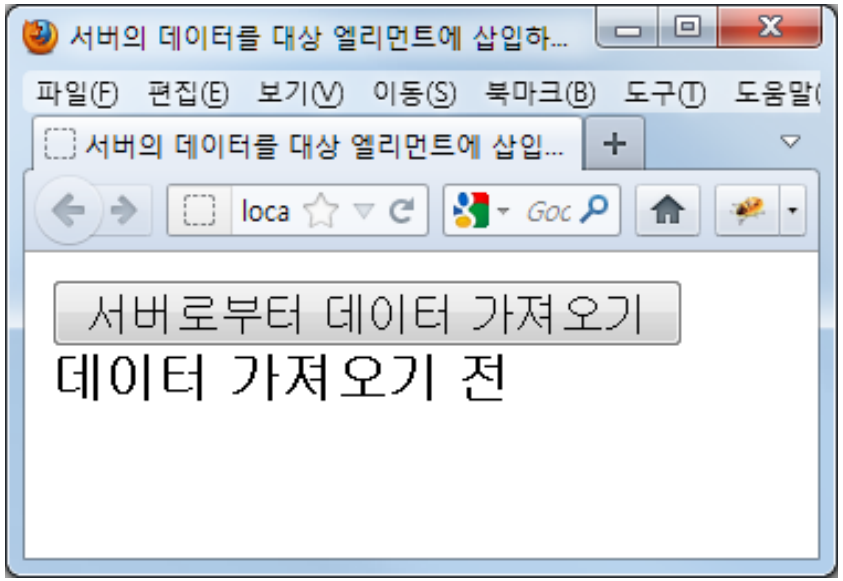
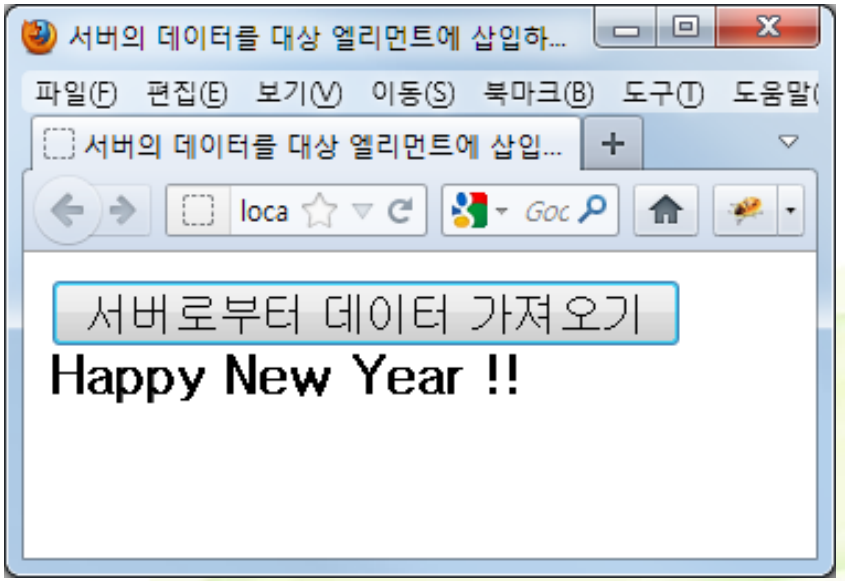
AJAX(jQuery)

서버의 데이터를 대상 엘리먼트에 삽입하기-load()

페이지를 동적으로 로드하기 위해서는 load() 메소드를 호출한다.





서버의 데이터를 대상 엘리먼트에 삽입하기-load()

```
load(url[, data][, success])
```

- url은 요청할 URL 주소이고, data는 요청할 때 서버에 보낼 자바스크립트 객체 맵이나 문자열 형식의 데이터이다.
- success는 요청이 성공했을 때 호출할 콜백 함수이다.
- load() 메소드는 완료된 응답을 일치하는 집합에 있는 엘리먼트의 콘텐츠로 삽입한다.

```
$("#container").load(“resource");
```

```
<!DOCTYPE html>
<html>
  <head>
      <meta charset="UTF-8">
      <title>서버의 데이터를 대상 엘리먼트에 삽입하기</title>
       <script src="../js/jquery.js" type="text/javascript"></script>
       <script type="text/javascript">
          $(function() {
                $('button').click(function() {
                     $('#container').load('resource.html');
                });
           });
        </script>
  </head>
  <body>
       <button> 서버로부터 데이터 가져오기 </button>
        <div id="container"> 데이터 가져오기 전 </div>
   </body>
</html>
```

menu.html

```
<p> 중 식 메 뉴 </p>
<ul>
<li> 짜장면 </li>
<li> 짬뽕 </li>
<li> 기스면 </li>
<li> 탕수육 </li>
</ul>
<p> 메뉴를 골라주세요.</p>
```

