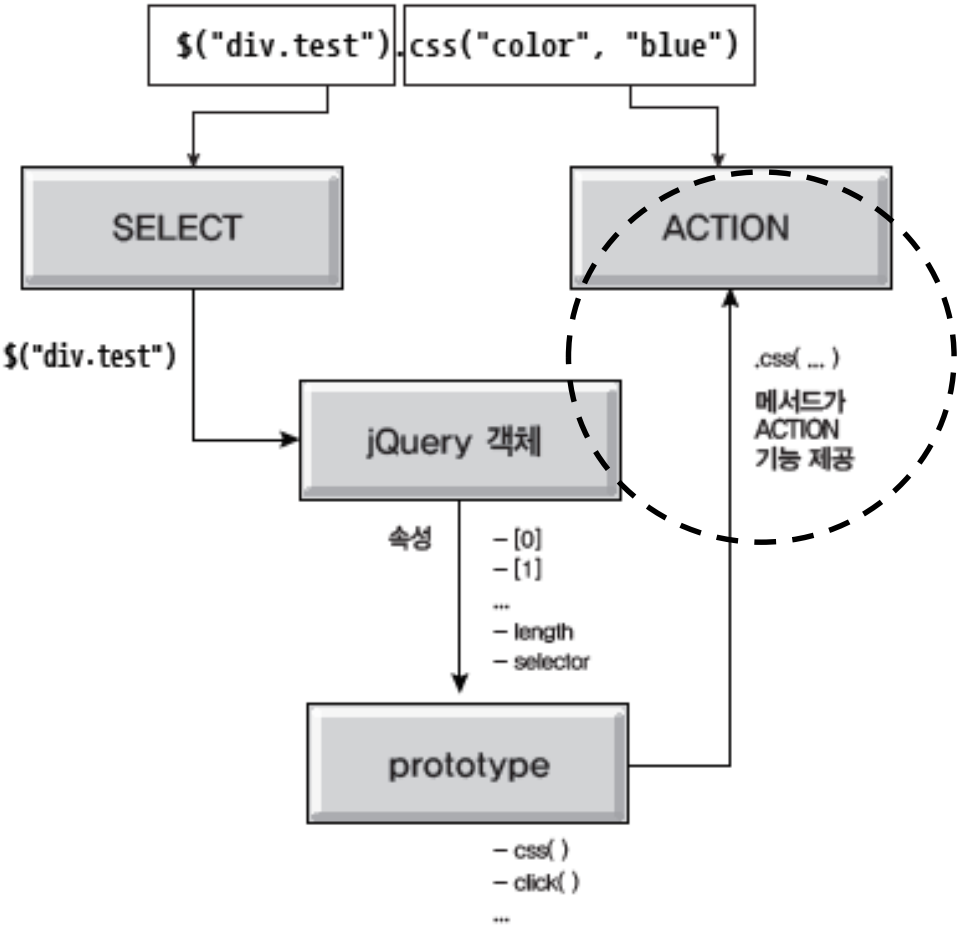
요소 조작하기

- 선택된 요소에 대한 작업은?
  - jQuery 객체의 prototype에 정의된 메서드를 이용한다.
  - 이벤트, 효과, 조작 메서드가 제공됨
  - 은연중에 이미 사용해본 경험!! --> css() 메서드!!



요소 조작 메서드의 특징(1)

쓰기, 읽기 메서드의 구분이 없다.

- 하나의 메서드로 읽기, 쓰기 모두 수행
  - css(name) : 읽기
  - css(name, value) : 쓰기
- 자바스크립트 언어는 메서드 오버로딩 개념이 없으므로 전달된 파라미터의 존재여부와 데이터 타입을 확인해서 작업 기능을 구분함.
- 대개 쓰기 작업이 읽기 작업에 비해 파라미터가 하나씩 더 많음

선택된 요소가 여러개인 경우의 기본 작업

- 설정 : 선택된 여러 요소를 모두 동일한 값으로 설정한다.
- 읽기 : 선택된 요소 중에서 첫 번째 요소의 값을 읽어낸다.

요소 조작 메서드의 특징(2)

예제 10-01(예제 08-01 소환!)

- 이 예제를 실행한 후 브라우저 콘솔에서 직접 실행

김수한무
거북이와 두루미
삼천갑자 동방삭
치치카포 사리사리센타
div.test.main 232px × 21px
무두셀라 구름이
허리케인에 담벼락
담벼락에 서생원
서생원에 고양이
고양이에 바둑이
바둑이는 돌돌이

Elements Console Sources Network Timeline Profiles

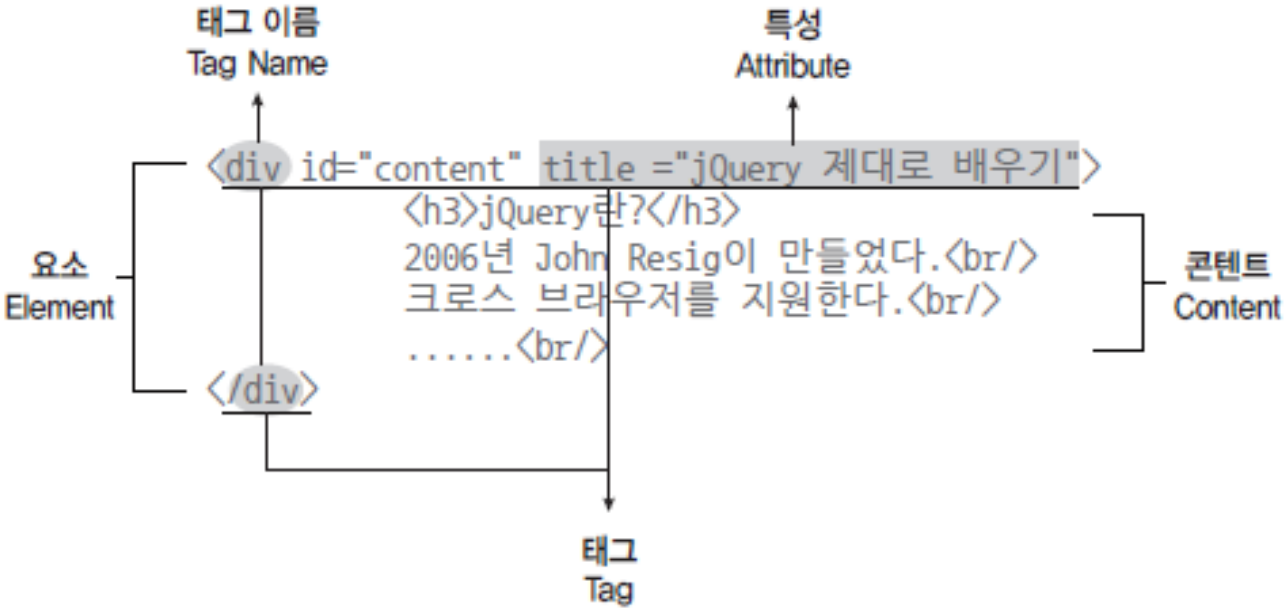
<top frame> Preserve log

```
> $("#content > div.test").css("background-color", "yellow");
< [  <div class="test" style="background-color: yellow;">삼천갑자 동방삭</div>,
   <div class="test  main" style="background-color: yellow;">치치카포 사리사리센
   타</div>
,   <div class="test" style="background-color: yellow;">서생원에 고양이</div>]
> $("#content > div.test").html();
< "삼천갑자 동방삭"
```

특성과 속성 조작하기(1)

특성(Attribute)과 속성(Property)

- 특성 : HTML 요소(Element)의 일부분
  - 요소는 태그와 다른 개념임



- 속성 : 객체가 가지고 있는 이름과 값의 쌍, 즉 객체의 속성이다

특성과 속성 조작하기(2)

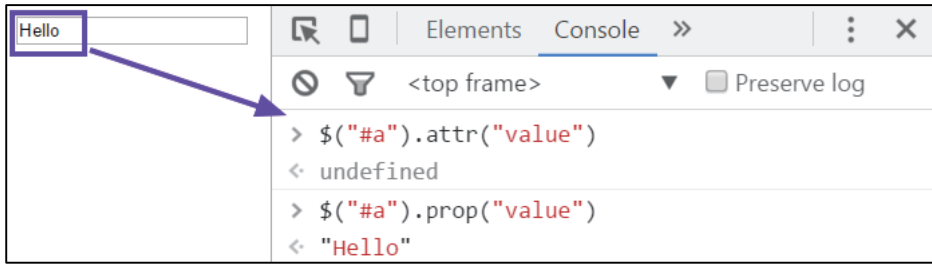
예제 10-02

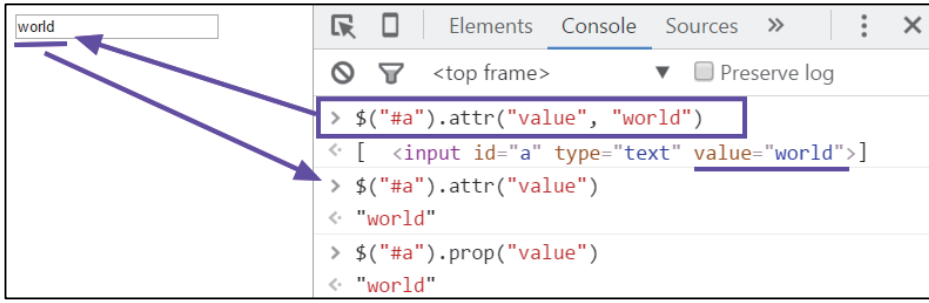
[예제 10-02]

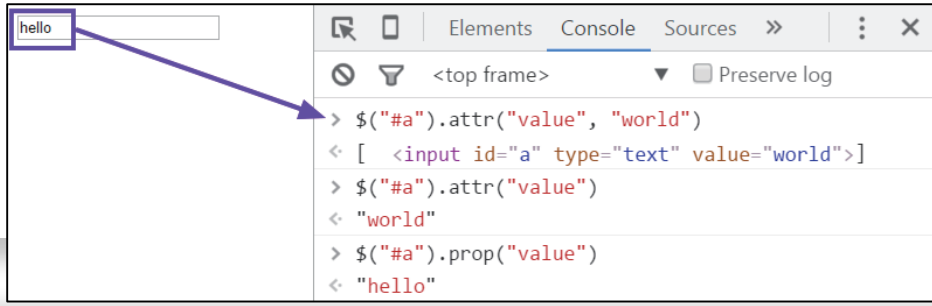
```
01: <!DOCTYPE html>
02: <html lang="">
03: <head>
04: <meta charset="utf-8">
05: <title>10-02</title>
06: <script type="text/javascript" src="https://code.jquery.com/jquery-3.1.0.js"></script>
07: <script type="text/javascript">
08: $(document).ready(function() {
09:
10: });
11: </script>
12: </head>
13: <body>
14:   <input id="a" type="text" />
15: </body>
16: </html>
17:
```

그림 10-04~07

```
> $("#a").attr("value")
<- undefined
> $("#a").prop("value")
<- ""
```







특성과 속성 조작하기(3)

attr( ), prop( ) 메서드

- 특성과 속성을 조작하는 메서드
- 사용 방법은 동일하므로 attr() 메서드를 중심으로 살펴본다.

[**표 10-02** : attr() 메서드의 사용 형태]

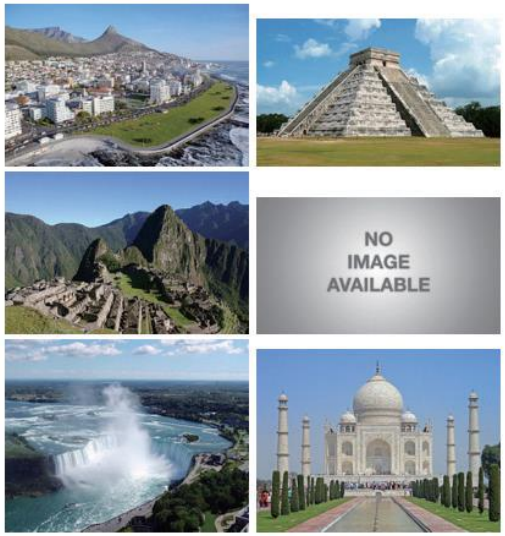
메서드	읽기/쓰기	설명
1. attr(name)	읽기	특성명으로 값을 읽어낸다.
2. attr(name, value)	쓰기	특성명에 대한 값을 설정한다.
3. attr(object)	쓰기	여러 특성값을 한번에 설정한다.
4. attr(name, function)	쓰기	선택된 요소마다 함수의 리턴값으로 특성값을 설정한다.

