웹과 대화를 합시다
외향적인 애플리케이션
(extroverted application)

Ajax 란? – Asynchronous JavaScript and XML

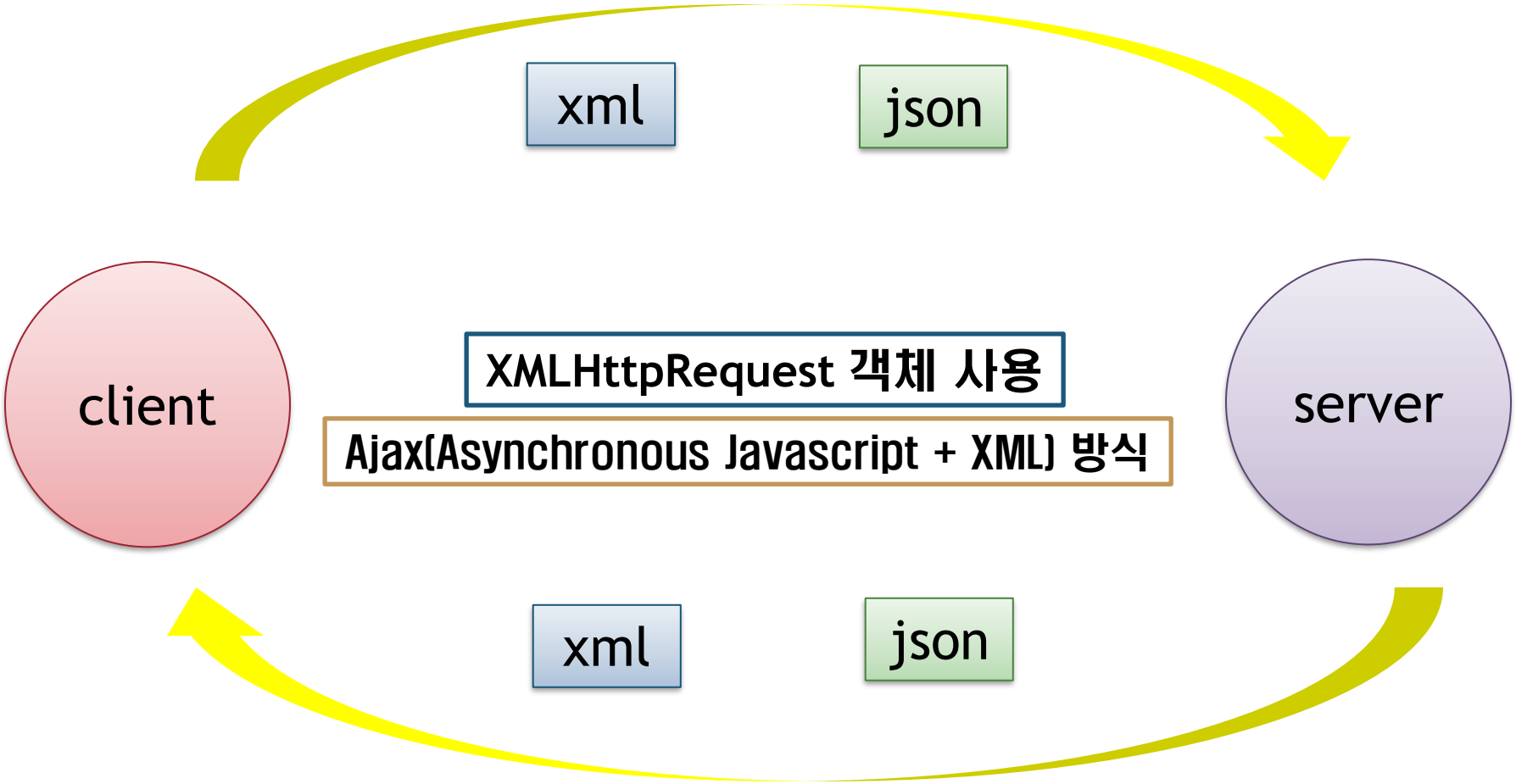
- **Ajax(Asynchronous Javascript + XML)란?**
  - **웹 어플리케이션 개발 시에 클라이언트와 서버의 통신방법에 대한 형태로 자바스크립트와 XML에 기반한 비동기 통신기법을 사용한다.**
  - **XMLHttpRequest()를 사용해서 비동기식으로 회수하는 데이터의 패턴을 보통 Ajax(Asynchronous JavaScript and XML) 라고 한다.**
  - **이때 사용하는 XMLHttpRequest(XHR)은 JavaScript API 이다.**

Defining Ajax
Ajax isn't a technology.
It's really several technologies, each flourishing in its own right,
coming together in powerful new ways. Ajax incorporates:
standards-based presentation using XHTML and CSS;
dynamic display and interaction using the Document Object Model;
data interchange and manipulation using XML and XSLT;
asynchronous data retrieval using XMLHttpRequest;
and JavaScript binding everything together.

Ajax 정의하기
Ajax는 기술이 아닙니다.
그것은 정말로 여러 가지 기술입니다. 각각의 기술은 그 자체로 번성합니다.
강력한 새로운 방식으로 모입니다.
Ajax는 다음을 통합합니다. XHTML 및 CSS를 사용하는 표준 기반 프레젠테이션.
문서 객체 모델을 사용한 동적 표시 및 상호 작용; XML 및 XSLT를 사용한 데이터 교환 및 조작;
XMLHttpRequest를 사용한 비동기 데이터 검색; 그리고 JavaScript 바인딩 모든 것을 함께.

Jesse James Garrett
(제시 제임스 가렛)

Ajax(Asynchronous Javascript +(and) XML)



XMLHttpRequest

- 서버와 비동기 통신을 하는 객체
  - 생성 방법
    - new XMLHttpRequest()
- XMLHttpRequest 주요 메소드

메소드	설 명
open (method, url, async)	요청의 초기화로 요청방식, 요청 URL, 비동기 여부를 지정 method : HTTP 요청방식. GET 또는 POST값을 주로 사용 url : 요청하는 자원의 URL. async : true나 생략할 경우 비동기, false일 경우 동기방식
send()	서버에 요청을 보냅니다. GET 요청에 사용
send(*string*)	서버에 요청을 보냅니다. POST 요청에 사용
abort()	요청 취소

참고 : https://www.w3schools.com/js/js_ajax_http.asp

http://www.tcpschool.com/ajax/ajax_server_request

XMLHttpRequest 의 속성

속성	설명
readyState	요청의 상태를 나타내는 정수 값 - 0 : 초기화 이전 상태. open() 호출 전 - 1 : 로드 되지 않은 상태. send() 메소드 호출 전 - 2 : 로드 된 상태. end() 메소드 호출 후 응답헤더와 상태 받음 - 3 : 상호작용 상태. 데이터를 받고 있는 상태 - 4 : 완료 상태. 모든 데이터를 받은 상태
status	서버로부터 받은 응답의 상태를 나타내는 HTTP 상태코드 - 200, 404, 500 등
statusText	서버로부터 받은 응답의 상태 메세지
responseText	응답으로 받은 문자열
responseXML	응답으로 받은 XML DOM 객체
onreadystatechange	readyState 속성이 변경될 때마다 실행할 이벤트 핸들러(콜백함수) 지정