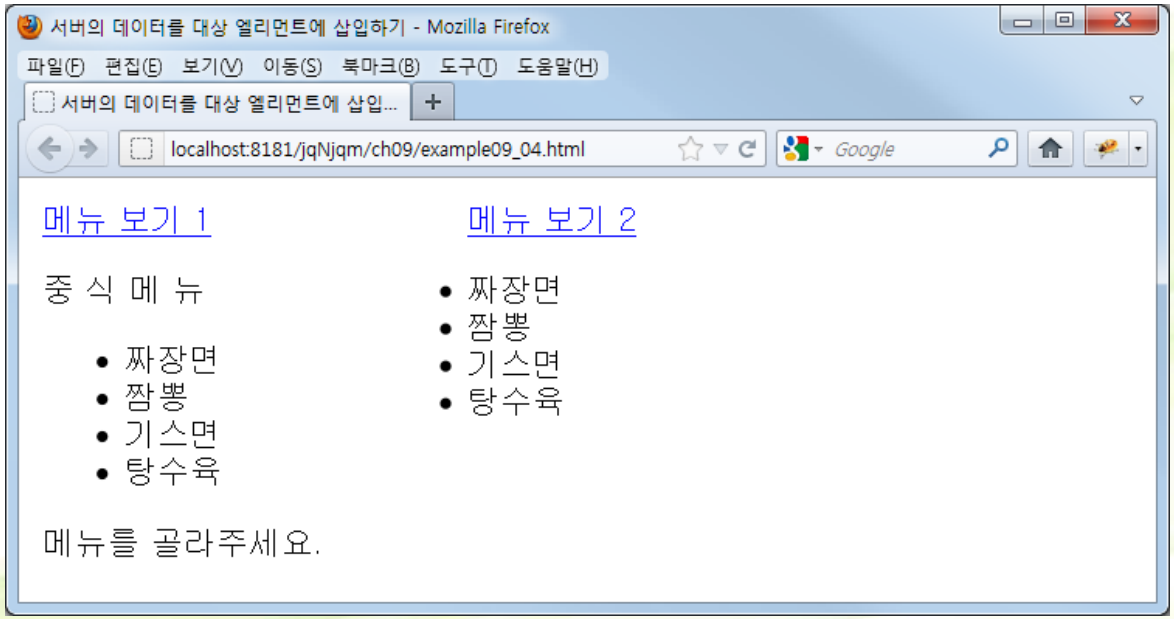
```
<!DOCTYPE html><html><head><meta charset="UTF-8">
<title>서버의 데이터를 대상 엘리먼트에 삽입하기</title>
<style type="text/css">
div {  width: 180px;  height: 80px;        margin: 3px;  float: left;       }
</style>
<script src="../js/jquery.js" type="text/javascript"></script>
<script type="text/javascript">
$(function() {
   $('#menu1').click(function () {
           $('#message1').load('menu.html');
            return false;
   });
   $('#menu2').click(function () {
            $('#message2').load('menu.html li');
            return false;
   });
});
</script> </head> <body>
<div> <a href="#" id="menu1">메뉴 보기 1</a><p>
        <span id="message1"></span>
</div>
<div> <a href="#" id="menu2">메뉴 보기 2</a><p>
        <span id="message2"></span>
</div>
</body>
</html>
```

Ajax 요청 직접 제어하기

- jQuery에서는 $.ajax()라는 유틸리티 함수로 Ajax 요청을 설정 제어한다.

```
$.ajax(options)
```

- 요청의 생성 방법과 통보 받을 콜백을 제어하고자 전달된 options 를 사용하여 Ajax 요청을 전송한다. 반환값은 XHR 인스턴스이다.



Ajax 요청 직접 제어하기

- url (String)
  - 요청 URL
- type (String)
  - 사용할 HTTP 메서드, 일반적으로 POST나 GET을 사용한다. 생략하면 기본값으로 GET을 사용한다.
- data (Object)
  - 요청에 전달되는 프로퍼티를 가진 객체. GET 요청이면 데이터는 쿼리 문자열로 제공된다. POST 요청이면 데이터는 요청의 본문으로 제공된다. 두 경우 모두 $.ajax() 유틸리티 함수가 값의 인코딩을 처리한다.

Ajax 요청 직접 제어하기

□ dataType (String)

- 응답의 결과로 반환되는 데이터의 종류를 식별하는 키워드
  - xml – 응답 텍스트는 XML 문서로 파싱되며, 콜백에 결과로 생성된 XML DOM을 전달한다.
  - html – 응답 텍스트는 처리 과정 없이 콜백 함수로 전달된다. 반환된 HTML 코드에 있는 모든 <script> 블록이 평가된다.
  - json – 응답 텍스트는 JSON 문자열로 평가되며, 생성된 객체는 콜백에 전달된다.
  - jsonp – 원격 스크립트를 허용한다는 점을 제외하고는 json 과 유사하다. 원격 서버가 이와 같은 방식을 지원한다고 가정한다.
  - script – 응답 텍스트는 콜백에 전달된다. 응답은 모든 콜백의 호출보다 먼저 자바스크립트 구문으로 처리된다.
  - text – 응답 텍스트는 일반 텍스트다.