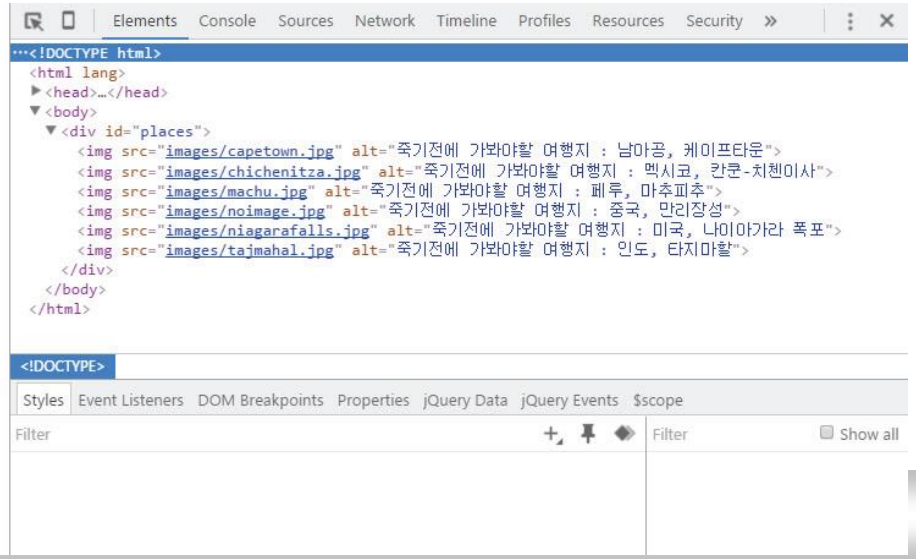
특성과 속성 조작하기(4)

- attr(name, function(i, old) {})
  - i : 인덱스 번호, old : 기존 특성값
  - 함수의 리턴값으로 새로운 특성값을 설정

```
21:   $("#places").find("img").attr("src", function(i) {
22:      if (places[i].image) {
23:         return places[i].image;
24:      }
25:   }).attr("alt", function(i) {
26:      return "죽기 전에 가봐야 할 여행지 : " + places[i].title;
27:   });
28: });
```

예제 10-30



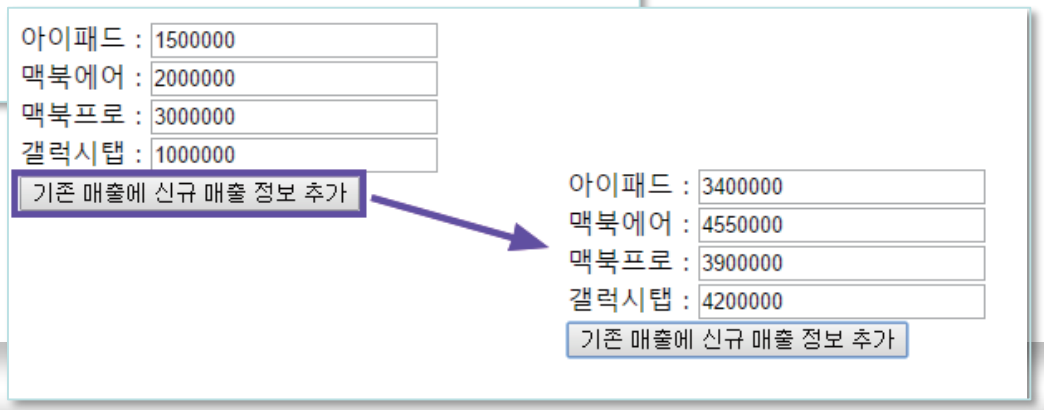


특성과 속성 조작하기(5)

예제 10-04

```
24: <body>
25:     <div id="sales">
26:         아이패드 : <input type="text" id="a" value="1500000" /><br />
27:         맥북에어 : <input type="text" id="b" value="2000000" /><br />
28:         맥북프로 : <input type="text" id="c" value="3000000" /><br />
29:         갤럭시탭 : <input type="text" id="d" value="1000000" /><br />
30:         <button id="addSalesInfo">기존 매출에 신규 매출 정보 추가</button>
31:     </div>
32: </body>
```

```
15:   //discountrate : 할인율, amount: 매출액
16:   $("#addSalesInfo").click(function() {
17:       $("#sales").find("input").attr("value", function(i, old) {
18:           return (sales[i].amount * (1-sales[i].discountrate)) + parseInt(old);
19:       });
20:   });
```



특성과 속성 조작하기(6)

- attr(object)
  - object : 한번에 설정할 특성들을 객체로 구성함
- 아래 두 코드는 무슨 차이가 있을까?

```
$("img#place1").attr("alt", "멕시코 칸쿤").attr("title", "칸쿤의 저녁 노을");
$("img#place1").attr({ alt: "멕시코 칸쿤", title: "칸쿤의 저녁 노을" });
```

- 잘 모르겠다면 아래 코드를 보자.

```
var options = {};
options.alt = "멕시코 칸쿤";
options.title = "칸쿤의 저녁 노을";
$("img#place1").attr(options);
```

  - 미리 객체를 생성하고 한번에 인자를 전달할 수 있음
  - 특성 값들이 자주 변경될 때 편리함.

특성과 속성 조작하기(7)

예제 10-05

```
17:     <div id="bestplace">
18:         <img />
19:     </div>
```

```
09:     var options  = {};
10:     options.src = "images/machu.jpg";
11:     options.alt = "죽기전에 가봐야할 베스트 여행지 : 페루, 마추피추";
12:     $("#bestplace").find("img").attr(options );
```

특성과 속성 조작하기(8)

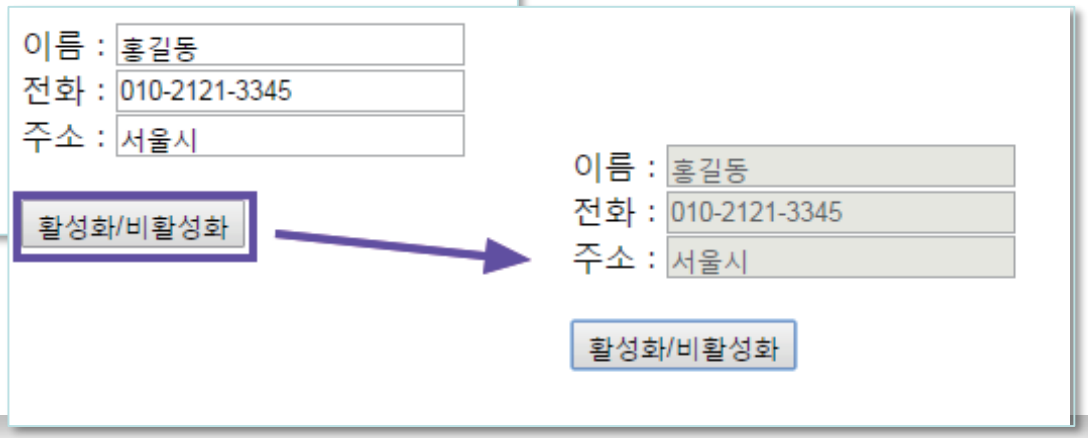
- removeAttr(), removeProp() 메서드
  - 특성과 속성을 제거하는 기능
  - removeAttr() 메서드
    - removeAttr(name) > attr(name, null)
  - removeProp() 메서드
    - 이 메서드를 호출하더라도 속성이 삭제되지 않을 수 있음
    - 개발자가 prop() 메서드를 이용해 추가한 임의의 속성만 삭제할 수 있음
      - jQuery 객체가 선택하고 있는 요소의 속성 중 내장 속성은 함부로 삭제되지 않음
      - delete 키워드는 삭제를 시도하지만 삭제되지 않아도 오류를 발생하지 않음

특성과 속성 조작하기(9)

예제 10-06

```
22:   <div id="content">
23:      이름 : <input id="name" class="test" type="text" value="홍길동"/><br />
24:      전화 : <input id="tel" class="test" type="text" value="010-2121-3345" /><br />
25:      주소 : <input id="address" class="test" type="text" value="서울시" /><br />
26:      <br/>
27:      <button id="toggle">활성화/비활성화</button>
28:   </div>
```

```
09:   $("#toggle").click(function() {
10:      var inputs = $("#content").find("input.test");
11:      if (inputs.is(":enabled") == true) {
12:         inputs.attr("disabled", "disabled");
13:      }  else {
14:         //inputs.attr("disabled", null);
15:         inputs.removeAttr("disabled");
16:      }
17:   });
```



특성과 속성 조작하기(10)

val() 메서드

- value 속성을 조작하는 메서드

[**표 10-03** : val( ) 메서드의 사용 형태]

메서드	읽기/쓰기	설명
1. val( )	읽기	value 속성값을 읽어낸다.
2. val(value)	쓰기	value 속성값을 설정한다.
3. val(function)	쓰기	선택된 요소에 대해 함수의 리턴값으로 value 속성값을 설정한다. ex) $("input").val(function(i, old) { });

- 예제 07-11을 참조

스타일 조작하기(1)

요소의 디자인 변경

- 특성 값 변경 --> CSS 스타일 값 변경

```
<table id="a" width="400px">
   ......
</table>
```

```
<table id="b" style="width:400px">
   ......
</table>
```

[**표 10-04** : 스타일 조작 메서드]

메서드	설명
addClass( )	클래스를 추가한다.
removeClass( )	클래스를 제거한다.
toggleClass( )	지정한 클래스가 있으면 제거하고, 없으면 추가한다.
css( )	스타일 속성을 읽어내거나 설정하는 메서드이다.
hasClass( )	선택된 요소에 파라미터값으로 전달한 클래스가 지정되어 있는지 조사한다.

스타일 조작하기(2)

스타일 기초

스타일 작성 방법

- 인라인 스타일

```
<div style="width:300px; height:200px; border:solid 1px gray;"></div>
```

- css 클래스 활용

[예제 10-08 : test.css]

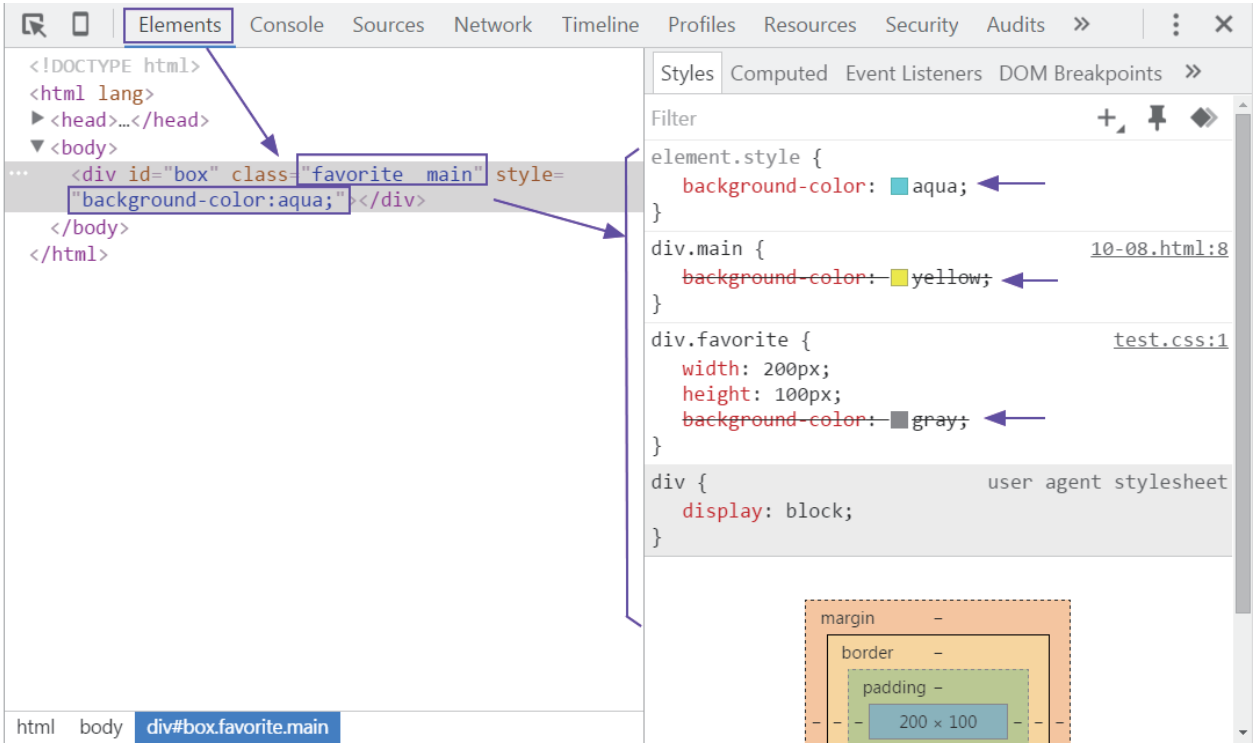
```
div.favorite {
   width:200px; height:100px; background-color: gray;
}
```

```
03: <head>
04: <meta charset="utf-8">
05: <title>10-08</title>
06: <link rel="stylesheet" type="text/css" href="css/test.css" />
07: <style>
08:   div.main { background-color:yellow; }
09: </style>
```

스타일 조작하기(3)

- 예제 10-08
  - 스타일 적용 우선순위

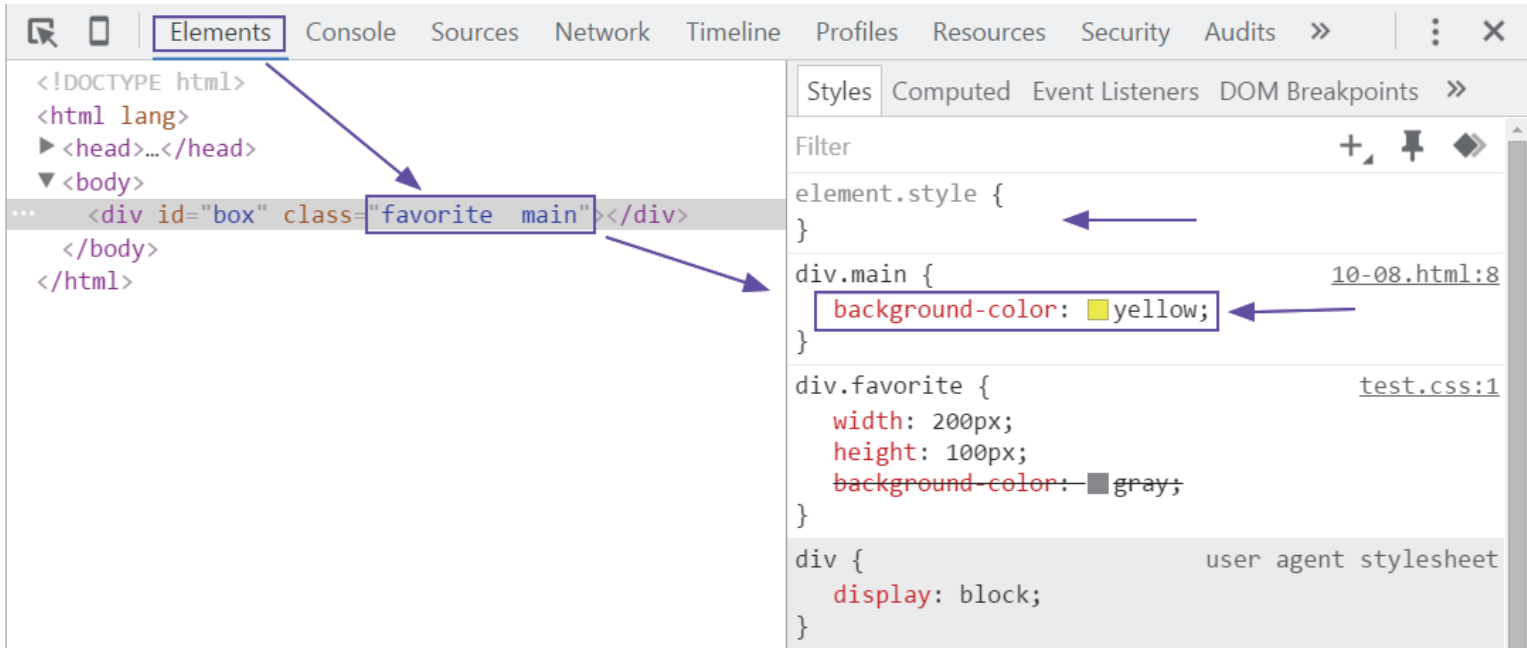
    인라인 스타일 > CSS 클래스2 > CSS 클래스1 > HTML 요소 기본 스타일



