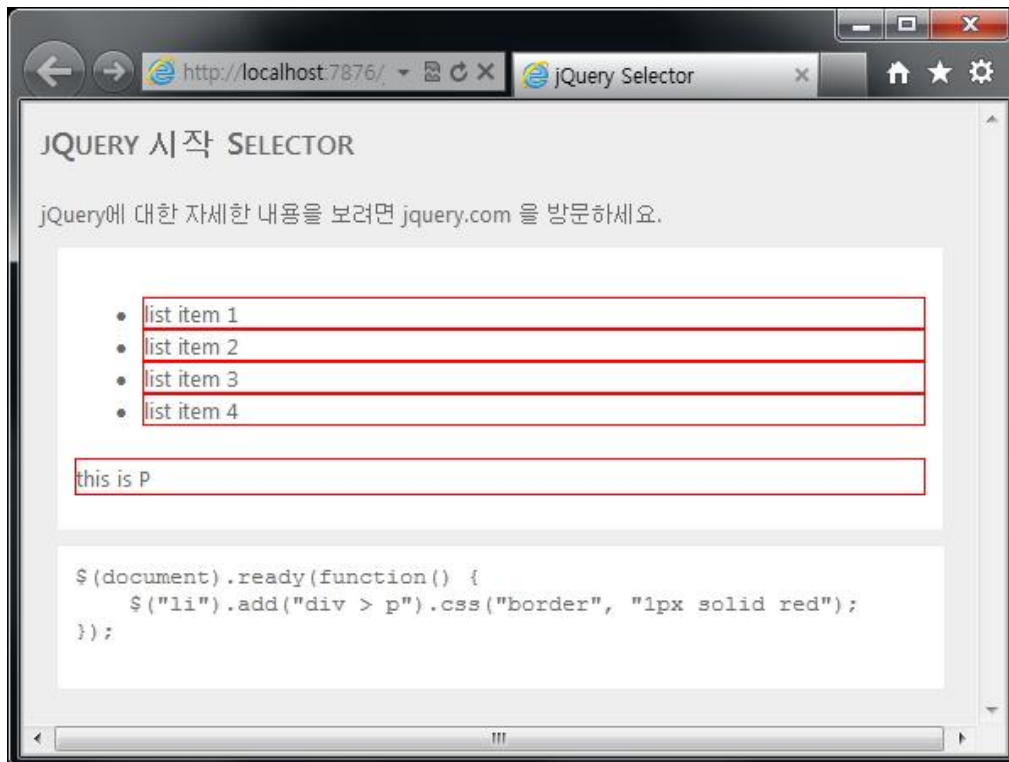
셀렉터를 통해 선택한 개체에 ()안에 값을 더하는 메서드로 selector 를 통해 가져온 개체, span 과 같은 요소, testSpan과 같은 html 요소가 올 수 있다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
```

```

    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
</style>
<script src="http://code.jquery.com/jquery-latest.js"></script>
<script type="text/javascript">
    $(document).ready(function() {
        $("li").add("div > p").css("border", "1px solid red");
    });
</script>
</head>
<body style="padding:10px;">
    <h2>jQuery 시작 Selector</h2>
    <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
    <div>
        <ul>
            <li>list item 1</li>
            <li>list item 2</li>
            <li>list item 3</li>
            <li>list item 4</li>
        </ul>
        <p>this is P</p>
    </div>
</body>
</html>

```



2. .andSelf()

사용빈도가 상당히 낮을 것으로 추측을 하는 메서드로 저 또한 이번 강좌를 준비하면서 알게 된 메서드이다. 앞서 설명한 부분을 다시 보면 "현재 설정 스택에 요소의 이전 설정을 추가한다." 말이 좀 어렵다.

우선 jQuery 문서에 있는 예제를 살펴 보기로 하겠다.

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      p, div { margin:5px; padding:5px; }
      .border { border: 2px solid red; }
      .background { background:yellow; }
    </style>
    <script src="http://code.jquery.com/jquery-latest.js"></script>
  </head>
  <body>
    <div>
      <p>First Paragraph</p>
      <p>Second Paragraph</p>
    </div>
```

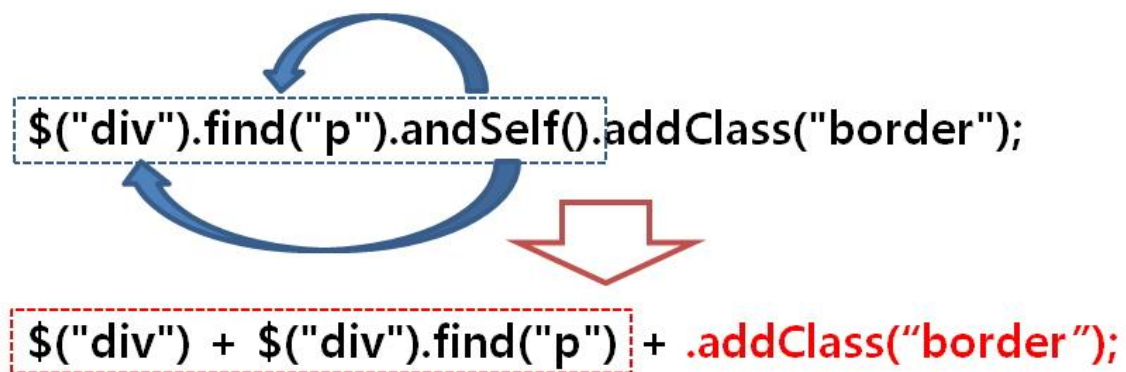
```

<script>
$("div").find("p").andSelf().addClass("border");
$("div").find("p").addClass("background");
</script>
</body>
</html>

```



다음 그림을 보면 좀더 이해가 쉽다.



3. contents()

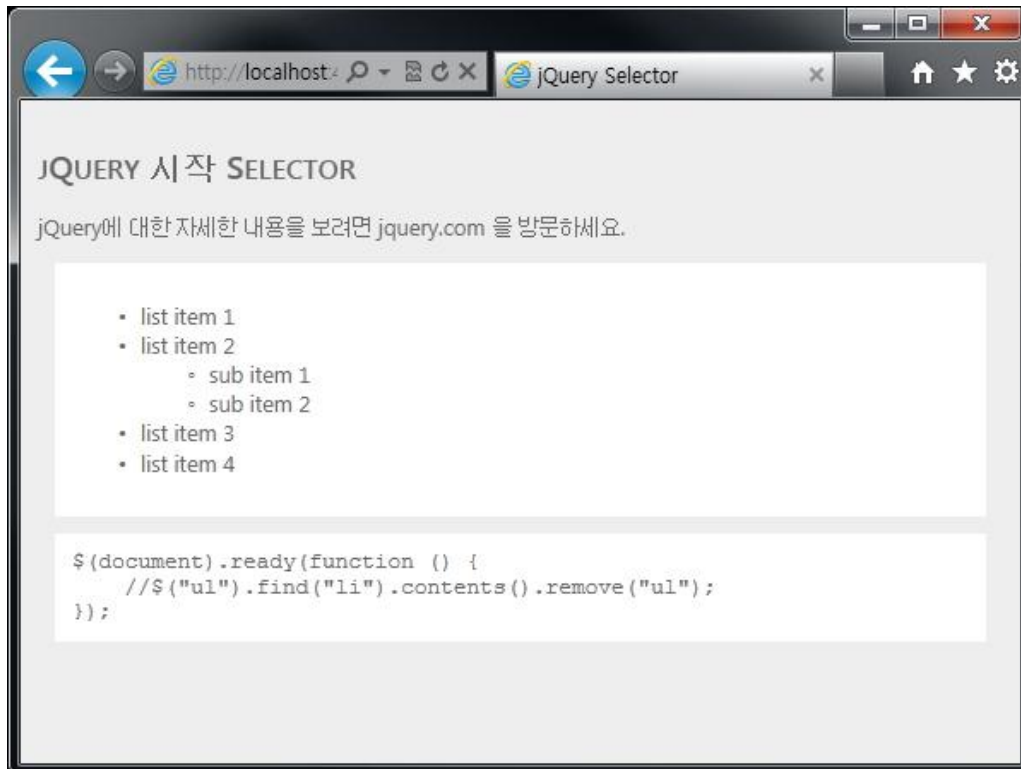
셀렉터를 통해 선택된 개체 안에 있는 모든 요소를 반환한다.

단어의 뜻대로 선택된 개체 안에 있는 요소의 집합을 반환하며, 이 메서드의 경우 filter 메서드와 조합을 통해 많이 사용을 하고 있다.

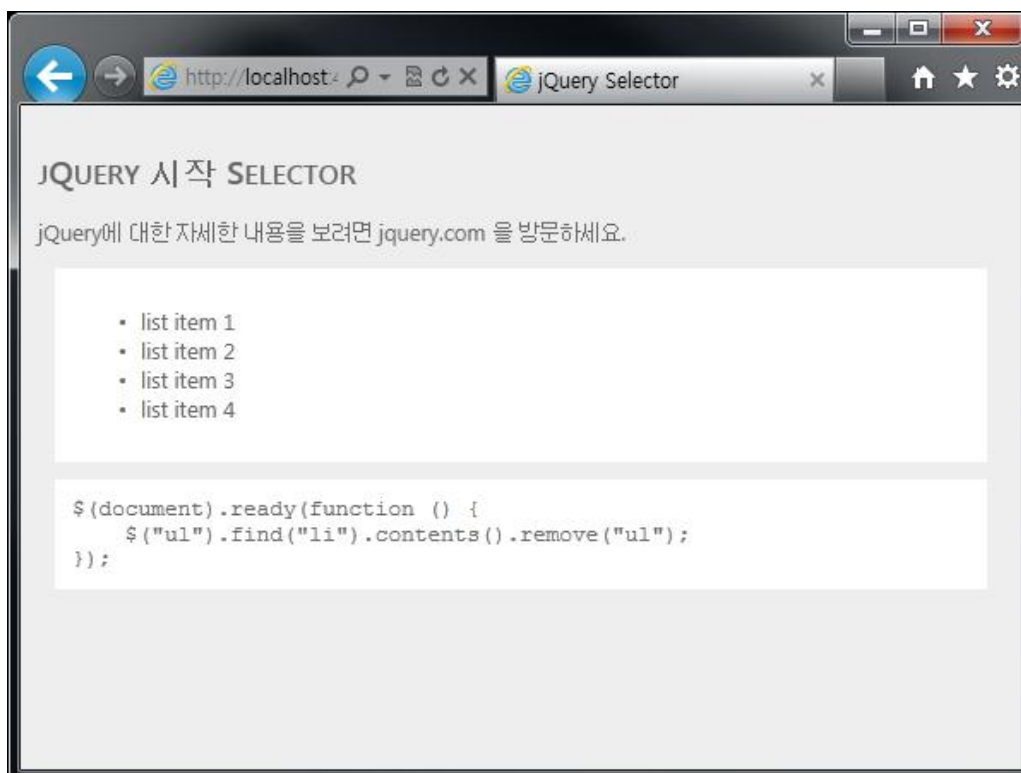
다음 예제를 통해 간단한 사용법을 살펴 보도록 하겠다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("ul").find("li").contents().remove("ul");
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <ul>
      <li>list item 1</li>
      <li>list item 2
        <ul>
          <li>sub item 1</li>
          <li>sub item 2</li>
        </ul>
      </li>
      <li>list item 3</li>
      <li>list item 4</li>
    </ul>
  </div>
</body>
```


</html>



[메서드 적용전의 모습]



[메서드 적용 후의 모습]

remove(selector) 메서드는 "selector"와 일치하는 내용을 삭제한다.

예제를 간단히 설명하면, 선택터를 통해 ul 요소를 찾은 후 find()를 통해 "li" 요소를 찾는다. 마지막으로 contents() 메서드를 통해 li 요소에 담겨있는 모든 요소를 찾은 후 remove() 메서드를 통해 "ul"이 제거가 되었다.

4. .end()

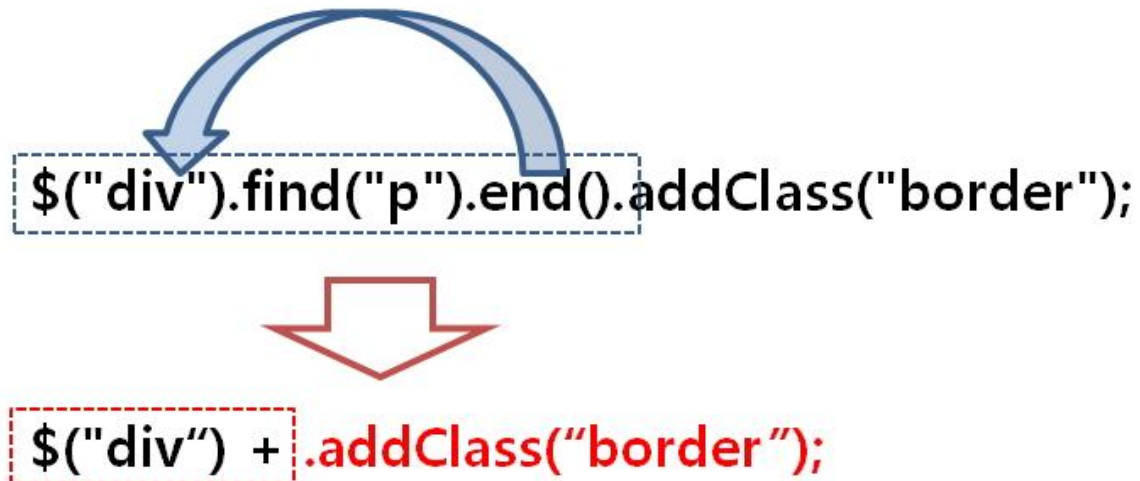
이 메서드는 정말 중요하고 많은 곳에서 사용이 되며, 실제로 복잡한 스크립트를 작성할 때 마법과 같은 역할을 해 주는 메서드이다.

.end() 메서드를 간단히 설명 드리자면 .end() 메서드가 호출 후 바로 이전의 상태로 돌아가는 것을 의미한다.

\$(div).find(p).end().addClass("myClass");와 같은 구문이 있다고 한다면 myClass 라는 class 속성이 div 에 적용이 된다.

.end() 메서드가 없다면 "div" 요소 안에 있는 "p" 요소에 class 속성이 적용이 되지만 .end() 메서드의 호출로 인해 "p" 요소 이전의 개체인 \$("div") 개체가 다시 호출이 되고 호출된 \$("div")에 class 속성이 적용이 되는 것이다.

그림으로 다시 설명을 드리자면 아래와 같이 동작을 한다고 보시면 된다.



.end() 메서드의 경우 앞에 강좌에서 설명한 jQuery 의 특징 중에 플러그인과 메서드체인을 이용한 스크립트 작성에 빠질 수 없는 요소로 정확히 이해를 하신다면 간결하고 클라이언트에 부담 줄일 수 있는 프로그래밍이 가능하므로 해당 메서드에 대해 정확히 이해를 하시기 바란다.

[jQuery 강좌] 12. jQuery Traverse - Tree Traversal

웹 프런티어와 함께 하는 jQuery 기초강좌

12th - jQuery Traverse(탐색) - 세 번째 이야기 - Tree Traversal

탐색(Traverse) 메서드의 마지막 시간으로 이번에는 Tree 와 관련된 메서드에 대해 알아 보도록 하겠다.

<Tree Traversal>

Tree 하면 생각나는 그 강좌가 있지 않으셨나요 ?

Hierarchy Selector 진행 시 HTML DOM 에 대한 내용 기억하시나요? 이번 시간도 DOM 을 이용하는 부분이니, 혹시라도 기억이 나지 않으시면 미리 한번 확인을 해 보시고 이번 강좌를 보시면 도움이 많이 되지 않을까 한다.

이전에 작성했던 강좌와 동일하게 우선 jQuery 에서 지원하고 있는 메서드의 목록을 알아 보도록 하겠다.

아래에 나열된 목록과 설명은 jQuery 에서 제공하는 메서드의 일부이며, 더 자세한 내용은 <http://docs.jquery.com/Traversing> 에서 확인 하시기 바란다.

메서드	메서드 설명
.children([selector])	선택된 개체의 자식 중 Selector 와 동일한 요소를 가져옵니다.
.closest([selector])	선택된 개체에서 DOM Tree 를 통해 가장 가까운 조상 요소를 가져옵니다.
.find([selector])	선택된 개체에서 selector 와 일치하는 요소를 가져옵니다.
.next([selector])	선택된 개체에서 selector 와 일치하는 형제 요소를 가져옵니다. 일치하는 항목이 없을 경우 다음 항목으로 이동해 요소를 찾는다.
.parent([selector])	선택된 개체에서 selector 와 일치하는 부모의 요소를 가져옵니다.
.prev([selector])	선택된 개체에서 selector 와 일치하는 바로 앞의 형제 요소를 가져옵니다.
.siblings([selector])	선택된 개체에서 selector 와 일치하는 형제 요소를 가져옵니다. 자신은 제외한다.
표 1. [jQuery Tree 메서드 종류]	

selector 는 일종의 옵션으로 존재를 하며, selector 의 사용여부는 일종의 필터링과 관계가 있으며 앞으로 설명 드릴 메서드는 모두 동일한 형태를 취하고 있다. Selector 가 있을 경우 selector 와 일치하는 요소를 탐색하고, 없을 경우 가장 가까운 요소를 가져옵니다.

설명을 보시면 앞서 설명을 드렸드시피 HTML DOM 에 대한 이해만 되어 있으시면 너무나도 쉽게 이해가 되리라 생각이 된다.

API 명이 너무나 명확해서 해당 명칭만 번역해 보셔도 메서드의 역할은 대부분 유추가 가능하다. 정말 자주 사용하는 find() 메서드와 .siblings() 메서드에 알아 보도록 하겠다.

1. .find()

이 메서드는 앞에 예제에서 알게 모르게 등장을 했었다. 예제를 앞서 간단히 설명을 붙이자면, \$("div").find("a")라고 했을 경우 div 요소를 우선 탐색하여 가져온 개체에서 "a" 태그가 존재하는 요소를 가져옵니다.

다른 자료를 찾아보니, filter() 메서드와 find() 메서드의 차이점에 대해 언급한 부분이 있어 저도 살짝 이야기를 하고 넘어가도록 하겠다.

\$("div").filter("a") : filter() 메서드를 사용할 경우는 div 요소의 집합에서 "a"를 찾는다.

\$("div").find("a") : find() 메서드를 사용할 경우 div 요소의 집합 내용에서 "a"의 요소를 가져옵니다.

다음 예제를 통해 차이점과 기본적인 사용방법에 대해 알아 보도록 하겠다.

