

# 16장 HTML5와 제이쿼리

---

16.1 HTML5 주요 개념

16.2 HTML5 시맨틱 웹을 위한 구성 요소

16.3 제이쿼리 주요 개념

16.4 제이쿼리의 여러 가지 기능

16.5 제이쿼리 Ajax 기능

16.6 제이쿼리에서 JSON 사용하기

## 16.1 HTML5 주요 개념

### HTML5의 여러 가지 기능

기능	설명
WebForm	입력 형태를 보다 다양하게 제공합니다.
Video	동영상 재생을 위한 API를 제공합니다.
Audio	음성 재생을 위한 API를 제공합니다.
Offline Web	인터넷 연결이 되지 않은 상태에서도 정상적인 기능을 지원하는 API를 제공합니다.
WebDataBase	표준 SQL을 사용해 데이터를 저장할 수 있는 기능을 제공합니다.
WebStorage	웹 애플리케이션에서 데이터를 저장할 수 있는 기능을 제공합니다.
Canvas	2차원 그래픽 그리기 및 객체에 대한 각종 효과를 주는 기능을 제공합니다.
SVG	XML 기반 2차원 벡터 그래픽을 표현하기 위한 SVG 언어를 지원합니다.
Geolocation	디바이스의 지리적 위치 정보를 가져오는 기능을 제공합니다.
WebWorker	웹 애플리케이션을 위한 스레드 기능을 제공합니다.
WebSocket	애플리케이션과 서버 간의 양방향 통신 기능을 제공합니다.
CSS3	웹 애플리케이션의 다양한 스타일 및 효과를 나타내기 위한 CSS3를 제공합니다.

## 16.1 HTML5 주요 개념

- HTML4와 HTML5의 문서 구조

### 코드 HTML4의 문서 구조

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  <title>HTML4 구조</title>
</head>
<body>
  hello world!!!
</body>
</html>
```

## 16.1 HTML5 주요 개념

### 코드 HTML5의 문서 구조

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8" />
  <title>HTML5 구조</title>
</head>
<body>
  hello world!!!
</body>
</html>
```

## 16.2 HTML5 시맨틱 웹을 위한 구성 요소

### 시맨틱 웹(Semantic Web) 정의

- 기계가 이해할 수 있고 처리할 수 있는 웹 콘텐츠를 만드는 것
- 1998년 월드와이드웹(WWW)의 창시자인 팀 버너스 리(Tim Berners Lee)에 의해 개발되고 정의된 개념

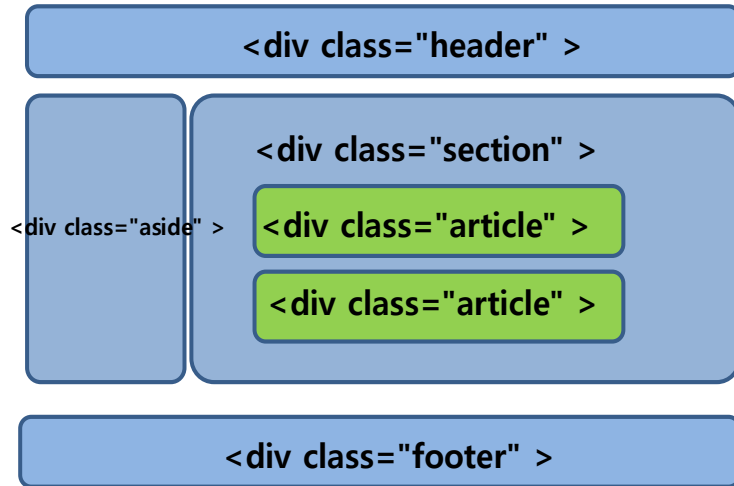
### • 16.1.1 JSP 페이지에 이미지 포함 실습

#### HTML5에 추가된 여러 가지 태그

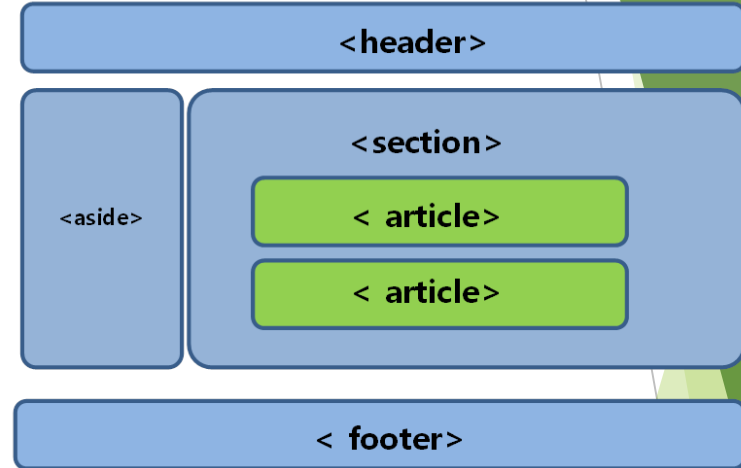
태그	설명
<header>	머리말을 나타내는 태그
<hgroup>	제목과 부제목을 묶는 태그
<nav>	메뉴 부분을 나타내는 태그
<section>	제목별로 나눌 수 있는 태그
<article>	개별 콘텐츠를 나타내는 태그
<aside>	왼쪽 또는 오른쪽에 위치하는 사이드 바를 나타내는 태그
<footer>	하단의 정보를 표시하는 태그

## 16.2 HTML5 시맨틱 웹을 위한 구성 요소

HTML4를 이용한 화면 레이아웃



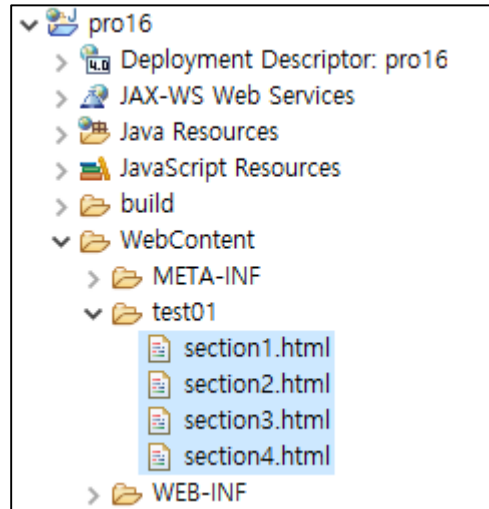
HTML5를 이용한 화면 레이아웃



## 16.2 HTML5 시맨틱 웹을 위한 구성 요소

- 16.2.2 HTML5 웹 페이지 구조 관련 태그 사용

1. 새 프로젝트 pro16을 만들고, test01 폴더를 만든 다음 section1.html, section2.html, section3.html, section4.html을 추가합니다.



## 16.2 HTML5 시맨틱 웹을 위한 구성 요소

2. 다음과 같이 section1.html을 작성합니다.

**코드 16-1** pro16/WebContent/test01/section1.html

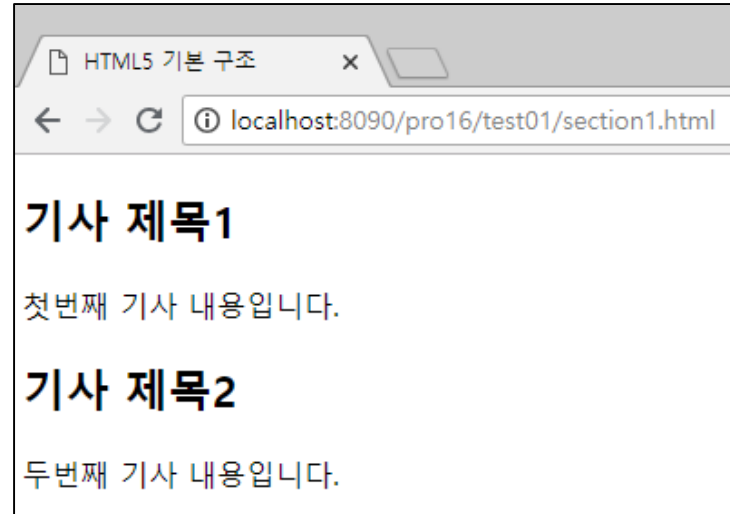
```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>HTML5 기본 구조</title>
</head>
<body>
  <section>
    <h1> 기사 제목1 </h1>
    <p>첫 번째 기사 내용입니다.</p>
  </section>
  <section>
    <h1> 기사 제목2 </h1>
    <p>두 번째 기사 내용입니다.</p>
  </section>
</body>
</html>
```

— <section> 태그로 제목을 표시합니다.



## 16.2 HTML5 시맨틱 웹을 위한 구성 요소

3. <http://localhost:8090/pro16/test01/section1.html>로 요청하여 결과를 확인합니다.



## 16.2 HTML5 시맨틱 웹을 위한 구성 요소

4. section2.html을 다음과 같이 작성합니다. <section> 태그 안에 <article> 태그를 사용해 본문을 표시합니다.

코드 16-2 pro16/WebContent/test01/section2.html

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>...</head>
```

```
<body>
```

```
  <section>
```

```
    <article>
```

```
      <h1>기사 제목1</h1>
```

```
      <p>첫 번째 기사의 내용 </p>
```

```
    </article>
```

```
    <article>
```

```
      <h1>기사 제목2</h1>
```

```
      <p>두 번째 기사의 내용</p>
```

```
    </article>
```

```
  </section>
```

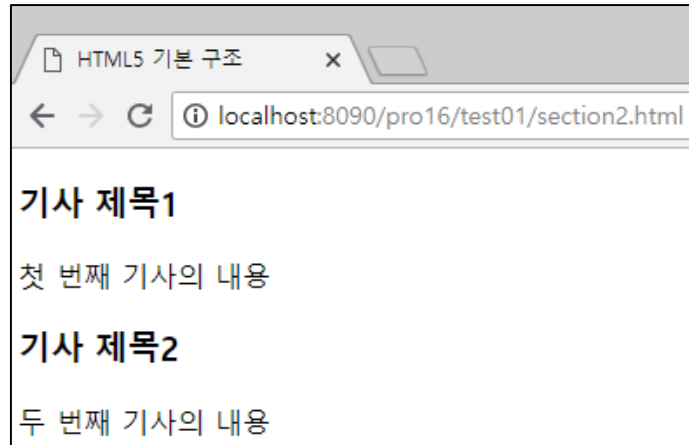
```
</body>
```

```
</html>
```

— <section> 태그 안의 <article> 태그를 이용해 콘텐츠를 나눕니다.

## 16.2 HTML5 시맨틱 웹을 위한 구성 요소

5. <http://localhost:8090/pro16/test01/section2.html>로 요청하여 결과를 확인합니다.



## 16.2 HTML5 시맨틱 웹을 위한 구성 요소

6. 이번에는 section3.html을 다음과 같이 작성합니다. 여러 가지 시맨틱 웹의 태그를 이용해 화면의 레이아웃을 구성하는 내용입니다.

코드 16-3 pro16/WebContent/test01/section3.html

```
<!DOCTYPE html>
<html>
<head>...</head>
<body>
```

```
  <header>
    <h1>HTML5 강좌를 시작합니다.</h1>
  </header>
```

〈header〉 태그를 이용해 로고나 메뉴를 표시합니다.

```
  <nav>
    <ul>
      <li>1. 개념 익히기</li>
      <li>2. 문법 익히기</li>
      <li>3. 실전 응용하기</li>
    </ul>
  </nav>
```

〈nav〉 태그를 이용해 상단의 내비게이션 메뉴를 표시합니다.

```
  <aside>
    사이트 메뉴
  </aside>
```

〈aside〉 태그를 이용해 사이드 메뉴를 표시합니다.