스타일 조작하기(4)

- 인라인 스타일을 제거한다면?
  - 인라인 스타일 제거 방법 : 빈문자열("") 할당

    $("#box").css("background-color", "");
  - 인라인스타일의 제거 결과 가려졌던 우선순위가 낮은 스타일이 적용됨



스타일 조작하기(5)

- 다양한 스타일에 대한 학습을 위한 참조 정보
  - https://www.kobzarev.com/wp-content/uploads/cheatsheets/css/css3-cheat-sheet.pdf
  - https://developer.mozilla.org/ko/docs/Web/CSS/CSS_Reference

스타일 조작하기(6)

addClass(), removeClass(), toggleClass()

- addClass() : 기존 클래스 뒤에 새로운 클래스 추가
  - 클래스 적용 순서는 스타일 우선순위에 영향을 줌

$("#a").addClass("main")을 실행하면 아래와 같이 변경됨.
$("#a").removeClass("test")을 실행하면 아래와 같이 변경됨.

```
addClass(className)

addClass(function(index, oldClass) { })
```

스타일 조작하기(7)

- 예제 10-09
  - mouseenter, mouseleave 이벤트를 이용한 hover 기능 구현

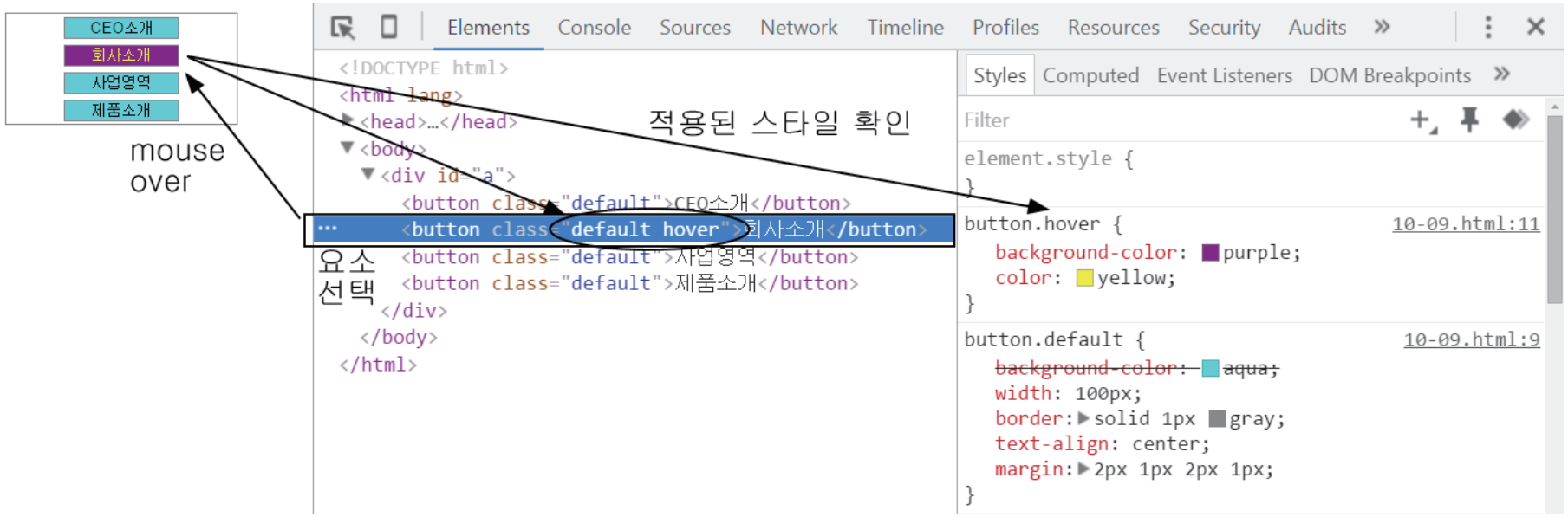
```
24:   <div id="a">
25:      <button class="default">CEO소개</button>
26:      <button class="default">회사소개</button>
27:      <button class="default">사업영역</button>
28:      <button class="default">제품소개</button>
29:   </div>
```

```
06: <style>
07:   button.default { background-color: aqua; width:100px;
08:     border: solid 1px gray; text-align: center;  }
09:   button.hover { background-color: purple; color:yellow; }
10: </style>
```

```
14:   //hover!!!
15:   $("#a").find("button.default").mouseenter(function() {
16:      $(this).addClass("hover");
17:   }).mouseleave(function() {
18:      $(this).removeClass("hover");
19:   });
```

스타일 조작하기(8)

- 예제 10-09 실행 결과



스타일 조작하기(9)

- toggleClass()

[표 10-05 : toggleClass( ) 메서드]

메서드	설명
toggleClass(class [, state])	지정한 클래스가 부여되어 있지 않으면 추가하고, 부여되어 있다면 제거한다. state는 옵션값이며, 이 값이 true면 지정된 클래스를 추가하고, false면 지정된 클래스를 삭제한다.
toggleClass([state])	HTML 요소에 주어진 모든 클래스를 추가하거나 삭제한다. state 옵션값은 앞의 내용과 동일하다.
toggleClass(function [, state])	선택된 요소마다 토글할 클래스명이 각기 다를 때 이 방법을 사용한다. 인자로 전달하는 함수가 리턴하는 클래스를 추가하거나 삭제한다. 함수의 형태는 다음과 같다. function(index, class, state)=>className index : 인덱스 번호 class : 클래스명 state : 전달된 state 옵션값 리턴값 : 토글 처리할 클래스명

스타일 조작하기(10)

예제 10-10

```
31:   <div id="a">
32:      <button class="default">CEO</button>
33:      <button class="default">회사소개</button>
34:      <button class="default">사업영역</button>
35:      <button class="default">제품소개</button>
36:   </div>
37:   <div id="b">
38:      <button class="default">한식</button>
39:      <button class="default">분식</button>
40:      <button class="default">중식</button>
41:      <button class="default">양식</button>
42:   </div>
```

- 각각 라디오 버튼과 체크박스 기능 구현
- hover 클래스를 추가/삭제를 반복함.

```
16:   //라디오 버튼!!
17:   $("#a").find("button.default").click(function() {
18:      if ($(this).is(".hover") == false) {
19:         $(this).parent().find(".hover").removeClass("hover");
20:         $(this).addClass("hover");
21:      }
22:   });
23:   //체크박스!!
24:   $("#b").find("button.default").click(function() {
25:      $(this).toggleClass("hover");
26:   });
```

스타일 조작하기(11)

- toggleClass() 메서드의 state 옵션과 익명함수를 사용한 예제
- (예제 10-11)

```
32:  <button id="addsep">5줄마다 라인 구분 추가</button>
33:  <button id="toggle">5줄마다 라인 구분 토글!!</button>
34:    <table id="nations">
35:        <thead>
36:            <tr><th>번호</th><th>국가명</th><th>수도</th></tr>
37:        </thead>
38:        <tbody>
39:            <tr><td>1</td><td>미국</td><td>워싱턴DC</td></tr>
40:            <tr><td>2</td><td>프랑스</td><td>파리</td></tr>
41:            <tr><td>3</td><td>영국</td><td>런던</td></tr>
42:            <tr><td>4</td><td>중국</td><td>베이징</td></tr>
43:            <tr><td>5</td><td>태국</td><td>방콕</td></tr>
44:            <tr><td>6</td><td>모로코</td><td>라바트</td></tr>
45:            <tr><td>7</td><td>라오스</td><td>비엔티안</td></tr>
46:            <tr><td>8</td><td>베트남</td><td>하노이</td></tr>
47:            <tr><td>9</td><td>피지</td><td>수바</td></tr>
48:            <tr><td>10</td><td>솔로몬 제도</td><td>호니아라</td></tr>
49:            <tr><td>11</td><td>자메이카</td><td>킹스턴</td></tr>
50:            <tr><td>12</td><td>나미비아</td><td>빈트후크</td></tr>
51:            <tr><td>13</td><td>통티모르</td><td>딜리</td></tr>
52:            <tr><td>14</td><td>멕시코</td><td>멕시코시티</td></tr>
53:            <tr><td>15</td><td>베네수엘라</td><td>카라카스</td></tr>
54:            <tr><td>16</td><td>서사모아</td><td>아피아</td></tr>
55:        </tbody>
56:    </table>
```

```
$("#addsep").click(function() {
  $("#nations > tbody > tr").each(function(i, item) {
    $(item).toggleClass("separator", i%5 == 4);
  });
});

$("#toggle").click(function() {
  $("#nations > tbody > tr").toggleClass(function(i) {
    if (i%5 == 4) {
      return "separator";
    }
  });
});
```

스타일 조작하기(12)

- 예제 10-11 실행 결과

5줄마다 라인 구분 추가 | 5줄마다 라인 구분 토글!!

번호	국가명	수도
1	미국	워싱턴DC
2	프랑스	파리
3	영국	런던
4	중국	베이징
5	태국	방콕
6	모로코	라바트
7	라오스	비엔티안
8	베트남	하노이
9	피지	수바
10	솔로몬 제도	호니아라
11	자메이카	킹스턴
12	나미비아	빈트후크
13	동티모르	딜리
14	멕시코	멕시코시티
15	베네수엘라	카라카스
16	서사모아	아피아

스타일 조작하기(13)

css() 메서드

- CSS 스타일을 설정하거나 읽어내는 메서드
- 예제 10-12로 기능 확인

```
06: <style>
07:    .test { color:"brown"; font-weight:bold; font-size:20pt; }
08: </style>
```

```
17:    <div id="a" class="test">Hello</div>
```

- 아래 명령문 실행

```
$("#a").css("color");
$("#a").css("color", "green");
```

- 설정은 인라인 스타일로!!
- 읽기는 인라인 스타일뿐만 아니라 클래스에 적용된 스타일까지 적용하여 읽어냄

스타일 조작하기(14)

- css() 메서드 사용법
  - attr() 메서드와 유사하다.

[**표 10-06** : css( ) 메서드의 사용 형태]

메서드	읽기/쓰기	설명
1. css(name)	읽기	스타일 속성명으로 값을 읽어낸다.
2. css(name, value)	쓰기	스타일 속성명에 대한 값을 설정한다.
3. css(object)	쓰기	여러 스타일 속성 값을 한번에 설정한다.
4. css(name, function)	쓰기	선택된 요소마다 함수의 리턴값으로 스타일 속성값을 설정한다.
5. css(names)	읽기	names 배열값에 해당하는 스타일 속성을 한번에 읽어낸다.

  - 5번 방식의 읽기는 스타일 정보를 JS 객체 형태로 한번에 읽어냄

```
> var obj = $("#a").css(["color", "background-color", "font-size"])
< undefined
> console.dir(obj)
  ▼Object i                                                VM645:2
     background-color: "rgba(0, 0, 0, 0)"
     color: "rgb(0, 128, 0)"
     font-size: "26.6667px"
    ▶ __proto__ : Object
```

스타일 조작하기(15)

hasClass() 메서드

- 선택된 요소에 지정한 클래스가 적용되었는지를 true/false로 리턴하는 메서드
- 예제 10-01을 이용해 브라우저 콘솔에서 테스트
  - hasClass("main")과 is(".main")은 사실상 같은 기능 수행

```
<div id="content">
    <div id="a">김수한무</div>
    <div>거북이와 두루미</div>
    <div class="test">삼천갑자 동방삭</div>
    <div class="test  main">치치카포 사리사리센타</div>
    <div>워리워리 세브리깡</div>
    <div>무두셀라 구름이</div>
    <div id="b">허리케인에 담벼락</div>
    <div class="main">담벼락에 서생원</div>
    <div class="test">서생원에 고양이</div>
    <div id="a">고양이에 바둑이</div>
    <div>바둑이는 돌돌이</div>
</div>
```

```
> $("#content > div").hasClass("main")
< true
> $("#content > div").is(".main")
< true
```

요소의 크기와 위치 조작하기(1)

RIA의 변화

- Rich Internet Application
  - Flash, ActiveX Control --> HTML5 + JavaScript
- 따라서 요소의 크기와 위치를 조작하는 기능을 알아두어야 함.

요소의 크기 조작

[**표 10-07** : 요소 크기 조작 메서드]

메서드	설명
width( )	요소의 폭을 구하거나 설정한다.
height( )	요소의 높이를 구하거나 설정한다.
innerWidth( )	요소의 테두리 선 안쪽의 폭을 구하거나 설정한다.
innerHeight( )	요소의 테두리 선 안쪽의 높이를 구하거나 설정한다.
outerWidth( )	요소의 테두리 선 밖의 폭을 구하거나 설정한다.
outerHeight( )	요소의 테두리 선 밖의 높이를 구하거나 설정한다.

요소의 크기와 위치 조작하기(2)

- 편의상 width() 메서드 하나만 설명

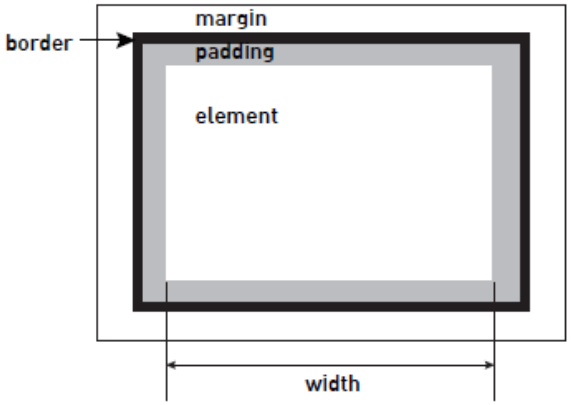
width() : 요소의 폭을 구한다.

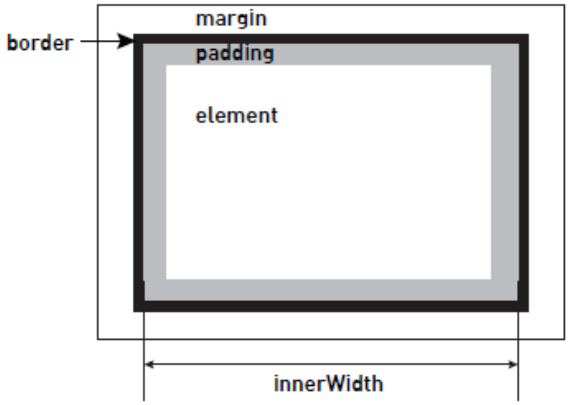
width(value) : 요소의 폭을 설정한다.