참고 : https://www.w3schools.com/js/js_ajax_http.asp

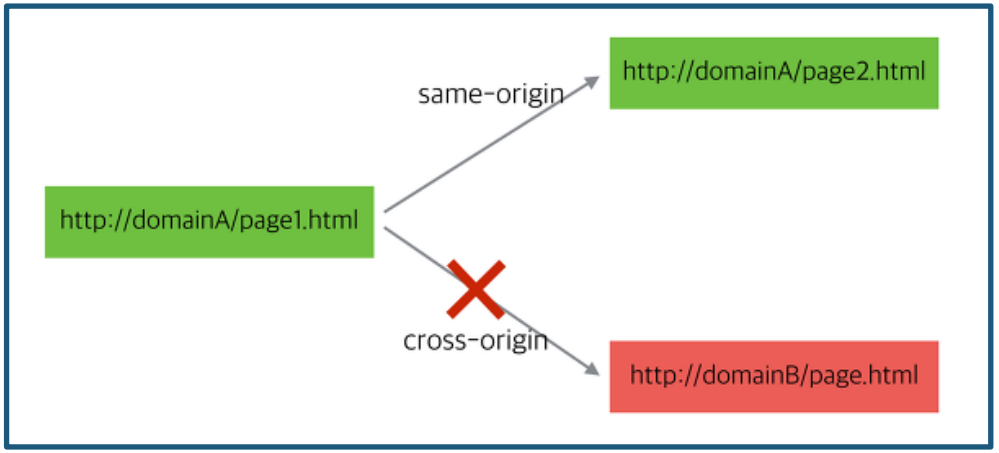
HTTP 상태 코드 참조 https://ko.wikipedia.org/wiki/HTTP_%EC%83%81%ED%83%9C_%EC%BD%94%EB%93%9C

XMLHttpRequest Level2

- HTML5 에서 추가된 스팩 중 XHR2 가 있습니다.
- XHR 은 이미 요즘엔 흔하게 사용되는 Ajax 의 기반이 되는 XMLHttpRequest 를 줄여 부른겁니다.
- 이것이 HTML5 스팩에서 좀 더 많은 기능들을 포함하면서 Level 2 로 발표되었고, 이를 지원하는 브라우저들이 늘어나고 있습니다.
- cross-origin resource sharding 이 포함되었습니다.
- 바이너리나 폼 데이터를 post 방식으로 전송 가능합니다.
- 요청과 응답 상태를 확인할 수 있는 이벤트 추가되었습니다.

- Cross-Origin Resource Sharing(CORS)
  - ✓ Ajax 는 아시다시피 브라우저의 Same-Origin Policy 정책에 의해 도메인이 다른 호스트로의 송수신이 금지되었습니다.
  - ✓ 하지만, OpenAPI 나 여러 공개 API 를 제공하거나 부득이 다른 호스트에서 데이터를 송수신 받을 필요가 있을 때 매우 곤란한 상황이 됩니다. 이것을 회피하기 위해 여러가지 방법들이 모색되었고, JSONP 라는 편법적인 전송 방식까지 널리 사용되고 있는 상황입니다.

▶ Cross-Domain 지원



<실습1> 파일의 내용을 Ajax로 가져오기

luckyornot.html
luckyornot.js
luckyornot.txt

요청객체 XMLHttpRequest를 사용해서 회수하는
데이터의 패턴을 보통 Ajax 또는 XHR이라고 합니다.

<실습1-1> 파일의 내용을 Ajax로 가져오기

luckyornot.html

```
<!doctype html>
<html lang="en">
<head>
<title>운이 좋은지 나쁜지</title>
<meta charset="utf-8">
<script src="luckyornot.js"></script>
</head>
<body>

<h1>행운인가 불행인가</h1>
<form>
<input type="button" id="amilucky" value="당신은 운이 좋은가요?">
</form>

<p id="luck">
</p>

</body>
</html>
```

<실습1-2> 파일의 내용을 Ajax로 가져오기

luckyornot.js

```
 6 window.onload = init;
 7 function init() {
 8     var button = document.getElementById("amilucky");
 9     button.onclick = getLuck;
10 }
11
12 function getLuck() {
13     var url = "luckyornot.txt";
14     var request = new XMLHttpRequest();
15     request.open("GET", url);
16     request.onload = function() {
17         if (request.status == 200) {
18             displayLuck(request.responseText);
19         }
20     };
21     request.send(null);
22 }
23
24 function displayLuck(luck) {
25     var p = document.getElementById("luck");
26     p.innerHTML = "당신은 오늘 " + luck + "합니다";
27 }
```

luckyornot.txt

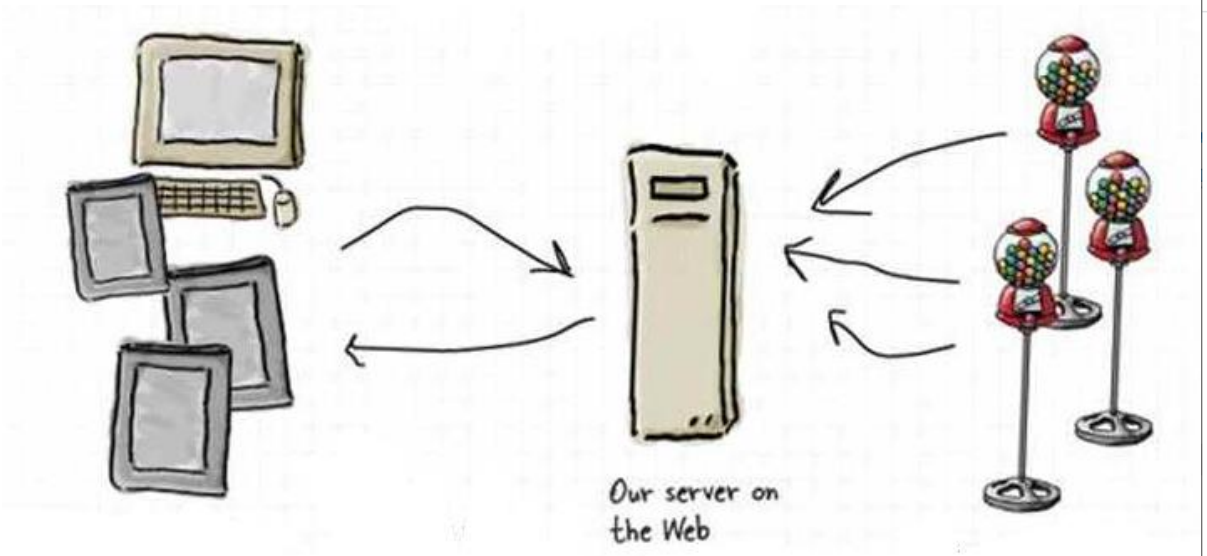
```
1 안개가득
```

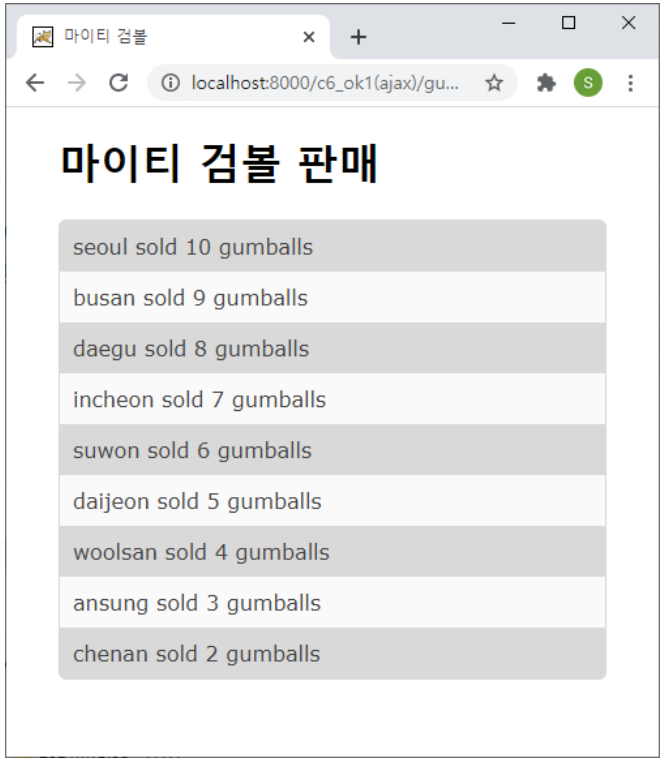
http://localhost:8000/js/c6_ok1(ajax)/luckyornot.html

<실습2> JSON 파일의 내용을 Ajax로 가져오기 – gumball.html

sales.json을 만들고 gumball 의 판매 목록이 보이도록 실행이 된다.