```
  <section>
    <article>
      <h1> 첫 번째 강좌 제목 </h1>
      <p> 첫 번째 강좌 내용 </p>
    </article>
```

〈section〉 태그를 이용해 본문을 표시합니다.

## 16.2 HTML5 시맨틱 웹을 위한 구성 요소

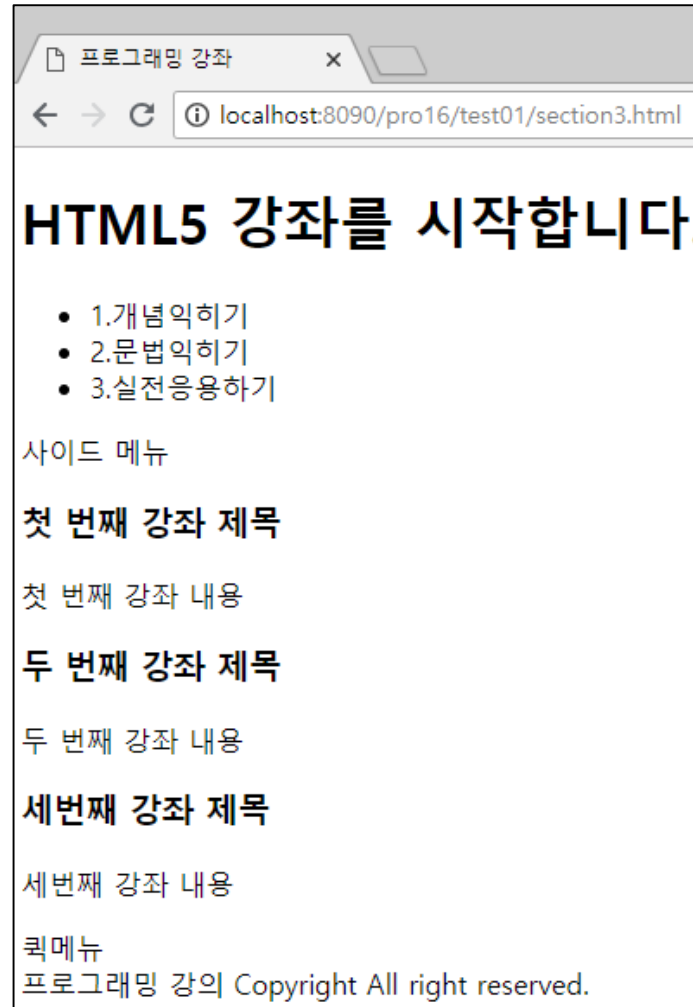
```
<article>
  <h1> 두 번째 강좌 제목 </h1>
  <p> 두 번째 강좌 내용 </p>
</article>
<article>
  <h1> 세번째 강좌 제목 </h1>
  <p> 세번째 강좌 내용 </p>
</article>
</section>
<aside>
  쿼메뉴
</aside>
<footer>
  프로그래밍 강의 Copyright All right reserved.
</footer>
</body>
</html>
```

• <aside> 태그를 이용해 쿼 메뉴를 표시합니다.

• <footer> 태그를 이용해 하단 정보를 표시합니다.

## 16.2 HTML5 시맨틱 웹을 위한 구성 요소

7. <http://localhost:8090/pro16/test01/section3.html>로 요청하여 결과를 확인합니다.



## 16.2 HTML5 시맨틱 웹을 위한 구성 요소

section4.html은 HTML 태그에 CSS를 적용한 코드 예입니다

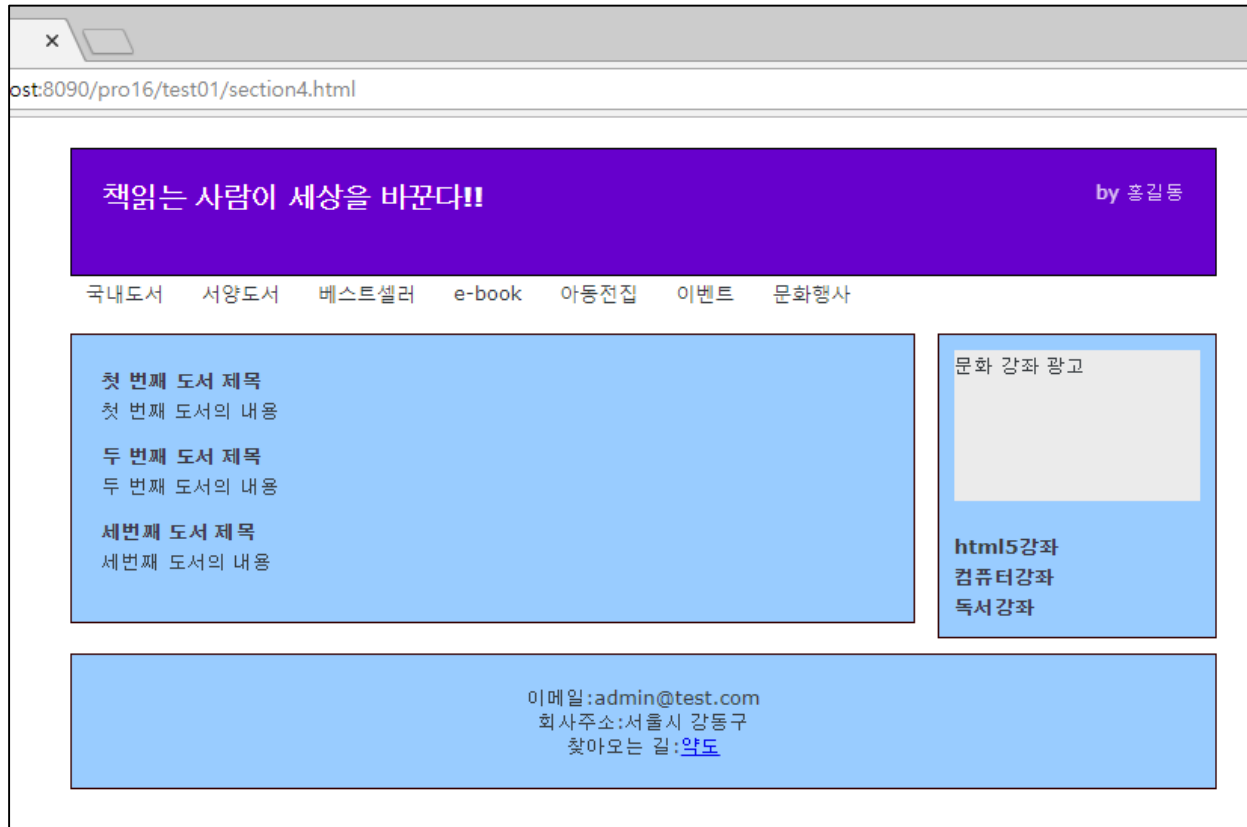
**코드 16-4** pro16/WebContent/test01/section4.html 일부

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>도서쇼핑몰</title>
  <style type="text/css">
    html,body{width:100%;height:70%}
    html{overflow-y:scroll}
    body,div,dl,dt,dd,ul,ol,li,h1,h2,h3,h4,h5,form,fieldset,p,button{margin:0;padding:0}
    body,h1,h2,h3,h4,input,button{font-family:NanumGothicWeb,verdana,dotum,
                                     sans-serif;font-size:13px;color:#383d41}
    ...
  </style>
  ...
</html>
```

CSS 속성을 HTML 태그에 적용해 레이아웃을 구성합니다.

## 16.2 HTML5 시맨틱 웹을 위한 구성 요소

http://localhost:8090/pro16/test01/section4.html로 요청하여 실행 결과를 확인합니다.





## 16.3 제이쿼리 주요 개념

### 제이쿼리(jQuery)

- 화면의 동적 기능을 자바스크립트보다 좀 더 쉽고 편리하게 개발할 수 있게 해주는 자바스크립트 기반 라이브러리

### 제이쿼리(jQuery) 특징

- CSS 선택자를 사용해 각 HTML 태그에 접근해서 작업하므로 명료하면서도 읽기 쉬운 형태로 표현함
- 메서드 체인 방식으로 수행하므로 여러 개의 동작(기능)이 한 줄로 수행할 수 있음
- 풍부한 플러그인을 제공하므로 이미 개발된 많은 플러그인을 쉽고 빠르게 이용할 수 있음
- 크로스 브라우징을 제공하므로 브라우저 종류에 상관 없이 동일하게 기능을 수행함

### 제이쿼리(jQuery) 사용 방법

- ① [www.jquery.com](http://www.jquery.com)에서 다운로드해서 사용하는 방법
- ② 네트워크로 CDN 호스트를 설정해서 사용하는 방법

## 16.3 제이쿼리 주요 개념

### 제이쿼리(jQuery) CDN 호스트 설정 방법

- `<script src="http://code.jquery.com/jquery-2.2.1.min.js"></script>`  
: 지정한 버전의 제이쿼리를 사용합니다.
- `<script src="http://code.jquery.com/jquery-latest.min.js"></script>`  
: 가장 최신 버전의 제이쿼리를 사용합니다.

## 16.4 제이쿼리의 여러 가지 기능

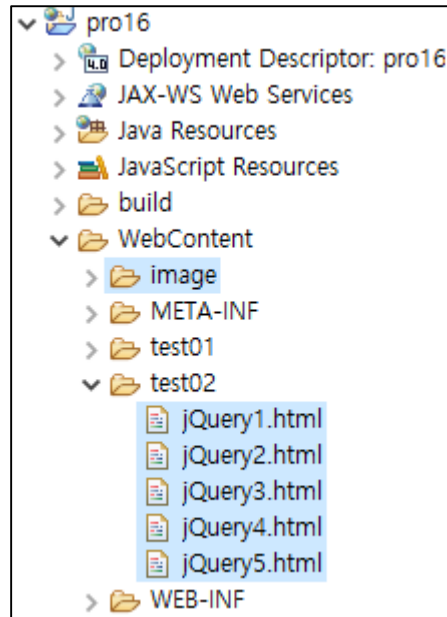
### 제이쿼리의 여러 가지 선택자

선택장 종류	선택자 표현 방법	설명
All selector	<code>\$("*")</code>	모든 DOM을 선택합니다.
ID selector	<code>\$("#id")</code>	해당되는 id를 가지는 DOM을 선택합니다.
Element selector	<code>\$("elementName")</code>	해당되는 이름을 가지는 DOM을 선택합니다.
class selector	<code>\$(".className")</code>	CSS 중 해당되는 클래스 이름을 가지는 DOM을 선택합니다.
Multiple selector	<code>\$("selector1,selector2,selector3, ..., selectorN")</code>	해당되는 선택자를 가지는 모든 DOM을 선택합니다.

## 16.4 제이쿼리의 여러 가지 기능

- 16.4.1 제이쿼리 선택자 사용 실습

1. 다음과 같이 test02 폴더에 제이쿼리 실습 html을 생성합니다.



## 16.4 제이쿼리의 여러 가지 기능

2. jQuery1.html을 다음과 같이 작성합니다.

코드 16-5 pro16/WebContent/test02/jQuery1.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>ID 선택자 연습1</title>
  <script src="http://code.jquery.com/jquery-latest.min.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      alert($("#unique2").html());
    });
  </script>
</head>

<body>
  <div class="class1">안녕하세요.</div>
  <div id="unique2">제이쿼리입니다!</div>
  <div id="unique3">
    <p>제이쿼리는 아주 쉽습니다!!! </p>
  </div>
</body>