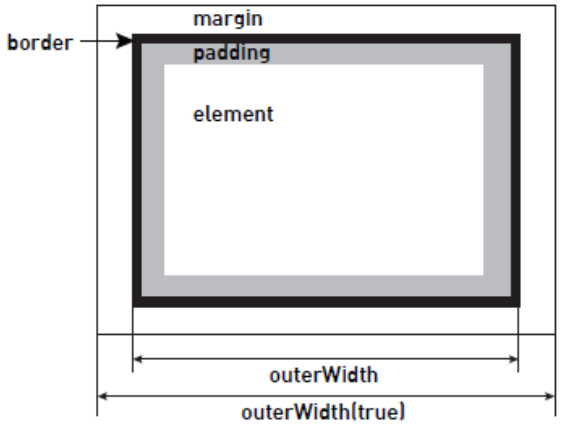
width(function) : 선택된 요소마다 각기 다른 값으로 폭을 설정한다.

function(index, oldValue) → 리턴 값은 새로이 설정하는 값

- width(), outerWidth(), innerWidth()





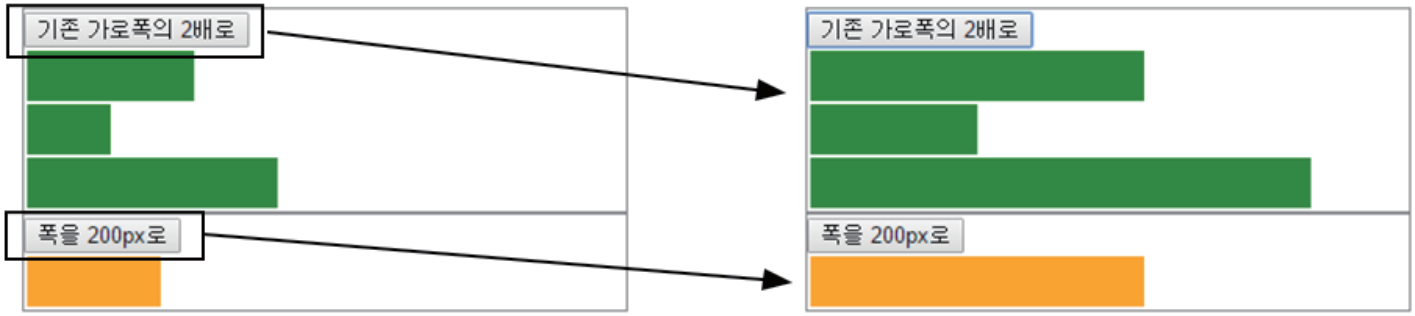


요소의 크기와 위치 조작하기(3)

예제 10-13

```
29:   <div id="content1" class="box">
30:      <button id="enlarge1">기존 가로폭의 2배로</button>
31:      <div id="a1"></div>
32:      <div id="a2"></div>
33:      <div id="a3"></div>
34:   </div>
35:   <div id="content2" class="box">
36:      <button id="enlarge2">폭을 200px로</button>
37:      <div id="b"></div>
38:   </div>
```

```
16:   $("#enlarge1").click(function() {
17:      $("#content1 > div").outerWidth(function(i, old) {
18:        return old * 2;
19:      });
20:   });
21:
22:   $("#enlarge2").click(function() {
23:      $("#b").width(200);
24:   });
```



- width(), width(200) == css("width"), css("width", "200px")
  - 숫자를 할당하느냐 문자열을 할당하느냐...

요소의 크기와 위치 조작하기(4)

요소의 위치 조작

- 요소의 위치 조작은 드래그 앤 드롭(drag and drop) 처리나 애니메이션 처리와 같은 기능을 구현할 때 반드시 알아야 함.
- 순수 자바스크립트가 실행 속도는 빠르겠지만 생산성이 낮음
- jQuery의 위치 조작 기능은 쉽게 개발 가능

[**표 10-08** : 요소 위치 조작 메서드]

메서드	기능	
offset( )	읽기/쓰기	선택된 요소의 HTML 문서 내에서의 좌표를 획득하거나 설정한다(절대 좌표).
offsetParent( )	읽기 전용	선택된 요소의 가장 가까운 조상 요소를 획득한다. 선택된 요소를 포함하고 있는 부모 요소를 가리킨다.
position( )	읽기 전용	선택된 요소의 부모 요소 영역에서의 좌표를 획득한다(상대 좌표).
scrollLeft( )	읽기/쓰기	좌우 스크롤바가 왼쪽에서부터 스크롤된 픽셀(px)값을 획득하거나 설정한다.
scrollTop( )	읽기/쓰기	상하 스크롤바가 위에서부터 스크롤된 픽셀(px)값을 획득하거나 설정한다.

요소의 크기와 위치 조작하기(5)

CSS 를 이용한 위치 조작

- jQuery도 결국 내부적으로 이 방법을 사용함.

position 속성

static (기본값) 위치 지정을 사실상 하지 않은 상태이다. 요소를 작성한 순서대로 차곡차곡 아래로 나타낸다.

relative static과 유사하지만, static이라면 위치하게 될 좌표로부터 상대적으로 위치를 나타내기 위해 top, left, bottom, right 속성을 사용한다.

fixed 브라우저 화면(viewport)상에서 좌표를 지정한다. 따라서 스크롤이 되더라도 요소는 항상 같은 곳에 위치한다. 좌표는 top, left, bottom, right 스타일 속성으로 표현한다.

absolute 절대 좌표, 절대 위치가 지정된 가장 가까운 조상 요소로부터의 좌표를 지정한다. 만일 절대 좌표가 지정된 요소가 없다면 HTML 문서(document) 안에서의 좌표를 지정하게 된다. 좌표는 top, left, bottom, right 스타일 속성으로 표현한다.

top, left, bottom, right 속성

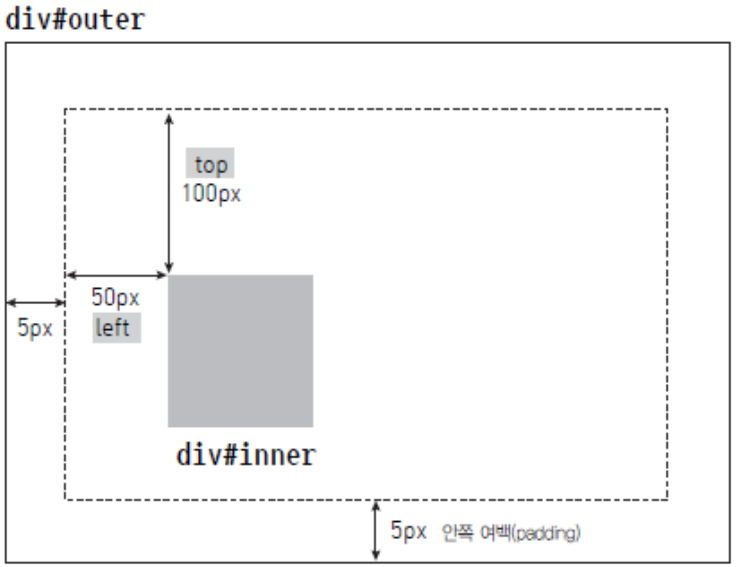
픽셀(px) 단위로 지정한다. top, left는 위쪽, 왼쪽에서부터의 좌표로 설정하는 방법이고, bottom, right는 아래쪽, 오른쪽에서부터의 좌표로 설정한다.

요소의 크기와 위치 조작하기(6)

예제 10-14 확인

```
07:     div.bg { width:800px; height:300px; background-color:aqua;
08:        margin: 5px 5px 5px 5px; }
09:     div#outer { border:solid 1px black; background-color: yellow; z-index:3;
10:         width:600px; height:400px; padding: 10px 10px 10px 10px;
11:       position:absolute; top: 100px; left:50px; }
12:     div#inner { border:solid 1px black; background-color: orange;
13:         width:100px; height:100px; position:relative; top:100px; left:50px; }
```

```
23:   <div class="bg"></div><div class="bg"></div>
24:   <div class="bg"></div><div class="bg"></div>
25:   <div id="outer">
26:      <div id="inner"></div>
27:   </div>
```

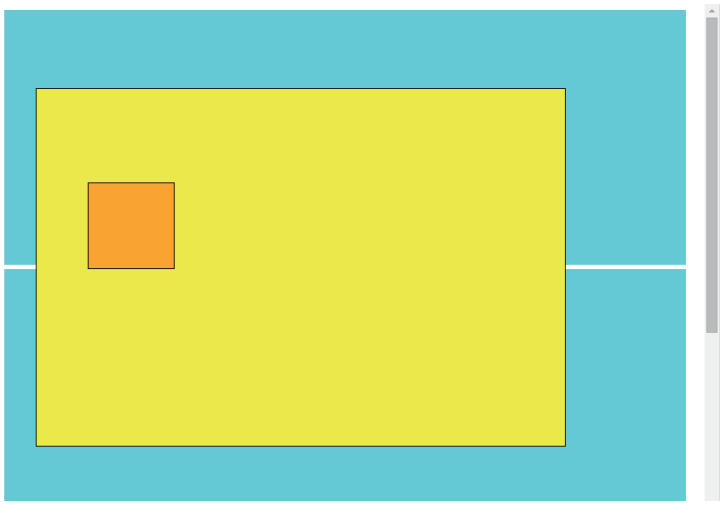


padding : 5px 5px 5px 5px
top right bottom left

```
div#inner{
   position : relative;
   top : 100px;
   left : 50px;
}
```

요소의 크기와 위치 조작하기(7)

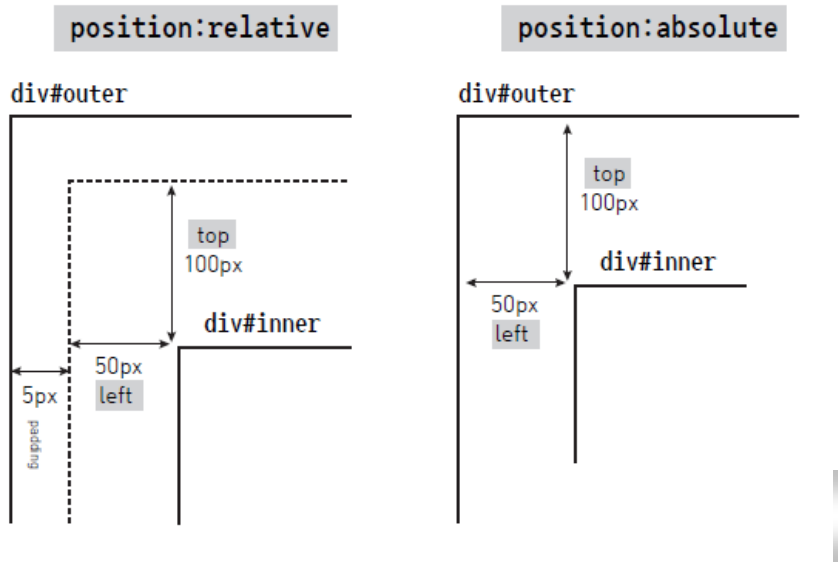
- 예제 10-14 실행 결과
  - 브라우저의 오른쪽 스크롤바를 이동시키면 div#outer가 따라 올라감.
    - div#outer가 HTML 문서(document) 영역에서의 좌표로 표현되었기 때문에...



  - 예제 10-14의 11행의 positio을 fixed로 변경하면?
    - 이제는 브라우저 화면(viewport)에서의 좌표로 설정했기에 스크롤바를 올리더라도 div#outer 요소의 화면상의 위치에는 변함이 없다

요소의 크기와 위치 조작하기(8)

- 예제 10-14에 대한 position 변경
  - div#outer : absolute
  - div#inner : relative
  - 이 결과 div#inner는 div#outer의 안쪽 여백(padding)을 제외한 영역에서의 좌표로 나타난다. (그림 10-25 참조)
- 예제 10-14에 대한 position 다시 변경
  - div#inner : absolute
  - 이 결과 div#outer 안쪽 영역에서의 좌표로 나타난다. padding 무시함.