```

<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js" ></script>
  <script type="text/javascript" >
    $(document).ready(function () {
      $("div").find("p").css("border", "1px solid red");
      // $("div").filter("p").css("border", "1px solid red");
      // $("div").filter(".myClass").css("border", "1px solid red");
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <p>this is 1st p</p>
  </div>
  <div>
    <p>this is 2nd p</p>
  </div>
  <div class="myClass">
    <p>this is 3rd p</p>
  </div>
</body>
</html>

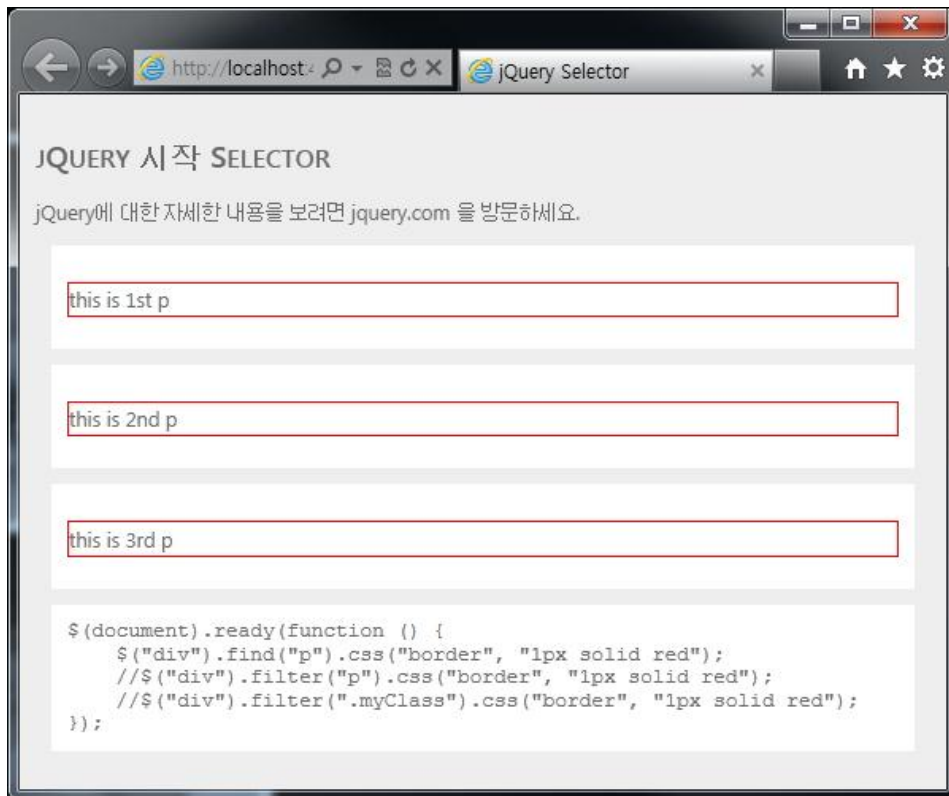
```

예제 소스를 보시면 아래의 모습처럼 세 가지의 스크립트가 존재 한다.

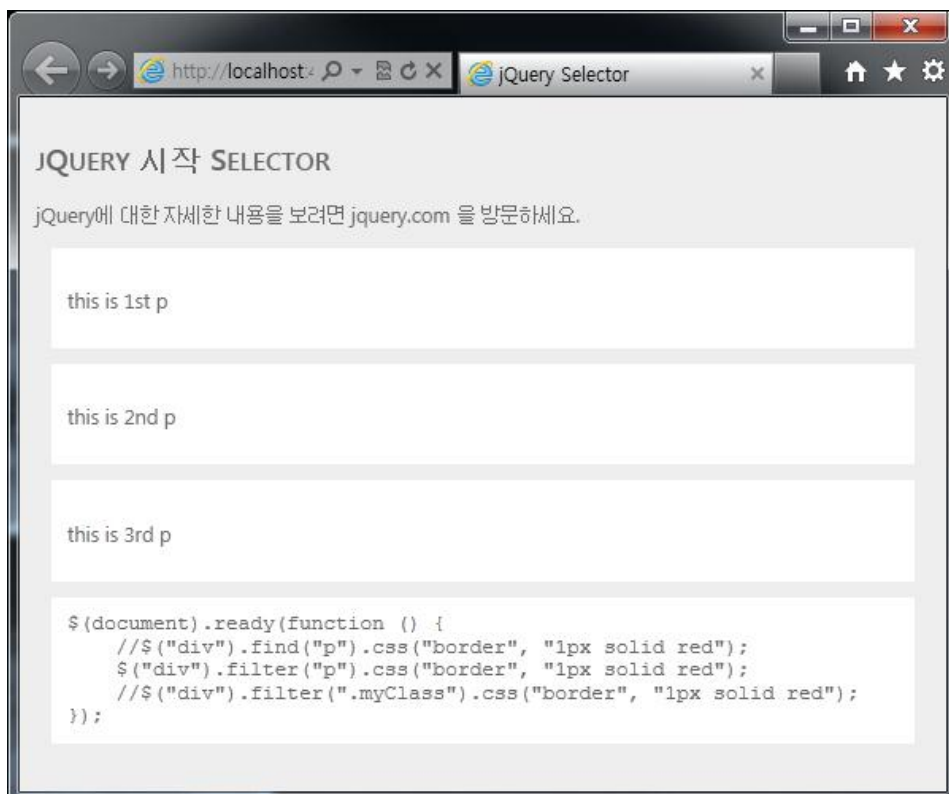
```

$("div").find("p").css("border", "1px solid red");
$("div").filter("p").css("border", "1px solid red");
$("div").filter(".myClass").css("border", "1px solid red");

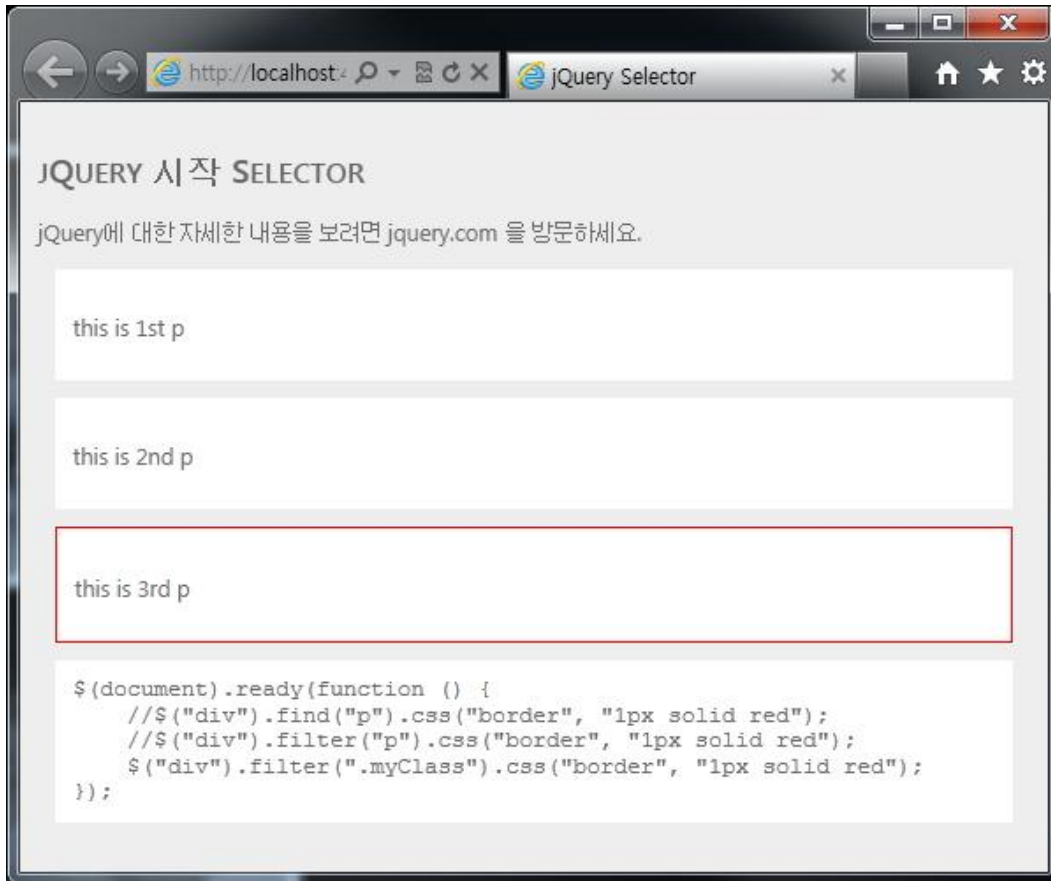
```



`[$("div").find("p").css("border", "1px solid red");]`



`[$("div").filter("p").css("border", "1px solid red");]`



`[$("div").filter(".myClass").css("border", "1px solid red");]`

화면과 같이 나열된 세 가지 메서드가 모두 다르게 동작하는 걸 알 수 있다. `find()`의 경우 이미 탐색된 개체의 내부 요소에서 selector 와 동일한 요소를 가져오며, `filter()`의 경우는 처음 탐색 시에 selector 에 해당되는 값으로 탐색을 도와주는(?) 역할을 한다고 보면 된다.

2. `.siblings()`

개인적으로 상당히 많은 도움을 받은 메서드이며, 주로 목록에서 마우스로 클릭한 요소만 강조할 경우 매우 쉽게 프로그래밍을 가능하도록 해준 메서드이다. 도움을 받았던 부분을 정리해 예제를 만들어 보았다.

`.siblings()` 메서드를 통해 목록에서 사용자가 선택(클릭)한 부분을 어떻게 처리하는지 알아보겠다.

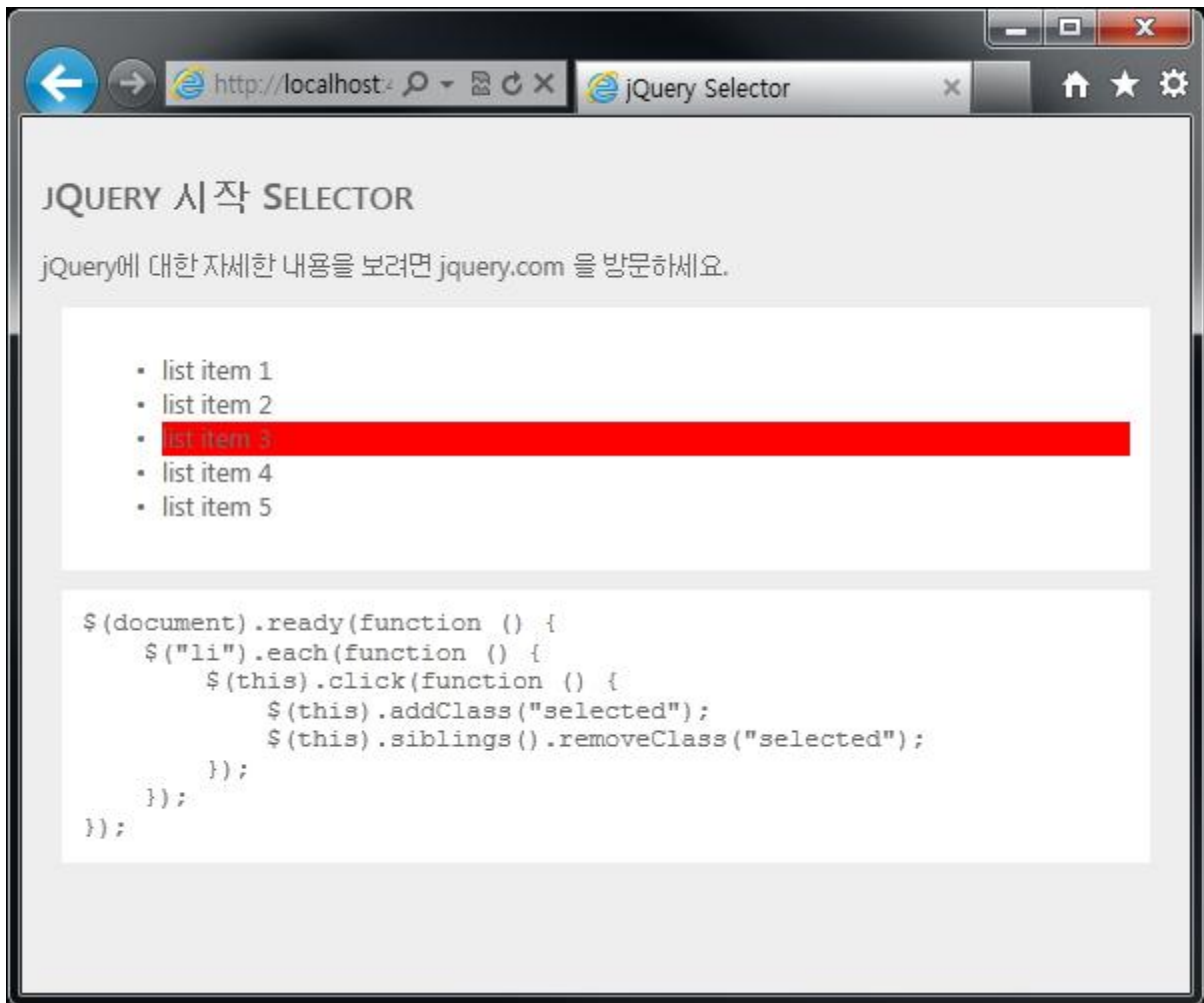
```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
```