</html>
```

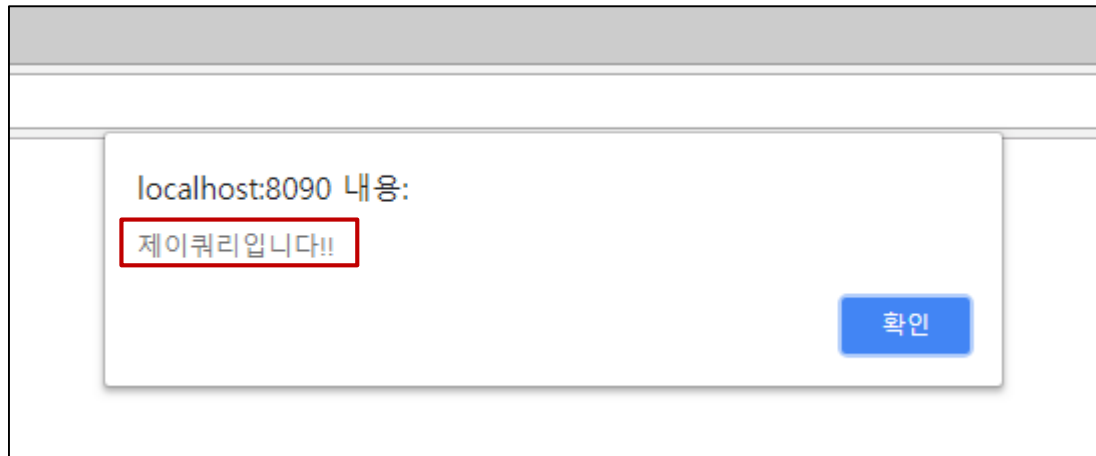
제이쿼리를 사용하기 위해 반드시 설정해 줍니다.

document에 DOM(Document Object Model)이 로드되는 이벤트 처리 함수입니다.

페이지 로드 시 id가 unique2인 태그를 검색한 후 html() 메서드를 이용해 태그의 값을 가져옵니다.

## 16.4 제이쿼리의 여러 가지 기능

3. <http://localhost:8090/pro16/test02/jQuery1.html>로 요청하여 실행 결과를 확인합니다. 웹 페이지가 브라우저에 로드되는 즉시 id에 해당되는 엘리먼트의 값을 출력합니다.



## 16.4 제이쿼리의 여러 가지 기능

4. 다음은 제이쿼리의 id 선택자를 이용해 해당 id를 가지는 <p> 엘리먼트에 접근하여 동적으로 텍스트를 추가해 보겠습니다.

코드 16-6 pro16/WebContent/test02/jquery2.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>ID 선택자 연습2</title>
  <script src="http://code.jquery.com/jquery-latest.min.js"></script>
  <script type="text/javascript">
    function addHtml() {
      $("#article").html('안녕하세요' + '<br>');
    }
  </script>
</head>
<body>
  <div>
    <p id="article"></p>
  </div>
  <input type="button" value="추가하기" onClick="addHtml()" />
</body>

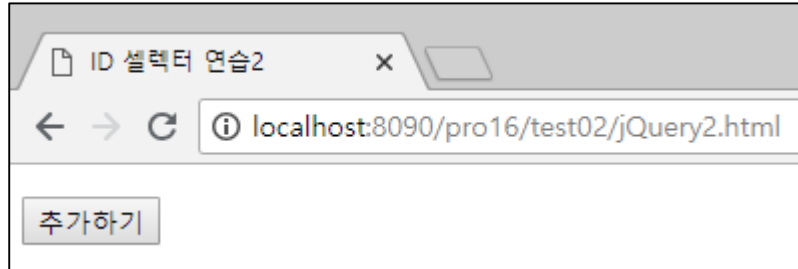
</html>
```

제이쿼리로 id가 article인 태그를 찾아서 html() 메서드의 인자 값을 태그에 설정합니다.

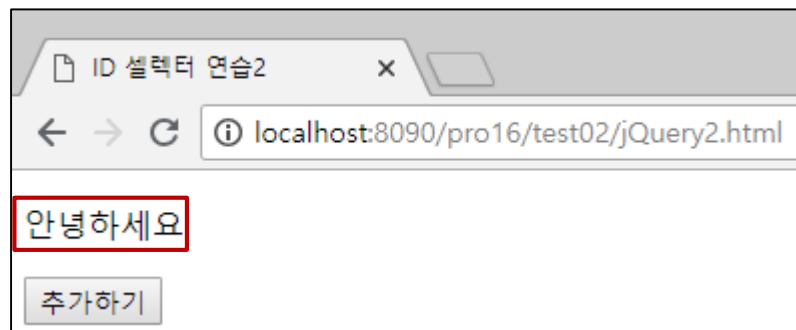
버튼 클릭 시 addHtml() 함수를 호출합니다.

## 16.4 제이쿼리의 여러 가지 기능

5. <http://localhost:8090/pro16/test02/jQuery2.html>로 요청한 후 추가하기를 클릭합니다.



6. <p> 태그에 "안녕하세요"라는 텍스트를 추가하고 결과를 확인합니다.





## 16.4 제이쿼리의 여러 가지 기능

7. 이번에는 class 선택자로 <div> 태그에 접근하여 기능을 수행해 보겠습니다.

코드 16-7 pro16/WebContent/test02/jQuery3.html

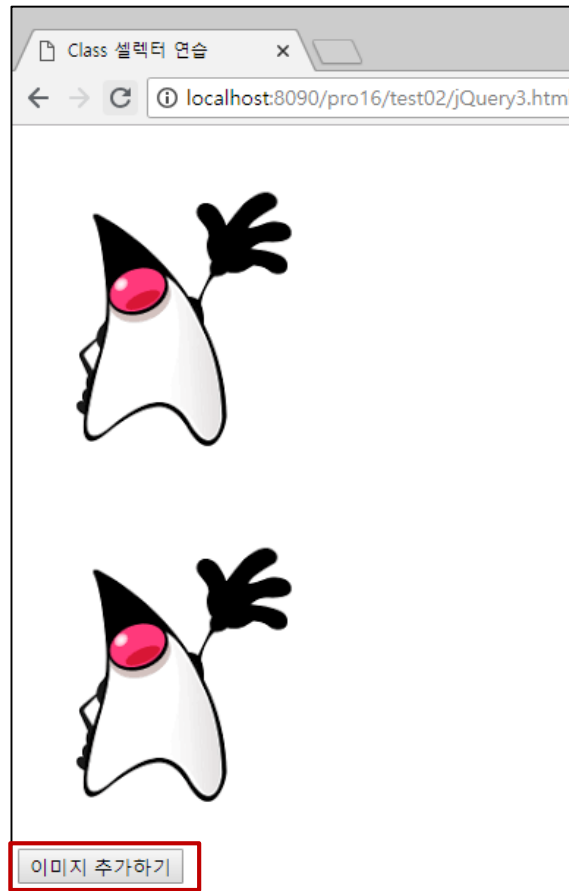
```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Class 선택자 연습</title>
  <script src="http://code.jquery.com/jquery-latest.min.js"></script>
  <script type="text/javascript">
    function addImage() {
      $(".class1").html("<img src='../image/duke.png'>");
    }
  </script>
</head>
<body>
  <div class="class1"></div>
  <div class="class1"></div>
  <input type="button" value="이미지 추가하기" onClick="addImage()" />
</body>
</html>
```

버튼 클릭 시 클래스 이름이 class1인 태그를 찾아서 <img> 태그를 추가합니다.

클래스 이름이 class1로 지정되어 있습니다.

## 16.4 제이쿼리의 여러 가지 기능

8. <http://localhost:8090/pro16/test02/jQuery3.html>로 요청하여 이미지 추가하기를 클릭하면 이미지가 두 개의 <div>에 추가됩니다.



## 16.4 제이쿼리의 여러 가지 기능

9. 다음은 제이쿼리에서 <div> 엘리먼트에 직접 접근하여 이미지를 추가해 보겠습니다.

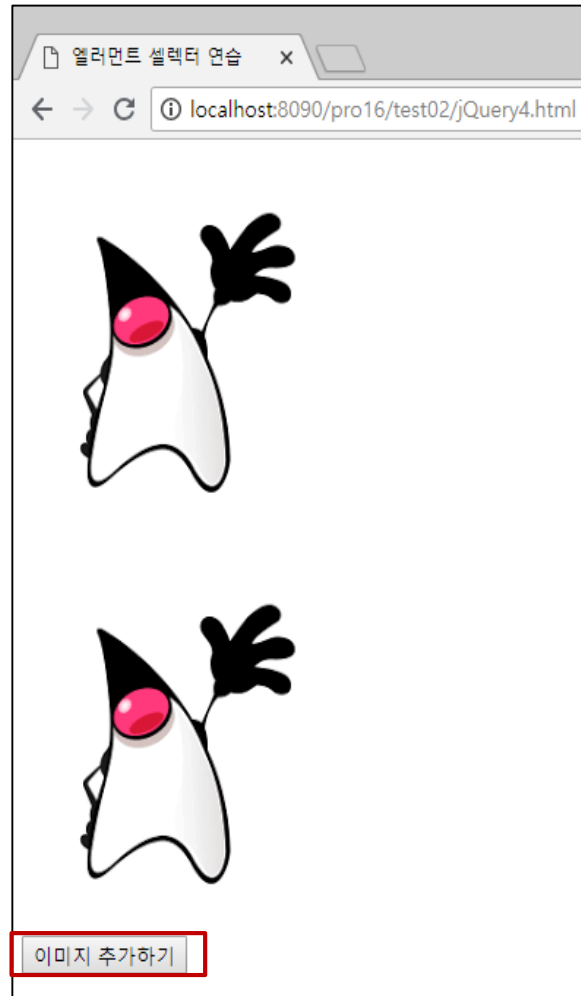
**코드 16-8** pro16/WebContent/test02/jquery4.html

```
<script src="http://code.jquery.com/jquery-latest.min.js"></script>
<script type="text/javascript">
    function addImage() {
        $("div").html("<img src='../image/duke.png'>");
    }
</script>
</head>
<body>
    <div></div>
    <div></div>
    <input type="button" value="이미지 추가하기" onClick="addImage()" />
</body>
```

제이쿼리로 <div> 엘리먼트에 직접 접근하여 <img> 태그를 추가합니다.

## 16.4 제이쿼리의 여러 가지 기능

10. <http://localhost:8090/pro16/test02/jQuery4.html>로 요청하여 <div> 엘리먼트에 이미지를 추가합니다.



## 16.4 제이쿼리의 여러 가지 기능

11. jQuery5.html을 다음과 같이 작성합니다.

**코드 16-9** pro16/WebContent/test02/jQuery5.html

```
<script type="text/javascript">
  function fn_process() {
    var value = $("#t_input").val();
    $("#t_output").val(value);
  }
</script>
...

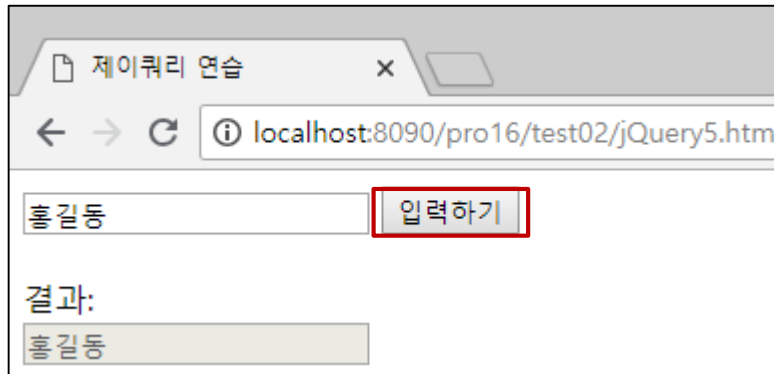
<body>
  <input type="text" id="t_input" />
  <input type="button" value="입력하기" onClick="fn_process()" /><br><br>
  <div>
    결과:<br>
    <input type="text" id="t_output" disabled />
  </div>
</body>
```

제이쿼리에서 id로 텍스트 박스에 접근하여 val() 메서드를 이용해서 입력 값을 가져옵니다.

제이쿼리에서 id로 텍스트 박스에 접근하여 val() 메서드를 이용해서 값을 출력합니다.

## 16.4 제이쿼리의 여러 가지 기능

12. <http://localhost:8090/pro16/test02/jQuery5.html>로 요청하여 텍스트 박스에 홍길동이라고 이름을 입력한 후 입력하기를 클릭하면 입력한 이름이 다른 텍스트 박스에 표시됩니다.