Ajax 요청 직접 제어하기

- timeout (Number)
  - Ajax 요청의 제한 시간을 밀리초 단위로 설정한다. 제한 시간 안에 요청이 완료되지 않으면 요청을 취소하거나, error 콜백이 정의되어 있다면 호출된다.
- global (Boolean)
  - true나 flase에 따라 전역 함수(global function)를 활성화하거나 비활성화한다. 전역 함수는 엘리먼트에 덧붙일 수 있으며 Ajax 호출 동안 다양한 위치나 조건에서 실행된다. 값이 생략되면 기본 값으로 전역 함수를 활성화한다.
- contentTypeString
  - 요청에 명시되는 콘텐츠 타입. 생략하면 'application/x-www-form-urlencoded'가 기본으로 설정되며, 이는 폼 전송이 기본으로 사용하는 타입과 동일하다.

Ajax 요청 직접 제어하기

□ success (Function)

- 응답이 성공 사태 코드를 반환하면 호출되는 함수. 응답 본문은 이 함수의 첫 번째 매개변수로 전달되며, dataType 프로퍼티에 명시한 형태로 구성된다. 두 번째 매개변수는 상태 값을 나타내는 문자열이며, 이번 경우에는 항상 'success' 다.

□ error (Funcction)

- 응답이 에러 상태 코드를 반환하면 호출되는 함수. 매개변수가 세 개 전달되는데, 각각 XHR 인스턴스, 상태 값이 항상 'error' 인 메시지 문자열, 선택사항으로 XHR 인스턴스가 반환한 예외 객체다.

Ajax 요청 직접 제어하기

- complete (Function)
  - 요청이 완료되면 호출되는 함수, 매개변수 두 개가 전달되는데, 각각 XHR 인스턴스와 'success' 혹은 'error'를 나타내는 상태 메시지 문자열이다. success' 혹은 'error'를 나타내는 상태 메시지 문자열이다. success 나 error 롤백을 명시했다면, 이 함수는 해당 콜백이 호출된 후에 실행된다.
- beforeSend (Function)
  - 요청이 전송되기에 앞서 먼저 호출되는 함수. 이 함수는 XHR 인스턴스를 전달 받으며, 사용자 정의 헤더를 설정하거나 요청 전에 필요한 연산을 수행하는 데 사용할 수 있다.

Ajax 요청 직접 제어하기

- async (Boolean)
  - false 이면 요청이 동기 호출로 전송된다. 기본은 비동기 요청이다.
- processDataBoolean
  - false로 설정되면, URL 인코딩된 형태로 처리되어 전달된 데이터를 금지한다. 기본 값은 데이터가 'application/x-www-form-urlencoded' 타입의 요청에 사용하는 형태의 URL로 인코딩된다.
- ifModified Boolen
  - true 일 때 Last-Modified 헤더를 확인하여 마지막 요청 이후에 응답 콘텐츠가 변경되지 않았다면 요청이 성공한다. 만일 생략하면 헤더를 확인하지 않는다.

```
<!DOCTYPE html>
<html>
    <head>
    <meta charset="UTF-8">
     <title>서버의 데이터를 대상 엘리먼트에 삽입하기</title>
     <style type="text/css">
          div {
               width: 200px;  height: 80px;
               margin: 3px;  float: left;
          }
      </style>
      <script src="../js/jquery.js" type="text/javascript"></script>
      <script type="text/javascript">
          $(function() {
                $('#menu1').click(function () {
                     $.ajax({
                               url:'menu.html',
                               dataType:"html",
                               success: function(data){
                                    $('#message1').html(data);
                               }
                      });
                      return false;
                 });
```

```
        $('#menu2').click(function () {
                $.ajax({
                        url:'menu.html',
                        dataType:"html",
                         success: function(data){
                                $('#message2').html($(data).find('li'));
                         }
                 });
                 return false;
          });
      });
    </script>
    </head>
    <body>
       <div>
            <a href="#" id="menu1">메뉴 보기 1</a><p>
            <span id="message1"></span>
        </div>
        <div>
             <a href="#" id="menu2">메뉴 보기 2</a><p>
             <span id="message2"></span>
        </div>
     </body>
</html>
```

JSON 이용하기

- JavaScript Object Notion의 약어
- XML 데이터를 대신하기 위해서 사용한다.
- 키와 값을 쌍으로 가지는 구조
- 배열을 사용할 경우에는 대괄호 안에 중괄호를 사용하여 조합