```

    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
    .selected { background:red }
</style>
<script src="http://code.jquery.com/jquery-latest.js"></script>
<script type="text/javascript">
    $(document).ready(function () {
        $("li").each(function () {
            $(this).click(function () {
                $(this).addClass("selected");
                $(this).siblings().removeClass("selected");
            });
        });
    });
</script>
</head>
<body style="padding:10px;">
    <h2>jQuery 시작 Selector</h2>
    <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
    <div>
        <ul>
            <li>list item 1</li>
            <li>list item 2</li>
            <li>list item 3</li>
            <li>list item 4</li>
            <li>list item 5</li>
        </ul>
    </div>
</body>
</html>

```

each() 메서드를 통해 "li"에 click 이벤트를 할당하고, 이벤트 발생시 이벤트가 발생한 요소의 class 를 추가 시키며, 자신을 제외한 나머지 요소에 대해서는 강조 클래스 셋팅을 제거 한다.



[선택한 아이템만 붉은색으로 강조 되는걸 확인 할 수 있다.]

이번 강좌를 끝으로 Traverse 에 대한 내용을 마치도록 하겠다.

[jQuery 강좌] 13. jQuery Core

웹 프런티어와 함께 하는 jQuery 기초강좌

13th - jQuery Core

< jQuery Core? >

지금까지 jQuery 의 셀렉터나 API 를 통한 jQuery 사용법을 알아 보았다.

jQuery 의 기본은 selector 라고 말씀을 드렸지만, 이번 강좌는 제목부터가 "CORE" 이다. 기본 중에 기본이라고 해야 할까요?

강좌를 시작하면서 언급했던 내용으로 jQuery 의 약자로 "\$"를 사용한다고 말씀 드렸다.

jQuery 소스를 확인 해 보시면 jQuery 문자 대신에 "\$"를 사용할 수 있도록 설정하는 부분을 어렵지 않게 찾을 수 있다. (소스코드 마지막에 위치한다.)

첫 번째로 jQuery(=\$)를 사용하여, jQuery object 를 생성하는 방법에 대해 알아 보도록 하겠다. 셀렉터 강좌 처음에 셀렉터를 통해 가져온 값은 jQuery object(개체)라는 말은 언급 했었다. jQuery 개체는 셀렉터를 통해서 가져오거나, 개발자가 () 안에 특별한 인자 값을 넣는 것으로도 개체를 만들 수 있다.

\$(html)

\$("span tag 입니다.")

\$("")

위와 같은 형태로 사용자의 마음대로 jQuery 개체를 만들 수 있다.

jQuery()

Contents:Categories: [Core](#)

jQuery(selector, [context])

jQuery(selector, [context])

jQuery(element)

jQuery(elementArray)

jQuery(jQuery object)

jQuery()

jQuery(html, [ownerDocument])

jQuery(html, [ownerDocument])

jQuery(html, props)

jQuery(callback)

jQuery(callback)

`$()`는 `()`안에 들어가는 표현 식에 따라 서로 전혀 다른 기능을 하고 있다.

셀렉터 문법을 사용하면 `$`는 셀렉터가 되며, 표현 식에 함수가 들어갈 경우 `document.onload` 와 대응이 되는 역할을 하게 되며, 많은 예제에서 사용했던 `$(document).ready(function() { });` 의 함축적인 표현으로 `$(function() { });`를 사용 한다.

jQuery 개체는 Javascript 의 개체와는 전혀 다르며, 사용법 또한 많은 부분에서 차이가 난다.

간단한 예로 `$("#id")`와 `document.getElementById("id")`를 통해 가져온 개체를 `alert()` 메서드를 통해 알아보면 `$("#id")`의 경우 object 형식으로, 후자의 경우 HTML Element 로 나타나는 걸 확인할 수 있다.

1. jQuery.noConflict()

예약어로 사용되는 `"$"`문자를 다른 라이브러리와 충돌이 나지 않게 하는 역할을 한다.

대표적으로 Prototype 이라는 라이브러리가 `"$"`를 사용하고 있으며, Prototype 에서 jQuery 로 옮겨가고 있는 사이트나, 개발 시 꼭 Prototype 과 혼용해서 사용하는 경우가 있다면 `noConflict()` 메서드를 통해 jQuery 의 `"$"`를 무력화 시키거나 다른 약어로 변경 할 수 있다.

우선 jQuery 의 `noConflict` 메서드의 내부를 보면 `"$"`를 `_`\$ 또는 `_jQuery` 로 변경하는 것을 확인할 수 있다.

```
noConflict: function( deep ) {  
    window.$ = _$;  
    if ( deep ) {  
        window.jQuery = _jQuery;  
    }  
    return jQuery;  
},
```

2. "\$" "X"로 변경하기

```
var X = jQuery.noConflict();
```

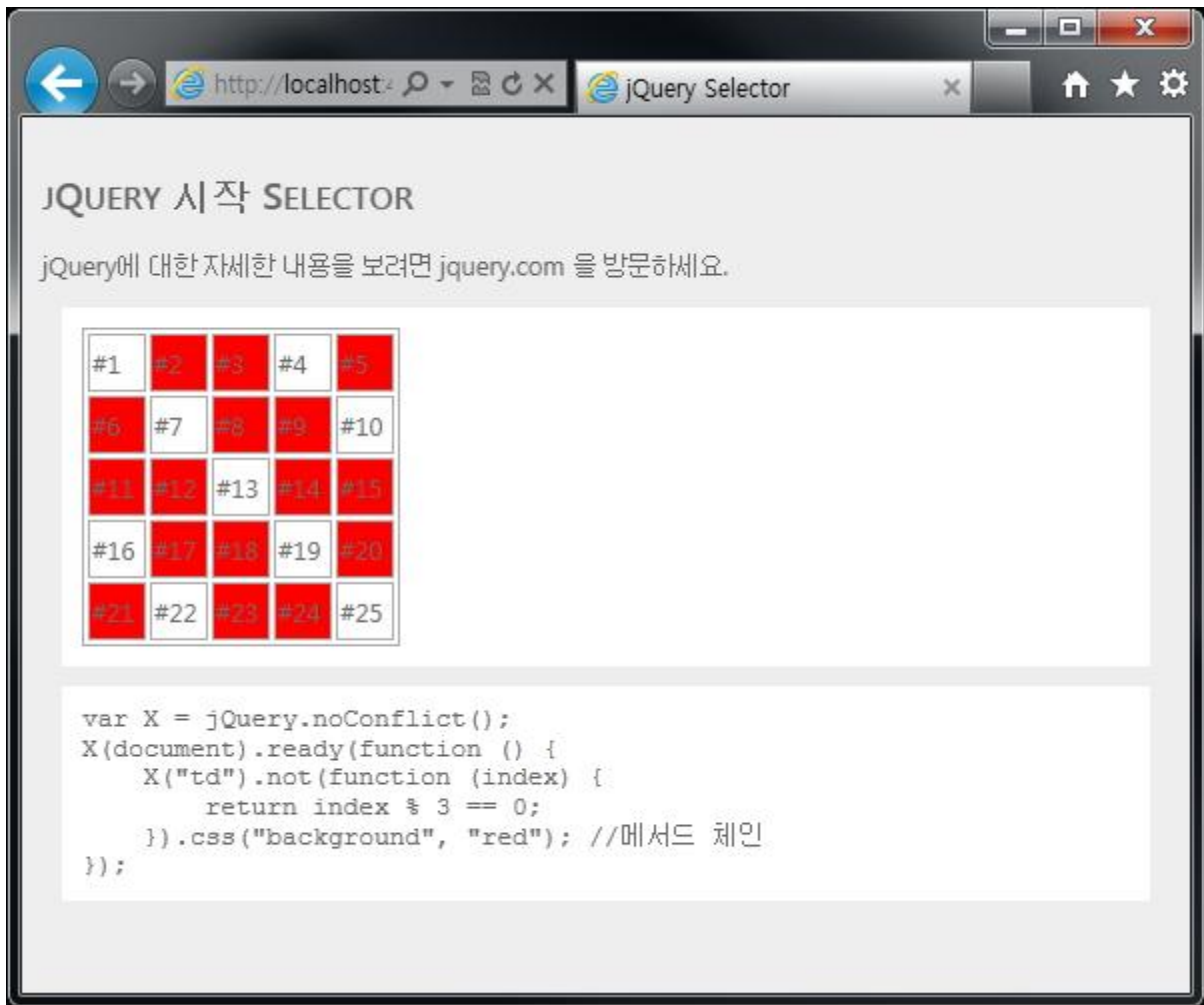
단순히 변수를 하나 지정하고, `noConflict()` 메서드를 실행 하기만 된다.

```
<html xmlns="http://www.w3.org/1999/xhtml">  
<head>  
  <title>jQuery Selector</title>  
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />  
  <style>
```

```

    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
</style>
<script src="http://code.jquery.com/jquery-latest.js"></script>
<script type="text/javascript">
    var X = jQuery.noConflict();
    X(document).ready(function () {
        X("td").not(function (index) {
            return index % 3 == 0;
        }).css("background", "red");
    });
</script>
</head>
<body style="padding:10px;">
    <h2>jQuery 시작 Selector</h2>
    <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
    <div>
        <table>
            <tr>
                <td>#1</td><td>#2</td><td>#3</td><td>#4</td><td>#5</td>
            </tr>
            <tr>
                <td>#6</td><td>#7</td><td>#8</td><td>#9</td><td>#10</td>
            </tr>
            <tr>
                <td>#11</td><td>#12</td><td>#13</td><td>#14</td><td>#15</td>
            </tr>
            <tr>
                <td>#16</td><td>#17</td><td>#18</td><td>#19</td><td>#20</td>
            </tr>
            <tr>
                <td>#21</td><td>#22</td><td>#23</td><td>#24</td><td>#25</td>
            </tr>
        </table>
    </div>
</body>
</html>

```



[$\$ \Rightarrow X$ 로 변경 후 전과 다름없이 동작에는 전혀 문제가 발생하지 않았다.]

지금까지 jQuery Core 에 대해 알아 보았으며, 다음 시간에는 CSS(Style)에 대해 알아보는 시간을 갖도록 하겠다.

[jQuery 강좌] 14. jQuery CSS - 스타일 관련 메서드에 대하여

웹 프런티어와 함께하는 jQuery 기초강좌

14th - jQuery CSS(스타일 관련 메서드 알아보기)

<HTML 문서의 스타일 제어하기>

jQuery 에서는 역시나 굉장히 많은 API 를 제공 하고 있다만, 여기서는 자주 사용되는 API 를 중심으로 설명을 드리고, 필요에 따라서 부수적인 API 에 대해 이야기 하도록 하겠다.

우선 스타일을 직접적으로 제어하는데 사용되는 API 를 알아 보도록 하겠다.

1. .css()



적용된 스타일을 가져오거나, 새로운 스타일을 적용한다.

`$("#div").css("background-color")`의 경우 해당 선택된 개체의 배경 색상을 가져옵니다.

기본적인 표현 식은 스타일에서 사용되는 구문을 동일하게 사용하시면 된다.

새로운 스타일을 적용하고 싶을 경우에는 `css(propertyName, value)` 메서드를 사용하시면 된다. 앞에서 진행한 예제를 보시면 대부분의 경우 이 메서드를 사용해서 강조하고 있다. 사용법 또한 스타일 관련 문법만 알고 계시면 어렵지 않게 활용 가능하다.

붉은색 배경을 넣고 싶을 경우 `$("#div").css("background-color", "red");` propertyName 에는 적용할 스타일의 이름과 value 에는 원하는 값을 넣기만 하면 된다.

`$("#div").css("background-color", function(index, value) { });` 와 같은 형식을 통해 자신이 원하는 기능을 확장하여 사용할 수 있다. `function` 의 파라미터를 살펴보면 `index` 와 `value` 가 존재하고 있다. `Index` 의 경우 선택한 개체의 순서이며, `value` 는 해당 개체의 현재 스타일 속성값이다. 이 형식을 사용하면, 한번에 개체의 스타일 값을 확인하고 변경할 수 있는 큰 장점이 있다.

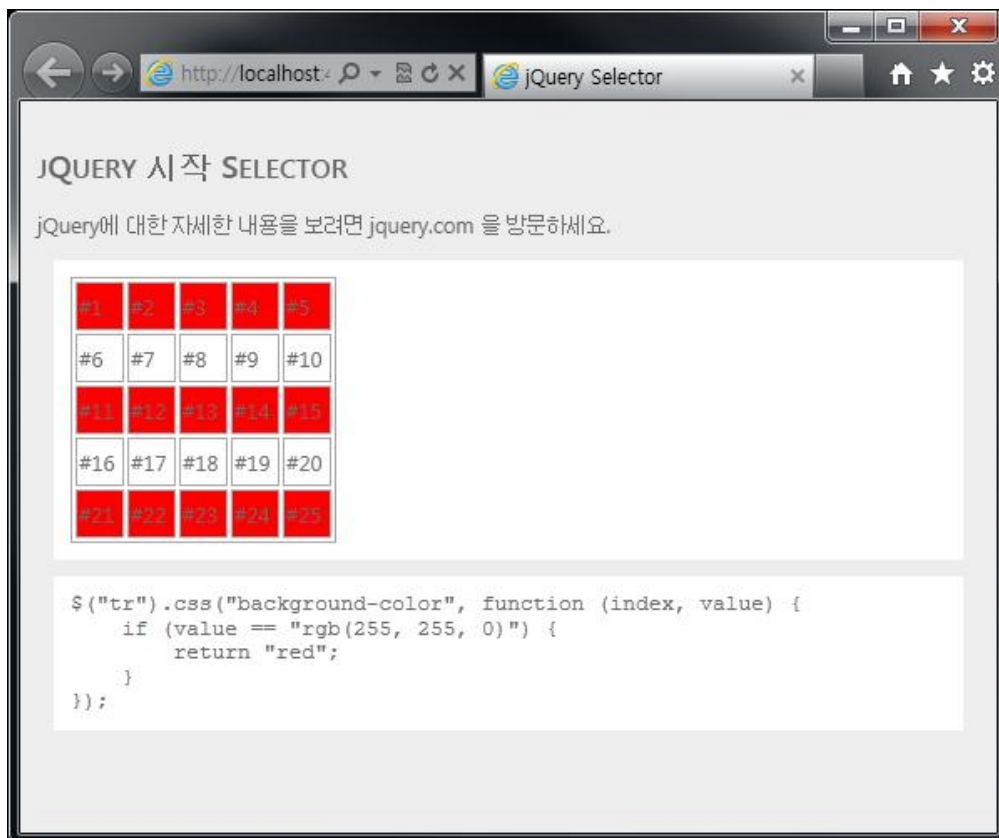
다음 예제를 통해 간단히 알아 보도록 하겠다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("#tr").css("background-color", function (index, value) {
        if (value == "rgb(255, 255, 0)") {
          return "red";
        }
      });
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <table>
      <tr style="background-color:rgb(255,255,0)">
        <td>#1</td> <td>#2</td> <td>#3</td> <td>#4</td> <td>#5</td>
      </tr>
      <tr>
        <td>#6</td> <td>#7</td> <td>#8</td> <td>#9</td> <td>#10</td>
```

```

</tr>
<tr style="background-color:rgb(255,255,0)">
    <td>#11</td><td>#12</td><td>#13</td><td>#14</td><td>#15</td>
</tr>
<tr>
    <td>#16</td><td>#17</td><td>#18</td><td>#19</td><td>#20</td>
</tr>
<tr style="background-color:rgb(255,255,0)">
    <td>#21</td><td>#22</td><td>#23</td><td>#24</td><td>#25</td>
</tr>
</table>
</div>
</body>
</html>

```



HTML 에서 노랑색으로 강조한 부분이 붉은 색으로 변경 되었다. Function 의 경우 파라미터가 하나이면 무조건 해당 개체의 index 가 반환되니 이점 참고 하시기 바란다. Index, value 에 대한 값이 필요 없을 경우는 파라미터를 사용하지 않으셔도 무방하다.

2. .hasClass(className)

.hasClass(className)

Returns: Boolean

Description: Determine whether any of the matched elements are assigned the given class.

.hasClass(className)

version added: [1.2](#)

className The class name to search for.

선택된 개체에 className 과 동일한 클래스가 존재 하는지에 대한 결과를 true/false 로 반환한다.

```
<div class="myClass">Content</div>
```

\$("#div").hasClass("myClass")의 경우 "true"를 반환한다.

\$("#div").hasClass("noClass")의 경우 "false"를 반환한다.

3. .addClass()

.addClass(className)

Returns: jQuery

Description: Adds the specified class(es) to each of the set of matched elements.

.addClass(className)

version added: [1.0](#)

className One or more class names to be added to the class attribute of each matched element.

.addClass(function(index, currentClass))

version added: [1.4](#)

function(index, currentClass) A function returning one or more space-separated class names to be added. Receives the index position of the element in the set and the old class value as arguments.

새로운 클래스를 추가하는 메서드이다.

\$("#div").addClass("className") 선택터를 통해 가져온 개체에 className 이라는 클래스를 추가한다.

이 부분은 더 드릴만한 설명이 없으므로 function 파라미터를 통한 클래스 제어에 대해 자세히 알아 보도록 하겠다.

앞서 설명 드린 "css(propertyName, function(index, value) {})"와 비슷한 형식을 보이는 걸로 봐서는 하는 역할도 비슷할 것이라 생각 할 수 있다.

css() 메서드의 경우 스타일에 대한 속성값을 제어하고, addClass()의 경우 속성값이 아닌 className 에 대한 부분만 제어 한다는 차이만 있을 뿐 거의 동일한 동작을 한다. function 의 파라미터 또한 css() 메서드에서 설명 드린 부분과 동일하다. Index 는 개체의 번호를 currentClass 는 value 와 같이 현재 개체의 className 을 반환한다.