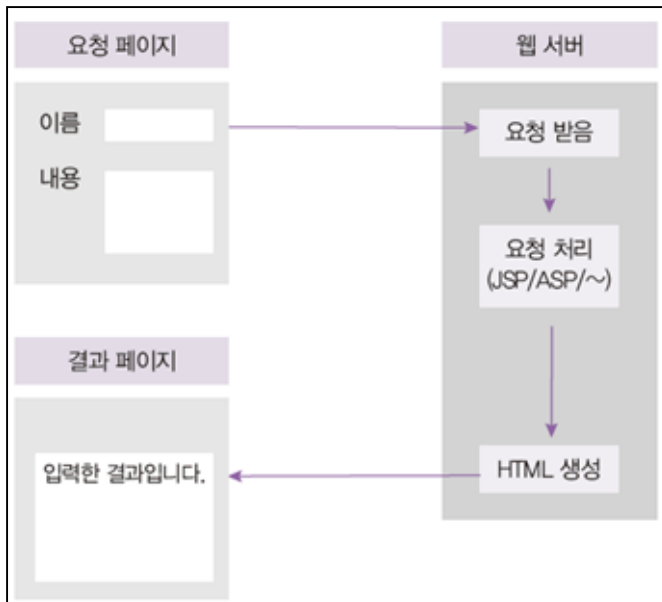


## 16.5 제이쿼리 Ajax 기능

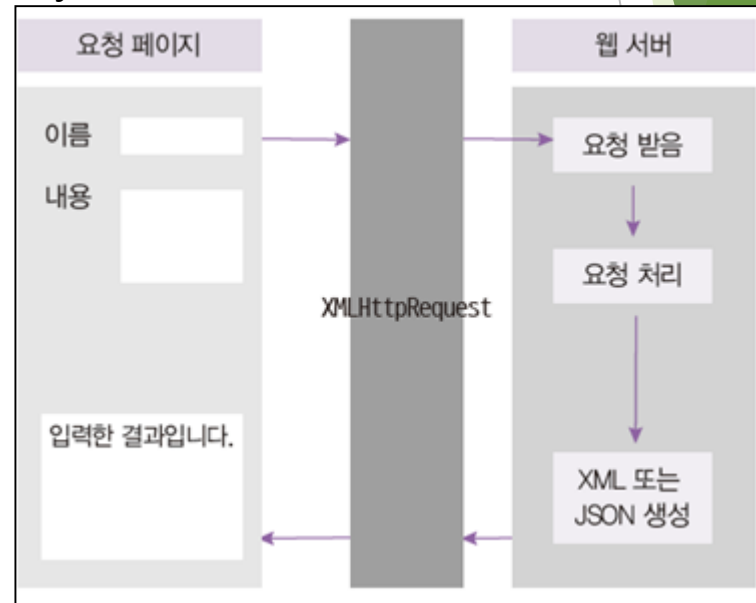
### Ajax의 정의

- Ajax란 Asynchronous Javascript(비동기 자바스크립트) + XML의 의미로 자바스크립트를 사용한 비동기 통신, 즉 클라이언트와 서버 간의 XML이나 JSON 데이터를 주고받는 기술

### 기존 웹 페이지 동작 방식



### Ajax 웹 페이지 동작 방식



## 16.5 제이쿼리 Ajax 기능

- 16.5.1 제이쿼리 Ajax 사용법

제이쿼리 Ajax 사용 형식

```
$.ajax({  
  type: "post 또는 get",  
  async:"true 또는 false",  
  url: "요청할 URL",  
  data: {서버로 전송할 데이터},  
  dataType: "서버에서 전송받을 데이터형식",  
  success:{  
    //정상 요청, 응답 시 처리  
  },  
  error: function(xhr,status,error){  
    //오류 발생 시 처리  
  },  
  complete:function(data,textStatus){  
    //작업 완료 후 처리  
  }  
});
```



## 16.5 제이쿼리 Ajax 기능

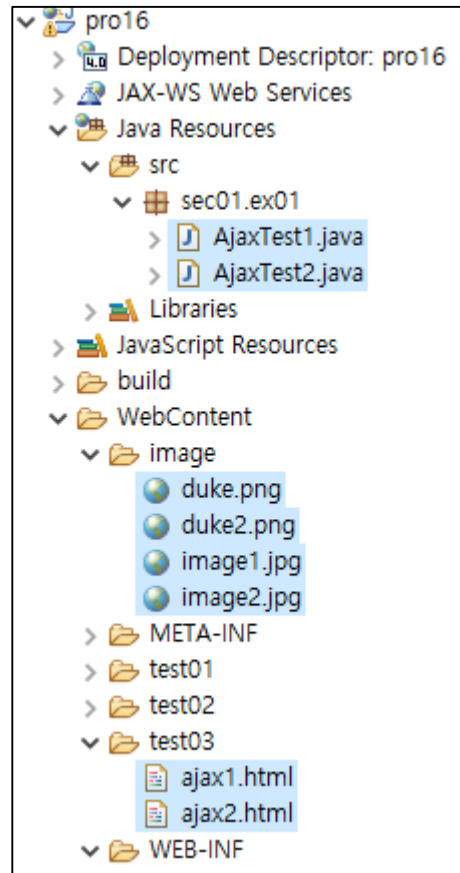
### 제이쿼리 Ajax 기능 관련 속성들

속성	설명
type	통신 타입을 설정합니다(post 또는 get 방식으로 선택합니다).
url	요청할 url을 설정합니다.
async	비동기식으로 처리할지의 여부를 설정합니다(false인 경우 동기식으로 처리합니다).
data	서버에 요청할 때 보낼 매개변수를 설정합니다.
dataType	응답 받을 데이터 타입을 설정합니다(XML, TEXT, HTML, JSON 등).
success	요청 및 응답에 성공했을 때 처리 구문을 설정합니다.
error	요청 및 응답에 실패했을 때 처리 구문을 설정합니다.
complete	모든 작업을 마친 후 처리 구문을 설정합니다.

## 16.5 제이쿼리 Ajax 기능

- 16.5.2 제이쿼리 Ajax 사용하기

1. sec01.ex01 패키지를 만들고 AjaxTest1.java, AjaxTest2.java를 생성합니다. 그리고 test03 폴더에 ajax1.html, ajax2.html을 추가합니다.



## 16.5 제이쿼리 Ajax 기능

2. AjaxTest1.java를 다음과 같이 작성합니다.

코드 16-10 pro16/src/sec01/ex01/AjaxTest1.java

```
package sec01.ex01;

...

@WebServlet("/ajaxTest1")
public class AjaxTest1 extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        doHandle(request, response);
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException{
        doHandle(request, response);
    }

    private protected void doHandle(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException{
        request.setCharacterEncoding("utf-8");
        response.setContentType("text/html; charset=utf-8");
        String param = (String)request.getParameter("param");
        System.out.println("param = " +param);
        PrintWriter writer = response.getWriter();
        writer.print("안녕하세요! 서버입니다.");
    }
}
```

getParameter() 메서드를 이용해 ajax로 전송된 매개변수를 가져옵니다.

PrintWriter의 print() 메서드를 이용해 브라우저에 응답 메시지를 보냅니다.

## 16.5 제이쿼리 Ajax 기능

3. ajax1.html을 다음과 같이 작성합니다.

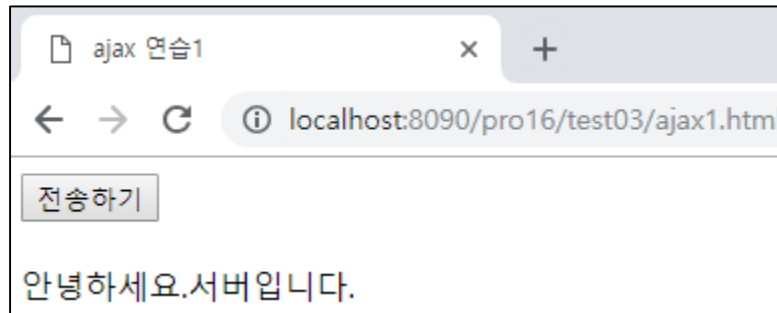
코드 16-11 pro16/src/sec01/ex01/ajax1.html

```
<script type="text/javascript">
  function fn_process() {
    $.ajax({
      type:"get",
      dataType:"text",
      async:false,
      url:"http://localhost:8090/pro16/ajaxTest1",
      data: {param:"Hello, jquery"},
      success:function (data,textStatus) {
        $('#message').append(data);
      },
      error:function(data,textStatus) {
        alert("에러가 발생했습니다.");
      },
      complete:function(data,textStatus) {
        alert("작업을 완료했습니다.");
      }
    });
  }
</script>
</head>
<body>
  <input type="button" value="전송하기" onClick="fn_process()" /><br><br>
  <div id="message"></div>
</body>
```

type:"get", get 방식으로 전송합니다.  
 dataType:"text", 응답 데이터를 텍스트로 지정합니다.  
 async:false, false인 경우 동기식으로 처리합니다.  
 url:"http://localhost:8090/pro16/ajaxTest1", 전송할 서버를 지정합니다.  
 data: {param:"Hello, jquery"}, 서버로 매개변수와 값을 설정합니다.  
 success:function (data,textStatus) { 전송과 응답이 성공했을 경우의 작업을 설정합니다.  
 \$('#message').append(data); 서버 응답 메시지를 <div> 엘리먼트에 표시합니다.  
 }, 작업 중 오류가 발생했을 경우에 수행할 작업을 설정합니다.  
 error:function(data,textStatus) {  
 alert("에러가 발생했습니다.");  
 }, 완료 시 수행할 작업을 설정합니다.  
 complete:function(data,textStatus) {  
 alert("작업을 완료했습니다.");  
 }
 }
}

## 16.5 제이쿼리 Ajax 기능

4. `http://localhost:8090/pro16/test03/ajax1.html`로 요청하여 전송하기를 클릭하면 서버에서 ajax로 전송된 데이터를 `<div>` 엘리먼트에 표시합니다.



5. 서버의 서블릿에서는 ajax로 전달된 매개변수 값을 콘솔로 출력합니다.

```
Tomcat v9.0 Server at localhost [Apache To  
param = Hello,jquery
```

## 16.5 제이쿼리 Ajax 기능

### • 16.15.3 XML 데이터 연동하기

1. 다음과 같이 AjaxTest2 클래스를 작성합니다.

코드 16-12 pro16/src/sec01/ex01/AjaxTest2.java

```
package sec01.ex01;
...
@WebServlet("/ajaxTest2")
public class AjaxTest2 extends HttpServlet {
    ...
    protected void doHandle(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        request.setCharacterEncoding("utf-8");
        response.setContentType("text/html; charset=utf-8");
        String result="";
        PrintWriter writer = response.getWriter();
        result="<main><book>"+
            "<title><![CDATA[초보자를 위한 자바 프로그래밍]]></title>" +
            "<writer><![CDATA[인포북스 저 | 이병승]]></writer>" +
            "<image><![CDATA[http://localhost:8090/pro16/image/image1.jpg]]></image>"+
            "</book>"+
            "<book>"+
            "<title><![CDATA[모두의 파이썬]]></title>" +
            "<writer><![CDATA[길벗 저 | 이승찬]]></writer>" +
            "<image><![CDATA[http://localhost:8090/pro16/image/image2.jpg]]></image>"+
            "</book></main>";
        ...
    }
}
```

도서 정보를 XML로 작성한 후  
클라이언트로 전송합니다.

pro16으로 접근 시 WebContent 하위  
image 폴더의 image1.jpg를 표시합니다.

## 16.5 제이쿼리 Ajax 기능

2. 브라우저에서는 XML 데이터를 받은 후 제이쿼리의 find() 메서드에 <title>, <writer>, <image> 태그 이름으로 호출하여 각각의 도서 정보를 가져옵니다.

코드 16-13 pro16/WebContent/test03/ajax2.html

```
<script type="text/javascript">
function fn_process() {
    $.ajax({
        type:"post",
        async:false,
        url:"http://localhost:8090/pro16/ajaxTest2",
        dataType:"xml",
        success:function (info,textStatus) {
            $(info).find("book").each(function() {
                var title=$(this).find("title").text();
                var writer=$(this).find("writer").text();
                var image=$(this).find("image").text();
                $("#bookInfo").append(
                    "<p>" +title+ "</p>" +
                    "<p>" +writer + "</p>" +
                    ""
                );
            });
        },
        ...
    });
</script>
```

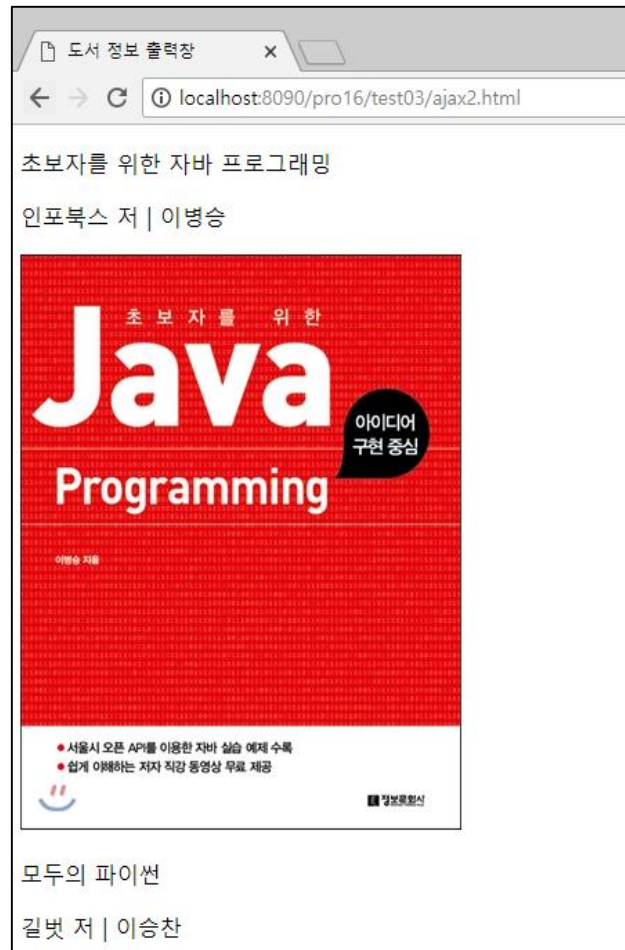
데이터를 XML 형태로 받습니다.