JSON 이용하기

□ **JSON**에 저장되는 정보의 형태

- 배열
  - 대괄호 안에 값을 콤마로 구분하여 나열하며, 대괄호 안에 나오는 순서대로 배열 요소의 순서가 매겨진다.
  - [1, 2, 3]
- 객체
  - 중괄호 안에 있는 이름: 값의 형태로 멤버 하나를 표현
  - 각 멤버는 콤마로 구분
  - 순서가 아닌 이름으로 읽기 때문에 멤버의 순서는 의미가 없다.
  - {"name": "레몬" }

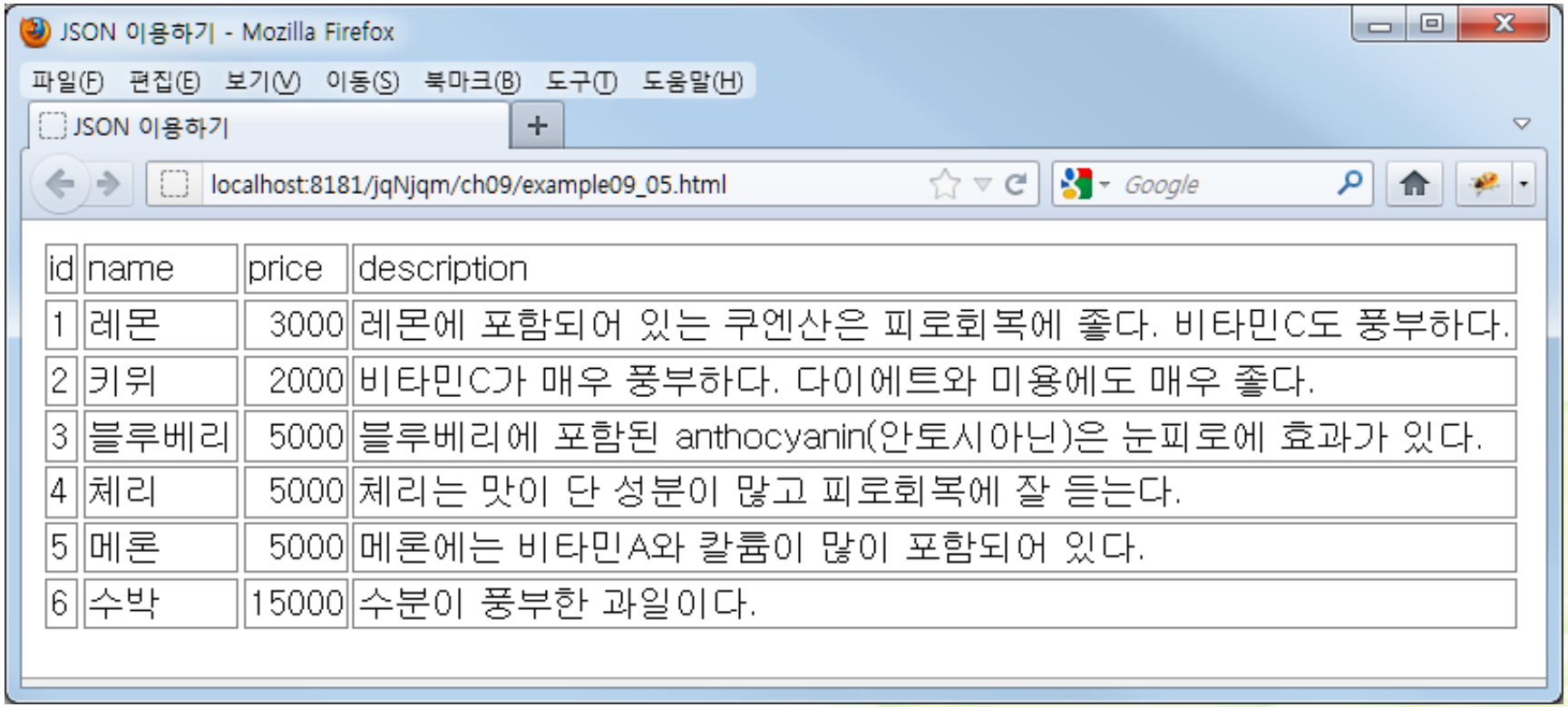
JSON 이용하기

```
[
  {
    "id": "1",
    "name": "레몬",
    "price": " 3000",
    "description": "레몬에 포함되어 있는 쿠엔산은 피로회복에 좋다. "
  },
  {
    "id": "2",
    "name": "키위",
    "price": " 2000",
    "description": "비타민C가 매우 풍부하다."
  }
]
```