▶ 데이터 수집방법





http://localhost:8000/js/c6_ok1(ajax)/gumball.html

XML에서 JSON을 다시 만나 봅시다.

- XML 은 사람이 읽고, 기계가 인식할 수 있는 모든 것을 데이터형식으로 읽으려고 하였습니다.
- 처음 XMLHttpRequest 가 발견되었을 당시 XML 이 데이터를 교환하는 방식 그래서 XMLHttpRequest로 이름 붙여졌습니다.

- Who's JSON?

✓ JavaScript Object Notation 가장 최신의 데이터 형식, 가장 읽기 쉬운 data format, JavaScript 객체와 값으로 빠르게 쉽게 파싱 될 수 있습니다.

✓ JSON을 네트워크를 경유해서 자바스크립트 데이터를 교환하고, 웹저장소 API를 사용해서 로컬저장소에 데이터를 저장하고, 웹 데이터에 접근하여 여러 다른 방식으로 사용할 것입니다.

JSON의 창시자인 Douglas Crokford

더글라스 크락포드
미국 프로그래머
출생: 1955년, 미국 미네소타 주
학력: 샌프란시스코 주립 대학교

How to set up your own Web Server

sales.json


```
[{"name":"seoul","time":1308774240669,"sales":10},
{"name":"busan","time":1308774240669,"sales":9},
{"name":"daegu","time":1308774240669,"sales":8},
{"name":"incheon","time":1308774240669,"sales":7},
{"name":"suwon","time":1308774240669,"sales":6},
{"name":"daijeon","time":1308774240669,"sales":5},
{"name":"woolsan","time":1308774240669,"sales":4},
{"name":"ansung","time":1308774240669,"sales":3},
{"name":"chenan","time":1308774240669,"sales":2}]
```

<실습2-1> gumball.html 구조 – 추가1, 추가2

```
 9 function init() {
10     getSales();
11 }
12
13 function getSales() {
14     var url = "sales.json";
15     var request = new XMLHttpRequest();
16     request.open("GET", url);
17     request.onload = function() {
18         if (request.status == 200) {
19             updateSales(request.responseText);
20         }
21     };
22     request.send(null);
23 }
24
25 function updateSales(responseText) {
26     var salesDiv = document.getElementById("sales");
27     var sales = JSON.parse(responseText);
28     for (var i = 0; i < sales.length; i++) {
29         var sale = sales[i];
30         var div = document.createElement("div");
31         div.setAttribute("class", "saleItem");
32         div.innerHTML = sale.name + " sold " + sale.sales + " gumballs";
33         salesDiv.appendChild(div);
34     }
35 }
```

```
80 <div id="sales">
81 </div>
```

<실습2-2> gumball1.html 의 실행화면 만들기

http://localhost:8000/js/c6_ok1(ajax)/gumball1.html

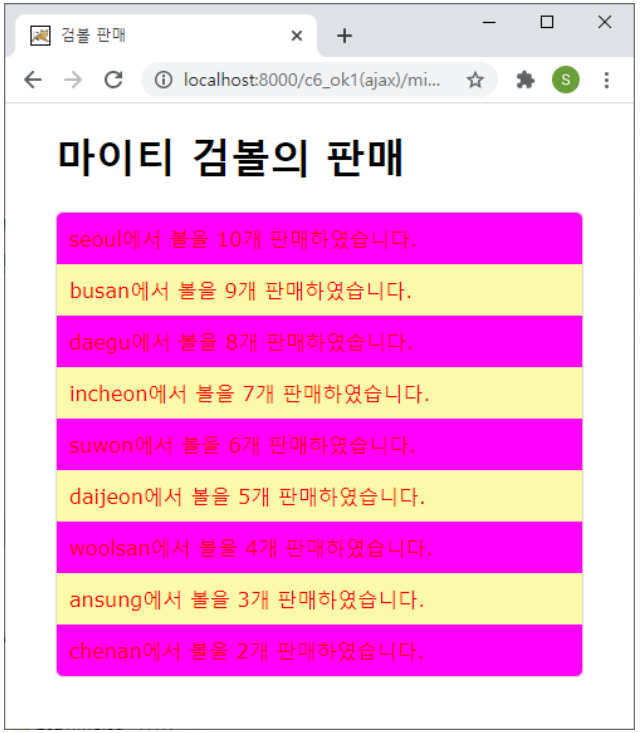
<실습2-3> gumball1.html 의 css 소스를 수정하여 보자

```
<style>
body {
    margin-left: 40px;
    margin-right: 40px;
}
/* 전체배경 핑크색#ff00ff , div id="sales"==># 으로 접근 */
 div#sales {
    background-color: #ff00ff;
    -webkit-border-radius: 6px;
    border-radius: 6px;
    margin: 10px 0px 0px 0px;
    padding: 0px;
    border: 1px solid #d9d9d9;
}
/* 핑크색위에,글자색#ff0000 , div의 saleItem은 class이므로 div.saleItem 점으로 접근 */
div.saleItem {
    font-family: Verdana, Helvetica, sans-serif;
    color: #ff0000;
    padding: 10px;
}
/* 이후 세개가 함께나와야 차례대로 행의 색이 바뀜 , 두번째행의 색:노란색#fafaaa */
div.saleItem:nth-child(2n) {
    background-color: #fafaaa;
}

div.saleItem:first-child {
    -webkit-border-top-left-radius: 6px;
    -webkit-border-top-right-radius: 6px;
    border-top-left-radius: 6px;
    border-top-right-radius: 6px;
}
div.saleItem:last-child {
    -webkit-border-bottom-left-radius: 6px;
    -webkit-border-bottom-right-radius: 6px;
    border-bottom-left-radius: 6px;
    border-bottom-right-radius: 6px;
}
</style>
```

<실습3> mightgumball.html, js, css로 소스 분리

- gumball1.html을 mightgumball.html, js, css로 분리하여 보자



mightgumball.html

mightygumball.js

mightygumball.css

mightygumball.html

```
div.innerHTML = sale.name + "에서 볼을 " + sale.sales + "개 판매하였습니다.";
```

http://localhost:8000/js/c6_ok1(ajax)/mightygumball.html

<실습3-1> mightgumball.html

```
 1 <!doctype html>
 2 <html lang="ko">
 3 <head>
 4 <title>검볼 판매</title>
 5 <meta charset="utf-8">
 6 <script src="mightygumball.js"></script>
 7 <link rel="stylesheet" href="mightygumball.css">
 8 </head>
 9 <body>
10
11 <h1>검볼의 판매</h1>
12
13 <div id="sales">
14 </div>
15
16 </body>
17 </html>
18
19
```