전송된 XML 데이터에서 엘리먼트 이름으로 데이터를 가져옵니다.

id가 bookInfo인 <div> 엘리먼트에 도서 정보를 표시합니다.

## 16.5 제이쿼리 Ajax 기능

3. <http://localhost:8090/pro16/test03/ajax2.html>로 요청하여 도서정보 요청을 클릭하면 도서 정보와 이미지가 나타납니다.

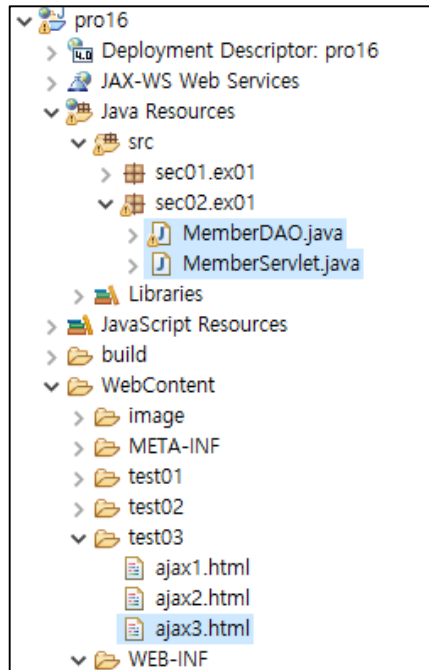




## 16.5 제이쿼리 Ajax 기능

- 16.5.4 ID 중복 여부 확인하기

1. sec02.ex01 패키지를 만들고 MemberDAO, MemberServlet 클래스를 만듭니다. 그리고 ajax3.html을 추가합니다.



## 16.5 제이쿼리 Ajax 기능

2. MemberServlet 클래스를 다음과 같이 작성합니다.

코드 16-14 pro16/src/sec02/ex01/MemberServlet.java

```
package sec02.ex01;

...

@WebServlet("/mem")
public class MemberServlet extends HttpServlet {
    ...

    protected void doHandle(HttpServletRequest request,
                             HttpServletResponse response)
        throws ServletException, IOException {
        request.setCharacterEncoding("utf-8");
        response.setContentType("text/html; charset=utf-8");
        PrintWriter writer = response.getWriter();
        String id = (String)request.getParameter("id");
        System.out.println("id = " + id);
        MemberDAO memberDAO=new MemberDAO();
        boolean overlappedID=memberDAO.overlappedID(id);

        if(overlappedID == true) {
            writer.print("not_usable");
        }else{
            writer.print("usable");
        }
    }
}
```

ID 중복 여부를 체크합니다.

결과를 메시지로 전송합니다.

## 16.5 제이쿼리 Ajax 기능

### 3. MemberDAO를 다음과 같이 작성합니다.

코드 16-15 pro16/src/sec02/ex01/MemberDAO.java

```
package sec01.ex02;

...
public class MemberDAO{
    ...
    public boolean overlappedID(String id)
    {
        boolean result = false;
        try
        {
            con = dataFactory.getConnection();
            String query = "select decode(count(*),1,'true','false') as result from t_member";
            query += " where id=?";
            System.out.println("prepareStatement: " + query);
            pstmt = con.prepareStatement(query);
            pstmt.setString(1, id);
            ResultSet rs = pstmt.executeQuery();
            rs.next();
            result = Boolean.parseBoolean(rs.getString("result"));
            pstmt.close();
        } catch (Exception e)
        {
            e.printStackTrace();
        }
        return result;
    }
}
```

오라클의 decode() 함수를 이용해 ID가 존재하면 true, 존재하지 않으면 false를 문자열로 조회합니다.

문자열을 불(Boolean) 자료형으로 변환합니다.

## 16.5 제이쿼리 Ajax 기능

4. ajax3.html을 다음과 같이 작성합니다.

코드 16-16 pro16/WebContent/test03/ajax3.html

```
<script type="text/javascript">
function fn_process() {
    var _id = $("#t_id").val();
    if (_id == '') {
        alert("ID를 입력하세요");
        return;
    }
    $.ajax({
        type: "post",
        async: false,
        url: "http://localhost:8090/pro16/mem",
        dataType: "text",
        data: { id: _id },
        success: function (data, textStatus) {
            if (data == 'usable') {
                $('#message').text("사용할 수 있는 ID입니다.");
                $('#btn_duplicate').prop("disabled", true);
            } else {
                $('#message').text("사용할 수 없는 ID입니다.");
            }
        }
    },
    ...
}
```

텍스트 박스에 입력한 ID를 가져옵니다.

ID를 입력하지 않으면 오류 메시지를 출력합니다.

ID를 서버로 전송합니다.

서버에서 전송된 결과에 따라 메시지를 표시합니다.

사용할 수 있는 ID면 버튼을 비활성화시킵니다.

## 16.5 제이쿼리 Ajax 기능

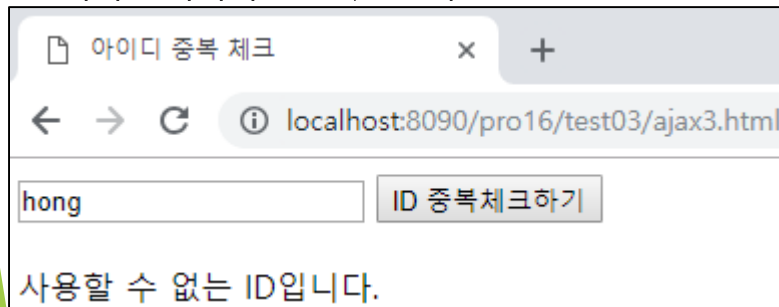
```
</script>

</head>
<body>
  <input type="text" id="t_id" />
  <input type="button" id="btn_duplicate" value="ID 중복체크하기"
    onClick="fn_process()" /><br><br>
  <div id="message"></div>
</body>
</html>
```

결과를 표시합니다.

5. <http://localhost:8090/pro16/test03/ajax3.html>로 요청하여 ID를 입력합니다.

존재하는 아이디를 입력한 경우



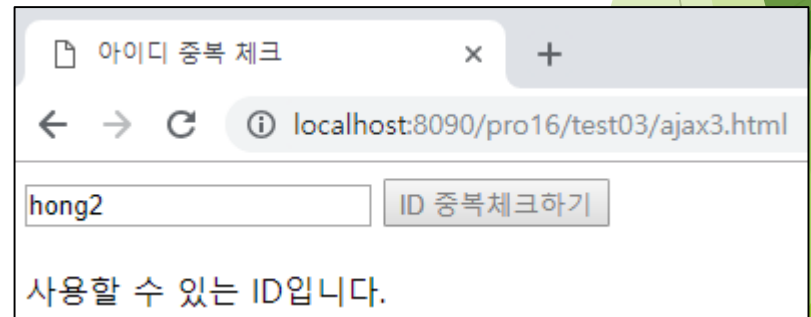
아이디 중복 체크

← → ↻ ⓘ localhost:8090/pro16/test03/ajax3.html

hong ID 중복체크하기

사용할 수 없는 ID입니다.

새 아이디를 입력한 경우



아이디 중복 체크

← → ↻ ⓘ localhost:8090/pro16/test03/ajax3.html

hong2 ID 중복체크하기

사용할 수 있는 ID입니다.

## 16.6 제이쿼리에서 JSON 사용하기

### JSON 정의

- JSON( Javascript Object Notation)은 name/value 쌍으로 이루어진 데이터 객체를 전달하기 위해 인간이 읽을 수 있는 텍스트를 사용하는 개방형 표준 데이터 형식
- 비동기 브라우저/서버 통신(Ajax)을 위해 XML을 대체하는 데이터 전송 형식 중 하나
- 자바스크립트에서 파생된 것이므로 자바스크립트의 구문 형식을 따르지만 프로그래밍 언어나 플랫폼에 독립적이어서 쉽게 사용할 수 있음

## 16.6 제이쿼리에서 JSON 사용하기

### JSON의 여러 가지 자료형

자료형	종류	예
수(Number)	정수	76,197,750,-11,-234
	실수(고정소수점)	3.14, -2.717, 45.78
	실수(부동소수점)	1e4, 2.5e34, 5.67e-9, 7.66E-3
문자열	문자열	"1234" "true" "apple-num" "사랑" "JSP"
	제어 문자	\\b (백스페이스) \\f (폼 피드) \\n (개행) \\r (캐리지 반환) \\t (탭) \\\" (따옴표) \\\" (슬래시) \\\\ (역슬래시)
배열	배열은 대괄호[]로 나타냅니다. 배열의 각 요소는 기본 자료형이거나 배열,객체입니다. 각 요소들은 콤마(,)로 구별합니다.	"name": ["홍길동", "이순신", "임꺽정"] // 대괄호 안에 배열 요소를 콤마(,)로 구분해서 나열합니다.
객체	JSON 객체는 중괄호{}로 둘러싸서 표현합니다. 콤마(,)를 사용해 여러 프로퍼티를 포함할 수 있습니다.	{ "name": "홍길동", "age": 16, "weight": 67 } // 중괄호 안에 name/value 쌍을 콤마(,)로 구분해서 나열합니다.

## 16.6 제이쿼리에서 JSON 사용하기

- 배열 이름이 **members**이고 JSON 객체를 배열 요소로 가지는 JSON 배열

배열 이름 members

배열 요소에는 name/value 쌍으로 회원 정보를 저장하는 객체를 저장합니다.

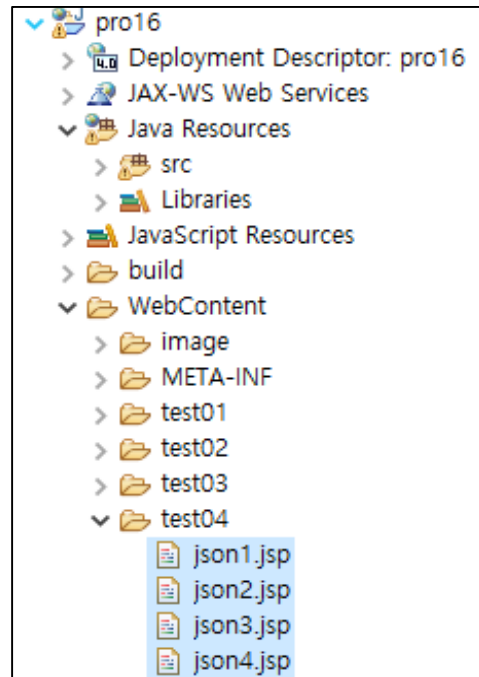
```
"members": [  
  {"name": "홍길동", "age": 22, "gender": "남", "nick": "날센돌이"},  
  {"name": "손흥민", "age": 30, "gender": "남", "nick": "탱크"},  
  {"name": "김연아", "age": 24, "gender": "여", "nick": "갈치"}  
]
```



## 16.6 제이쿼리에서 JSON 사용하기

- 16.6.1 JSON의 자료형 사용 실습

1. 다음과 같이 test04 폴더를 만들고 json1~4까지 jsp 파일을 생성합니다.