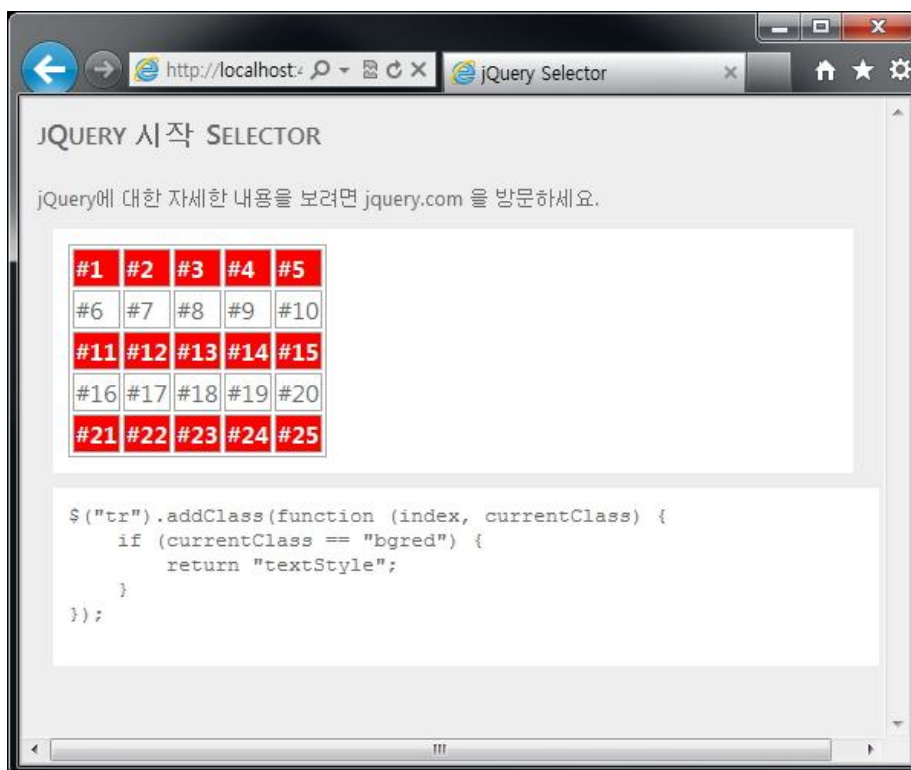
다음 예제를 통해 addClass(function(index, currentClass))의 사용법을 알아 보도록 하겠다.

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
  <title>jQuery</title>
  <link href="../../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
    .textStyle { font-weight:bold; color:White; }
    .bgred { background-color : Red; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js" ></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("tr").addClass(function (index, currentClass) {
        if (currentClass == "bgred") {
          return "textStyle";
        }
      });
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <table>
      <tr class="bgred">
        <td>#1</td> <td>#2</td> <td>#3</td> <td>#4</td> <td>#5</td>
```

```

</tr>
<tr>
    <td>#6</td><td>#7</td><td>#8</td><td>#9</td><td>#10</td>
</tr>
<tr class="bgred">
    <td>#11</td><td>#12</td><td>#13</td><td>#14</td><td>#15</td>
</tr>
<tr>
    <td>#16</td><td>#17</td><td>#18</td><td>#19</td><td>#20</td>
</tr>
<tr class="bgred">
    <td>#21</td><td>#22</td><td>#23</td><td>#24</td><td>#25</td>
</tr>
</table>
</div>
</body>
</html>

```



선택한 개체 중에 클래스 명이 "bgred"일 경우 해당 개체에 "textStyle"이라는 클래스를 추가 하였으며, 결과 화면에서 보듯이 하얀색의 글자와 bold 속성이 적용 되었다.

4. .removeClass()

.removeClass([className])

Returns: [jQuery](#)

Description: Remove a single class, multiple classes, or all classes from each element in the set of matched elements.

.removeClass([className])

version added: [1.0](#)

className One or more space-separated classes to be removed from the class attribute of each matched element.

.removeClass(function(index, class))

version added: [1.4](#)

function(index, class) A function returning one or more space-separated class names to be removed. Receives the index position of the element in the set and the old class value as arguments.

.addClass()와는 반대로 해당 className 을 삭제 한다.

.addClass()와 동작하는 형태와 사용법은 완전히 동일 하며, 반대로 선언한 클래스를 삭제 한다는 것만 알아 두시면 된다.

5. .toggleClass()

.toggleClass(className)

Returns: [jQuery](#)

Description: Add or remove one or more classes from each element in the set of matched elements, depending on either the class's presence or the value of the switch argument.

.toggleClass(className)

version added: [1.0](#)

className One or more class names (separated by spaces) to be toggled for each element in the matched set.

.toggleClass(className, switch)

version added: [1.3](#)

className One or more class names (separated by spaces) to be toggled for each element in the matched set.

switch A Boolean (not just truthy/falsy) value to determine whether the class should be added or removed.

.toggleClass(function(index, class), [switch])

version added: [1.4](#)

function(index, class) A function that returns class names to be toggled in the class attribute of each element in the matched set. Receives the index position of the element in the set and the old class value as arguments.

switch A boolean value to determine whether the class should be added or removed.

선택한 개체를 마치 스위치처럼 켜다(removeClass()), 껀다(addClass())하는 기능을 하는 메서드이다.

```
$("#div").toggleClass("myClass")
```

선택한 개체에 myClass 가 적용 시 myClass 를 삭제하고, 적용되지 않은 경우에는 myClass 를 적용하는 역할을 한다.

toggleClass(className, switch)의 두 번째 파라미터인 switch(Boolean)값을 통해 사용자가 toggle 상태를 지정할 수 도 있다. true 라면 addClass()와 같은 효과를 반대로 false 라면 removeClass()와 같은 효과를 나타낸다.

이 메서드 역시 function 을 통한 확장된 제어가 가능하다.

모양을 보아 하니 역시나 이전 메서드에서 설명 드린 부분과 동일한 형태를 취하고 있는걸 확인할 수 있다.

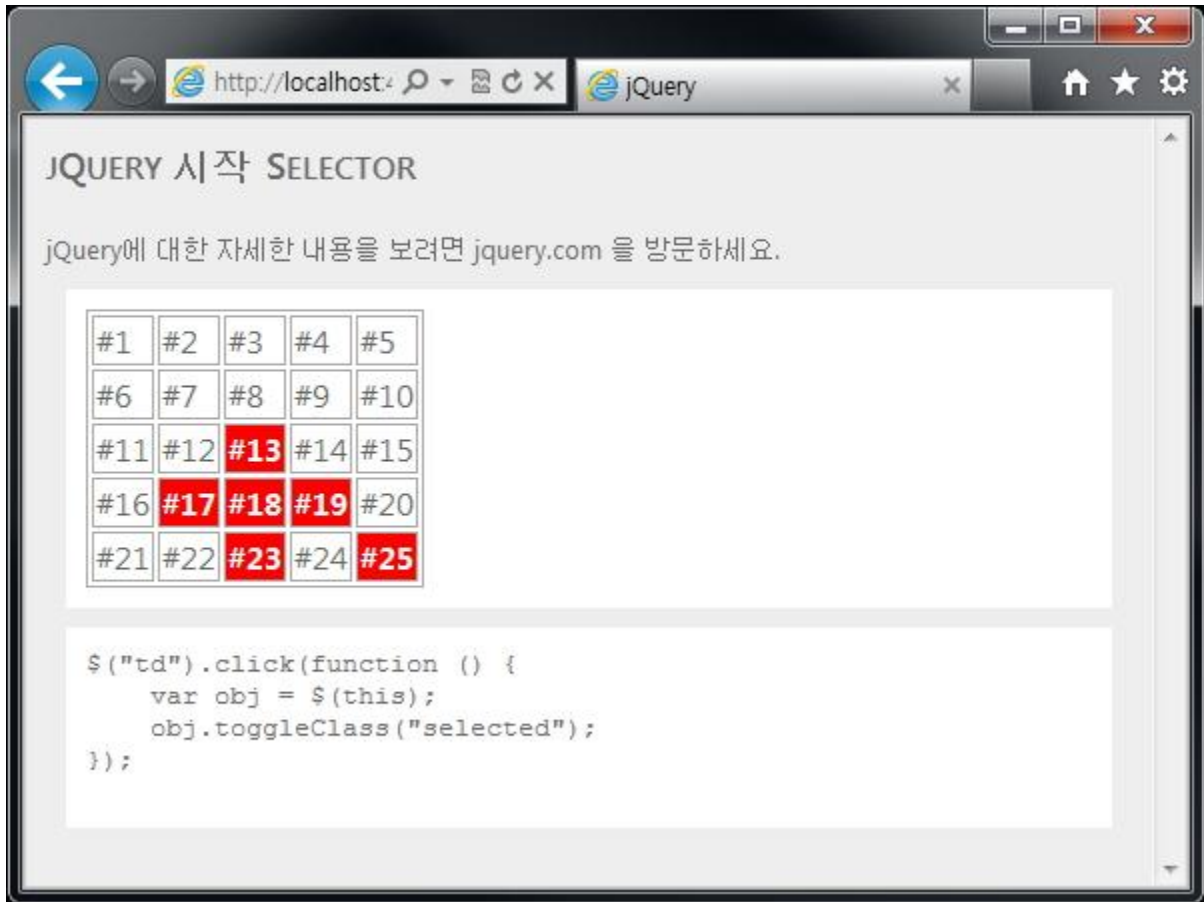
아마도 function 의 역할은 toggle 할 클래스명의 리턴이 되겠다. toggleClass()를 사용하여 목록에서 사용자가 선택한 항목을 강조하는 예제 이다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
    .selected { font-weight:bold; color:White; background-color : Red;}
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("#td").click(function () {
        var obj = $(this);
        obj.toggleClass("selected");
      });
    });
  </script>
</head>
```

```

<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <table>
      <tr>
        <td>#1</td><td>#2</td><td>#3</td><td>#4</td><td>#5</td>
      </tr>
      <tr>
        <td>#6</td><td>#7</td><td>#8</td><td>#9</td><td>#10</td>
      </tr>
      <tr>
        <td>#11</td><td>#12</td><td>#13</td><td>#14</td><td>#15</td>
      </tr>
      <tr>
        <td>#16</td><td>#17</td><td>#18</td><td>#19</td><td>#20</td>
      </tr>
      <tr>
        <td>#21</td><td>#22</td><td>#23</td><td>#24</td><td>#25</td>
      </tr>
    </table>
  </div>
</body>
</html>

```



예제 소스를 복사하셔서 브라우저로 확인을 해 보시면 한번 클릭하면 toggleClass()로 인해서 selected 클래스가 추가되면 이미 추가된 항목을 선택하면 selected 클래스가 삭제되어 이전의 모습으로 돌아 가는 모습을 확인 할 수 있다.

6. .position(), .width(), .height()

단어만 봐도 어떠한 역할을 하는지 알 수 있을 것으로 보인다. 우선 position()에 대해 살펴보면 단어의 뜻 그대로 선택된 개체의 위치값을 가지고 있는 개체를 반환한다. 반환되는 개체는 top, left 라는 속성값을 가지고 있으며, 다음과 같이 사용된다.

```

var p = $("div").position();
var pTop = p.top;
var pLeft = p.left;

```

```

var p = $("div").position();
var pTop = p.top;
var pLeft = p.left;

```

7. width() 그리고 height()

선택된 개체의 가로와 세로길이를 반환하거나 설정하는 메서드이다. 아무런 파라미터가 없을 경우 해당 값을 반환하며, 파라미터를 넣을 경우 각각의 값이 파라미터의 값으로 대체된다.

개체의 가로,세로 길이를 구할 때

```
var width = $("div").width();  
var height = $("div").height();
```

개체의 가로, 세로 길이를 지정 할 때

```
$("div").width(100);  
$("div").height(100);
```

개체의 가로,세로 길이를 구할 때

```
var width = $("div").width();  
var height = $("div").height();
```

개체의 가로, 세로 길이를 지정 할 때

```
$("div").width(100);  
$("div").height(100);
```


[jQuery 강좌] 15. jQuery Attribute - 요소의 속성 관련 메서드에 대하여

웹 프런티어와 함께하는 jQuery 기초강좌

15th - jQuery Attribute(속성 관련 메서드 알아보기)

<HTML 문서의 속성 제어하기>

이번 시간에는 jQuery 를 이용하여, 개체 속성을 제어하는 방법에 대해 알아 보도록 하겠다.

HTML 요소에서 이름을 제외한 나머지 요소는 모두 속성이 될 수 있다.

즉 이라는 요소가 있다면 src, alt 는 모두 요소가 되며, 정의되지 않은 값을 사용자 임의로 넣어 사용 가능하다. (사용자 정의 속성은 특별한 오류를 나타내지는 않으나, 간혹 브라우저 특성으로 인해 발생하는 오류가 있으니, 꼭 필요한 곳에만 사용을 하시기 바란다.)

1. .attr()

.attr()

Contents: Categories: [Attributes](#) | [Manipulation](#) > [General Attributes](#)

- attr(attributeName)
 .attr(attributeName)
- attr(attributeName, value)
 .attr(attributeName, value)
 .attr(map)
 .attr(attributeName, function(index, attr))

동작하는 모습도 클래스에서 속성으로만 변경되었지 동일하다.

요소의 속성값을 가져오기 위해서는 `.attr(attributeName)`을 사용하며, 속성값을 설정하기 위해서는 `.attr(attributeName, value)`를 사용하면 된다.

```

var imgSrc = $("img").attr("src"); //img 요소의 src 속성을 가져옵니다.
$("img").attr("src", "img/old.gif"); //img 요소의 src 속성을 "img/old.gif"로 변경한다.
```

속성값을 설정 할 경우 해당 속성이 존재하면 기존 속성에 덮어쓰기를 하며, 속성이 존재 하지 않을 경우에는 새로운 속성을 할당 하게 된다. Callback 메서드를 통한 속성값 제어는 이미 `css()` 메서드에서 자세히 설명 했던 부분이므로 넘어 가도록 하겠다.

일반적인 자바스크립트를 통해서도 속성에 접근이 가능하다. 하지만 브라우저마다 지원하는 방식이 다르기 때문에 속성값을 사용할 경우에는 jQuery 의 `attr()` 메서드를 사용하시는 것이 좋다.

2. `.removeAttr()`

`.removeAttr(attributeName)`

Returns: [jQuery](#)

Description: Remove an attribute from each element in the set of matched elements.

`.removeAttr(attributeName)`

version added: [1.0](#)

attributeName An attribute to remove.

`attr()` 메서드를 통해 속성값을 추가 했으니, 이제 삭제를 시키는 방법에 대해 알아 보도록 하겠다.

`attr("attributeName", "");` 와 같은 방법으로 해당 속성값을 삭제(?) 할 수도 있다만, FireBug 와 같은 툴을 사용해 확인을 해 보면 값만 빈 값으로 변경이 된 걸 확인 할 수 있다. `removeAttr()` 메서드를 사용시 해당 속성에 대한 이름과 값이 모두 제거되며, 역시나, 자바스크립를 통한 속성값 삭제 보다는 해당 메서드를 사용하길 추천 한다.

3. val()



선택된 요소의 값을 가져옵니다. jQuery 에서 셀렉터 다음으로 가장 많이 사용되는 메서드가 아닌가 싶다. Input, select 등의 요소에서 value 에 해당하는 값을 가져오거나 설정하며, 자바스크립트의 document.getElementById("txtName").value; 와 동일한 역할을 한다.

```
<input type="text" name="txtName" id="txtName" />
var name = $("#txtName").val(); //입력된 값 가져오기
$("#txtName").val("val 를 이용한 새로운 값 입력"); //값 입력하기
```

Callback 함수의 경우 역시 앞서 설명한 부분과 동작이 완전 동일하여, 자세한 설명을 생략한다.

4. .text() / .html()

선택된 개체의 text 또는 html 을 가져오거나 사용자가 지정하는 값으로 설정하는 역할을 하는 메서드 이다.



.html()

Contents:

Categories: [Attributes](#) | [Manipulation](#) > [DOM Insertion, Inside](#)

html()

.html()

html(htmlString)

.html(htmlString)

.html(function(index, oldhtml))

두 개의 메서드 역시 앞서 설명 드린 부분과 사용방법이 크게 다르지 않다. 파라미터 인자의 유무에 따라 값을 설정 하거나, 해당 값을 가져오거나 할 수 있다.