A quick example using JSON (1/2)

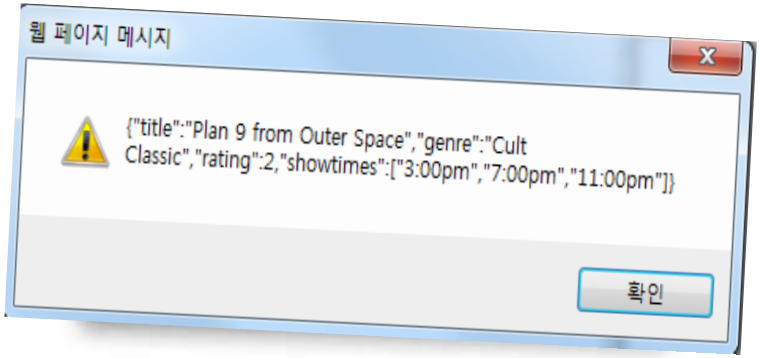
1. 객체를 JSON 문자열 형식으로 변환하는 간단한 예제를 살펴보도록 하죠.

```
var plan9Movie = new Movie("Plan 9 from Outer Space","Cult Classic", 2,
                           ["3:00pm", "7:00pm", "11:00pm"]);
```

여기 문자열, 숫자, 배열을 가진
멋진 movie 객체가 있군요

2. JSON.stringify method를
사용해서 JSON 문자열형식으로 변환

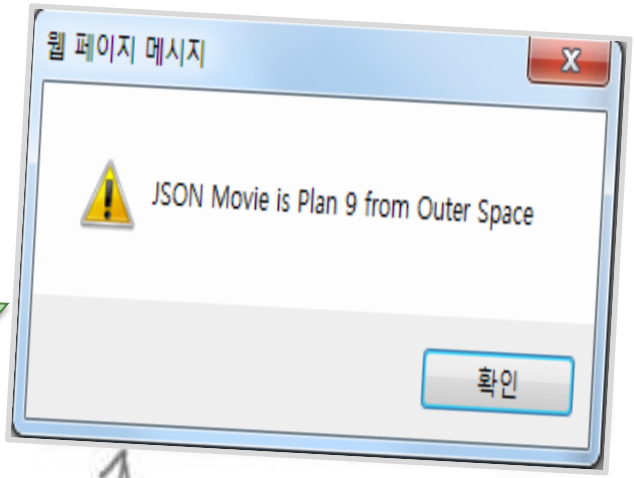
```
var jsonString = JSON.stringify(plan9Movie);
alert(jsonString);
```



Here's the result, a string version of the object displayed in the alert.

A quick example using JSON (2/2)

3. 어떻게 하면 이를 다시 객체로 되돌려서 다른 작업을 할 수 있을까요?
다음과 같이 'JSON.stringify'의 형제격인 JSON.parse 메서드를 사용




```
var jsonMovieObject = JSON.parse(jsonString);
alert("JSON Movie is " + jsonMovieObject.title);
```

Ah, and now we use this as a real object, accessing its properties.

http://www.json.org/

화면을 보기 좋게 만들기 위해 작업… 복습

- 아래의 단계를 따라 하여 변화하는 과정을 봅시다

A. 문자열 형태로 출력

B-1. 먼저 XMLHttpRequest 객체에서 받는 데이터(순수한JSON문자열 데이터)를 가져다가 자바스크립트 객체로 변환합니다.

B-2. 이제 이를 배열로 만들어서 DOM의 새로운 요소로 추가할 수 있습니다. 판매된 항목을 배열에 집어넣으면 됩니다. 여기서는 각 항목에 대해 새로운 를 생성합니다.

<실습4> mightygumball1.html – css가 적용 안 되는 경우(주석)

A. 문자열 형태로 출력

mightygumball1.html

```
<script src="mightygumball1.js"></script>
<link rel="stylesheet" href="mightygumball.css">
```

```
function updateSales(responseText) {
    var salesDiv = document.getElementById("sales");
    salesDiv.innerHTML = responseText;
    /*var sales = JSON.parse(responseText);
    //JSON.parse는 객체로 변환
    for (var i = 0; i < sales.length; i++) {
        var sale = sales[i];
        var div = document.createElement("div");
        div.setAttribute("class", "saleItem");
        div.innerHTML = sale.name + "에서 볼을 " + sale
        salesDiv.appendChild(div);
    }*/
}
```

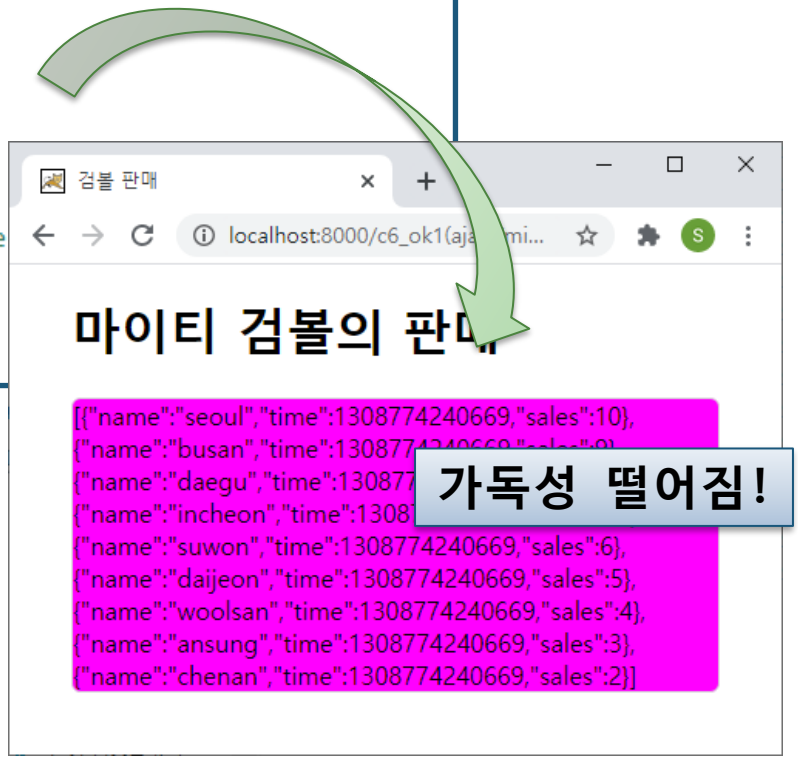
주석처리

mightygumball1.js 로 수정하여 결과확인

mightygumball1.html 만들어 수정

mightygumball1.js 만들어 수정

mightygumball.css 그대로 사용



가독성 떨어짐!

http://localhost:8000/js/c6_ok1(ajax)/mightygumball1.html

<실습5> mightygumball2.js – css를 바꾸어 보자

B-1. 먼저 XMLHttpRequest 객체에서 받는 데이터(순수한JSON문자열 데이터)를 가져다가 자바스크립트 객체로 변환합니다.

```
function updateSales(responseText) {
    var salesDiv = document.getElementById("sales");
    var sales = JSON.parse(responseText);
    //JSON.parse는 객체로 변환
    for (var i = 0; i < sales.length; i++) {
        var sale = sales[i];
        var div = document.createElement("div");
        div.setAttribute("class", "saleItem");
        div.innerHTML = sale.name + "에서 볼을 " + sale.sales + "개 판매하였습니다.";
        salesDiv.appendChild(div);
    }
}
```