item.json


```
[
 {
  "id": "1",    "name": "레몬",    "price": " 3000",
  "description": "레몬에 포함되어 있는 쿠엔산은 피로회복에 좋다. 비타민C도 풍부하다."
 },
 {
  "id": "2",    "name": "키위",    "price": " 2000",
  "description": "비타민C가 매우 풍부하다. 다이에트와 미용에도 매우 좋다."
 },
 {    "id": "3",    "name": "블루베리",    "price": " 5000",
  "description": "블루베리에 포함된 anthocyanin(안토시아닌)은 눈피로에 효과가 있다."
 },
 {    "id": "4",    "name": "체리",    "price": " 5000",
  "description": "체리는 맛이 단 성분이 많고 피로회복에 잘 듣는다."
 },
 {    "id": "5",    "name": "메론",    "price": " 5000",
  "description": "메론에는 비타민A와 칼륨이 많이 포함되어 있다."
 },
 {
  "id": "6",    "name": "수박",    "price": "15000",
  "description": "수분이 풍부한 과일이다."
 }
]
```

JSON 이용하기



id	name	price	description
1	레몬	3000	레몬에 포함되어 있는 쿠엔산은 피로회복에 좋다. 비타민C도 풍부하다.
2	키위	2000	비타민C가 매우 풍부하다. 다이에트와 미용에도 매우 좋다.
3	블루베리	5000	블루베리에 포함된 anthocyanin(안토시아닌)은 눈피로에 효과가 있다.
4	체리	5000	체리는 맛이 단 성분이 많고 피로회복에 잘 듣는다.
5	메론	5000	메론에는 비타민A와 칼륨이 많이 포함되어 있다.
6	수박	15000	수분이 풍부한 과일이다.

```
<!DOCTYPE html><html><head><meta charset="UTF-8">
<title>JSON 이용하기</title>
<style>
td {  border: 1px solid gray;  }
</style>
<script src="../js/jquery.js" type="text/javascript"></script>
<script type="text/javascript">
$(function() {
$.getJSON('item.json', function(data, textStatus) {
        $("#treeData").append(
                "<tr><td>id</td>" + "<td>name</td>"
              + "<td>price</td>" + "<td>description</td>" + "</tr>");
        $.each(data, function() {
              $("#treeData").append("<tr>" + "<td>"
                      + this.id + "</td>" + "<td>"
                      + this.name + "</td>" + "<td align='right'>"
                      + this.price + "</td>" + "<td>"
                      + this.description + "</td>" + "</tr>");
          });
 });
});
</script></head>
<body>
  <table id="treeData"></table>
</body>
</html>
```

JSON 이용하기

```
$.getJSON('item.json', function(data, textStatus) {
        $.each(data, function() {
                $("#treeData").append("<tr>" + "<td>"
                        + this.id + "</td>" + "<td>"
                        + this.name + "</td>" + "<td align='right'>"
                        + this.price + "</td>" + "<td>"
                        + this.description + "</td>" + "</tr>");
        });
});
```

□ $.getJSON(url[, data][, success])

- JSON으로 표현한 데이터를 파일에 저장해 두었다가 필요할 경우 이를 로드 하는 전역 메소드

XML 파일을 GET 방식으로 로드하기-$.get()

```
$.get('item.xml', function(data) {
        $(data).find('item').each(function() {
         var $item = $(this);
         $("#treeData").append("<tr>" + "<td>"
            + $item.attr("id") + "</td>" + "<td>"
            + $item.attr("name") + "</td>" + "<td align='right'>"
            + $item.find("price").text() + "</td>" + "<td>"
            + $item.find("description").text() + "</td>" + "</tr>");
      });
    });
```

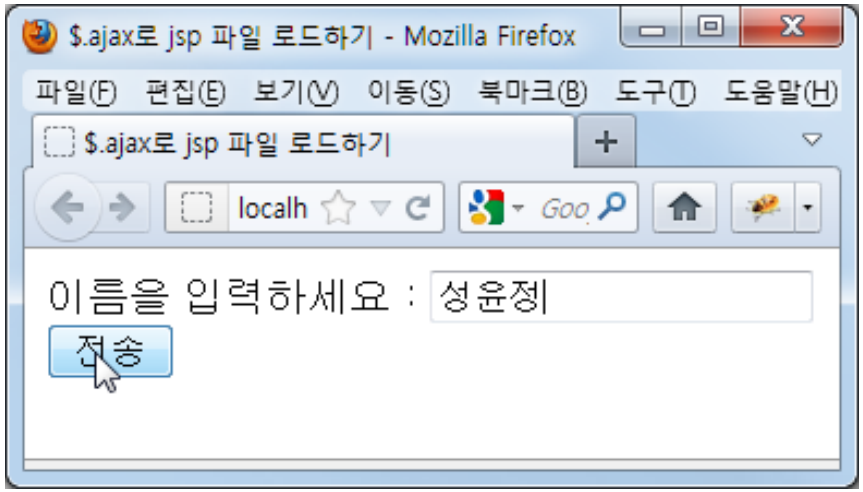
□ $.get(URL, fn)