## 16.6 제이쿼리에서 JSON 사용하기

2. json1.jsp를 다음과 같이 작성합니다.

코드 16-17 pro16/WebContent/test04/json1.jsp

```
<script src="http://code.jquery.com/jquery-latest.min.js"></script>
<script>
$(function () {
    $("#checkJson").click(function () {
        var jsonStr = '{ "name": ["홍길동", "이순신", "임꺽정"] }';
        var jsonInfo = JSON.parse(jsonStr);
        var output = "회원 이름<br>";
        output += "=====<br>";
        for (var i in jsonInfo.name) {
            output += jsonInfo.name[i] + "<br>";
        }
        $("#output").html(output);
    });
});
</script>
```

이름을 저장하는 JSON 배열을 name으로 선언합니다.

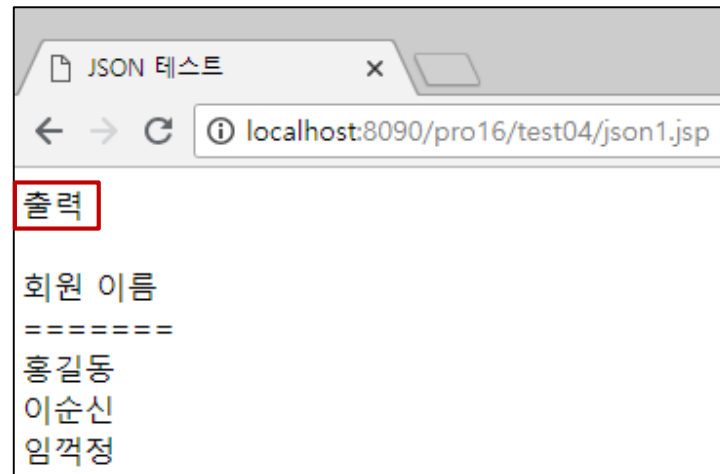
제이쿼리의 JSON 기능인 parse() 메서드를 이용해 JSON 자료형을 가져옵니다.

배열 이름 name으로 배열 요소에 반복 변수를 이용해 차례로 접근하여 값을 가져옵니다.

회원 이름을 출력합니다.

## 16.6 제이쿼리에서 JSON 사용하기

3. <http://localhost:8090/pro16/test04/json1.jsp>로 요청하여 출력을 클릭하면 배열 요소의 값을 출력합니다.



## 16.6 제이쿼리에서 JSON 사용하기

4. 이번에는 정수 자료형을 배열로 저장한 후 화면에 출력해 보겠습니다. json2.jsp를 다음과 같이 작성합니다.

코드 16-18 pro16/WebContent/test04/json2.jsp

```
<script>
$(function () {
    $("#checkJson").click(function () {
        var jsonStr = '{ "age": [22, 33, 44] }';
        var jsonInfo = JSON.parse(jsonStr);
        var output = "회원 나이<br>";
        output += "=====<br>";
        for (var i in jsonInfo.age) {
            output += jsonInfo.age[i] + "<br>";
        }
        $("#output").html(output);
    });
});
</script>
```

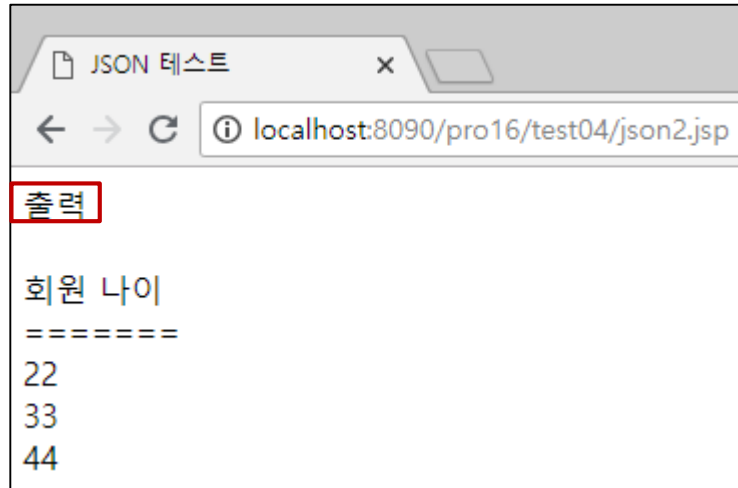
정수형 데이터를 가지는 이름이 age인  
배열을 선언합니다.

parse() 메서드로 배열을 구합니다.

배열 요소(나이)를 차례대로 출력합니다.

## 16.6 제이쿼리에서 JSON 사용하기

5. <http://localhost:8090/pro16/test04/json2.jsp>로 요청한 후 출력을 클릭하여 결과를 확인합니다.



## 16.6 제이쿼리에서 JSON 사용하기

6. 이번에는 JSON 객체에 회원 정보를 저장한 후 다시 회원 정보를 출력해 보겠습니다. json3.jsp를 다음과 같이 작성합니다.

코드 16-19 pro16/WebContent/test04/json3.jsp

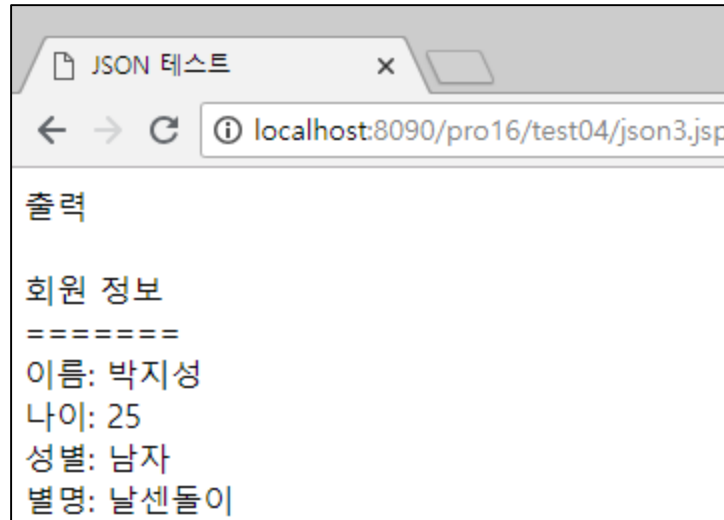
```
<script>
$(function () {
    $("#checkJson").click(function () {
        var jsonStr = '{"name":"박지성","age":25,"gender":"남자","nickname":"날센돌이"}';
        var jsonObj = JSON.parse(jsonStr);
        var output = "회원 정보<br>";
        output += "=====<br>";
        output += "이름: " + jsonObj.name + "<br>";
        output += "나이: " + jsonObj.age + "<br>";
        output += "성별: " + jsonObj.gender + "<br>";
        output += "별명: " + jsonObj.nickname + "<br>";
        $("#output").html(output);
    });
});
</script>
```

parse() 메서드로 JSON 데이터를 가져옵니다.

문자열에서 JSON 객체의 속성을 가져옵니다.

## 16.6 제이쿼리에서 JSON 사용하기

7. <http://localhost:8090/pro16/test04/json3.jsp>로 요청하여 실행 결과를 확인합니다.



## 16.6 제이쿼리에서 JSON 사용하기

8. 마지막으로 이번에는 JSON 배열의 요소에 JSON 객체를 저장한 후 다시 배열에 접근하여 JSON 객체의 속성 값을 출력해 보겠습니다. json4.jsp를 다음과 같이 작성합니다.

코드 16-20 pro16/WebContent/test04/json4.jsp

```
<script>
$(function () {
    $("#checkJson").click(function () {
        var jsonStr =
            '{"members":[{"name":"박지성","age":"25","gender":"남자","nickname":"날센돌이"}]
            + ', {"name":"손흥민","age":"30","gender":"남자","nickname":"탱크"}}';

        var jsonInfo = JSON.parse(jsonStr);
        var output = "회원 정보<br>";
        output += "=====<br>";
        for (var i in jsonInfo.members) {
            output += "이름: " + jsonInfo.members[i].name + "<br>";
            output += "나이: " + jsonInfo.members[i].age + "<br>";
            output += "성별: " + jsonInfo.members[i].gender + "<br>";
            output += "별명: " + jsonInfo.members[i].nickname + "<br><br><br>";
        }
        $("#output").html(output);
    });
});
</script>
```

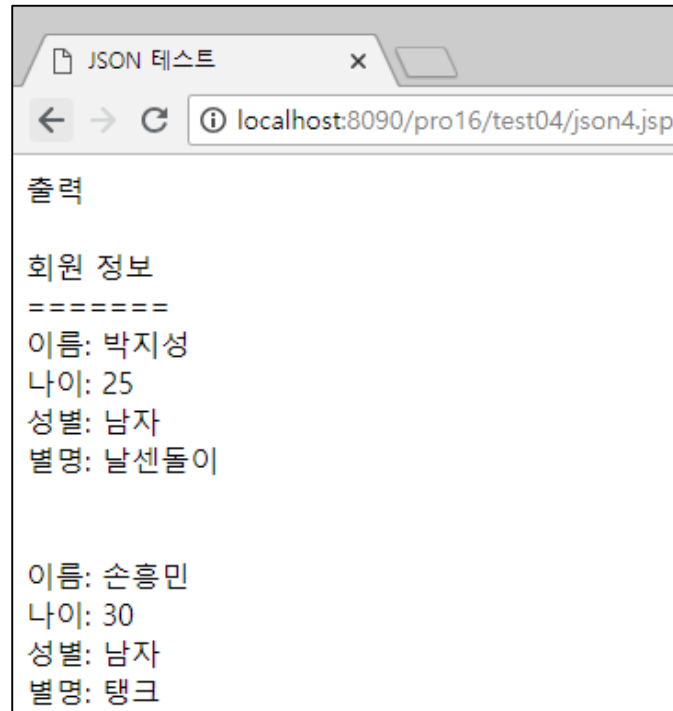
members 배열에 회원 정보를 객체의 name/value 쌍으로 저장합니다.

각 배열 요소에 접근하여 객체의 name으로 value를 출력합니다.



## 16.6 제이쿼리에서 JSON 사용하기

9. 다음은 실행 결과입니다.



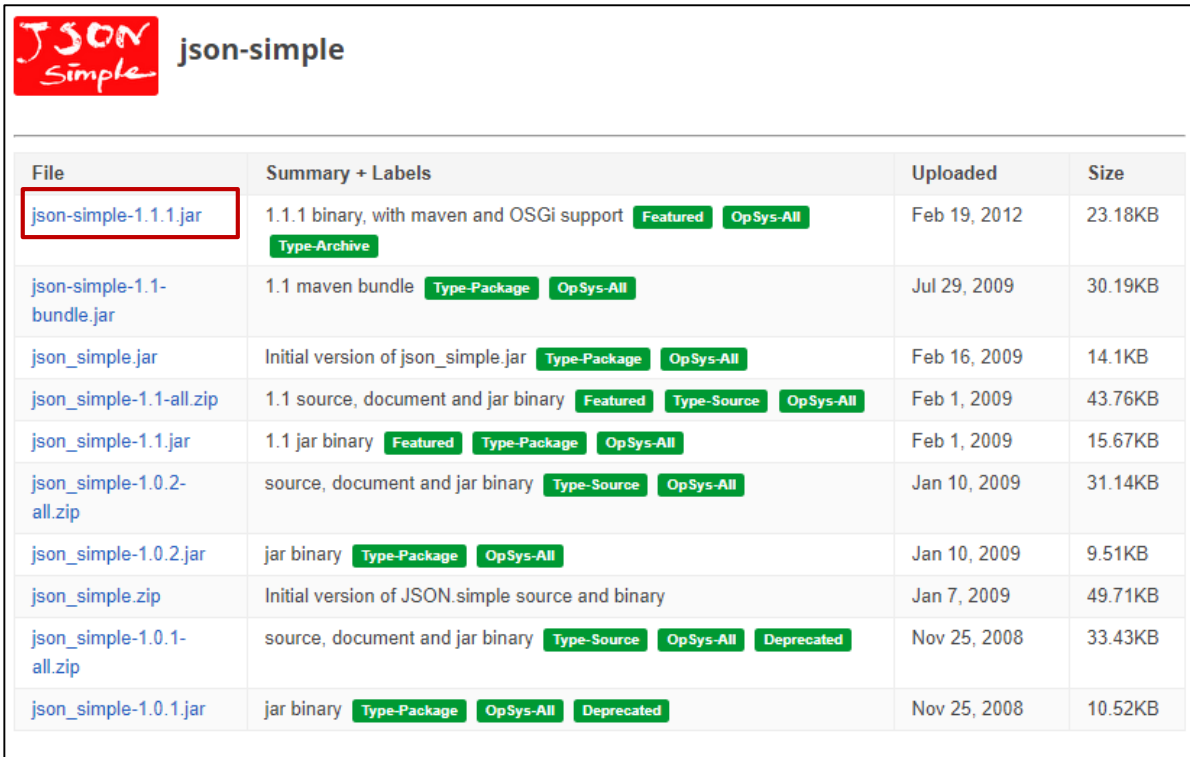
## 16.6 제이쿼리에서 JSON 사용하기

### • 16.6.2 Ajax 이용해 서버와 JSON 데이터 주고받기

1. 다음 사이트로 접속합니다.