요소의 크기와 위치 조작하기(9)

예제 10-15

- 예제 10-14에 볼드체 표현 부분만 추가 작성

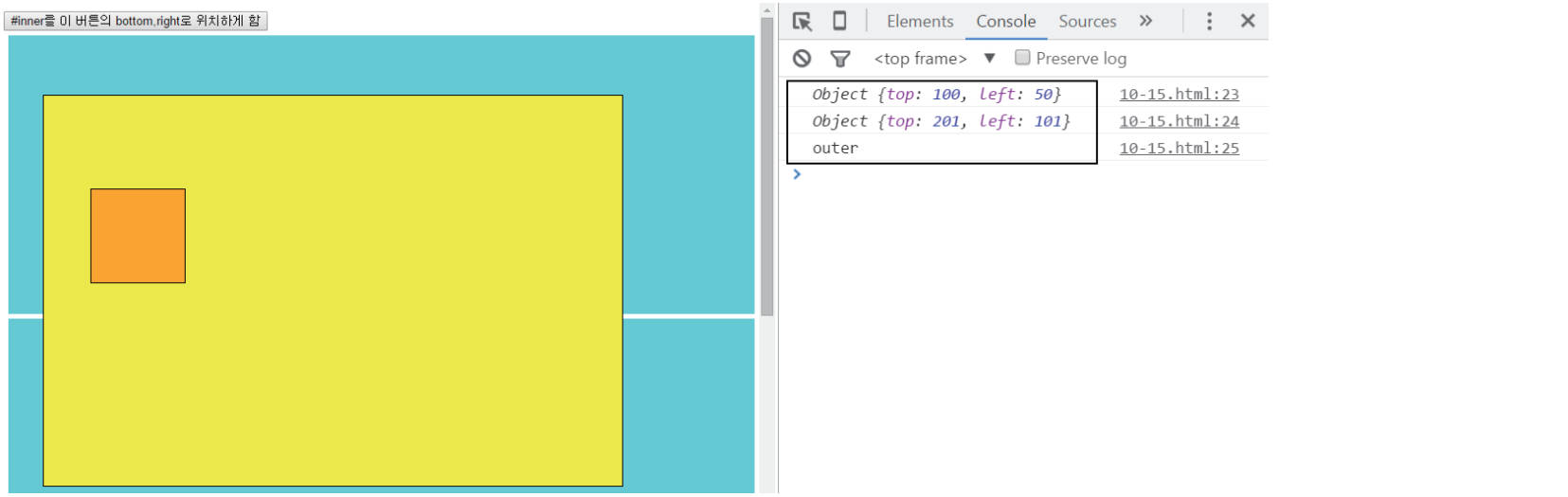
```
38:   <button id="changePos">#inner를 이 버튼의 bottom,right로 위치하게 함</button>
```

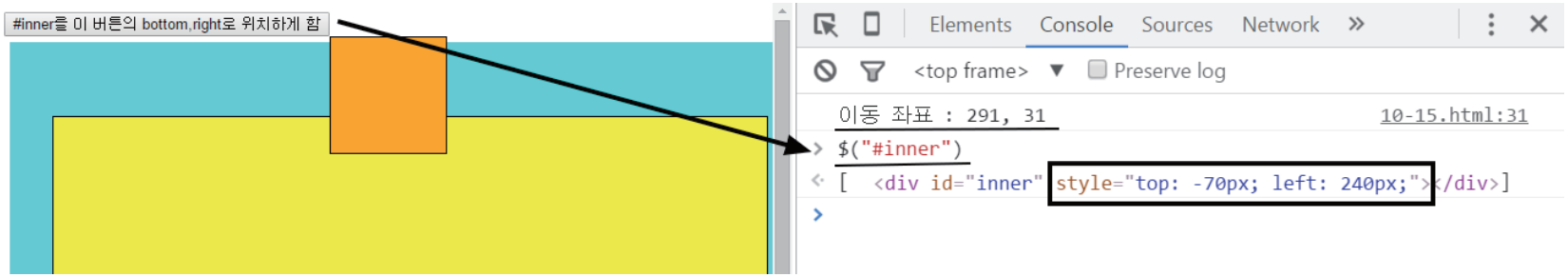
```
09:     div#outer { border:solid 1px black; background-color: yellow;
10:         width:600px; height:400px; padding: 10px 10px 10px 10px;
11:       position:absolute; top: 100px; left:50px; }
12:     div#inner { border:solid 1px black; background-color: orange;
13:         width:100px; height:100px; position:absolute; top:100px; left
```

```
18:   $("#inner").click(function() {
19:       var coor1 = $(this).position();
20:       var coor2 = $(this).offset();
21:       var parent = $(this).offsetParent();
22:
23:       console.log(coor1);
24:       console.log(coor2);
25:       console.log(parent.attr("id"));
26:   });
27:
28:   $("#changePos").click(function() {
29:       var x = $(this).offset().left + $(this).outerWidth();
30:       var y = $(this).offset().top + $(this).outerHeight();
31:       console.log("이동 좌표 : " + x + ", " + y);
32:       $("#inner").offset({ top:y, left:x });
33:   });
```

요소의 크기와 위치 조작하기(10)

- 예제 10-15 실행 결과1, 결과 2
  - 위치는 테두리(border) 두께를 고려해야 함
  - offset() 으로 좌표를 지정하면 상대 좌표로 지정됨.



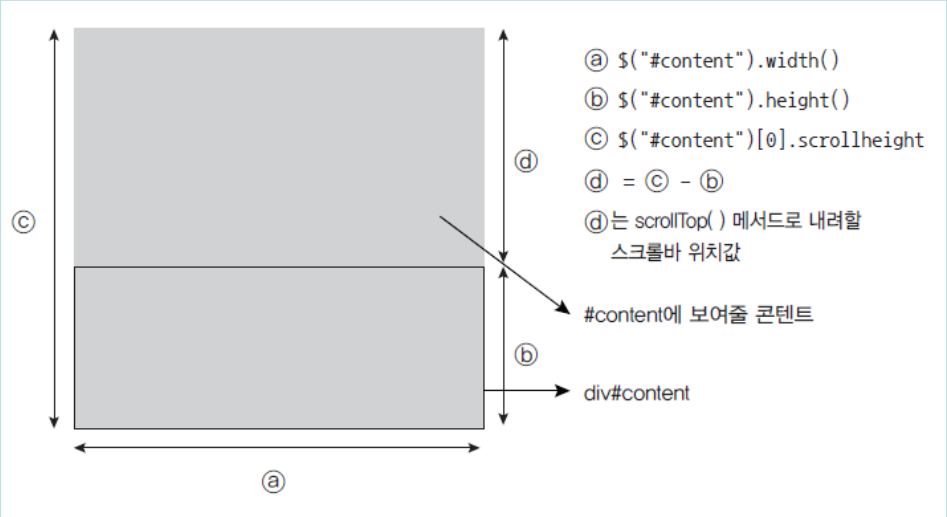


요소의 크기와 위치 조작하기(11)

scrollTop(), scrollLeft() 메서드 - 예제 10-16

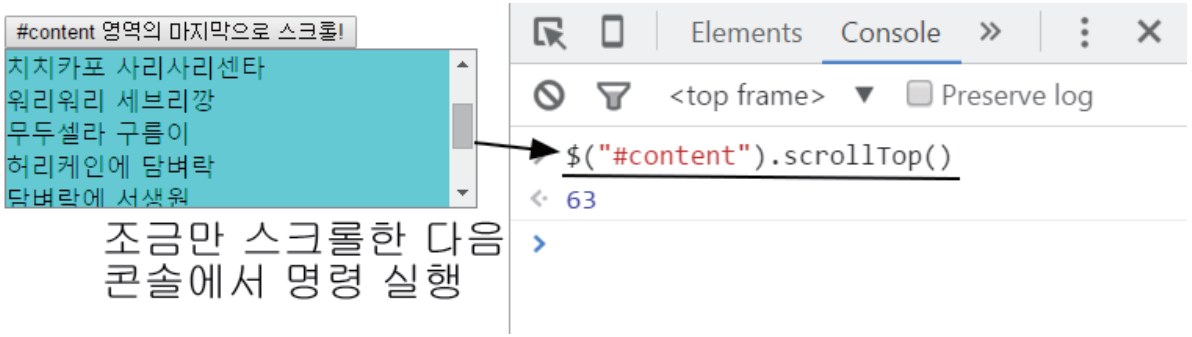
```
21:         <button id="toEnd">#content 영역의 마지막으로 스크롤!</button>
22:         <div id="content">
23:             <div id="a">김수한무</div>
24:             <div>거북이와 두루미</div>
25:             <div class="test">삼천갑자 동방삭</div>
26:             <div class="test  main">치치카포 사리사리센타</div>
27:             <div>워리워리 세브리깡</div>
28:             <div>무두셀라 구름이</div>
29:             <div id="b">허리케인에 담벼락</div>
30:             <div class="main">담벼락에 서생원</div>
31:             <div class="test">서생원에 고양이</div>
32:             <div id="a">고양이에 바둑이</div>
33:             <div>바둑이는 돌돌이</div>
34:         </div>
```

```
13:     $("#toEnd").click(function() {
14:         var st = $("#content")[0].scrollHeight - $("#content").height() + 20;
15:         $("#content").scrollTop(st);
16:     });
```



요소의 크기와 위치 조작하기(12)

- 예제 10-16 실행 결과



요소에 임의의 데이터 저장하기(1)

데이터 저장 메서드 3가지

- **data( ) 메서드** dataset 속성에 읽기와 쓰기를 한다.
  - data( ) 선택된 요소의 모든 데이터 정보를 객체로 리턴한다.
  - data(key) 선택된 요소의 데이터 중 key에 해당하는 값을 리턴한다.
  - data(key, value) key에 대한 value값을 저장한다.
  - data(object) dataset 속성에 한번에 여러 개의 키–값 쌍을 저장한다.
- **removeData( ) 메서드** 데이터 정보를 삭제한다.
  - removeData( ) 선택된 요소의 모든 데이터 정보를 삭제한다.
  - removeData(key) 특정 key에 대한 데이터값을 삭제한다.
- **$.hasData( ) 메서드** 선택된 요소가 데이터를 가지고 있는지를 판단한다.
  - $.hasData(element) 파라미터로 전달된 요소가 임의의 데이터를 저장하고 있는지를 판단한다.

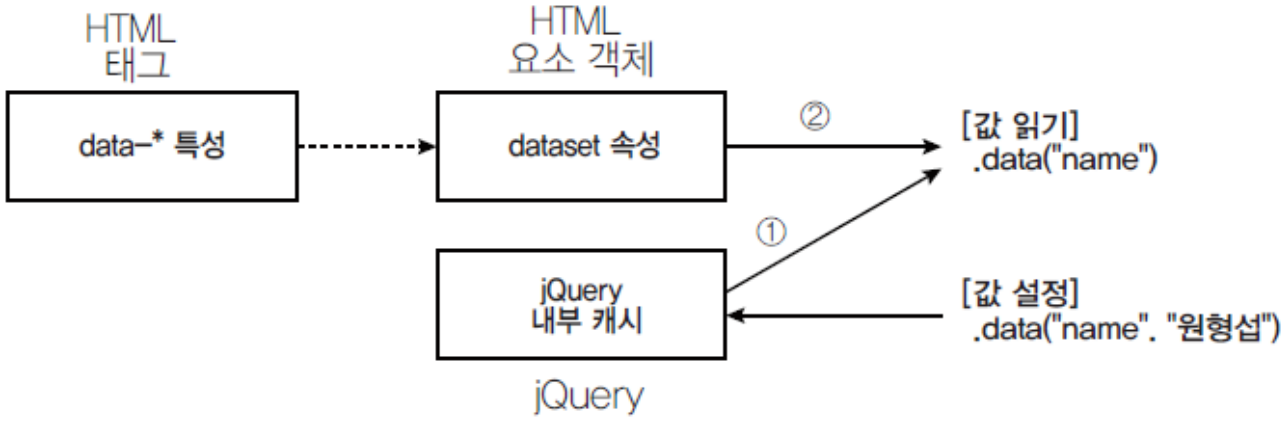
요소에 임의의 데이터 저장하기(2)

데이터 저장

- HTML 마크업에 데이터를 저장하기 위해 HTML5에서 data-* 특성을 사용하라고 가이드하고 있음

no	P-274651
name	아이패드
price	299000

...



- 값의 설정은 항상 jQuery 내부 캐시에 함
- 값의 읽기는 ①내부 캐시부터 조회를 시도하고
  ②키-값이 없으면 data-*로 초기화된 dataset 속성값 참조

요소에 임의의 데이터 저장하기(2)

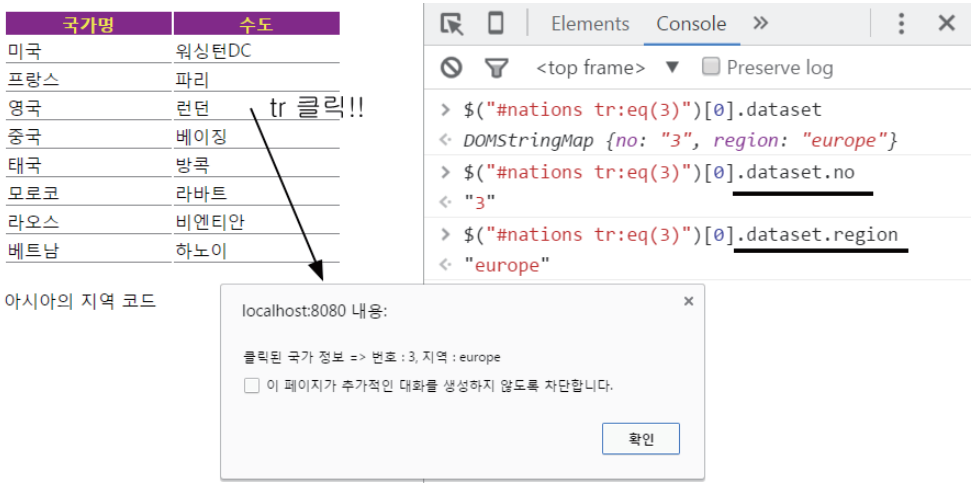
예제 10-17

```
30:     <tr data-no="1" data-region="america"><td>미국</td><td>워싱턴DC</td></tr>
31:     <tr data-no="2" data-region="europe"><td>프랑스</td><td>파리</td></tr>
32:     <tr data-no="3" data-region="europe"><td>영국</td><td>런던</td></tr>
33:     <tr data-no="4" data-region="asia"><td>중국</td><td>베이징</td></tr>
34:     <tr data-no="5" data-region="asia"><td>태국</td><td>방콕</td></tr>
35:     <tr data-no="6" data-region="africa"><td>모로코</td><td>라바트</td></tr>
36:     <tr data-no="7" data-region="asia"><td>라오스</td><td>비엔티안</td></tr>
37:     <tr data-no="8" data-region="asia"><td>베트남</td><td>하노이</td></tr>
```

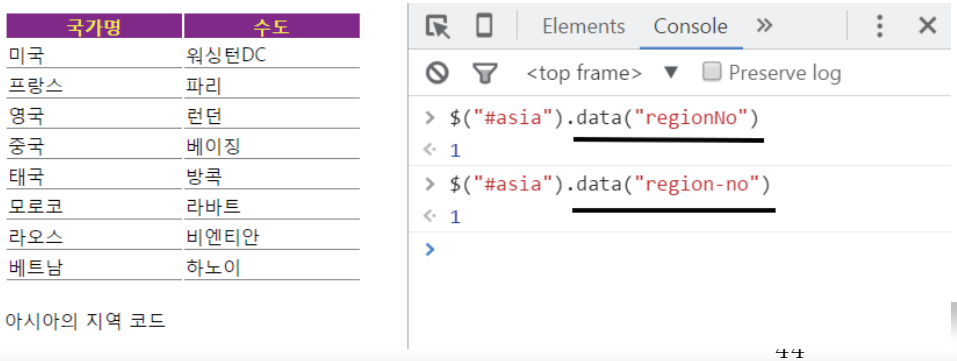
```
15:     $("#nations").find("tr:gt(0)").click(function() {
16:      //var info = $(this).data();
17:      var no = $(this).data("no");
18:      var region = $(this).data("region");
19:      alert("클릭된 국가 정보 => 번호 : " + no + ", 지역 : " +region);
20:   });
```

요소에 임의의 데이터 저장하기(3)