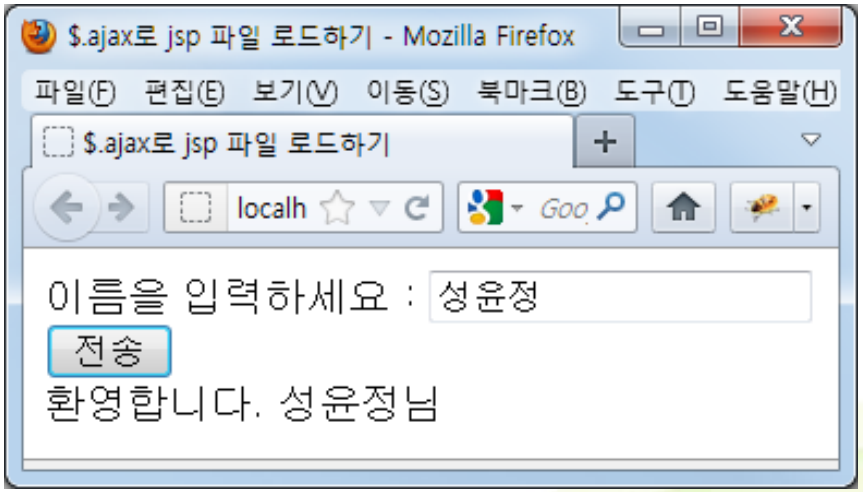
- GET 방식으로 서버와 통신하는 jQuery Ajax 함수
- URL로 지정한 파일을 로드해서 그 데이터를 텍스트 형식으로 콜백 함수(fn)에게 넘겨주는 기능을 한다.

```
<?xml version="1.0" encoding="UTF-8"?>
<items>
 <item id="1" name="레몬"><price>3000</price>
   <description> 레몬에 포함되어 있는 쿠엔산은 피로회복에 좋다. 비타민C도 풍부하다.
   </description>
 </item>
 <item id="2" name="키위"><price>2000</price>
   <description> 비타민C가 매우 풍부하다. 다이에트와 미용에도 매우 좋다. </description>
 </item>
 <item id="3" name="블루베리"><price>5000</price>
   <description> 블루베리에 포함된 anthocyanin(안토시아닌)은 눈피로에 효과가 있다.
   </description>
 </item>
 <item id="4" name="체리">
   <price>5000</price>
   <description> 체리는 맛이 단 성분이 많고 피로회복에 잘 듣는다. </description>
 </item>
 <item id="5" name="메론">
   <price>5000</price>
   <description> 메론에는 비타민A와 칼륨이 많이 포함되어 있다. </description>
 </item>
 <item id="6" name="수박">
   <price>15000</price>
   <description> 수분이 풍부한 과일이다.</description>
  </item>
</items>
```

```
<!DOCTYPE html><html><head><meta charset="UTF-8">
<title>XML 파일을 GET 방식으로 로드하기</title>
<style>
td {  border: 1px solid gray;  }
</style>
<script src="../js/jquery.js" type="text/javascript"></script>
<script type="text/javascript">
$(function() {
 $.get('item.xml', function(data) {
            $("#treeData").append(
                  "<tr><td>id</td>" + "<td>name</td>"
                + "<td>price</td>" + "<td>description</td>" + "</tr>");
            $(data).find('item').each(function() {
            var $item = $(this);
                $("#treeData").append("<tr>" + "<td>"
                        + $item.attr("id") + "</td>" + "<td>"
                        + $item.attr("name") + "</td>" + "<td align='right'>"
                        + $item.find("price").text() + "</td>" + "<td>"
                        + $item.find("description").text() + "</td>" + "</tr>");
            });
    });
});
</script></head>
<body>
  <table id="treeData"></table>
</body></html>
```

POST 방식으로 서버와 통신하기-$.post()





□ $.post(URL[, data][,fn][,dataType])

- POST 방식으로 서버와 통신하는 jQuery Ajax 함수이다. $.post()는 URL로 지정한 파일을 로드해서 그 데이터를 텍스트 형식으로 콜백 함수(fn)에게 넘겨주는 기능을 한다.

POST 방식으로 서버와 통신하기-$.post()

```
var username = $('.username').val();
var sendData = 'username=' + username;

$.post("welcome.jsp",
        sendData,
        function (data) {
                $('#message').html(data);
        });
```

welcome.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!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>Insert title here</title>
</head>
<body>
 환영합니다. ${param.username}님
</body>
</html>
```

```
<!DOCTYPE html><html><head><meta charset="UTF-8">
<title>$.ajax로 jsp 파일 로드하기</title>
<style>
td {  border: 1px solid gray;  }
</style>
<script src="../js/jquery.js" type="text/javascript"></script>
<script type="text/javascript">
$(function() {
    $('#submit').click(function () {
            var username = $('.username').val();
            var sendData = 'username=' + username;
            $.post(    "welcome.jsp",    sendData,
                  function (msg) {
                  $('#message').html(msg);
            });
            return false;
    });
});
</script></head>
<body>
    <form>
    <label> 이름을 입력하세요 : </label>
    <input type="text" name="username" class="username"/>  <br/>
    <input type="button" id="submit"  value="전송"/>
  </form>
  <div id="message"></div>
</body></html>
```

Ajax에 대한 global 옵션 설정-$.ajaxSetup()