- <https://code.google.com/archive/p/json-simple/downloads>

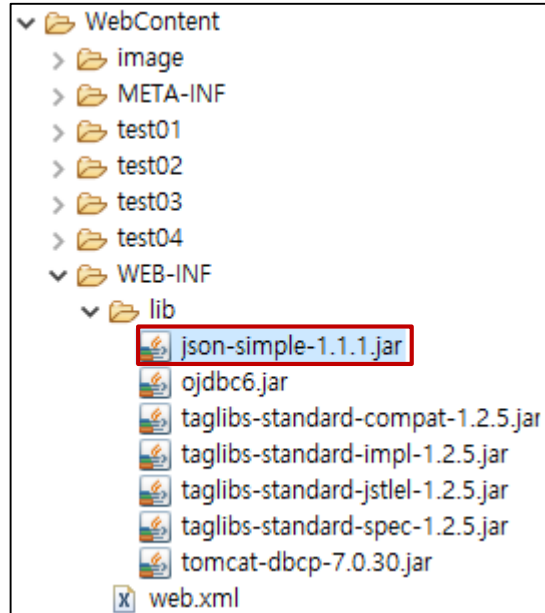
2. json-simple-1.1.1.jar를 클릭해 다운로드합니다.



File	Summary + Labels	Uploaded	Size
<a href="#">json-simple-1.1.1.jar</a>	1.1.1 binary, with maven and OSGi support <span>Type-Archive</span> <span>Featured</span> <span>OpSys-All</span>	Feb 19, 2012	23.18KB
<a href="#">json-simple-1.1-bundle.jar</a>	1.1 maven bundle <span>Type-Package</span> <span>OpSys-All</span>	Jul 29, 2009	30.19KB
<a href="#">json_simple.jar</a>	Initial version of json_simple.jar <span>Type-Package</span> <span>OpSys-All</span>	Feb 16, 2009	14.1KB
<a href="#">json_simple-1.1-all.zip</a>	1.1 source, document and jar binary <span>Featured</span> <span>Type-Source</span> <span>OpSys-All</span>	Feb 1, 2009	43.76KB
<a href="#">json_simple-1.1.jar</a>	1.1 jar binary <span>Featured</span> <span>Type-Package</span> <span>OpSys-All</span>	Feb 1, 2009	15.67KB
<a href="#">json_simple-1.0.2-all.zip</a>	source, document and jar binary <span>Type-Source</span> <span>OpSys-All</span>	Jan 10, 2009	31.14KB
<a href="#">json_simple-1.0.2.jar</a>	jar binary <span>Type-Package</span> <span>OpSys-All</span>	Jan 10, 2009	9.51KB
<a href="#">json_simple.zip</a>	Initial version of JSON.simple source and binary	Jan 7, 2009	49.71KB
<a href="#">json_simple-1.0.1-all.zip</a>	source, document and jar binary <span>Type-Source</span> <span>OpSys-All</span> <span>Deprecated</span>	Nov 25, 2008	33.43KB
<a href="#">json_simple-1.0.1.jar</a>	jar binary <span>Type-Package</span> <span>OpSys-All</span> <span>Deprecated</span>	Nov 25, 2008	10.52KB

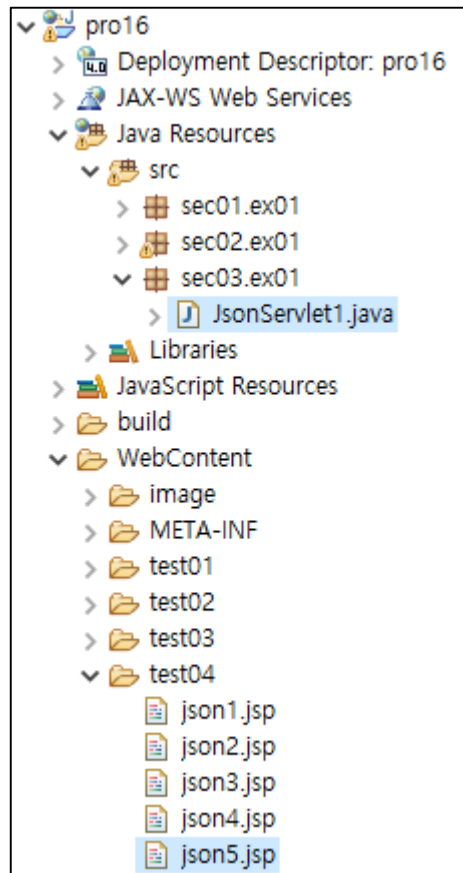
## 16.6 제이쿼리에서 JSON 사용하기

3. 이클립스 프로젝트의 /WebContent/lib 폴더에 붙여 넣습니다.



## 16.6 제이쿼리에서 JSON 사용하기

4. 이번에는 JSP에서 제이쿼리 Ajax 기능을 이용해 서버릿으로 JSON 데이터를 전송하기 위해 sec03.ex01 패키지를 만들고 JsonServlet1 클래스를 추가합니다.



## 16.6 제이쿼리에서 JSON 사용하기

5. JsonServlet1 클래스를 다음과 같이 작성합니다.

코드 16-21 pro16/src/sec03/ex01/JsonServlet1.java

```
package ex02;
...
@WebServlet("/json")
public class JsonServlet1 extends HttpServlet {
    ...
    privateprotected void doHandle(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        request.setCharacterEncoding("utf-8");
        response.setContentType("text/html; charset=utf-8");

        String jsonInfo = request.getParameter("jsonInfo");
        try {
            JSONParser jsonParser = new JSONParser();
            JSONObject jsonObject = (JSONObject) jsonParser.parse(jsonInfo);
            System.out.println("* 회원 정보*");
            System.out.println(jsonObject.get("name"));
            System.out.println(jsonObject.get("age"));
            System.out.println(jsonObject.get("gender"));
            System.out.println(jsonObject.get("nickname"));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

문자로 전송된 JSON 데이터를  
getParameter()를 이용해 가져옵니다.

JSON 데이터를 처리하기 위해  
JSONParser 객체를 생성합니다.

전송된 JSON 데이터를 파싱합니다.

JSON 데이터의 name 속성들을  
get()에 전달하여 value를 출력합  
니다.

## 16.6 제이쿼리에서 JSON 사용하기

6. json5.jsp를 다음과 같이 작성합니다. 자바스크립트에서 회원 정보를 JSON 객체로 만들어 매개변수 이름 jsonInfo로 ajax를 이용해 서블릿으로 전송합니다.

코드 16-22 pro16/WebContent/test04/json5.jsp

```
<script>
$(function() {
    $("#checkJson").click(function() {
        var _jsonInfo = '{"name":"박지성","age":"25","gender":"남자","nickname":"날센돌이"}';
        $.ajax({
            type:"post",
            async:false,
            url:"${contextPath}/json",
            data : {jsonInfo: _jsonInfo},
            success:function (data,textStatus) {
            },
            error:function(data,textStatus) {
                alert("에러가 발생했습니다.");
            },
            complete:function(data,textStatus) {
            }
        }); //end ajax
    });
});
</script>
```

전송할 회원 정보를 JSON 형식으로 만듭니다.

매개변수 이름 jsonInfo로 JSON 데이터를 ajax로 전송합니다.

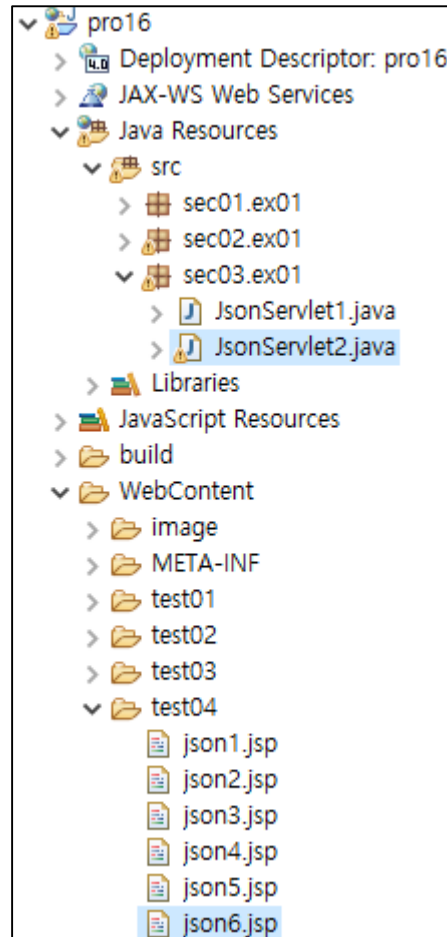
## 16.6 제이쿼리에서 JSON 사용하기

7. <http://localhost:8090/pro16/test04/json5.jsp>로 요청하여 JSP 페이지에서 전송을 클릭하면 이클립스 콘솔에 다음과 같이 회원 정보가 출력됩니다.

**\* 회원 정보\***  
박지성  
25  
남자  
날센돌이

## 16.6 제이쿼리에서 JSON 사용하기

8. 이번에는 반대로 서버의 서블릿에서 웹 페이지로 JSON 형식의 회원 정보를 전송해 보겠습니다. JsonServlet2 클래스를 생성합니다.





## 16.6 제이쿼리에서 JSON 사용하기

9. JsonServlet2 클래스를 다음과 같이 작성합니다.

### JSON 배열에 정보를 저장하는 과정

- ① memberInfo로 JSONObject 객체를 생성한 후 회원 정보를 name/value 쌍으로 저장
- ② membersArray의 JSONArray 객체를 생성한 후 회원 정보를 저장한 JSON 객체를 차례대로 저장
- ③ membersArray 배열에 회원 정보를 저장한 후 totalObject로 JSONObject 객체를 생성하여 name에는 자바스크립트에서 접근할 때 사용하는 이름인 members를, value에는 membersArray를 최종적으로 저장

## 16.6 제이쿼리에서 JSON 사용하기

코드 16-23 pro16/src/sec03/ex01/JsonServlet2.java

```
package ex02;

...

@WebServlet("/json2")
public class JsonServlet2 extends HttpServlet {
    ...
```

```
privateprotected void doHandle(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    request.setCharacterEncoding("utf-8");
    response.setContentType("text/html; charset=utf-8");
    PrintWriter writer = response.getWriter();
```

```
JSONObject totalObject = new JSONObject();
```

배열을 저장할 totalObject를 선언합니다.

```
JSONArray membersArray = new JSONArray();
```

memberInfo JSON 객체를 저장할 membersArray 배열을 선언합니다.

```
JSONObject memberInfo = new JSONObject();
```

```
memberInfo.put("name", "박지성");
```

회원 한 명의 정보가 들어갈 memberInfo JSON 객체를 선언합니다.

```
memberInfo.put("age", "25");
```

```
memberInfo.put("gender", "남자");
```

회원 정보를 name/value 쌍으로 저장합니다.

```
memberInfo.put("nickname", "날센돌이");
```

```
membersArray.add(memberInfo);
```

회원 정보를 다시 membersArray 배열에 저장합니다.

```
memberInfo = new JSONObject();
```

다른 회원 정보를 name/value 쌍으로 저장한 후 membersArray에 다시 저장합니다.