- 예제 10-17 실행 결과



- data-* 특성도 특성이다. 삭제를 위해서는 removeAttr() 메서드 사용
- data 메서드 사용시 dash 문자열은 생략 가능



콘텐트 조작 메서드(1)

콘텐트 조작 메서드

- HTML 마크업에서 콘텐트란 시작 태그와 종료 태그 사이의 내용 영역이라 할 수 있다

[**표 10-09** : 콘텐트 조작 메서드]

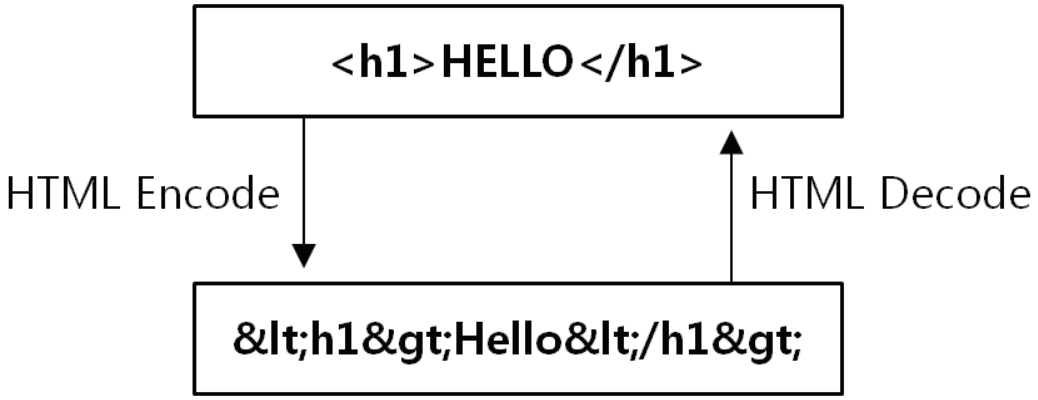
메서드	기능	설명
text( )	읽기/쓰기	선택된 요소의 콘텐트를 읽어내거나 설정한다. 이 메서드는 설정할 때는 HTML 인코딩, 읽어낼 때는 HTML 디코딩 기능을 수행한다.
html( )	읽기/쓰기	선택된 요소의 콘텐트를 읽어내거나 설정한다. text() 메서드와의 차이점은 HTML 인코딩, 디코딩을 수행하지 않는다.

- **html( )** 콘텐트를 읽어낸다
- **html(val)** 콘텐트 설정한다.
- **html(function(i, old) { })** 각 요소마다 각기 다른 값으로 콘텐트 설정한다. 함수의 old 파라미터는 요소의 기존 콘텐트이고, 새롭게 리턴하는 값으로 콘텐트가 설정된다.

콘텐트 조작 메서드(2)

text() 와 html()의 차이점

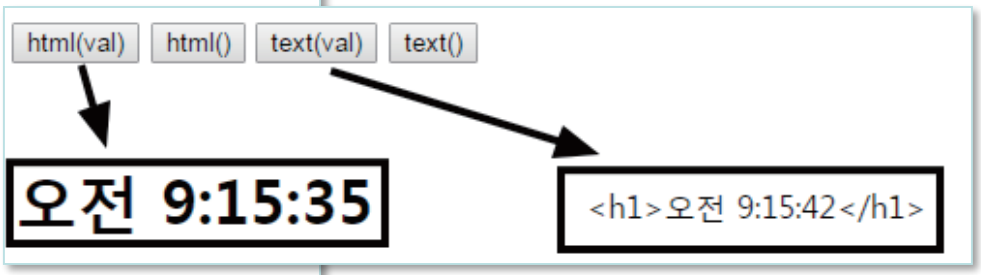
- text() : 읽기, 쓰기 시에 HtmlEncoding, HtmlDecoding을 수행한다.
- html() : 인코딩, 디코딩 하지 않는다.
- 사용 빈도는 html() 메서드가 더 높다



콘텐트 조작 메서드(3)

예제 10-18

```
09:     function createTimeString() {
10:      return "<h1>" + (new Date()).toLocaleTimeString() +"</h1>";
11:   }
12:
13:   $("#savehtml").click(function() {
14:      $("#content").html(createTimeString());
15:   });
16:   $("#readhtml").click(function() {
17:      var s = $("#content").html();
18:      alert(s);
19:   });
20:   $("#savetext").click(function() {
21:      $("#content").text(createTimeString());
22:   });
23:   $("#readtext").click(function() {
24:      var s = $("#content").text();
25:      alert(s);
26:   });
```



HTML DOM에 요소 추가하기(1)

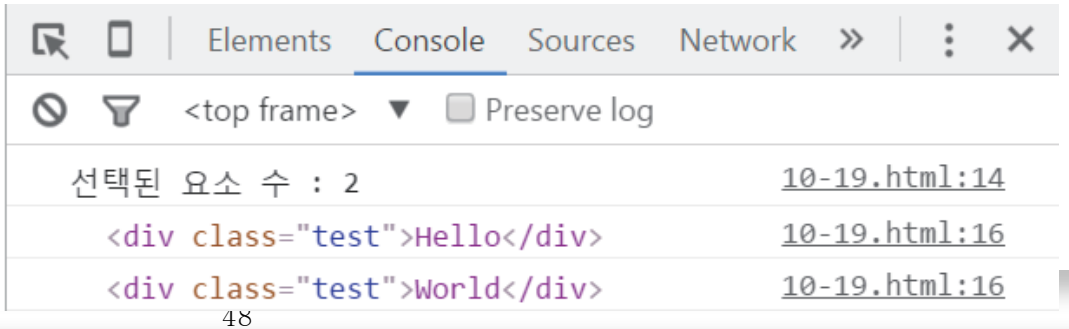
요소 생성

- jQuery() 함수의 또다른 기능
  - 태그 형태의 문자열을 첫번째 인자로 전달할 경우는 새로운 요소 객체를 생성하는 역할을 하며 가장 바깥쪽 요소를 선택함.
- 예제 10-19

```
13:   var temp = "<div class='test'>Hello</div><div class='test'>World</div>";
14:   var divs = $(temp);
15:   console.log("선택된 요소 수 : " + divs.length);
16:   for (var i=0; i < divs.length; i++) {
17:      console.log(divs[i]);
18:   }
19:   $("#content").html(divs);
```

Hello

World



HTML DOM에 요소 추가하기(2)

- 생성된 요소는 횡단탐색 메서드로 이동하면서 값을 조작할 수 있음.
- 예제 10-20

```
09:   //--bulk data 생성
10:   console.time("create bulk data");
11:   var data = [];
12:   for (var i=1; i <= 10000; i++) {
13:      data.push({
14:         no:i ,
15:         name:"홍길동"+i,
16:         score:50+i%50,
17:         tel : "010-2121-313"+(i%10)
18:      });
19:   }
20:   console.timeEnd("create bulk data");
```

```
22:   //--bulk data를 div#content에 출력
23:   console.time("create Element by $ function");
24:   var temp1 = "<div><span></span>" +
25:      "<button class='detail'>상세보기</button></div>";
26:   for (var i=0; i < data.length; i++) {
27:      var item = data[i];
28:      $(temp1).attr("id", "d_"+item.no)
29:         .find("span").html(item.name + " : " + item.score + ", " + item.tel)
30:         .next().attr("data-no", item.no).parent().appendTo("#content");
31:   }
32:   console.timeEnd("create Element by $ function");
```

console.time( ) 메서드와 console.timeEnd( ) 메서드

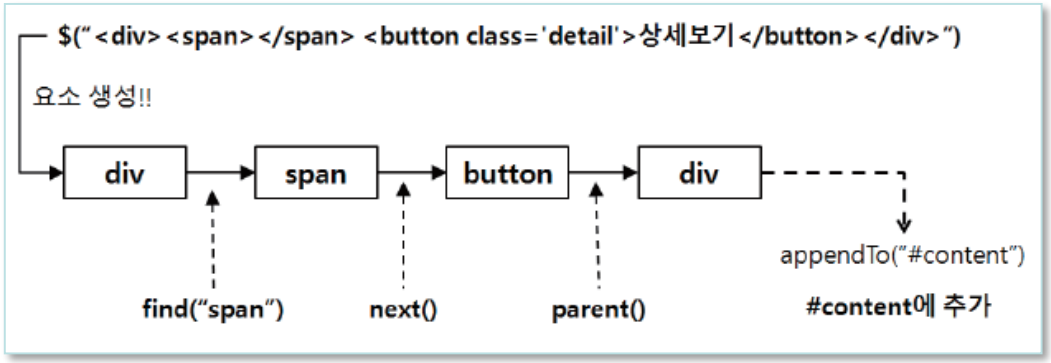
console.time("A"), console.timeEnd("A") 메서드는 실행 시간 측정을 위해서 사용한다. time("A")부터 timeEnd("A")까지 구간의 실행 시간을 측정해 콘솔에 출력한다.

HTML DOM에 요소 추가하기(3)

- 예제 10-20 실행 결과
  - 다음과 같은 형식의 HTML 요소를 10000개 생성

```
<div id="d_1">
   <span>홍길동1 : 51, 010-2121-3131</span>
   <button class="detail" data-no="1">상세보기</button>
</div>
```

  - 각각의 개별 요소는 다음과 같이 횡단탐색 메서드를 거치면서 요소 값 설정함



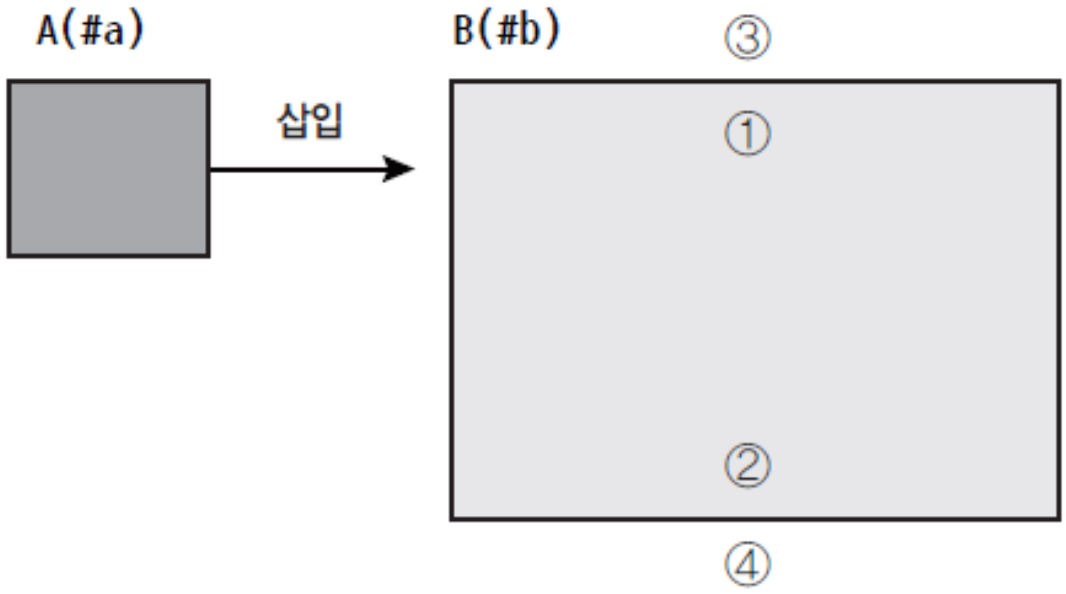
노트

개발 생산성 측면이나 코드의 유지 보수 측면, 성능 측면에서 보자면 예제 10-20은 그리 바람직한 코드는 아니다. HTML 요소를 자바스크립트 코드로 직접 조작하므로 개발자와 웹디자이너 모두 유지 보수하기 힘들 수밖에 없는 코드이다. 또한 $() 함수로 요소를 만드는 것은 성능상 바람직하지 않다. 오히려 문자열을 + 연산자로 이어 붙여 두었다가 한번에 요소의 콘텐트로 설정하는 것이 더 성능이 좋다.

HTML DOM에 요소 추가하기(4)

요소 추가하기

- 요소를 추가하는 관점은 두가지
  - 추가하는 요소를 중심으로 코드를 작성할 것인지(A를 중심으로)
  - 추가되는 위치의 요소를 중심으로 코드를 작성할 것인지(B를 중심으로)



HTML DOM에 요소 추가하기(5)

- 요소 추가 메서드 1
  - A를 중심으로 한 메서드

[**표 10-10** : 요소 추가 메서드 1]

메서드	위치	설명
$(A).prependTo(B)	①	A를 B 내부 콘텐트의 앞에 삽입한다. 기존 콘텐트가 있다면 뒤로 밀려난다.
$(A).appendTo(B)	②	A를 B 내부의 콘텐트 마지막에 추가한다.
$(A).insertBefore(B)	③	A를 B 요소와 동일한 수준의 앞 요소로 삽입한다.
$(A).insertAfter(B)	④	A를 B 요소와 동일한 수준의 다음 요소로 삽입한다.

- 요소 추가 메서드 2
  - B를 중심으로 한 메서드

[**표 10-11** : 요소 추가 메서드 2]

메서드	위치	설명
$(B).prepend(A)	①	B 안의 맨 앞에 A를 삽입한다.
$(B).append(A)	②	B 안의 마지막에 A를 추가한다.
$(B).before(A)	③	B 요소의 바로 앞에 A를 추가한다.
$(B).after(A)	④	B 요소의 바로 다음에 A를 추가한다.