주석풀기

mightygumball2.js

mightygumball2.css

mightygumball2.html

가독성 좋아짐!

http://localhost:8000/js/c6_ok1(ajax)/mightygumball2.html

```
<div id="sales">
</div>
```

<실습5-1> mightygumball2.js – css를 바꾸어 보자

B-2. 이제 이를 배열로 만들어서 DOM의 새로운 요소로 추가할 수 있습니다. 판매된 항목을 배열에 집어넣으면 됩니다. 여기서는 각 항목에 대해 새로운 를 생성

주석풀기

```
function updateSales(responseText) {
    var salesDiv = document.getElementById("sales");
    var sales = JSON.parse(responseText);
    //JSON.parse는 객체로 변환
    for (var i = 0; i < sales.length; i++) {
        var sale = sales[i];
        var div = document.createElement("div");
        div.setAttribute("class", "saleItem");
        div.innerHTML = sale.name + "에서 볼을 " + sale.sales + "개 판매하였습니다.";
        salesDiv.appendChild(div);
    }
}
```

mightygumball2.js

mightygumball2.css

mightygumball2.html

가독성 좋아짐!


```
<div id="sales">
</div>
```

http://localhost:8000/js/c6_ok1(ajax)/mightygumball2.html

<실습5-2> mightygumball2.css – css를 바꾸어 보자

```
/* 전체배경  div id="sales"==># 으로 접근 */
 div#sales {
    background-color: #00ff2a; // 연두색 설정
    -webkit-border-radius: 6px;
    border-radius: 6px;
    margin: 10px 0px 0px 0px;
    padding: 0px;
    border: 5px solid #ff3908; //  전체테두리  , 빨간색
}
/* 핑크색위에,글자색#ff0000 , div의 saleItem은 class이므로 div.saleItem 점으로 접근 */
div.saleItem {
    font-family: Verdana, Helvetica, sans-serif;
    color: #d400ff;
    padding: 10px;
}
/* 이후 세개가 함께나와야 차례대로 행의 색이 바뀜 , 두번째행의 색:노란색#fafaaa */
div.saleItem:nth-child(2n) { /* 추가 짝수색, 보라색 */
    background-color: #b9aafa;/*  보라색 설정*/
    margin:10px;/* 추가 테두리색 보이도록 수정 */
}
div.saleItem:nth-child(2n+1) {/* 추가 홀수색, 노란색 */
    background-color: #ecfc17;/*  노란색 설정*/
    margin:10px;/* 추가 테두리색 보이도록 수정 */
}
```

css 수정

가독성 좋아짐!