```
$.ajaxSetup({
        type:"POST",
        url:"logincheck.jsp",
        dataType : "text",
        success: function (msg) {
                $('#message').html(msg);
       }
});
$.ajax({
       data: sendData
});
```

□ $.ajaxSetup()

- jQuery Ajax에 대한 공통적인 기본 설정들을 지정하기 위해서 사용된다.

logincheck.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<!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>Insert title here</title>
</head>
<body>
입력한 아이디는 ${param.username}이며
<c:choose>
  <c:when test="${param.username=='admin' &&  param.password=='1234'}">
     당신은 접근 권한이 있습니다.
  </c:when>
  <c:otherwise>
    죄송합니다. 당신은 접근 권한이 없습니다.
  </c:otherwise>
</c:choose>
</body>
</html>
```

```
<!DOCTYPE html><html><head><meta charset="UTF-8">
<title>$.ajax로 jsp 파일 로드하기</title>
<style>
td {  border: 1px solid gray;  }
</style>
<script src="../js/jquery.js" type="text/javascript"></script>
<script type="text/javascript">
$(function() {
    $('#submit').click(function () {
        var username = $('.username').val();
        var pwd = $('.passwd').val();
        var sendData = 'username=' + username+'&password='+pwd;
        $.ajaxSetup({   type:"POST",  url:"logincheck.jsp", dataType : "text",
                    success: function (msg) {
                            $('#message').html(msg);
                    }
        });
        $.ajax({          data: sendData          });
        return false;
    });
});
</script></head>
```

```
<body>
    <form>
    <label> 이름을 입력하세요 : </label>
    <input type="text" name="username" class="username"/>  <br/>
    <label> 패스워드를 입력하세요 :</label>
    <input type="password"  name="password" class="passwd"/>  <br/>

    <input type="button" id="submit" value="전송"/>
  </form>
  <div id="message"></div>
</body>
</html
```

스크립트 로드하기-$.getScript()

```
$.getScript("test.js");
```

test.js

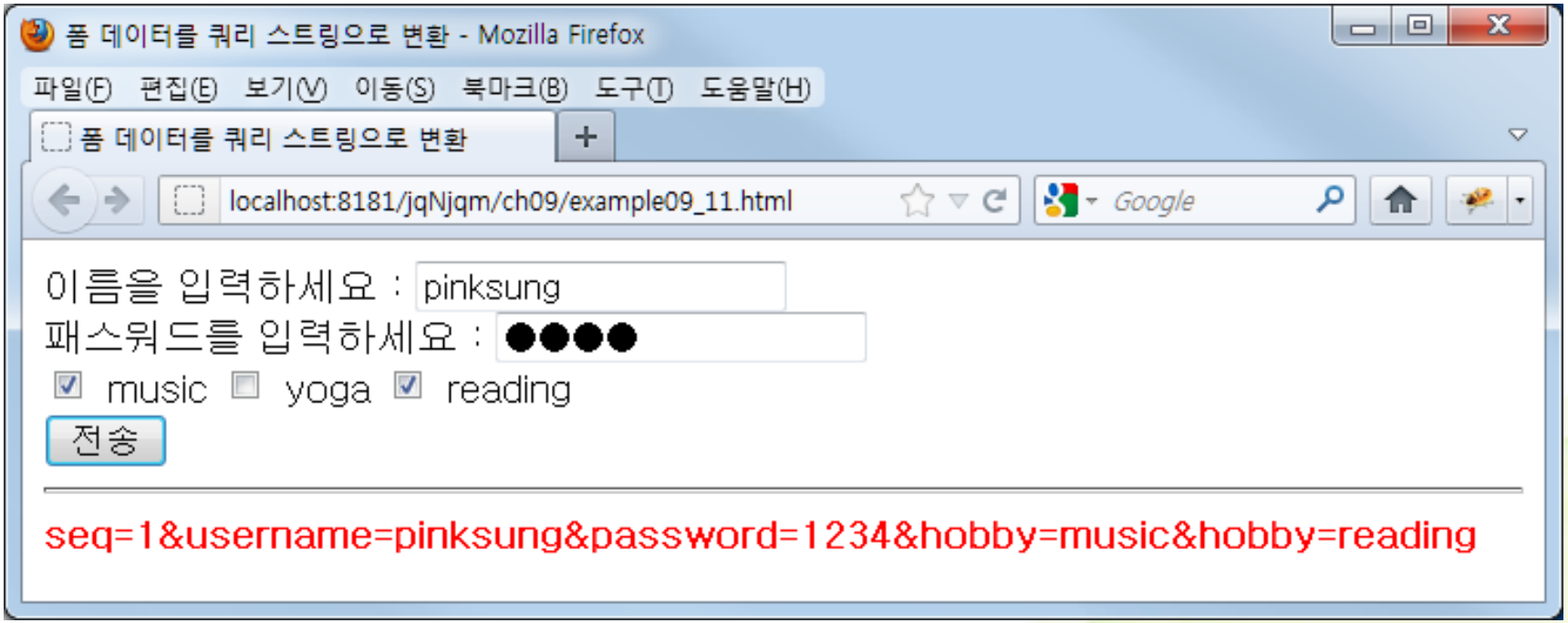
```
function call(param){
    return ("Hello, " + param);
}
```