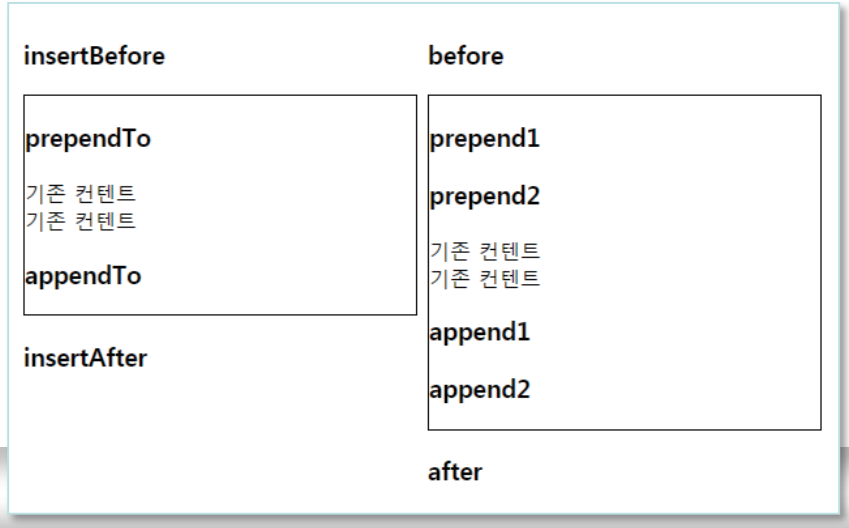
HTML DOM에 요소 추가하기(6)

- 예제 10-21

```
13:   $("<h3>prependTo</h3>").prependTo("#content1");
14:   $("<h3>appendTo</h3>").appendTo("#content1");
15:   $("<h3>insertBefore</h3>").insertBefore("#content1");
16:   $("<h3>insertAfter</h3>").insertAfter("#content1");
17:
18:   $("#content2").prepend("<h3>prepend1</h3>", "<h3>prepend2</h3>");
19:   $("#content2").append("<h3>append1</h3>", "<h3>append2</h3>");
20:   $("#content2").before("<h3>before</h3>");
21:   $("#content2").after("<h3>after</h3>");
```

```
26:   <div id="container1">
27:      <div id="content1">
28:         <div>기존 콘텐트</div>
29:         <div>기존 콘텐트</div>
30:      </div>
31:   </div>
32:   <div id="container2">
33:      <div id="content2">
34:         <div>기존 콘텐트</div>
35:         <div>기존 콘텐트</div>
36:      </div>
37:   </div>
```

- 실행 결과



HTML DOM에 요소 추가하기(7)

요소 교체, 제거하기

- 지원 메서드

[**표 10-12** : 요소 교체, 제거 메서드]

메서드	설명
replaceWith( )	선택된 요소를 새로운 콘텐트로 교체한다. $(A).replaceWith(B) : A를 B로 교체한다.
replaceAll( )	선택된 요소를 새로운 HTML 요소로 교체한다. $(B).replaceAll(A) : B로 A를 대체한다.
empty( )	선택된 요소의 콘텐트(모든 자식 노드)를 제거한다.
remove( )	선택된 요소를 DOM 구조에서 제거한다.
detach( )	선택된 요소를 DOM 구조에서 제거한다. remove( )와 차이점은 제거한 요소를 리턴받아 저장해둘 수 있고, 나중에 다시 DOM에 추가할 수 있다.

- replaceWith()나 replaceAll() 메서드로 요소를 문서 내의 기존 요소로 교체하면 기존 요소의 이동이 일어난다는 점을 주의해야 함

HTML DOM에 요소 추가하기(8)

- replaceWith(), replaceAll()
- 예제 10-22

```
16:   <div id="content">
17:       <div id="a">김수한무</div>
18:       <div>거북이와 두루미</div>
19:       <div class="test">삼천갑자 동방삭</div>
20:       <div class="test  main">치치카포 사리사리센타</div>
21:       <div id="d">워리워리 세브리깡</div>
22:       <div>무두셀라 구름이</div>
23:       <div id="b">허리케인에 담벼락</div>
24:       <div class="main">담벼락에 서생원</div>
25:       <div class="test">서생원에 고양이</div>
26:       <div id="a">고양이에 바둑이</div>
27:       <div id="c">바둑이는 돌돌이</div>
28:   </div>
```

```
09:   $("<h2>.test를 h2로 대체함.</h2>").replaceAll(".test");
10:   $("#b").replaceWith("<h2>#b를 h2로 대체함.</h2>");
11:   $("#c").replaceWith($("#d"));
```

김수한무
거북이와 두루미

.test를 h2로 대체함.

.test를 h2로 대체함.

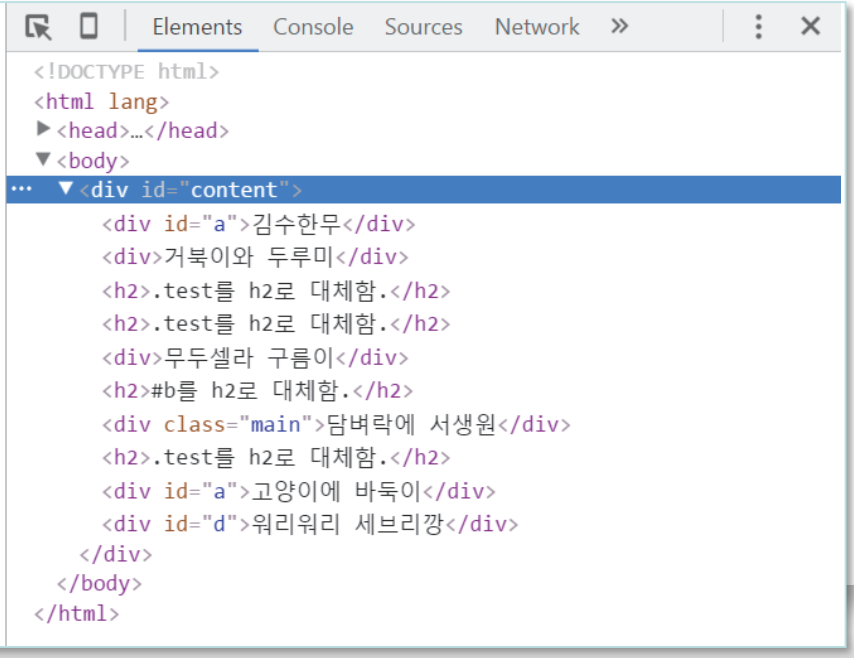
무두셀라 구름이

#b를 h2로 대체함.

담벼락에 서생원

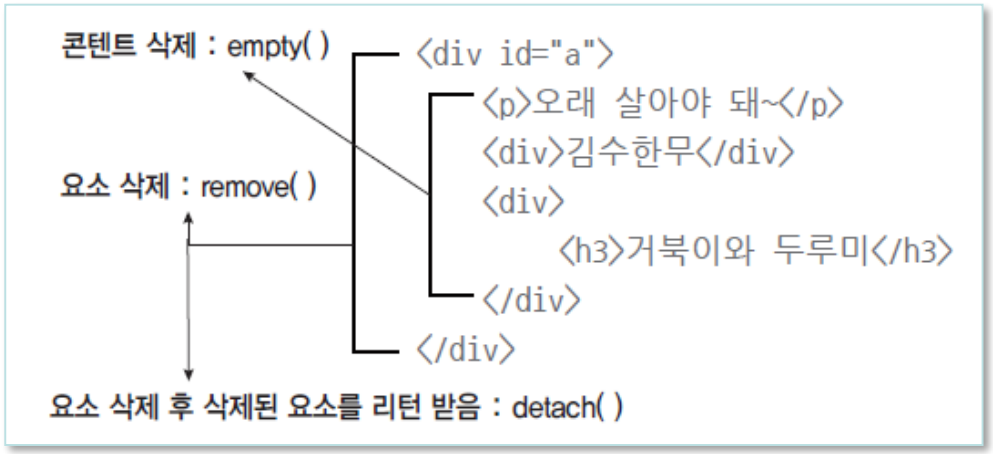
.test를 h2로 대체함.

고양이에 바둑이
워리워리 세브리깡



HTML DOM에 요소 추가하기(9)

- remove, empty(), detach()
  - 각 메서드의 삭제 대상(선택된 요소가 div#a라고 가정함)



HTML DOM에 요소 추가하기(10)

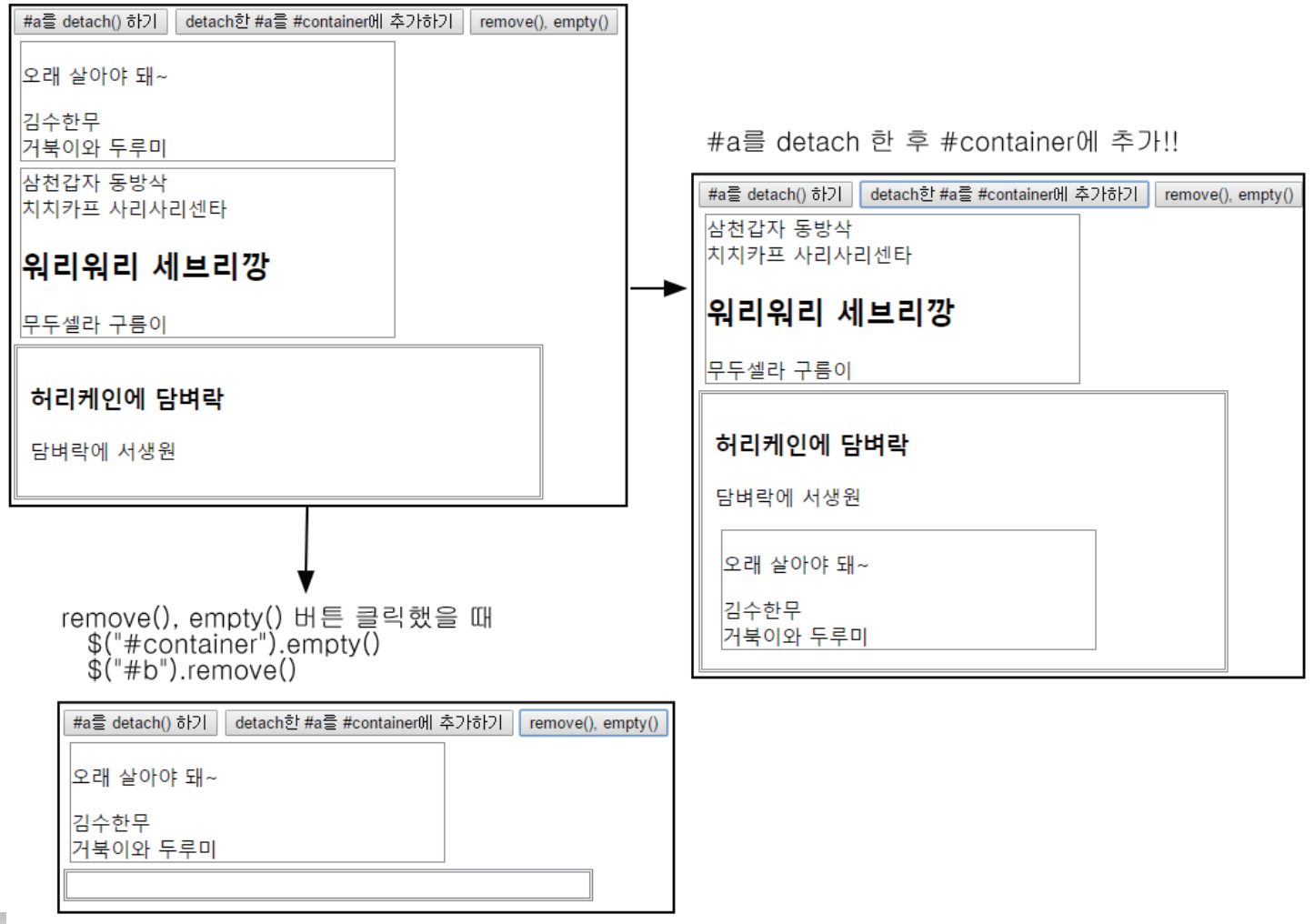
- 예제 10-23

```
33:  <button id="detach">#a를 detach() 하기</button>
34:  <button id="append">detach한 #a를 #container에 추가하기</button>
35:  <button id="delete">remove(), empty()</button>
36:    <div id="a">
37:        <p>오래 살아야 돼~</p>
38:        <div>김수한무</div>
39:        <div>거북이와 두루미</div>
40:    </div>
41:    <div id="b">
42:        <div>삼천갑자 동방삭</div>
43:        <div>치치카프 사리사리센타</div>
44:        <h2>워리워리 세브리깡</h2>
45:        <div>무두셀라 구름이</div>
46:    </div>
47:    <div id="container">
48:        <h3>허리케인에 담벼락</h3>
49:        <p>담벼락에 서생원</p>
50:    </div>
```

```
13: $(document).ready(function() {
14:   var detached;
15:   $("#delete").click(function() {
16:      $("#container").empty();
17:      $("#b").remove();
18:   });
19:   $("#detach").click(function() {
20:      detached = $("#a").detach();
21:   });
22:   $("#append").click(function() {
23:      $("#container").append(detached);
24:   });
25:   //이벤트 연결 정보 유실 여부를 확인하기 위한 코드
26:   $("#a > p").click(function() {
27:      alert("detach해두었다가 추가한 요소는 이벤트 연결 정보가 유실되지 않음.");
28:   });
29: });
```

HTML DOM에 요소 추가하기(11)

- 예제 10-23 실행 결과



요소를 다른 요소로 감싸기(1)

- 선택된 요소를 다른 요소로 감싸는 요소 조작 메서드
- 지원 메서드

[**표 10-13** : 요소 교체, 제거 메서드]

메서드	설명
wrap( )	선택된 요소를 파라미터로 전달한 요소로 감싼다.
wrapInner( )	선택된 요소의 콘텐트를 파라미터로 전달한 요소로 감싼다.
unwrap( )	선택된 요소의 태그 부분을 제거하고 콘텐트 영역만 남긴다.
wrapAll( )	선택된 요소를 한군데로 모아 한꺼번에 감싼다.