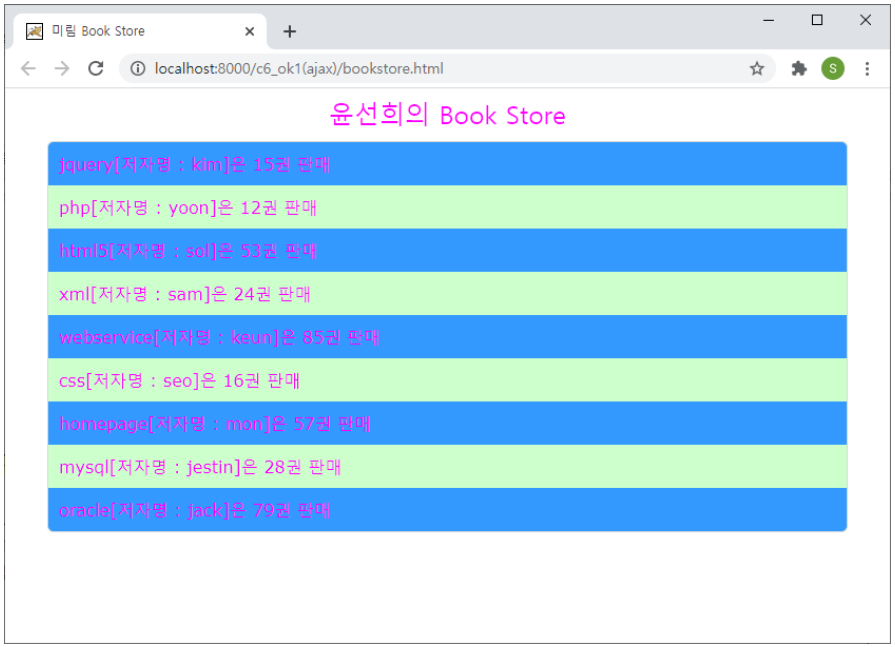
mightygumball2.js

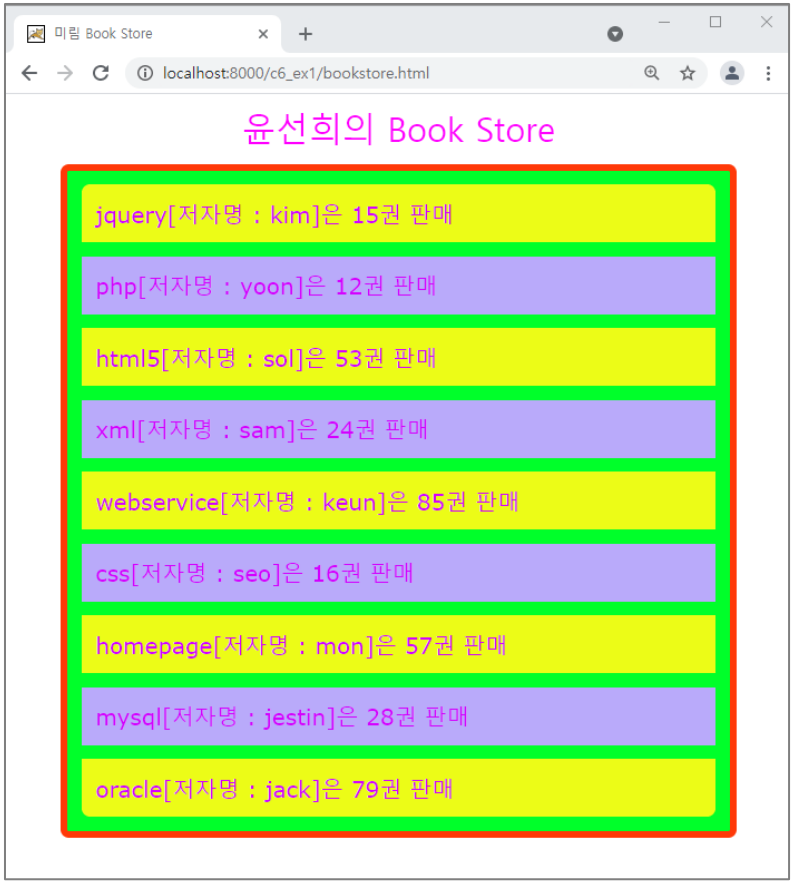
mightygumball2.css

mightygumball2.html

http://localhost:8000/js/c6_ok1(ajax)/mightygumball2.html

<실습6> json과 css파일을 바꾸어 bookstore.html을 만들자



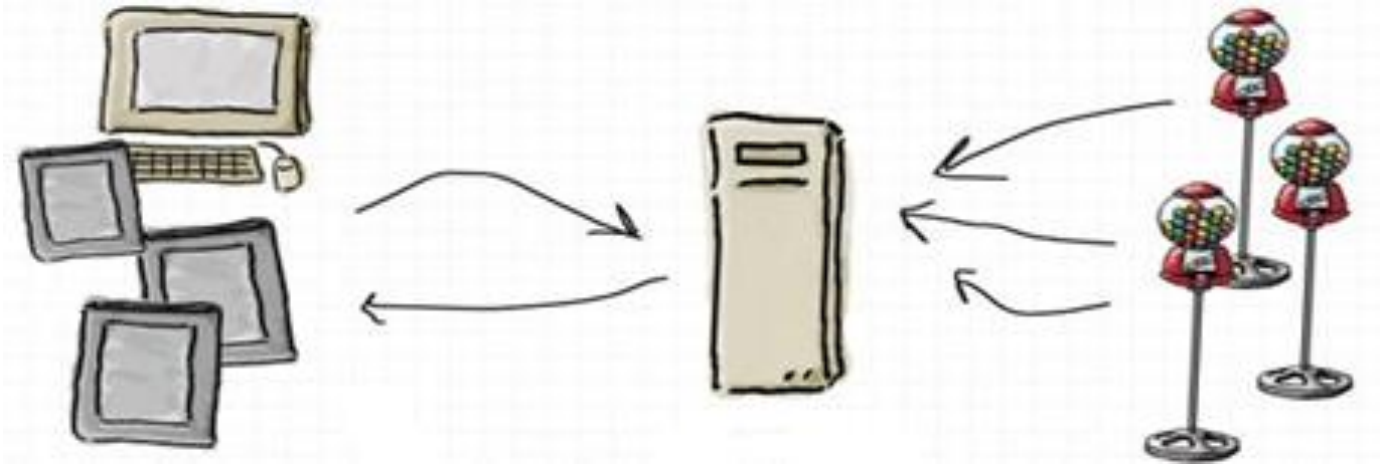


<실습6-1> bookstore.json

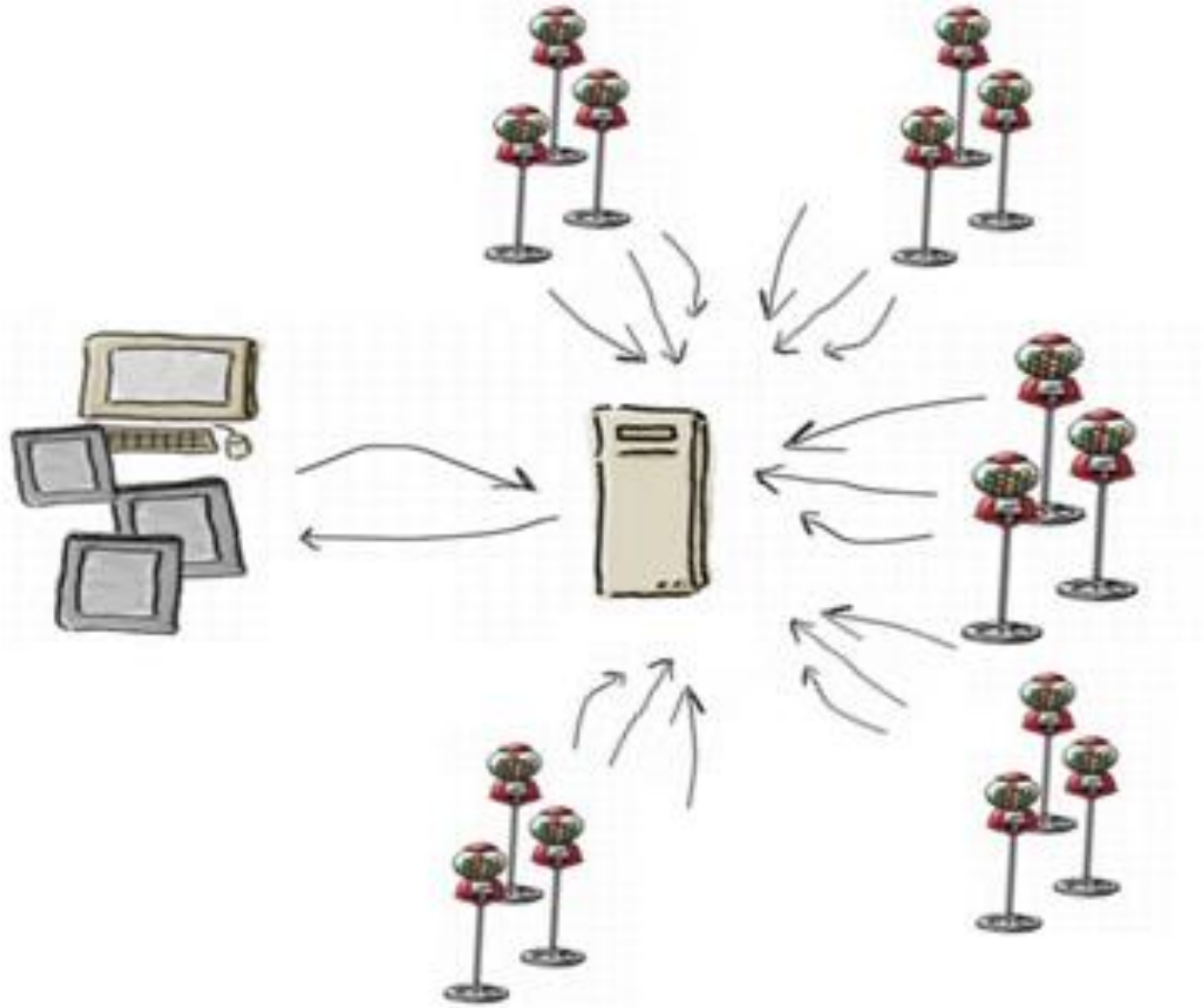
```
[{"name":"jquery","author":"kim","sales":15},
{"name":"php","author":"yoon","sales":12},
{"name":"html5","author":"sol","sales":53},
{"name":"xml","author":"sam","sales":24},
{"name":"webservice","author":"keun","sales":85},
{"name":"css","author":"seo","sales":16},
{"name":"homepage","author":"mon","sales":57},
{"name":"mysql","author":"jestin","sales":28},
{"name":"oracle","author":"jack","sales":79}]
```

Mighty Gumball 은 Web app을 간절히 원합니다

- 마이티 검볼은 검볼 머신을 제작해서 판매하는 혁신적인 회사로, 최근 도움을 요청하여 왔습니다.
- 그들은 최근에 직접 방문하지 않더라도 네트워크를 통해 거의 실시간으로 판매량을 추적할 수 있는 검볼 머신을 개발하였습니다.
- 이에 검볼 판매량을 파악할 수 있는 애플리케이션을 만드는데 우리의 도움을 원하고 있습니다.