```
memberInfo.put("name", "김연아");
```

```
memberInfo.put("age", "21");
```

```
memberInfo.put("gender", "여자");
```

```
memberInfo.put("nickname", "칼치");
```

```
membersArray.add(memberInfo);
```

## 16.6 제이쿼리에서 JSON 사용하기

```
totalObject.put("members", membersArray);
```

totalObject에 members라는 name으로 membersArray를 value로 저장합니다.

```
String jsonInfo = totalObject.toJSONString();
```

JSONObject를 문자열로 변환합니다.

```
System.out.print(jsonInfo);
```

```
writer.print(jsonInfo);
```

JSON 데이터를 브라우저로 전송합니다.

```
}
```

```
}
```

---

## 16.6 제이쿼리에서 JSON 사용하기

10. json6.jsp를 다음과 같이 작성합니다.

코드 16-24 pro16/WebContent/test04/json6.jsp

```
<script>
$(function () {
    $("#checkJson").click(function () {
        $.ajax({
            type: "post",
            async: false,
            url: "${contextPath}/json2",
            success: function (data, textStatus) {
                var jsonInfo = JSON.parse(data);
                var memberInfo = "회원 정보<br>";
                memberInfo += "=====<br>";
            }
        });
    });
});
```

서버에서 전송한 JSON 데이터를 파싱합니다.

## 16.6 제이쿼리에서 JSON 사용하기

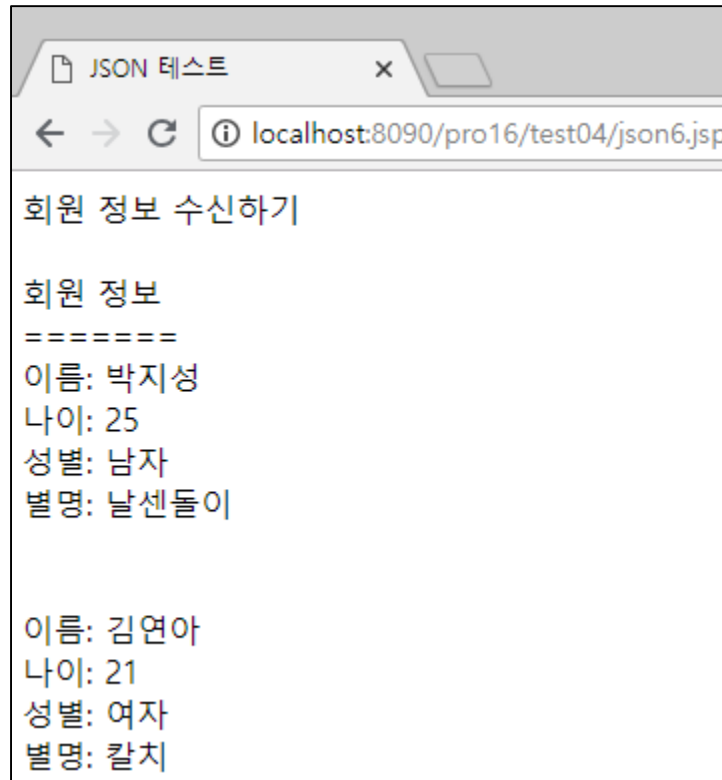
```
for (var i in jsonInfo.members) {  
    memberInfo += "이름: " + jsonInfo.members[i].name + "<br>";  
    memberInfo += "나이: " + jsonInfo.members[i].age + "<br>";  
    memberInfo += "성별: " + jsonInfo.members[i].gender + "<br>";  
    memberInfo += "별명: " +  
        jsonInfo.members[i].nickname + "<br><br><br>";  
}
```

```
$("#output").html(memberInfo);  
},  
error: function (data, textStatus) {  
    alert("에러가 발생했습니다.");  
},  
complete: function (data, textStatus) {  
}  
});  
});  
</script>
```

배열 이름 members로 각 배열 요소에 접근한 후  
name으로 value를 출력합니다.

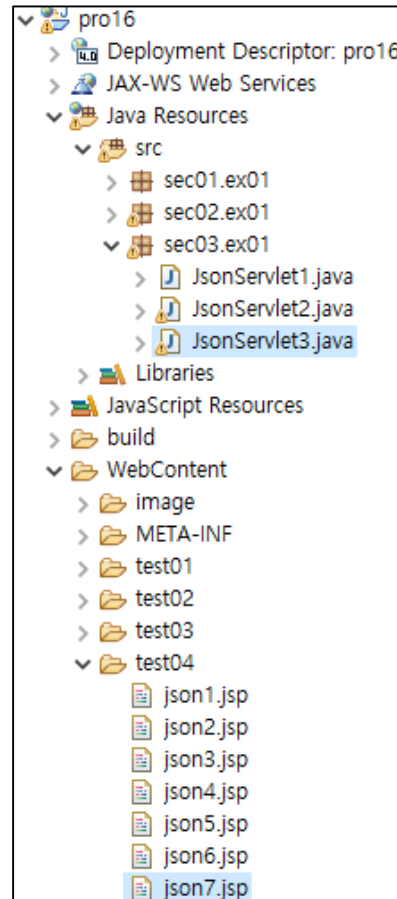
## 16.6 제이쿼리에서 JSON 사용하기

11. <http://localhost:8090/pro16/test04/json6.jsp>로 요청하여 회원 정보 수신하기를 클릭합니다.



## 16.6 제이쿼리에서 JSON 사용하기

12. 마지막으로 여러 개의 배열을 JSON으로 전달하는 예제를 만들어 보겠습니다. JsonServlet3 클래스를 추가로 생성합니다.



## 16.6 제이쿼리에서 JSON 사용하기

13. JsonServlet3 클래스를 다음과 같이 작성합니다.

### 여러 개의 배열을 전송하는 과정

- ① bookInfo의 JSONObject 객체를 생성한 후 도서 정보를 name/value 쌍으로 저장
- ② bookArray의 JSONArray 객체를 생성한 후 도서 정보를 저장한 bookInfo를 차례대로 저장
- ③ 이미 회원 배열을 저장하고 있는 totaObject의 name에는 배열 이름에 해당하는 books를, value에는 bookArray를 최종적으로 저장

코드 16-25 pro16/src/sec03/ex01/JsonServlet3.java

```
packate sec03.ex01;
```

```
...
```

```
@WebServlet("/json3")
```

```
public class JsonServlet3 extends HttpServlet {
```

```
...
```

```
private protected void doHandle(HttpServletRequest request, HttpServletResponse response)
```



## 16.6 제이쿼리에서 JSON 사용하기

```
throws ServletException, IOException {  
    request.setCharacterEncoding("utf-8");  
    response.setContentType("text/html; charset=utf-8");  
    PrintWriter writer = response.getWriter();
```

```
    JSONObject totaObject = new JSONObject();
```

```
    JSONArray membersArray = new JSONArray();
```

```
    JSONObject memberInfo = new JSONObject();
```

```
    memberInfo.put("name", "박지성");
```

```
    memberInfo.put("age", "25");
```

```
    memberInfo.put("gender", "남자");
```

```
    memberInfo.put("nickname", "날센돌이");
```

```
    membersArray.add(memberInfo);
```

```
    memberInfo = new JSONObject();
```

```
    memberInfo.put("name", "김연아");
```

```
    memberInfo.put("age", "21");
```

```
    memberInfo.put("gender", "여자");
```

```
    memberInfo.put("nickname", "칼치");
```

```
    membersArray.add(memberInfo);
```

```
    totaObject.put("members", membersArray);
```

배열을 최종적으로 저장할 JSONObject 객체를 생성합니다.

회원 정보를 저장한 배열을 배열 이름 members로 totaObject에 저장합니다

## 16.6 제이쿼리에서 JSON 사용하기

```
JSONArray bookArray = new JSONArray();
```

JSONArray 객체를 생성합니다.

```
JSONObject bookInfo = new JSONObject();
```

```
bookInfo.put("title", "초보자를 위한 자바 프로그래밍");
```

```
bookInfo.put("writer", "이병승");
```

```
bookInfo.put("price", "30000");
```

```
bookInfo.put("genre", "IT");
```

```
bookInfo.put("image", "http://localhost:8090/pro16/image/image1.jpg");
```

```
bookArray.add(bookInfo);
```

JSONObject 객체를 생성한 후 책 정보를 저장합니다.

bookArray에 객체를 저장합니다.

```
bookInfo = new JSONObject();
```

```
bookInfo.put("title", "모두의 파이썬");
```

```
bookInfo.put("writer", "이승찬");
```

```
bookInfo.put("price", "12000");
```

```
bookInfo.put("genre", "IT");
```

```
bookInfo.put("image", "http://localhost:8090/pro16/image/image1.jpg");
```

```
bookArray.add(bookInfo);
```

JSONObject 객체를 생성한 후 책 정보를 저장합니다.

bookArray에 객체를 저장합니다.

```
totaObject.put("books", bookArray);
```

도서 정보를 저장한 배열을 배열 이름

```
String jsonInfo = totaObject.toJSONString();
```

books로 totalObject에 저장합니다

```
System.out.print(jsonInfo);
```

```
writer.print(jsonInfo);
```

```
}
```

```
}
```

## 16.6 제이쿼리에서 JSON 사용하기

14. json7.jsp를 다음과 같이 작성합니다.

코드 16-26 pro16/WebContent/test04/json7.jsp

```
<script>
$(function () {
    $("#checkJson").click(function () {
        $.ajax({
            type: "post",
            async: false,
            url: "${contextPath}/json3",
            success: function (data, textStatus) {
                var jsonInfo = JSON.parse(data);
                var memberInfo = "회원 정보<br>";
                memberInfo += "=====<br>";
                for (var i in jsonInfo.members) {
                    memberInfo += "이름: " + jsonInfo.members[i].name + "<br>";
                    memberInfo += "나이: " + jsonInfo.members[i].age + "<br>";
                    memberInfo += "성별: " + jsonInfo.members[i].gender + "<br>";
                    memberInfo += "별명: " + jsonInfo.members[i].nickname + "<br><br><br>";
                }
            }
        });
    });
});
```

배열 이름 members로 회원 정보를 출력합니다.

## 16.6 제이쿼리에서 JSON 사용하기

```

var booksInfo = "<br><br><br>도서 정보<br>";
booksInfo += "=====<br>";
for (var i in jsonInfo.books) {
    booksInfo += "제목: " + jsonInfo.books[i].title + "<br>";
    booksInfo += "저자: " + jsonInfo.books[i].writer + "<br>";
    booksInfo += "가격: " + jsonInfo.books[i].price + "원 <br>";
    booksInfo += "장르: " + jsonInfo.books[i].genre + "<br>";
    imageURL = jsonInfo.books[i].image;
    booksInfo += "" + "<br><br><br>";
}
$("#output").html(memberInfo + "<br>" + booksInfo);
},
error: function (data, textStatus) {
    alert("에러가 발생했습니다.");
}
});
});
</script>

```

배열 이름 books로 도서 정보를 출력합니다.

이미지 URL을 구해 <img> 태그의 src 속성에 설정합니다.

## 16.6 제이쿼리에서 JSON 사용하기

15. <http://localhost:8090/pro16/test04/json7.jsp>로 요청하여 데이터 수신하기를 클릭하면 다음과 같이 회원 정보를 출력합니다. 이번에는 회원 정보는 물론 도서 정보도 배열로 전달받아 출력합니다.

### 데이터 수신하기

#### 회원 정보

=====

이름: 박지성

나이: 25

성별: 남자

별명: 날센들이

이름: 김연아

나이: 21

성별: 여자

별명: 칼치

#### 도서 정보

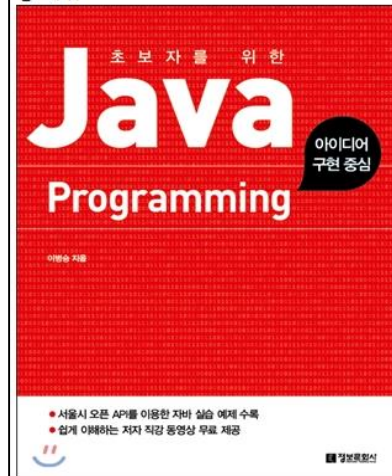
=====

제목: 초보자를 위한 자바 프로그래밍

저자: 이병승

가격: 30000원

장르: IT



제목: 모두의 파이썬

저자: 이승찬

가격: 12000원

장르: IT