□ $.getScript(url[, success])

- 자바 스크립트 파일을 로드한다.
- url : 요청할 서버 측 자원의 URL
- success : 요청이 성공했을 때 호출할 함수

```
<!DOCTYPE html><html><head><meta charset="UTF-8">
<title>스크립트 로드하기</title>
<script src="../js/jquery.js" type="text/javascript"></script>
<script type="text/javascript">
  $(document).ready(function() {
        $.getScript("test.js");
        $('#submit').click(function() {
            var msg=call($('.username').val());
            $('#message').html(msg);
            return false;
         });
  });
</script>
</head>
<body>
  <form>
    <label> 이름을 입력하세요 : </label>
    <input type="text" name="username" class="username"/>  <br/>
    <input type="button" id="submit" value="전송"/>
    <div id="message"></div>
  </form>
</body>
</html>
```

폼 데이터를 쿼리 스트링으로 변환-serialize()



폼 데이터를 쿼리 스트링으로 변환-serialize()

```
var form_data=$('form').serialize();
```

□ serialize()

- 폼 엘리먼트의 값을 쿼리 스트링으로 변환한다.
- 키/값의 쌍 형태의 문자열로 구성되는데 이를 쿼리 스트링이라고 한다.

```
seq=1&username=pinksung&password=1234&hobby=music&hobby=reading
```

```
<!DOCTYPE html><html><head><meta charset="UTF-8">
<title>폼 데이터를 쿼리 스트링으로 변환</title>
<style type="text/css">
div{  font-weight: bold; color:red;  }
</style>
<script src="../js/jquery.js" type="text/javascript"></script>
<script type="text/javascript">
$(document).ready(function() {
   $('#submit').click(function () {
       var form_data=$('form').serialize();
      $('#result').text(form_data);
  });
});
</script></head><body>
<form>
<input type="hidden" name="seq" value="1">
    <label> 이름을 입력하세요 : </label>
    <input type="text" name="username"/>  <br/>
    <label> 패스워드를 입력하세요 :</label>
    <input type="password"  name="password"/>  <br/>
    <input type="checkbox" name="hobby" value="music"> music
    <input type="checkbox" name="hobby" value="yoga"> yoga
    <input type="checkbox" name="hobby" value="reading"> reading <br/>
    <input type="button" id="submit" value="전송"/>
</form>
<hr><div id="result"></div></body></html>
```

엘리먼트 형식의 값을 배열 형태로 변환- serializeArray()

```
var form_data=$('form').serialize();
```

□ serializeArray()

- 폼 엘리먼트의 값을 객체 배열로 변환한다.
- 객체 배열은 {name=value}의 Array 형태

```
[ Object { name="seq", value="1"},
  Object { name="username", value="subean"},
  Object { name="password", value="4321"},
  Object { name="hobby", value="yoga"}  ]
```

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>폼 데이터를 배열 형태로 변환</title>
<style type="text/css">
div{
font-weight: bold; color:red;
}
</style>
<script src="../js/jquery.js" type="text/javascript"></script>
<script type="text/javascript">
   $(document).ready(function() {
         $('#submit').click(function () {
               $("#result").empty();
              var form_data=$('form').serializeArray();
              console.log(form_data);
              //$('#result').text(form_data);
              $.each(form_data, function(index, value) {
                        $("#result").append(value.name + "=" + value.value + "<br>");
              });
     });
});
</script>
</head>
```

```
<body>
<form>
<input type="hidden" name="seq" value="1">
    <label> 이름을 입력하세요 : </label>
    <input type="text" name="username"/>  <br/>
    <label> 패스워드를 입력하세요 :</label>
    <input type="password"  name="password"/>  <br/>
    <input type="checkbox" name="hobby" value="music"> music
     <input type="checkbox" name="hobby" value="yoga"> yoga
     <input type="checkbox" name="hobby" value="reading"> reading <br/>
    <input type="button" id="submit" value="전송"/>
</form>
<hr>
<div id="result"></div>
</body>
</html>
```