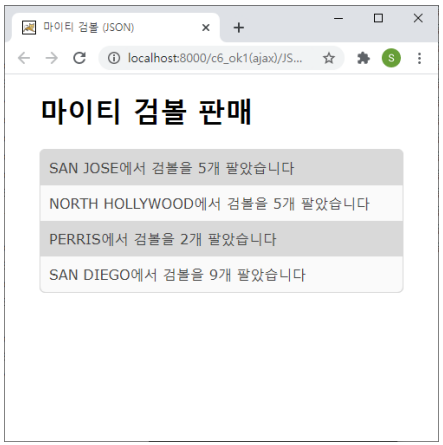


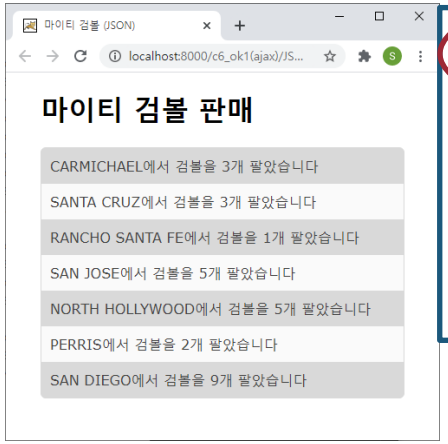
실제 외부서버에서 데이터 수집방법 – Ajax

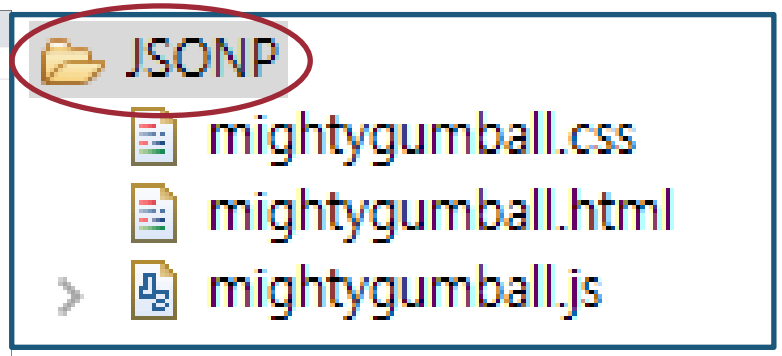


<실습7> 빠른 출발.. 웹을 경유하여 외부에서 가져오기

- 우리에게 필요한 것은 판매보고서를 보여 줄 공간입니다.
- 웹을 경유해서 뭔가를 가져오는 방법을 알아보도록 하죠.







mightygumball.html

```
4  <title>마이티 검볼 (JSON)</title>          <!--추가 1 js와 css연결-->
5  <meta charset="utf-8">
6  <script src="mightygumball.js"></script>
7  <link rel="stylesheet" href="mightygumball.css">
```

```
12 <h1>마이티 검볼 판매</h1>                  <!--추가 2 id="sales"-->
13
14 <div id="sales">
15 </div>
```

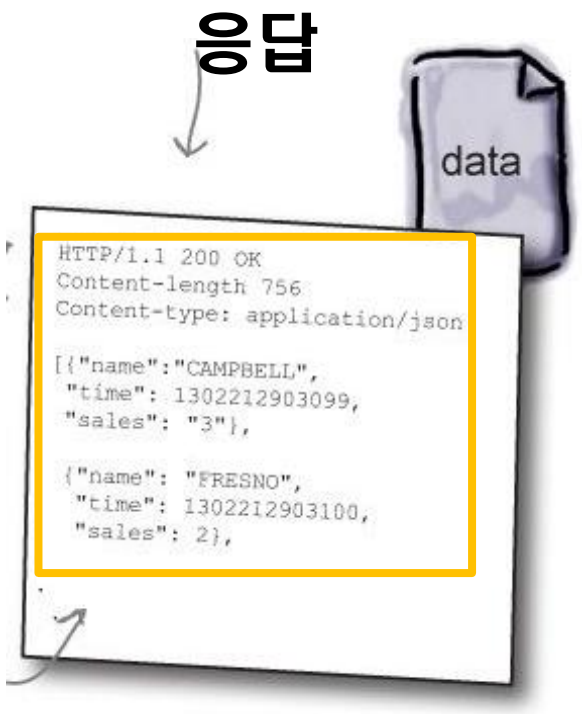
mightgumball.css – 그대로 사용

그럼, web services 에 대한 요청은 어떻게 만들까요?

- 브라우저는 HTML, 마크업, CSS, 자바스크립트가 포함된 마이티 검볼 애플리케이션을 띄웁니다.
- 애플리케이션은 검볼 서버로부터 집계된 판매 데이터를 회수하기 위해 웹 요청을 생성합니다.
- 애플리케에션이 검볼 서버로부터 데이터를 받습니다.

그럼, web services 에 대한 요청은 어떻게 만들까요?

- 요청 : 서버에 있는 우리 애플리케이션에 있는 리소스를 가져오기 위해 http 프로토콜을 사용합니다.
- 응답 : http 프로토콜 헤더가 제일 처음에 오고 이는 http 프로토콜을 사용하며 응답 코드를 제공한다는 의미입니다.



XMLHttpRequest를 사용해서 회수하는
데이터의 패턴을 보통 Ajax'또는 XHR 이라고 합니다.

JavaScript로 request 을 생성 하는 방법 1/6

HTTP로 데이터를 어떻게 회수할까요?

1. Starts with a URL
   데이터를 얻어올 곳을 브라우저에게 알려 줘야 합니다.

```
var url =“http://someserver.com/data.json”;
```

JavaScript로 request 을 생성 하는 방법 2/6

HTTP request 요청을 만들어 봅시다:

2. 아래와 같이 create a request object

```
var request = new XMLHttpRequest();
```

request object

JavaScript로 request 을 생성 하는 방법 3/6

Uses the request object's open method.

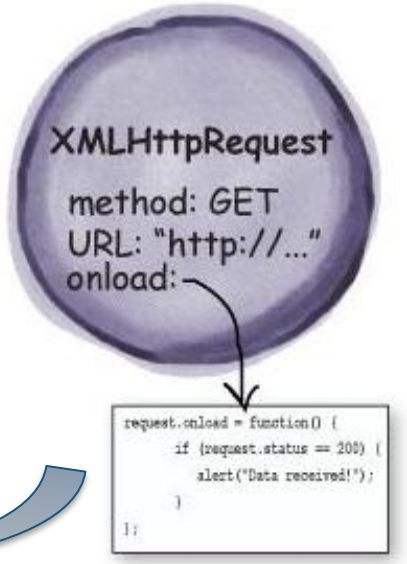
3. 이제 회수하려는 URL과 함께 사용해야 하는 요청의 종류를 요청객체에게 알려줘야 합니다. (표준 http GET방식 사용)

```
request.open("GET", url);
```

JavaScript로 request 을 생성 하는 방법 4/6

4. 마침내 데이터를 회수해 달라고 XMLHttpRequest 객체에 요청을 할 때 이 객체는 즉시 데이터를 받아 옵니다.

```
request.onload=function(){
    if (request.status == 200){
        alert(Data received!");
        updateSales(request.responseText);
    }
};
```




```
updateSales(request.responseText);
```
데이터가 도착할 때 호출되는 핸들러 함수는 만들어야 한다.