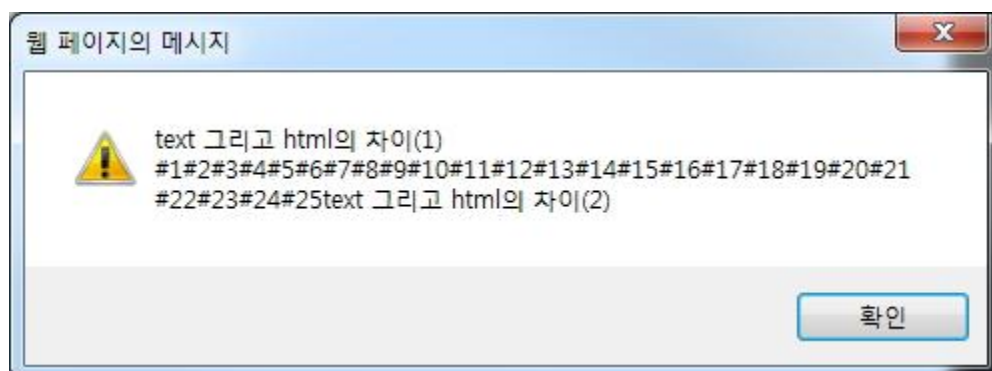
사용 방법에 대해 설명 보다는 text 와 html 에 대한 차이점에 대한 예제를 끝으로 이번 강좌를 마칠까 한다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      var text = $("div").text();
      var html = $("div").html();
      alert(text);
      alert(html);
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
```

```

<div>
text 그리고 html??? 차이(1)
<table>
  <tr>
    <td>#1</td><td>#2</td><td>#3</td><td>#4</td><td>#5</td>
  </tr>
  <tr>
    <td>#6</td><td>#7</td><td>#8</td><td>#9</td><td>#10</td>
  </tr>
  <tr>
    <td>#11</td><td>#12</td><td>#13</td><td>#14</td><td>#15</td>
  </tr>
  <tr>
    <td>#16</td><td>#17</td><td>#18</td><td>#19</td><td>#20</td>
  </tr>
  <tr>
    <td>#21</td><td>#22</td><td>#23</td><td>#24</td><td>#25</td>
  </tr>
</table>
text 그리고 html??? 차이(2)
</div>
</body>
</html>

```



```
[$("div").text()]
```



[\$("div").html()]

text()의 경우 html 을 제외한 순수 text 만 가져오는 반면에 html()의 경우는 text 를 포함한 모든 html 의 내용까지 가져 오는걸 확인 할 수 있다.

[jQuery 강좌] 16. jQuery Form API - 폼 지원 메서드에 대하여

16th - jQuery FORM API

<폼(form)을 위한 API>

이번 시간에는 Form 에 대한 이야기를 진행한다.

웹 프로그램에서 Form 은 매우 중요한 요소이다. 사용자의 데이터를 서버로 전송을 하는 막중한 임무를 띄고 있다.

ASP.NET 에서는 단일 폼의 형태로 구성이 되어 있었으며, 많은 부분이 Form 에 대해 크게 신경을 쓰지 않아도 되는 부분이 있었다. 하지만 ASP.NET MVC 가 나오면서 상황은 바뀌었다.

ASP, PHP 처럼 Form 을 직접적으로 컨트롤 하고, 다중 폼을 이용한 프로그래밍이 쉬워졌기 때문이다.

물론 ASP.NET 도 Form 을 통한 데이터 전달이나 이벤트 발생을 하고 있지만 대부분 개발자가 크게 신경을 써야 하는 부분이 적었던 반면에 ASP.NET MVC 의 경우는 ASP 처럼 많은 부분에 Form 에 대해 신경을 써야 한다.

ASP.NET 이 단일 폼만을 지원하는 것은 아니며, 흔히 말하는 폼수를 사용하면 다중 폼도 아무런 문제 없이 사용이 가능하다.

Form 하면 꼭 등장하는 것이 바로 submit() 이다. "submit"은 form 에 있는 데이터를 사용자가 지정한 곳에 보내는 역할을 하게 된다. jQuery 에서 지원하는 .submit() 동일한 역할을 하고 있으며 다음과 같은 형태를 가지고 있다.

.submit(handler(eventObject))

Returns: jQuery

Description: Bind an event handler to the "submit" JavaScript event, or trigger that event on an element.

.submit(handler(eventObject))

version added: [1.0](#)

handler(eventObject) A function to execute each time the event is triggered.

.submit([eventData], handler(eventObject))

version added: [1.4.3](#)

eventData A map of data that will be passed to the event handler.

handler(eventObject) A function to execute each time the event is triggered.

.submit()

version added: [1.0](#)

.submit(handler(eventObject)) 를 살펴 보면, 뒤에 handler 라는 것이 존재를 하고 있다. .submit()을 호출하게 되면 정의된 handler 를 실행하고 리턴 받은 Boolean 값을 통해 서버로 데이터 전송 유/무를 결정 하게 된다.

일반적으로 아래와 같은 방법을 사용을 하고 있다.

```
function sendSubmit() {  
    if (document.getElementById("txtName").value == "")  
    {  
        alert("이름을 입력 하세요.");  
        return;  
    }  
    else  
    {  
        document.form.submit();  
    }  
}
```

위 방법을 jQuery 에서 제공하는 .submit(handler);를 사용하게 되면 다음과 같다.

```
$('#form').submit(function() {  
    if($('#txtName').val() == "") {  
        alert("이름을 입력 하세요.");  
        return false;  
    }  
    else {  
        return true;  
    }  
});
```


약간의 형태만 달라졌지, 기본적인 개념은 기존과 동일하기 때문에 큰 어려움 없이 바로 사용 가능하다.

서버간의 데이터 통신을 위해 form 에 있는 데이터를 직렬화 하거나 url 을 통해 전달을 하는 경우가 있다.

특히나 Ajax 를 이용한 비동기 통신에서는 데이터의 직렬화를 상당히 많이 사용하고 있다. 이럴 때를 위해서 jQuery 에서는 form 데이터를 쉽게 직렬화 시킬 수 있는 메서드를 제공하고 있다.

바로 지금부터 설명드릴 아래의 두 메서드이다.

.serialize()
Encode a set of form elements as a string for submission.
[Helper Functions](#)

.serializeArray()
Encode a set of form elements as an array of names and values.
[Helper Functions](#)

메서드 이름부터가 직렬화이다. 두 메서드의 역할은 동일하며, 반환되는 값에서 약간의 차이를 보이고 있다.

.serialize()의 경우 선택한 폼의 값을 "a=1&b=2&c=3"등의 형태로 반환하며, .serializeArray() 메서드의 경우 jQuery 배열 개체로 반환한다.

쿼리스트링을 통한 Ajax 관련 데이터를 만들 때는 .serialize()를 사용하시면 된다.

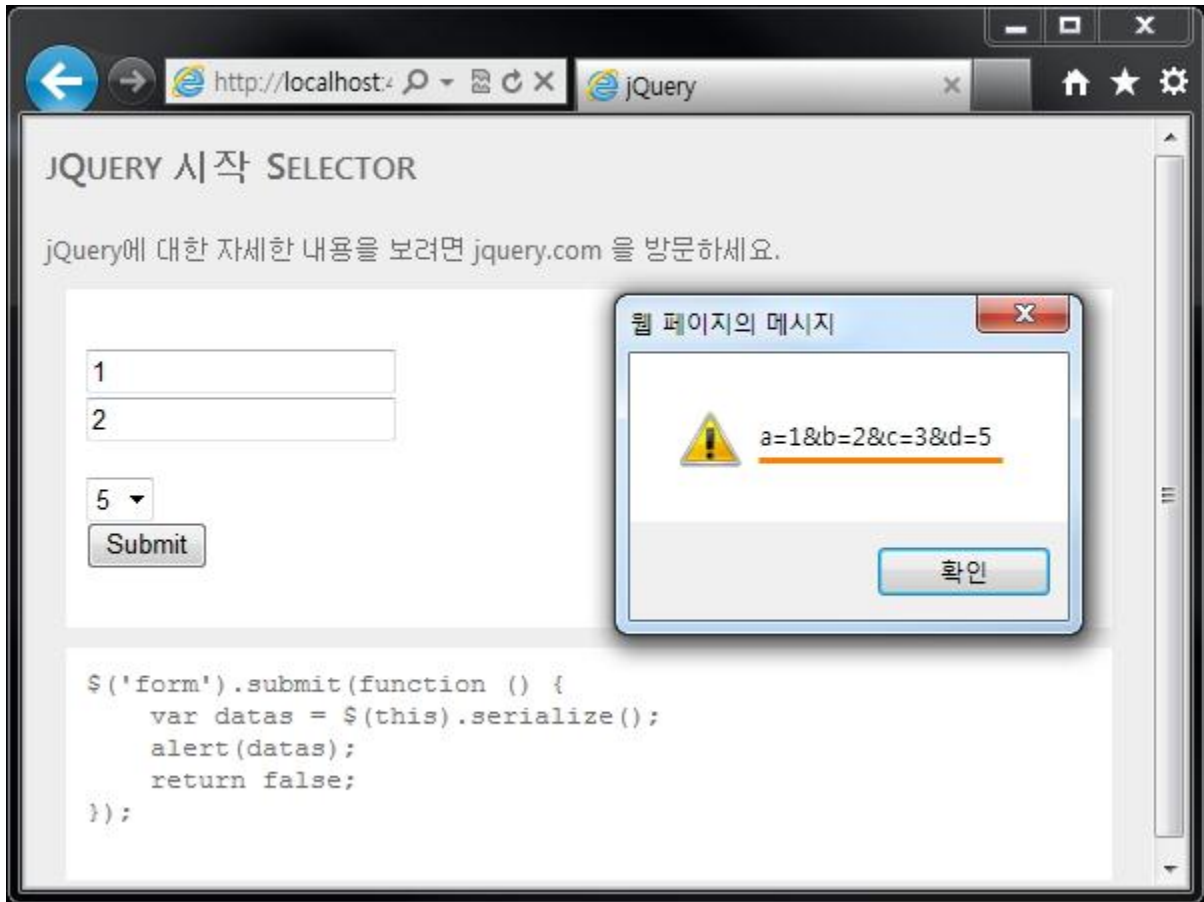
다음 예제를 통해 .serialize() 메서드 동작을 확인해 보도록 하겠다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $('form').submit(function () {
```

```

        var datas = $(this).serialize();
        alert(datas);
        return false;
    });
});
</script>
</head>
<body style="padding:10px;">
    <h2>jQuery 시작 Selector</h2>
    <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
    <div>
        <form>
            <input type="text" name="a" value="1" id="a" /> <br />
            <input type="text" name="b" value="2" id="b" /> <br />
            <input type="hidden" name="c" value="3" id="c" /> <br />
            <select name="d">
                <option value="5" selected="selected">5</option>
                <option value="6">6</option>
                <option value="7">7</option>
            </select> <br />
            <input type="submit" name="f" value="Submit" id="f" />
        </form>
    </div>
</body>
</html>

```



[폼 데이터에 대한 값이 serialize 된 결과를 확인 할 수 있다.]

이제 더 이상 쿼리스트링을 만들기 위해서 노가다 작업을 하지 않아도 된다는 아주 큰 희소식이다.

물론 일반적인 Form 데이터를 전송하게 되면 자동으로 직렬화 형태로 값이 전달이 되지만, Ajax 와 같은 특수한 상황에서는 정말로 편리한 메소드가 아닌가 싶다.

[jQuery 강좌] 17. jQuery Event - 이벤트 지원 메서드

17th - jQuery Events 이벤트 지원 메서드 살펴보기

jQuery에서는 사용자가 발생시키는 이벤트를 쉽고, 간단하게 핸들링 할 수 있는 메소드를 지원하고 있다. 마우스의 클릭과 키보드의 입력은 물론이고 심지어 더블클릭과 마우스 휠에 대한 이벤트를 쉽게 처리 할 수 있다.

jQuery에서 제공하는 이벤트 관련 메소드의 종류는 정말로 너무나 많다. 그 만큼 개발자가 많은 이벤트를 쉽게 이용할 수 있고 좀더 나은 UX를 만들 수 있다는 큰 장점이 있다. 이벤트 중에서 이번 시간을 통해 알아볼 내용은 사용자의 입력이나 브라우저의 상태가 변경되었을 경우 발생하는 이벤트 메서드이다.

이벤트 종류(형식)	이벤트 설명
.blur()	요소에서 포커스를 잃을 경우에 발생하는 이벤트이다.
.change()	<input />, <textarea />, <select /> 요소의 값 변경시 발생하는 이벤트이다.
.click()	마우스 클릭 시 발생하는 이벤트이다.
.dblclick()	마우스를 더블클릭 했을 경우 발생하는 이벤트이다.
.focus()	요소에 포커스 되었을 때 발생하는 이벤트이다.
.hover()	마우스가 요소 위에 위치했을 때 발생하는 이벤트이다.
.keydown()	키 입력 시 발생하는 이벤트이며, 모든 키에 대해 적용이 된다.
.keypress()	keydown 이벤트와 동일하게 키 입력 시 발생이 되지만 enter, tab 등의 특수키에는 이벤트가 발생되지 않는다.
.keyup()	키 입력 후 발생하는 이벤트이다.
.mousedown()	마우스 클릭 시 발생하는 이벤트이다.
.mouseenter()	선택한 요소의 영역에 마우스가 위치했을 때 발생하는 이벤트이다.
.mouseleave()	선택한 요소의 영역에서 마우스가 벗어 났을 때 발생하는 이벤트이다. 인터넷익스플로러에서만 발생하는 이벤트지만 jQuery는 브라우저 관계없이 사용할 수 있도록 시뮬레이터 된다.
.mouseout()	선택한 요소의 영역에서 마우스가 벗어 났을 때 발생하는 이벤트이다.
.mouseup()	마우스 클릭 후 발생하는 이벤트이다.
.ready()	DOM이 모두 준비 되었을 때 발생하는 이벤트이다.

.resize()	resize 될 경우 발생하는 이벤트이다.
.scroll()	HTML 문서가 스크롤 되었을 때 발생하는 이벤트이다.
.select()	선택한 개체를 마우스를 통해 선택 하였을 때 발생하는 이벤트이다.
.submit()	Submit 이 일어날 때 발생하는 이벤트이다. (Form 메서드 참고)
표 1. [jQuery Event 관련 메서드 종류]	

우선 jQuery 에서 지원하는 이벤트의 목록을 살펴 보면 정말(완전 대박!) 많은 이벤트를 지원하고 있는걸 확인 할 수 있다.

목록은 많으나 사용법이 대부분 동일하기 때문에 jQuery 에서 지원하고 있는 이벤트가 이런 것이 있구나 하는 정도만 알아 두시면 된다.

이벤트 중에서 가장 많이 사용하는 마우스 클릭에 대한 예제를 통해 HTML 요소에 이벤트를 어떻게 할당하고, 처리하는지 알아 보도록 하겠다.

예제를 진행하기 전에 예전 웹 프로그램 개발 시에는 HTML 과 자바스크립트를 혼용(흔히 말하는 스파게티)하여 사용을 하였으나, 웹 표준이 자리를 잡고 코드 분리를 통한 유지보수 반복적인 패턴의 코드중복을 피하기 위해 스크립트를 통해 HTML 요소와 이벤트를 맵핑하는 방식으로 개발을 하고 있다.

<이전방식>

```

<script type="text/javascript">
    function tdClick(obj) {
        alert(obj);
    }
</script>

<tr>
    <td onclick="tdClick('1');">#1</td><td>#2</td><td>#3</td><td>#4</td><td>#5</td>
</tr>
<tr>
    <td onclick="tdClick('6');">#6</td><td>#7</td><td>#8</td><td>#9</td><td>#10</td>
</tr>

```

[필요한 함수를 만들고, 해당 이벤트가 필요한 요소에 일일이 적용]

<추천방식>

```
<script type="text/javascript">
  $(document).ready(function () {
    $("td").click(function () {
      alert($(this).text());
    });
  });
</script>
<tr>
  <td>#1</td><td>#2</td><td>#3</td><td>#4</td><td>#5</td>
</tr>
<tr>
  <td>#6</td><td>#7</td><td>#8</td><td>#9</td><td>#10</td>
</tr>
```

[HTML 요소에는 아무런 작업을 하지 않았다.]

처음에는 어색하고 오히려 불편하다고 느낄 수 있으나, HTML 에서 사용하는 이벤트가 많아지고 동적으로 이벤트와 HTML 의 요소를 맵핑하는 기능 구현 시에는 정말 없어서는 안되는 반복적인 작업을 깔끔하게 해결하는 최고의 방법이다.

1. .click()

.click(handler(eventObject))

Returns: jQuery

Description: Bind an event handler to the "click" JavaScript event, or trigger that event on an element.

.click(handler(eventObject))

version added: 1.0

handler(eventObject) A function to execute each time the event is triggered.

.click([eventData], handler(eventObject))

version added: 1.4.3

eventData A map of data that will be passed to the event handler.

handler(eventObject) A function to execute each time the event is triggered.

.click()

version added: 1.0

마우스 클릭 시 동작하는 이벤트로 선택한 객체에 마우스 클릭 이벤트가 발생할 경우 해당 정의한 메서드가 실행 되며, 인자 값으로는 eventData 로 이벤트에 대한 데이터 값이 전달이 된다. `$("#div").click(function() { alert('click'); })` 이라고 정의 할 경우 HTML 문서에 존재하는 모든 DIV 요소에 "click" 이벤트를 할당하고, 정의한 메서드의 결과인 `alert('click')`을 실행하게 된다.