요소를 다른 요소로 감싸기(2)

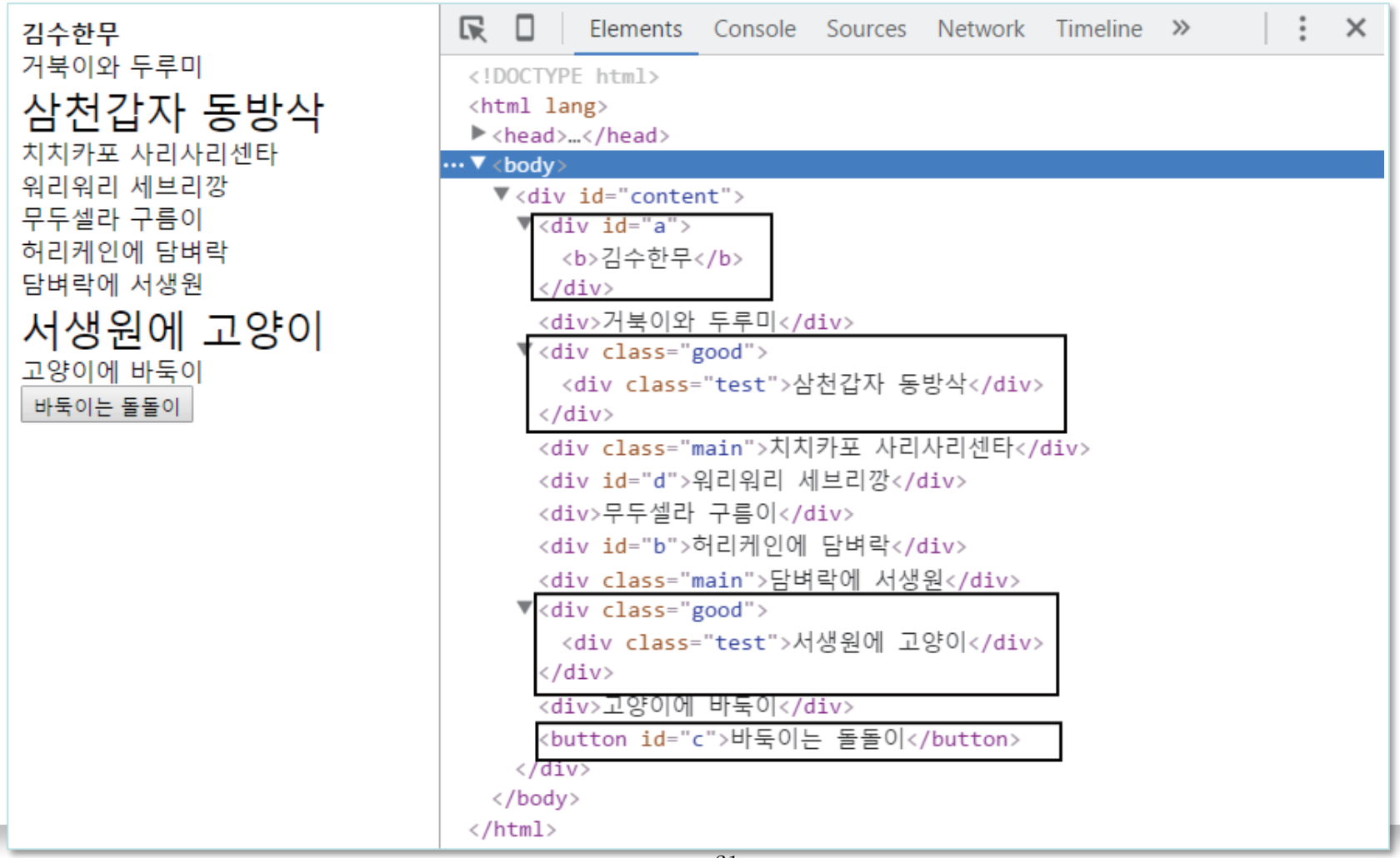
- 예제 10-24

```
19:  <div id="content">
20:      <div id="a">김수한무</div>
21:      <div>거북이와 두루미</div>
22:      <div class="test">삼천갑자 동방삭</div>
23:      <div class="main">치치카포 사리사리센타</div>
24:      <div id="d">워리워리 세브리깡</div>
25:      <div>무두셀라 구름이</div>
26:      <div id="b">허리케인에 담벼락</div>
27:      <div class="main">담벼락에 서생원</div>
28:      <div class="test">서생원에 고양이</div>
29:      <div>고양이에 바둑이</div>
30:      <p>
31:        <button id="c">바둑이는 돌돌이</button>
32:      </p>
33:  </div>
```

```
12:  $("#a").wrapInner("<b></b>");
13:  $(".test").wrap("<div class='good'></div>");
14:  $("#c").unwrap();
```

요소를 다른 요소로 감싸기(3)

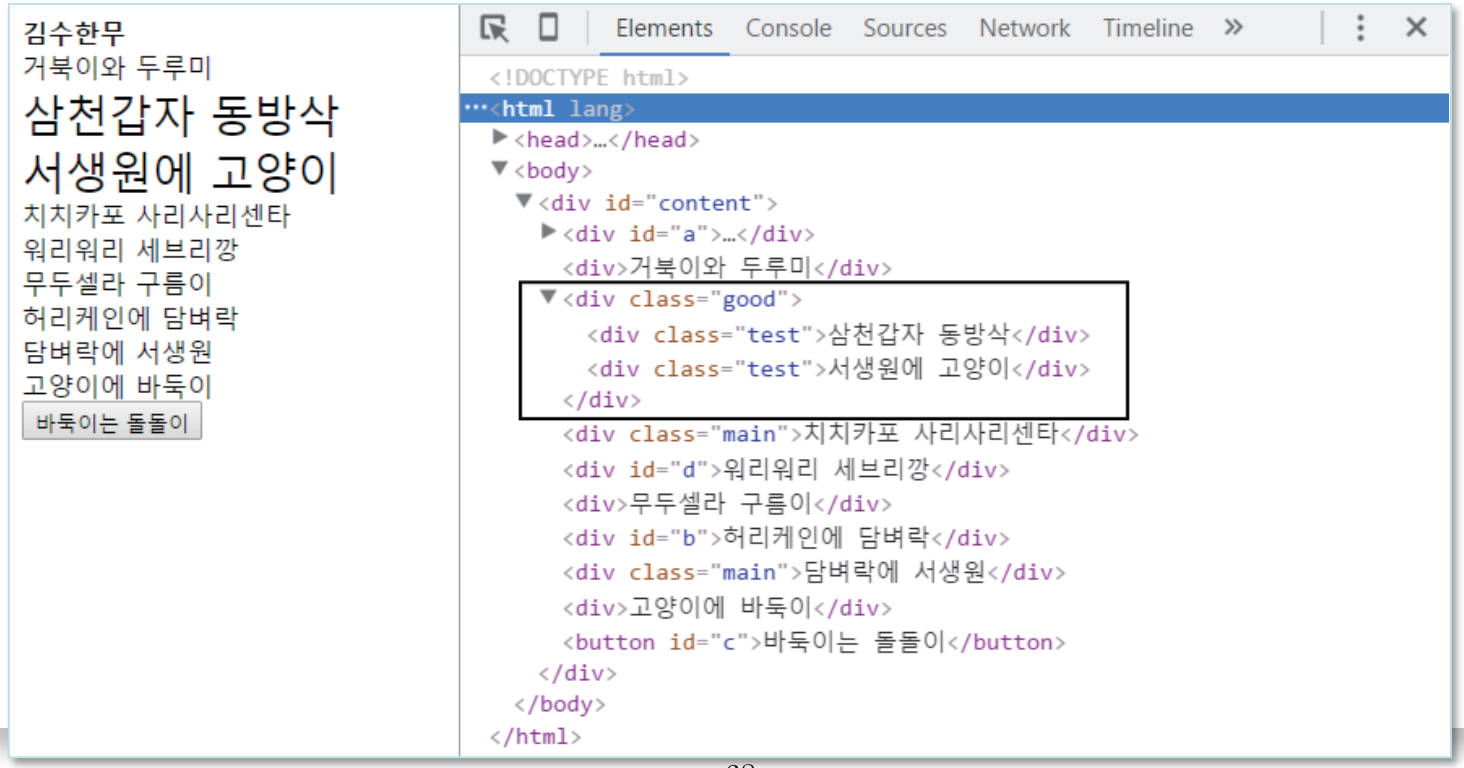
- 예제 10-24 실행 결과



요소를 다른 요소로 감싸기(4)

- wrapAll() 메서드는 wrap()과 작동 방식이 조금 다르다.
  - wrapAll() 메서드는 선택된 요소를 한군데로 모아서 지정한 태그로 감싼다.

```
$(".test").wrapAll("<div class='good'></div>");
```



요소의 이동과 복제(1)

요소의 이동

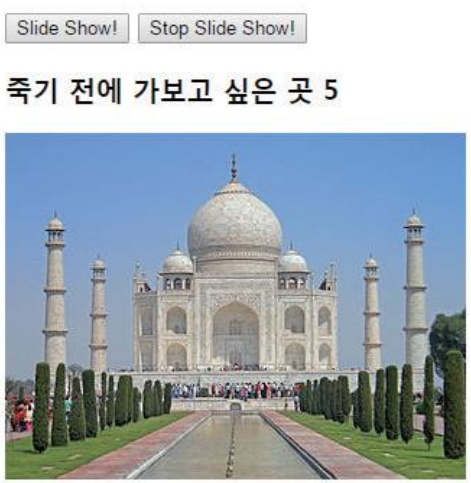
- HTML 문서의 기존 요소를 선택해 추가할 때는 이동!!
- 예제 10-25 : 슬라이드 쇼

```
31:  <button id="slideStart">Slide Show!</button>
32:  <button id="slideStop">Stop Slide Show!</button>
33:  <h3>죽기 전에 가보고 싶은 곳 5</h3>
34:  <div id="slider">
35:     <img class="scape_hidden" src="images/capetown.jpg" />
36:     <img class="scape_hidden" src="images/chichenitza.jpg" /
37:     <img class="scape_hidden" src="images/machu.jpg" />
38:     <img class="scape_hidden" src="images/niagarafalls.jpg"
39:     <img class="scape_hidden" src="images/tajmahal.jpg" />
40:  </div>
```

```
12: $(document).ready(function() {
13:    var handle;
14:    function viewSlide() {
15:       var slider = $("#slider");
16:       slider.find("img.scape_visible").removeClass("scape_visible");
17:       slider.find("img:first").appendTo(slider).addClass("scape_visible");
18:    }
19:
20:    $("#slideStart").click(function() {
21:       handle = setInterval(viewSlide, 2000);
22:    });
23:    $("#slideStop").click(function() {
24:       clearInterval(handle);
25:    });
26:    viewSlide();
27: });
```

요소의 이동과 복제(2)

- 예제 10-25 실행 결과




```
<!DOCTYPE html>
<html lang>
▶ <head>…</head>
▼ <body>
    <button id="slideStart">Slide Show!</button>
    <button id="slideStop">Stop Slide Show!</button>
    <h3>죽기 전에 가보고 싶은 곳 5</h3>
  ▼ <div id="slider">
      <img class="scape_hidden" src="images/capetown.jpg">
      <img class="scape_hidden" src="images/chichenitza.jpg">
      <img class="scape_hidden" src="images/machu.jpg">
      <img class="scape_hidden" src="images/niagarafalls.jpg">
      <img class="scape_hidden scape_visible" src="images/tajmahal.jpg">
    </div>
  </body>
</html>
```

- 핵심은 viewSlide() 함수
  - #slider 내에서 현재 화면에 보여지는 img(클래스가 scape_visible인 요소)를 찾아서 removeClass를 호출하여 보이지 않게 처리함
  - 그후 img 중에서 첫 번째 요소를 선택하여 appendTo(slider) 메서드를 호출함으로써 마지막에 추가함.
  - 이동한 요소는 scape_visible 클래스를 추가함으로써 화면에 나타남

요소의 이동과 복제(3)

요소의 복제

- clone() 메서드의 3가지 방법

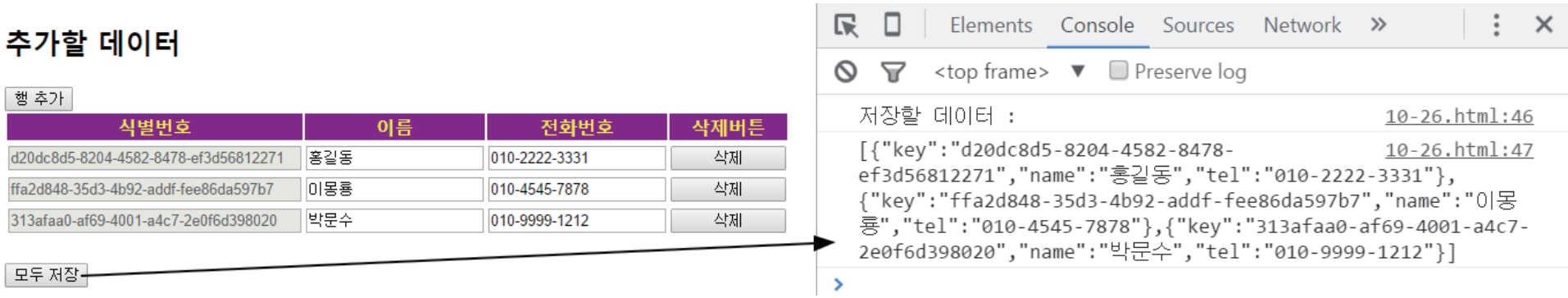
[**표 10-14** : clone( ) 메서드]

메서드	설명
clone( )	파라미터 없이 호출하면 이벤트와 요소 관련 데이터 정보는 복제하지 않는다.
clone(true)	true값을 첫 번째 파라미터로 전달하면 이벤트와 요소 관련 데이터 정보를 함께 복제한다.
clone(true, true)	두 번째 파라미터로 true를 전달하면 모든 후손 요소에 설정된 이벤트와 요소 관련 데이터 정보를 복제한다.

- 예제 10-26
  - 코드량이 조금 많다.
  - 보이지 않는 테이블(#temptable)의 row(#temprow)를 선택해 복제(clone)한 후 화면에 보여줄 테이블의 tbody 요소(#rowbody)에 추가한다.
  - 추가된 테이블에 고유키(UUID)를 생성해 할당해주고 나머지 값을 입력받는다.
  - 모두 입력되면 입력 "모두 저장" 버튼을 클릭하면 객체 배열 값으로 저장하고 전송한다.

요소의 이동과 복제(4)

- 예제 10-26 실행 결과



- 서버로 전송하는 기능을 구현하는 대신 콘솔에 객체 배열을 json으로 변환한 다음 출력하도록 하였다.

정리

- **모든 메서드를 암기하고 적용할 필요는 없다.**
  - 예를 들어 empty() 메서드는 html("") 메서드로도 대체할 수 있음
- **메서드의 본래의 목적과 기능을 이해하는 것이 중요하다.**
  - 예를 들면 attr(), prop() 메서드의 기능적 차이
  - 메서드 본래의 기능을 알아두는 것이 개발에 훨씬 더 도움
- **요소의 설정이나 읽기가 요소 자체의 어떤 정보를 변경하거나 이용하는지도 알아두자.**
  - 브라우저 개발자 도구를 적극 활용해야 한다. 디버깅에도 도움이 된다.
- **모든 메서드의 용법을 기억할 필요 없다.**
  - 대부분 일정한 패턴을 가지고 있다.

"메서드의 원래 목적과 기능, 요소에 직접 끼치는 영향을 기억하고, 메서드 호출 패턴을 잘 익혀두도록 하자"