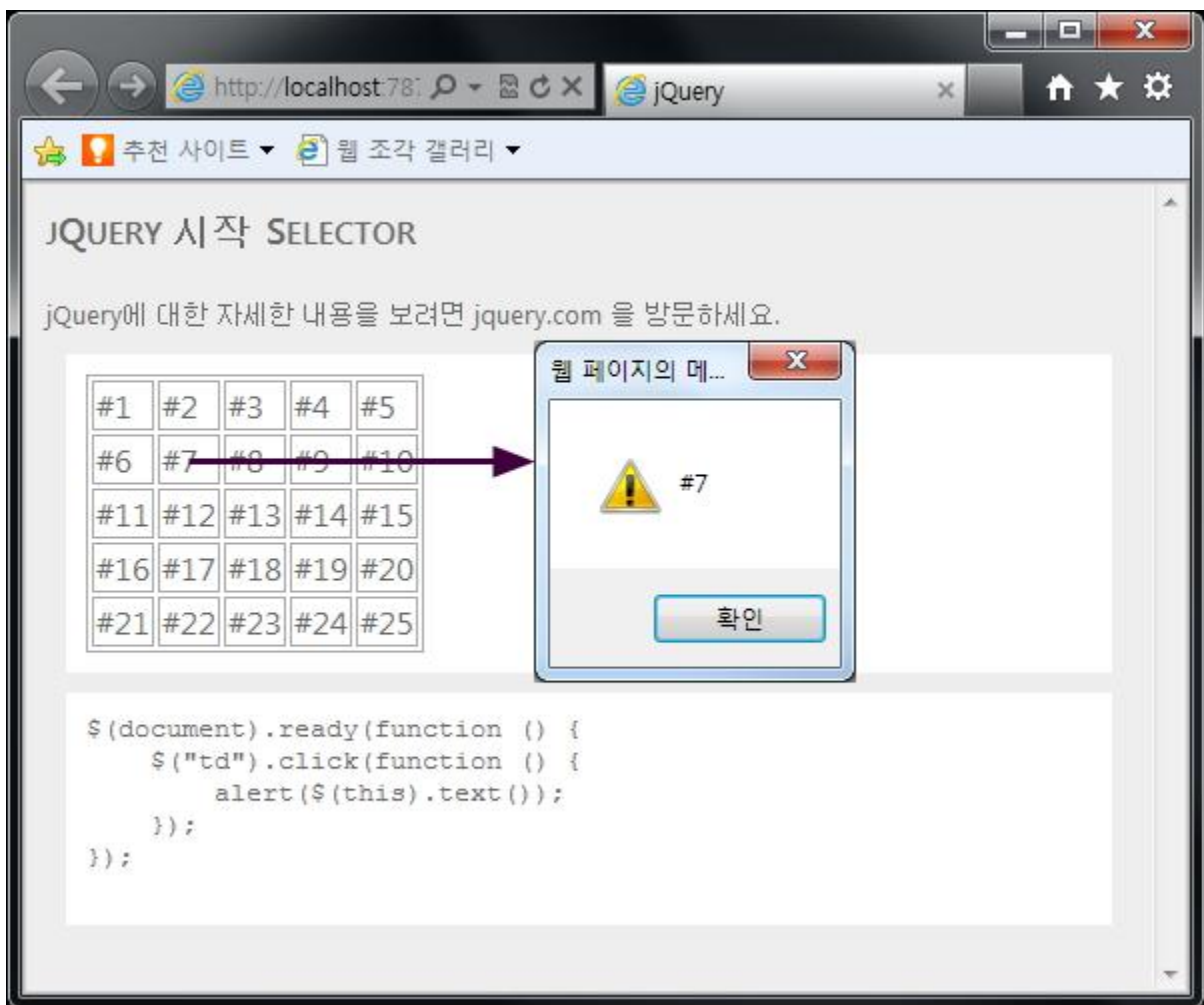
다음 예제를 통해 자세히 알아 보도록 하겠다.

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
  <title>jQuery</title>
  <link href="../../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
    .selected { font-weight:bold; color:White; background-color : Red;}
  </style>
  <script src="http://code.jquery.com/jquery-latest.js" ></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("td").click(function () {
        alert($(this).text());
      });
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <table>
      <tr>
        <td>#1</td><td>#2</td><td>#3</td><td>#4</td><td>#5</td>
      </tr>
      <tr>
        <td>#6</td><td>#7</td><td>#8</td><td>#9</td><td>#10</td>
      </tr>
      <tr>
        <td>#11</td><td>#12</td><td>#13</td><td>#14</td><td>#15</td>
      </tr>
    </table>
  </div>
</body>
</html>
```

```

</tr>
<tr>
    <td>#16</td><td>#17</td><td>#18</td><td>#19</td><td>#20</td>
</tr>
<tr>
    <td>#21</td><td>#22</td><td>#23</td><td>#24</td><td>#25</td>
</tr>
</table>
</div>
</body>
</html>

```



클릭 이벤트가 필요한 요소를 선택터를 통해 선택하고, 선택된 모든 개체에 일괄적으로 이벤트를 할당한다.

이런 패턴의 개발은 코드의 양을 줄이고, 관련 메서드를 수정할 때 번거로움이 줄어들어 든다는 이점이 있다. 또한 클라이언트로 전송되는 데이터도 적어지기 때문에 트래픽에 대한 부담도 많이 줄일 수 있다.

웹에서 흔히 사용이 되지는 않지만 jQuery 를 통해서 쉽게 구현이 가능한 마우스 더블클릭을 이용하고 싶다면, 위의 예제에서 .click() 메서드를 .dblclick()으로 변경만 하면 아무런 문제없이 사용이 가능하다.

```
<script type="text/javascript">
    $(document).ready(function () {
        $("td").dblclick(function () {
            alert($(this).text());
        });
    });
</script>
```

대부분의 이벤트 관련 메서드는 이처럼 관련 메서드 이름만 다를 뿐 사용하는 방법에 대한 차이가 거의 존재 하지 않는다.

마지막으로 키보드 입력 시 키보드의 값을 알아내는 예제를 통해 eventData 의 간단한 사용법을 알아 보도록 하겠다.

.keydown(handler(eventObject))**Returns: jQuery**

***Description:** Bind an event handler to the "keydown" JavaScript event, or trigger that event on an element.*

.keydown(handler(eventObject))**version added: 1.0**

handler(eventObject) A function to execute each time the event is triggered.

.keydown([eventData], handler(eventObject))**version added: 1.4.3**

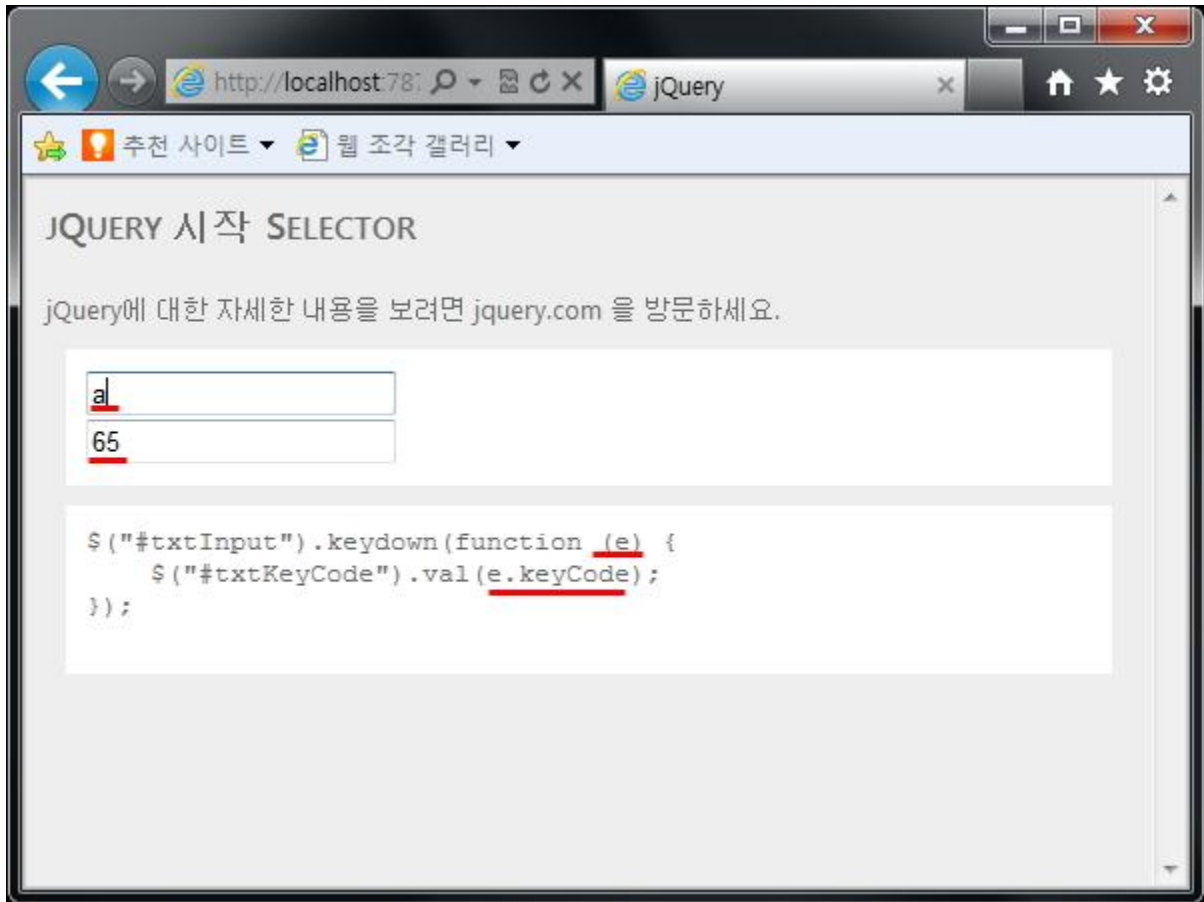
eventData A map of data that will be passed to the event handler.

handler(eventObject) A function to execute each time the event is triggered.

.keydown()**version added: 1.0**

2. **.keydown()** 이벤트를 사용하는 이유는 엔터키와 같은 특수 입력키에 대한 내용도 체크하기 위함이며, 특수키에 대한 값이 필요 없을 경우에는 **.keypress()** 를 사용하시면 된다.

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
  <title>jQuery</title>
  <link href="../../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
    .selected { font-weight:bold; color:White; background-color : Red;}
  </style>
  <script src="http://code.jquery.com/jquery-latest.js" ></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("#txtInput").keydown(function (e) {
        $("#txtKeyCode").val(e.keyCode);
      });
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <input type="text" id="txtInput" /> </br />
    <input type="text" id="txtKeyCode" />
  </div>
</body>
</html>
```



입력되는 키에 따른 해당 "keyCode"값이 노출되는걸 확인 할 수 있으며, "Enter", "Ctrl", "Shift"등의 특수 키보드의 값도 확인 할 수 있다. 앞서 말씀 드린 봐와 같이 jQuery 에서 지원하고 있는 이벤트 관련 메서드의 종류는 상당히 많은 편이나, 이번 강좌에서 예제로 보여드린 부분만 이해를 하신다면 다른 이벤트 관련 메서드를 사용하시는데 큰 문제가 없을 것으로 생각이 된다.

[참고자료 및 참고링크]

jQuery Events: <http://api.jquery.com/category/events/>

[jQuery 강좌] 18. jQuery Event - bind() 메서드

웹 프런티어와 함께하는 jQuery 기초강좌

18th - jQuery Event - bind() 메서드를 통한 이벤트 연결

이번 시간에는 jQuery 에서 지원하는 이벤트를 좀더 고급스럽게 사용하는 방법에 대해 알아보도록 하겠다.

이번 강좌에서는 고급스런 내용의 첫 번째 주자로 나선 bind() 메서드에 대하여 알차게 이야기를 해 보도록 하겠다.

1. .bind()

.bind(eventType, [eventData], handler(eventObject))

Returns: jQuery

Description: Attach a handler to an event for the elements.

.bind(eventType, [eventData], handler(eventObject)) version added: [1.0](#)

eventType A string containing one or more JavaScript event types, such as "click" or "submit," or custom event names.

eventData A map of data that will be passed to the event handler.

handler(eventObject) A function to execute each time the event is triggered.

.bind(eventType, [eventData], false) version added: [1.4.3](#)

eventType A string containing one or more JavaScript event types, such as "click" or "submit," or custom event names.

eventData A map of data that will be passed to the event handler.

false Setting the third argument to false will attach a function that prevents the default action from occurring and stops the event from bubbling.

.bind(events) version added: [1.4](#)

events A map of one or more JavaScript event types and functions to execute for them.

.bind() 메서드는 말 그대로 개체와 이벤트를 묶어주는 역할을 한다.

앞에서 진행한 event 관련 메서드의 경우 해당 메서드를 직접 호출하지만, bind()의 경우 파라미터의 값으로 이벤트 이름을 넣음으로써 해당 이벤트를 체크하게 된다.

```
$("#div").click(function() { alert('click'); })
```

```
$("#div").click(function() { alert('click'); })
```

event 메서드에서 이렇게 사용했다면, bind() 메서드를 통해서도 다음과 같이 표현 된다.

```
$("#div").bind('click', function() { alert('bind click'); })
```

```
$("#div").bind('click', function() { alert('bind click'); })
```

두 메서드의 차이점이 보이시나요 ?

.click() 메서드의 직접호출이 아닌 해당 이벤트의 이름을 넘김으로써 동일한 효과를 얻을 수 있다.

이러한 기능을 통해 조건에 따라 매우 간단히 이벤트를 동적으로 할당 할 수 있다. 또한, 여러 가지의 이벤트를 손쉽게 선택한 개체에 적용할 수 있다. 두 개의 이벤트를 적용한 다음 예제를 통해 좀더 자세히 알아 보도록 하겠다.

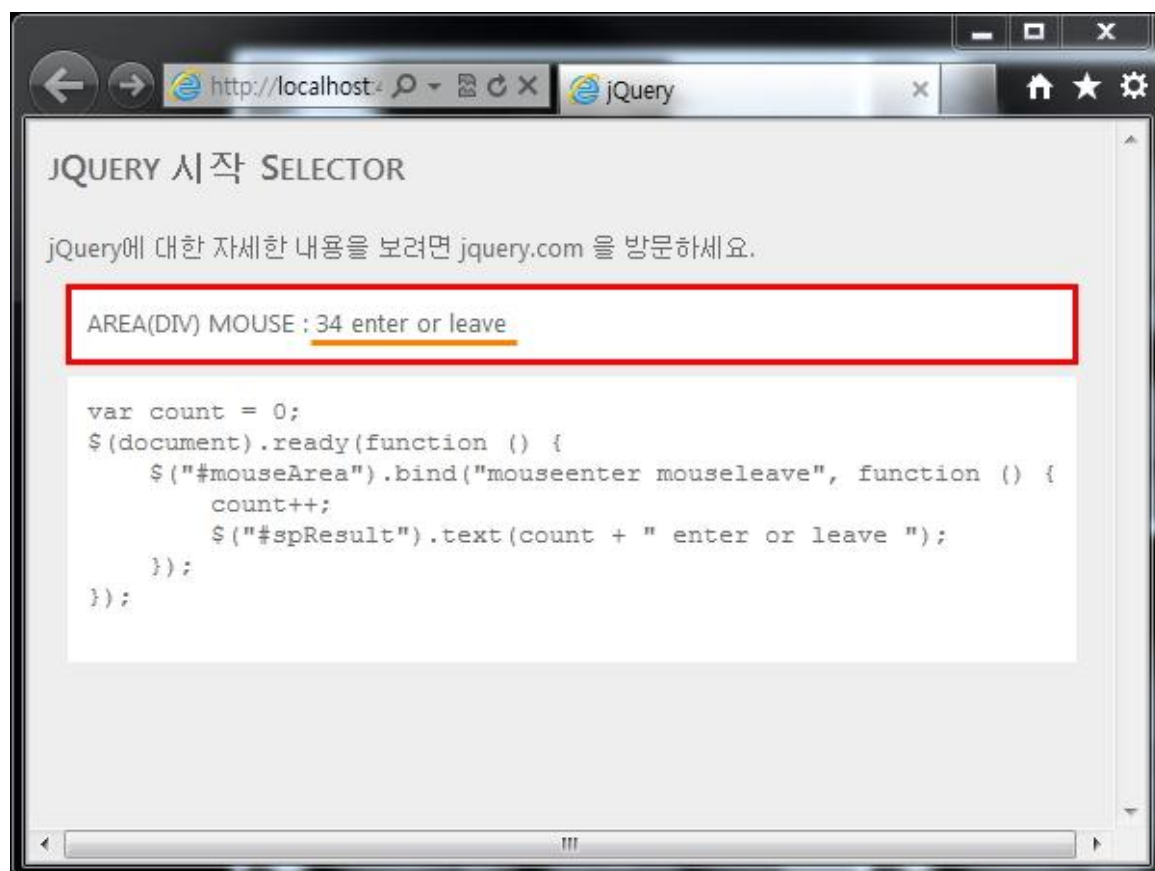
이번 예제는 마우스의 포인터가 지정한 요소에 들어 왔을 때와, 나갔을 때의 이벤트를 체크하여 상태를 표시하며, bind()에 "mouseenter, mouseleave" 이벤트를 이용하였다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery</title>
  <link href="../../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
    .selected { font-weight:bold; color:White; background-color : Red;}
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    var count = 0;
    $(document).ready(function () {
```

```

    $("#mouseArea").bind("mouseenter mouseleave", function (e) {
        count++;
        $("#spResult").text(count + " enter or leave ");
    });
});
</script>
</head>
<body style="padding:10px;">
    <h2>jQuery 시작 Selector</h2>
    <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
    <div id="mouseArea">
        AREA(DIV)
        MOUSE : <span id="spResult"></span>
    </div>
</body>
</html>

```



붉은색으로 표시된 부분에 마우스 포인터를 위치할 때 마다 카운터가 올라가는 것을 확인 할 수 있다. 이렇게 여러 개의 이벤트를 지정 시에도 문제없이 동작을 하는걸 확인 할 수 있다.

이번에는 이렇게 복수의 이벤트를 지정 후 각각의 이벤트에 따른 다른 동작을 구현을 하기 위해서는 어떠한 방법이 있는지 알아 보도록 하겠다. bind() 메서드에서 매우 중요한 부분이니 주의 깊게 보시기 바란다.

다음 예제에서는 bind() 메서드의 파라미터와 eventData 를 사용하여 각각의 이벤트를 어떻게 확인하고 처리하는지 알아 보도록 하겠다.

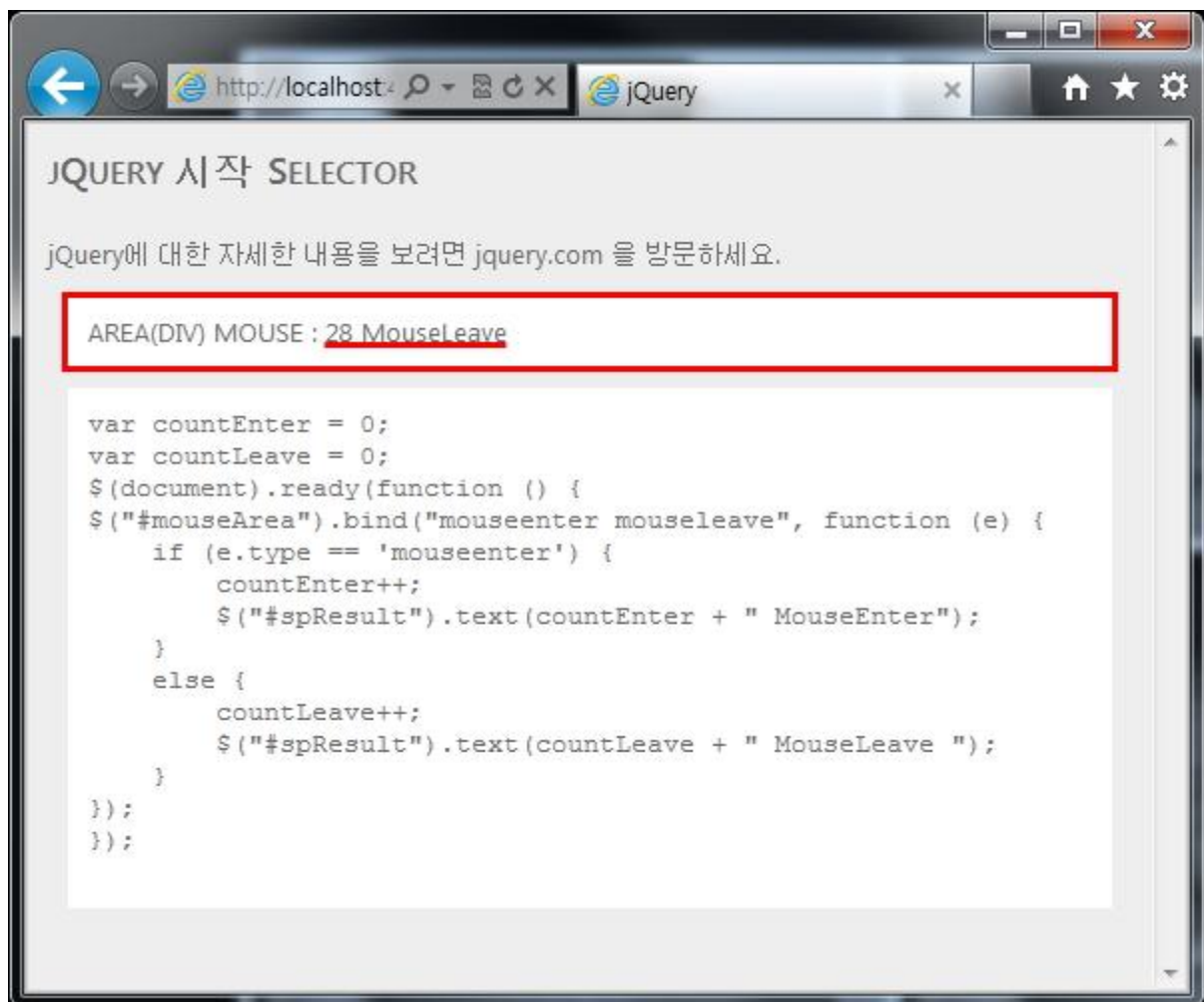
```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery</title>
  <link href="../../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
    table { border:1px solid #AAA; }
    td { border:1px solid #AAA; width:25px; height:25px; }
    .selected { font-weight:bold; color:White; background-color : Red;}
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    var countEnter = 0;
    var countLeave = 0;
    $(document).ready(function () {
      $("#mouseArea").bind("mouseenter mouseleave", function (e) {
        if (e.type == 'mouseenter') {
          countEnter++;
          $("#spResult").text(countEnter + " MouseEnter");
        }
        else {
          countLeave++;
          $("#spResult").text(countLeave + " MouseLeave ");
        }
      });
    });
  </script>
</head>
<body style="padding:10px;">
```

```

<h2>jQuery 시작 Selector</h2>
<p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
<div id="mouseArea">
  AREA(DIV)
  MOUSE : <span id="spResult"></span>
</div>
</body>
</html>

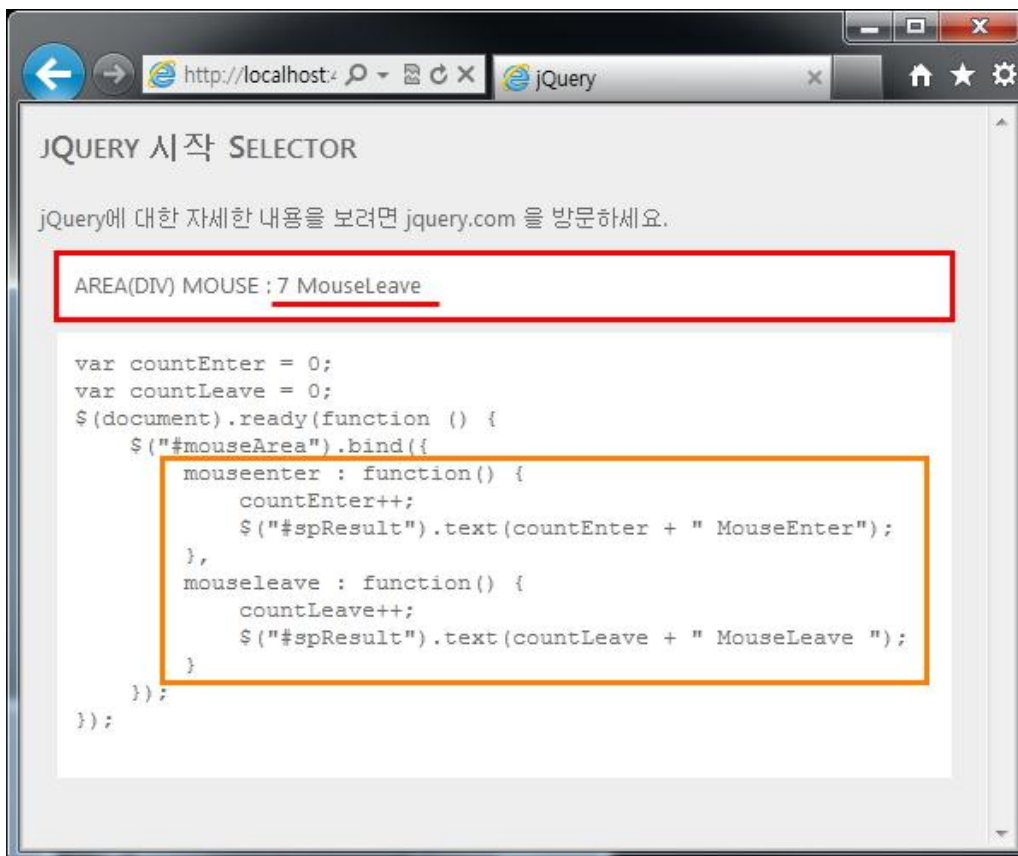
```

[eventData 를 통한 이벤트 확인]



각각의 이벤트에 따라 값이 변경되는걸 확인 할 수 있다.

```
<script type="text/javascript">
    var countEnter = 0;
    var countLeave = 0;
    $(document).ready(function () {
        $("#mouseArea").bind({
            mouseenter : function() {
                countEnter++;
                $("#spResult").text(countEnter + " MouseEnter");
            },
            mouseleave : function() {
                countLeave++;
                $("#spResult").text(countLeave + " MouseLeave ");
            }
        });
    });
</script>
```



위 소스에서 스크립트 부분만 변경하였으며, eventData 대신에 각각의 이벤트에 관련 함수를 연결하여 해당 이벤트를 구분 하고 있다. 역시나 동일한 동작을 하고 있는 모습을 확인 할 수 있다. 이벤트와 요소를 연결하고, 요소에 여러가지 이벤트를 연결하고 구분 짓는 고급스러운 jQuery 표현 방식을 알아 보았다. 이벤트를 제어하기 위해서 필수적으로 알아야 하는 메서드이니, 꼭 숙지 하시기를 부탁 드립니다.

[jQuery 강좌] 19. jQuery Event - 이벤트에 생명을~

19th - jQuery Event(이벤트에 생명을)

이번 시간에는 지난 시간 보다는 좀더 고품격 이벤트 처리에 대해 진행을 할까 한다. 메서드에도 명품이 있다면 이번에 알려드릴 메서드는 정말 최고의 명품이 아닐까 한다. 그럼 거두절미하고 바로 강의를 시작해 보도록 하겠다.

첫 번째로 알아 볼 메서드는 이벤트에 생명을 주는 **live()** 이다.

.live(eventType, handler)	Returns: <u>jQuery</u>
<i>Description: Attach a handler to the event for all elements which match the current selector, now and in the future.</i>	
.live(eventType, handler)	version added: 1.3
eventType A string containing a JavaScript event type, such as "click" or "keydown." As of jQuery 1.4 the string can contain multiple, space-separated event types or custom event names, as well.	
handler A function to execute at the time the event is triggered.	
.live(eventType, eventData, handler)	version added: 1.4
eventType A string containing a JavaScript event type, such as "click" or "keydown." As of jQuery 1.4 the string can contain multiple, space-separated event types or custom event names, as well.	
eventData A map of data that will be passed to the event handler.	
handler A function to execute at the time the event is triggered.	
.live(events)	version added: 1.4.3
events A map of one or more JavaScript event types and functions to execute for them.	

live() 메서드의 경우 원래는 jQuery 의 플러그인으로 개발되어 사용이 되었으나, 너무나 좋은 기능 때문인지 "1.3" 이후 버전의 jQuery 에는 기본 기능으로 추가 되었다.

.click(), .bind() 와 같은 이벤트 메서드에서는 이미 로드가 완료된 개체에 이벤트를 주었다면, .live() 메서드의 경우 동적으로 생성될 개체나 요소에 대해서도 이벤트를 맵핑 할 수 있다.

`$("#a").click(function() { alert('click a'); });` 의 경우 HTML 문서에 존재하는 "a"요소를 찾아 마우스 클릭 이벤트를 맵핑 하였으나, 이후에 동적으로 추가된 "a" 요소에는 영향을 주지 못한다. 하지만 `$("#a").live('click', function() { alert('live click a'); });`를 사용 한다면 처음 로드 된 요소는 물론 로드 후에 동적으로 생성되는 "a" 요소에 대해서도 동일한 이벤트가 적용된다.

.bind() 메서드와 전체적인 구문이 비슷하다. .bind() 메서드에서 강조 드렸던 여러 이벤트를 한번에 등록 하는 것도 똑같이 적용 가능하며, 메서드명과 하는 역할만 조금 다를 뿐 구문은 동일하니 사용상의 구문은 생략을 하도록 하겠다. 좀 더 자세한 내용이 알고 싶으시다면 이전 강좌인 "jQuery Event - bind()"를 참고 부탁 드립니다. 간단한 예제를 통해 .live() 메서드에 대해 좀 더 알아 보도록 하겠다.

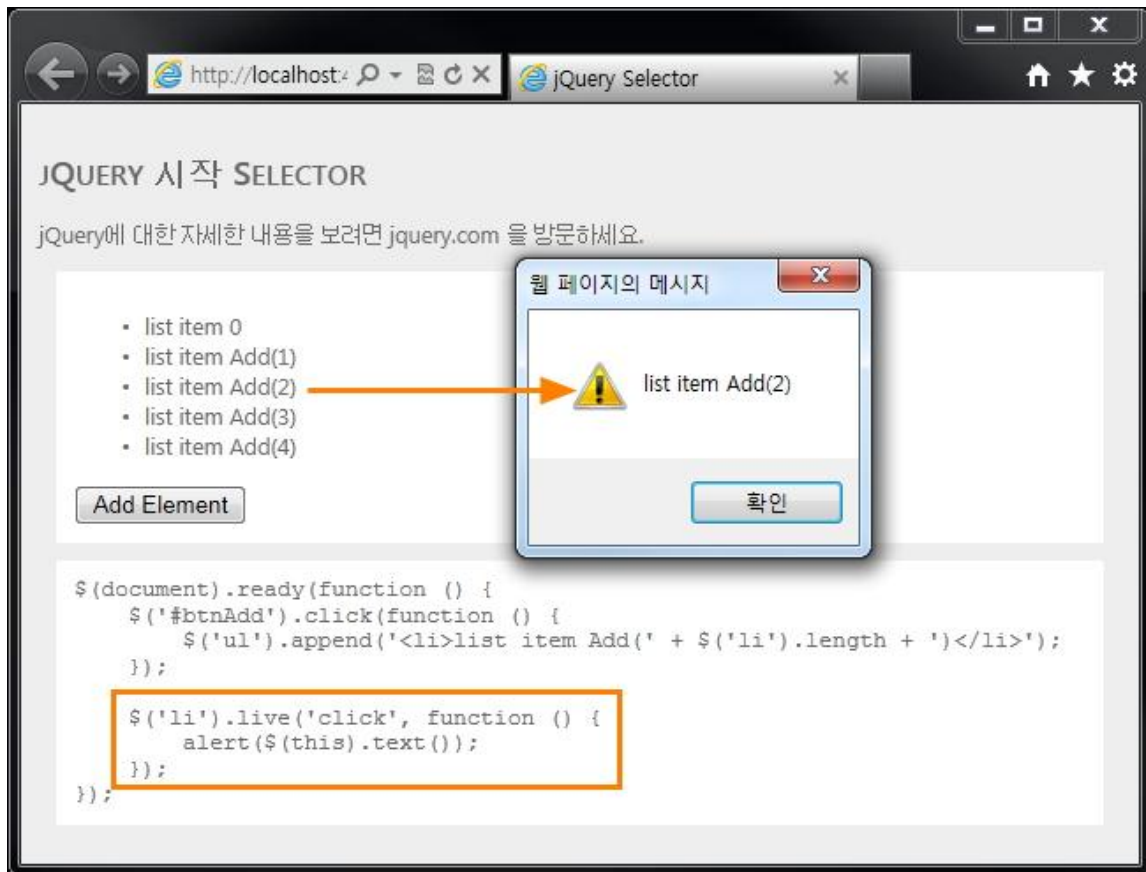
```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div.pre { background : #FFF; padding:10px; margin:10px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $('#btnAdd').click(function () {
        $('ul').append('<li>list item Add(' + $('li').length + ')</li>');
      });
    });
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
  <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
  <div>
    <ul>
      <li>list item 0</li>
    </ul>
    <input type="button" id="btnAdd" value="Add Element" />
  </div>
</body>
</html>
```

버튼을 클릭할 때 마다 ""요소가 추가 되는 내용 이다. 결과 화면을 보면 "" 요소가 추가되는 거 외에는 별다른 역할을 하고 있지 않는다.



이제 live() 메서드를 통해 이벤트를 추가 해 보도록 하겠다. 동적으로 생성되는 "" 요소를 클릭할 경우 해당 요소의 text 를 경고 창을 통해 보여주도록 하였다.

```
$('li').live('click', function () {  
    alert($(this).text());  
});
```



위 예제를 실행을 해 보시면, 동적으로 추가된 "" 요소에 자동적으로 이벤트가 적용 되어 있는걸 확인 할 수 있다. 이제는 동적으로 구성되는 요소에 일일이 하드코딩 하지 않고 몇 줄의 코드로 깔끔하게 처리 할 수 있다.

1. .die()

.die(eventType, [handler])

Returns: jQuery

Description: Remove an event handler previously attached using .live() from the elements.

.die(eventType, [handler])

version added: [1.3](#)

eventType A string containing a JavaScript event type, such as click or keydown.

handler The function that is no longer to be executed.

.die(eventTypes)

version added: [1.4.3](#)

eventTypes A map of one or more event types, such as click or keydown and their corresponding functions that are no longer to be executed.

조금은 잔인한 이야기가 될지 모르지만 생명을 주었으니 죽일 수 도 있다.

기껏 살려놓은 이벤트를 왜 죽여야 하는지에 대해 아주 살짝 힌트로 말씀을 드리면, 클라이언트에서 돔의 변화를 계속 감지하기 때문에 클라이언트에 부하를 줄 수 있으며, 어느 순간 해당 기능이 필요 없어지는 경우 불필요한 부하를 막거나 계속되는 이벤트 맵핑으로 인한 오류를 미연에 방지하기 위함이다.

앞서 설명한 live() 메서드의 예제에 해당 메서드를 추가 하여 어떻게 동작을 하는지 알아 보겠다.

die() 메서드를 호출할 버튼과 "eventDie" 메서드를 을 하나 추가 했다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $('#btnAdd').click(function () {
        $('ul').append('<li>list item Add(' + $('li').length + ')</li>');
      });

      $('li').live('click', function () {
        alert($(this).text());
      });

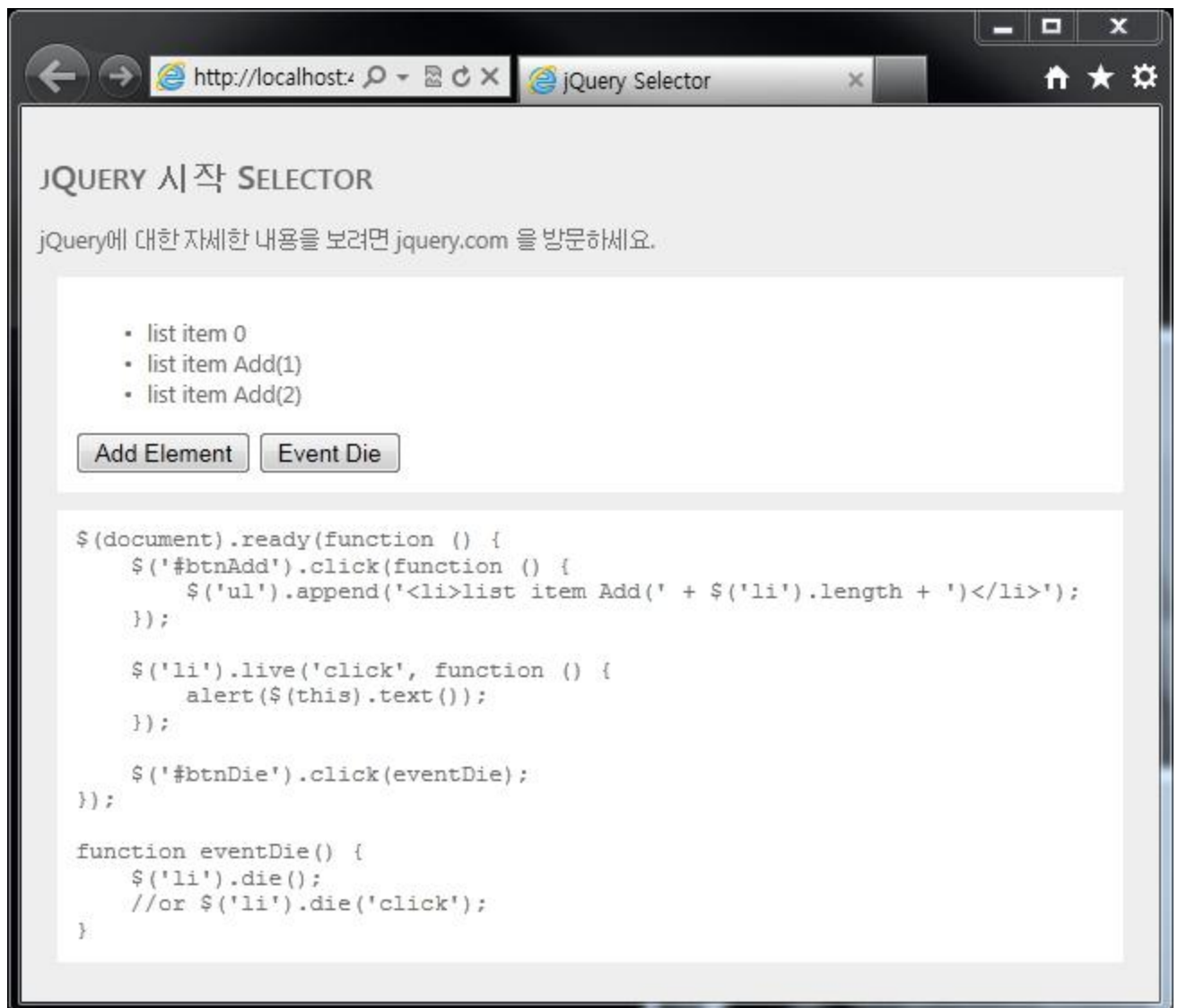
      $('#btnDie').click(eventDie);
    });

    function eventDie() {
      $('li').die();
      //or $('li').die('click');
    }
  </script>
</head>
<body style="padding:10px;">
  <h2>jQuery 시작 Selector</h2>
```

```

<p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
<div>
  <ul>
    <li>list item 0</li>
  </ul>
  <input type="button" id="btnAdd" value="Add Element" /> <input type="button"
id="btnDie" value="Event Die" />
</div>
</body>
</html>

```



"Add Element"를 통해 요소를 동적으로 추가하고 해당 요소를 클릭시에 이전과 동일하게 동작을 하나 "Event Die" 버튼을 클릭하시면 더 이상 경고 창이 뜨지 않는 것을 확인 할 수 있다.

`$(li).die()` 의 경우는 요소 "``"과 연결된 모든 이벤트를 삭제하며, `$(li).die('click');`이라고 지정할 경우에는 "``"요소와 연결된 'click' 이벤트를 삭제 한다. 너무나 간단하게 이벤트를 추가하고, 삭제할 수 있다니 그것도 동적으로 생성되는 요소에 정말 완소 아이템 아니 메서드가 아닌가 싶다.

바로 이어서 마지막으로 설명드릴 메서드는 더욱더 많은 기능을 하는 `.one()` 이다.

`.one(eventType, [eventData], handler(eventObject))` Returns: *jQuery*

Description: Attach a handler to an event for the elements. The handler is executed at most once per element.

`.one(eventType, [eventData], handler(eventObject))` version added: **1.1**

eventType A string containing one or more JavaScript event types, such as "click" or "submit," or custom event names.

eventData A map of data that will be passed to the event handler.

handler(eventObject) A function to execute at the time the event is triggered.

`.bind()` 메서드와 동일한 방식으로 이벤트를 추가한다. 하지만 `.one()` 메서드를 통해 추가된 이벤트는 딱 한번만 실행이 되고 사라지게 된다. 이런 것이 jQuery 에서 제공하는 메서드의 호출만으로 적용 할 수 있다는 사실이 놀라울 따름이다.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Selector</title>
  <link href="../../Styles/Site.css" rel="stylesheet" type="text/css" />
  <style>
    div,pre { background : #FFF; padding:10px; margin:10px; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $('li').one('click', function () {
        alert($(this).text());
      });
    });
  </script>
</head>
<body>
  <div>
    <pre>
      <code>
        <div>
          <ul>
            <li>jQuery Selector
            </li>
          </ul>
        </div>
      </code>
    </pre>
  </div>
</body>
</html>
```

```

    });
</script>
</head>
<body style="padding:10px;">
    <h2>jQuery 시작 Selector</h2>
    <p>jQuery 에 대한 자세한 내용을 보려면 jquery.com 을 방문할 수 있다.</p>
    <div>
        <ul>
            <li>list item 1</li>
            <li>list item 2</li>
            <li>list item 3</li>
            <li>list item 4</li>
            <li>list item 5</li>
        </ul>
    </div>
</body>
</html>

```

바로 앞에 진행한 예제와 기본 동작은 동일하다. "" 요소를 클릭하면 해당 내용을 경고 창을 통해서 보여준다. 설명을 드린 내용과 동일하게 동작을 하는지 확인을 해 보도록 하겠다.



확인을 위해서라면 <http://api.jquery.com/one> 를 참고하면 되겠다.

이번 시간에는 jQuery 에서 제공하는 고급 이벤트 처리에 대해서 알아 보았다.

동적인 웹 페이지 구성에는 정말 감초 또는 가뭄에 내리는 단비와 같은 역할을 하는 고마운 요소들이니 사용법을 익혀 두시면 정말 큰 도움이 될 거라고 강력하게 외치며 강좌를 마치도록 하겠다.

[jQuery 강좌] 20. jQuery Performance

20th - jQuery Performance - jQuery 성능에 대한 낚두리

이번 시간에는 jQuery 성능에 대해서 이야기를 하려고 한다.

jQuery 는 정말로 많은 기능을 개발자가 사용하기 쉬운 형태로 제공하고 있다. 하지만 얻는 게 있다면 잃는 것도 있는 게 세상의 이치이듯 jQuery 의 강력한 기능을 사용함으로써 개발에 드는 공수는 줄어 들지만 클라이언트의 부하는 증가하게 된다. 하지만 앞으로 설명드릴 내용을 이해 하신다면 어느 정도의 부하를 줄일 수 있다.

말은 거창하게 시작했지만 크게 어렵거나 한 부분은 없으니, 부담 없이 강좌를 봐 주시면 되겠다.

<첫 번째 - 셀렉터의 구체화>

다른 사람을 이해시키기 위해서 자세하게 설명 하듯이 jQuery 도 마찬가지로 원하는 기능에 대해 자세히 알려주면 성능이 올라간다.

`$('#elementID')`, `$(elementName)`는 조금 상황이 다르지만, "class" 또는 "attribute"를 통한 셀렉트시에는 최대한 자세히 명시를 해 주는 것이 성능에 큰 도움이 된다. 셀렉터의 "class"를 이용할 경우에는 단순히 `$(".myClass")` 보다는 `$(div > ul > li.class)`처럼 많은 정보를 주면 jQuery 가 HTML 문서에서 탐색하는 범위가 줄어들어 성능향상의 효과가 있다.

<두 번째 - 셀렉터의 사용 자제>

셀렉터를 사용하다 보면 같은 요소를 여러 번 셀렉트 하는 경우가 발생 하곤 한다. 이런 경우에는 기존에 사용한 셀렉터를 재사용을 하거나, `.end()` 메서드를 통해 반복 사용하지 않는 것이 좋다.



<세 번째 - 순수 자바스크립트 사용>

jQuery 는 javascript 의 복잡한 함수로 이루어져 있다. jQuery 의 함수를 실행할 경우 내부에서 구현된 복잡한 로직을 통해 javascript 를 실행하고 있기 때문에 해당 로직을 피하게 되면 그만큼의 성능이 올라간다고 생각을 하시면 된다.

자주 사용하는 기능 중에 2 개를 예를 들어 설명을 드리면 다음과 같다.

1. `$('#id')` 보다는 `document.getElementById("id")`
2. `$("#id").css("color", "red")` 보다는 `document.getElementById("id").style.color = "red";`
`$("#id").hide()` 보다는 `document.getElementById("id").style.display = 'none';`

또한, DOM 처리를 할 때에도 jQuery 보다는 순수 javascript 를 사용하는 것이 성능 면에서 큰 이득이니, 이 부분도 참고 하시기 바란다.

```
$(document).ready(function () {  
    var $htmlData = $("<ul/>");  
    for (var i = 0; i < 10; i++) {  
        $htmlData.append($("<li/>").text('승연아빠(' + i.toString() + ')'));  
    }  
    $htmlData.appendTo('#contents');  
});
```

```
$(document).ready(function () {  
    var htmlData = '<ul>';  
    for (var i = 0; i < 10; i++) {
```

```

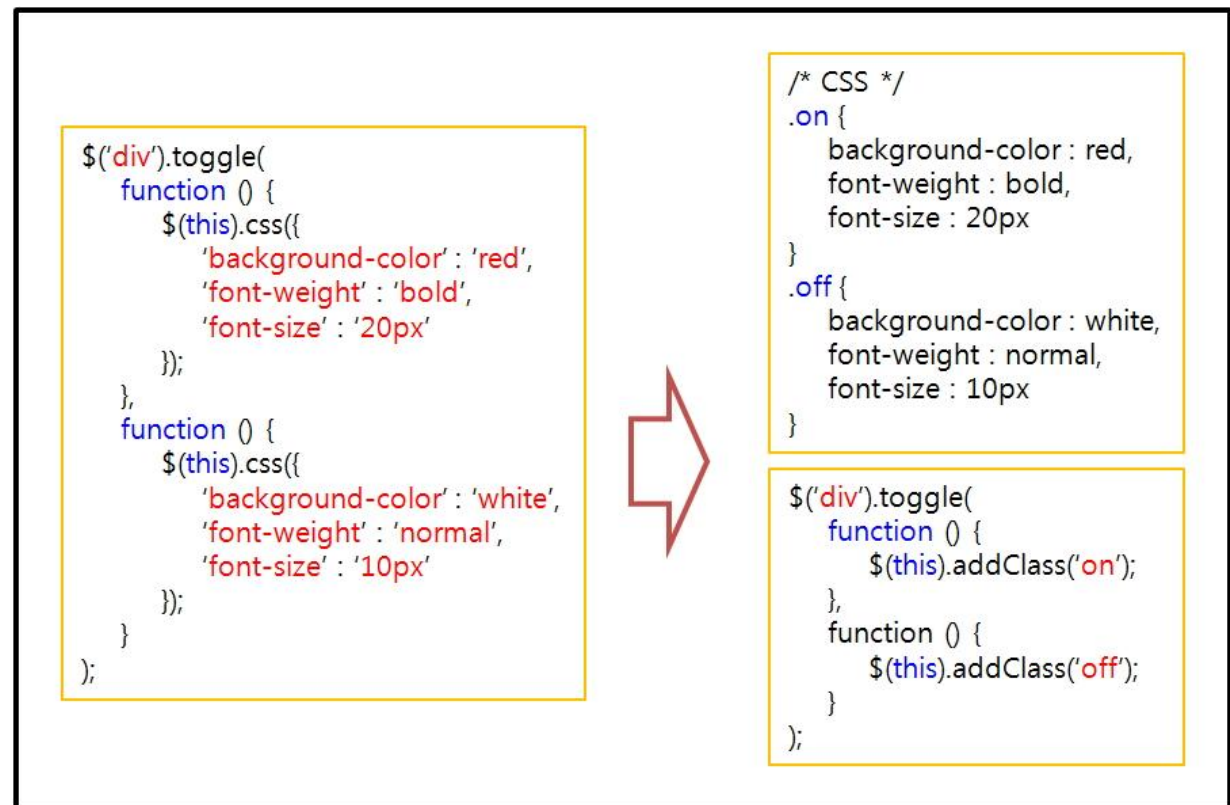
        htmlData += '<li>승연아빠(' + i.toString() + ')';
    }
    htmlData += '</ul>';
    $('#contents').html(htmlData);
});

```

[첫 번째 방법 보다는 두 번째 방법이 성능에 좋다.]

<네 번째 - .css(>

.css() 메서드를 사용하여, 선택한 개체에 특정 스타일을 적용할 경우 내부에서 많은 처리를 하게 된다. 이런 경우가 많다면 스타일시트를 통한 각각의 스타일 클래스를 작성하고, 해당 클래스를 적용 하시기 바란다.



이 부분은 jQuery 에서만 적용되는 부분이 아닌, 일반적인 javascript 도 해당하는 부분이니 참고 하시기 바란다.