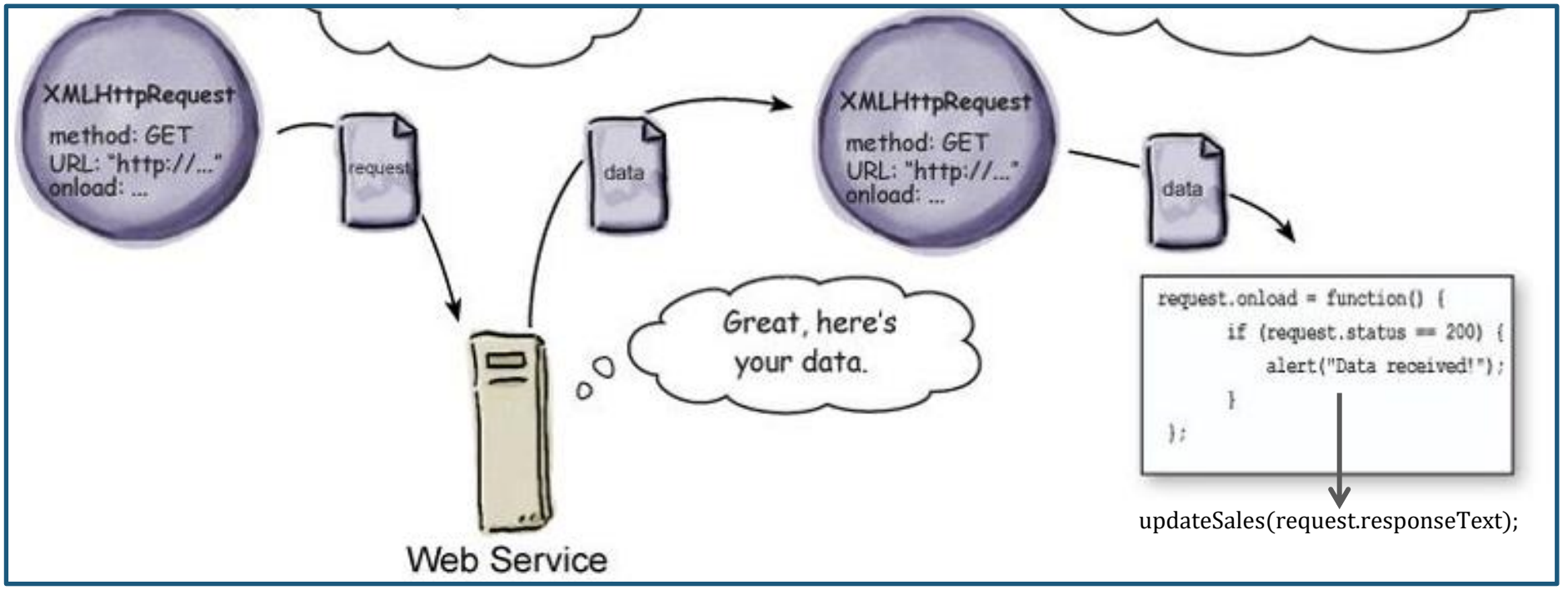
JavaScript로 request 을 생성 하는 방법 5/6

5. 우리는 여전히 요청 객체에게 나가서 데이터를 가져오라고 말해야 하는데 send method를 사용합니다. 데이터를 보내지 않을 것이라면 null을 전달합니다.

```
request.send(null);
```

마지막으로 요청을 전송합니다

JavaScript로 request 을 생성 하는 방법 6/6



XMLHttpRequest object를 생성해서 URL과 HTTP 요청
요청 후 데이터가 도착하기 기다렸죠, 데이터가 도착하면 핸들러가 호출됨

updateSales(request.responseText); 에서 데이터 받기 – 핸들러 함수

HTTP GET 방식으로 받은 데이터는 request 객체의 responseText 속성에 들어 있고 따라서 다음과 같이 코드를 작성할 수 있습니다.

```
request.onload=function(){
    if (request.status == 200){
        updateSales(request.responseText);      // 핸들러 함수
    }
};
```

반환된 코드가 200 이라면 모든 것이 정상이고
Request객체의 responseText부터 응답을 받을 받을 수 있습니다.
데이터가 도착하면 핸들러가 호출됩니다.-updateSales

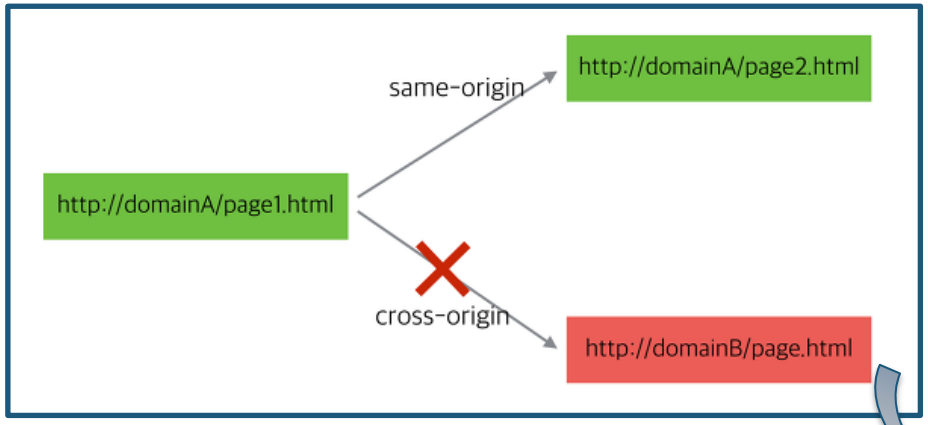
이번에는 운영서버를 이관하기 위해 알아야 할것들

- 마이트검볼에서는 로컬에서 테스트 하길 요구했고, 그대로 수행하였습니다.
- 그리고 실제 서버에서 테스트할 준비가 됐습니다.
- 이번에는 정적인 JSON데이터 파일 대신에 마이트 검볼 서버에서 동적으로 생성된 JSON 데이터를 받아 보도록 합시다.
- XMLHttpRequest가 사용하는 URL을 마이티검볼 서버를 가리키도록 수정해 봅시다.

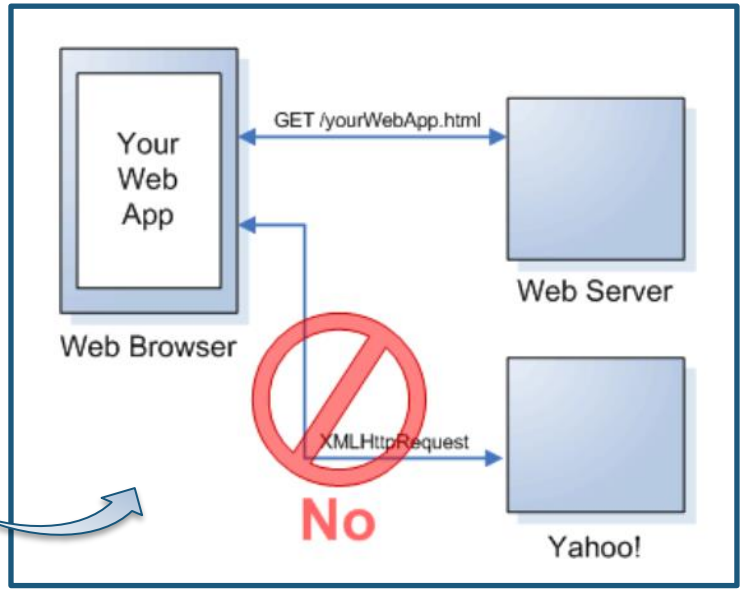
```
window.onload = function() {
    var url = "http://gumball.wickedlysmart.com";
    var request = new XMLHttpRequest();
    request.open("GET", url);
    request.onload = function() {
        if (request.status == 200) {
            updateSales(request.responseText);
        }
    };
    request.send(null);
}
```

브라우저 보안정책 – 크로스도메인 이슈(CORS, Cross Origin Resources Sharing)

- 페이지가 속해있는 도메인과 다른 도메인에 있는 서버에서 데이터를 가져 올 수 없다.



- Same-origin policy : cross-origin 에서는 데이터를 가져올 수 없다.
- 동일 출처 정책(same-origin policy)은 한 출처(origin)에서 로드된 문서나 스크립트가 다른 출처 자원과 상호작용하지 못하도록 제약한다.



XMLHttpRequest가 다른 페이지에서 데이터를 가져오는 것을 막고 있습니다.

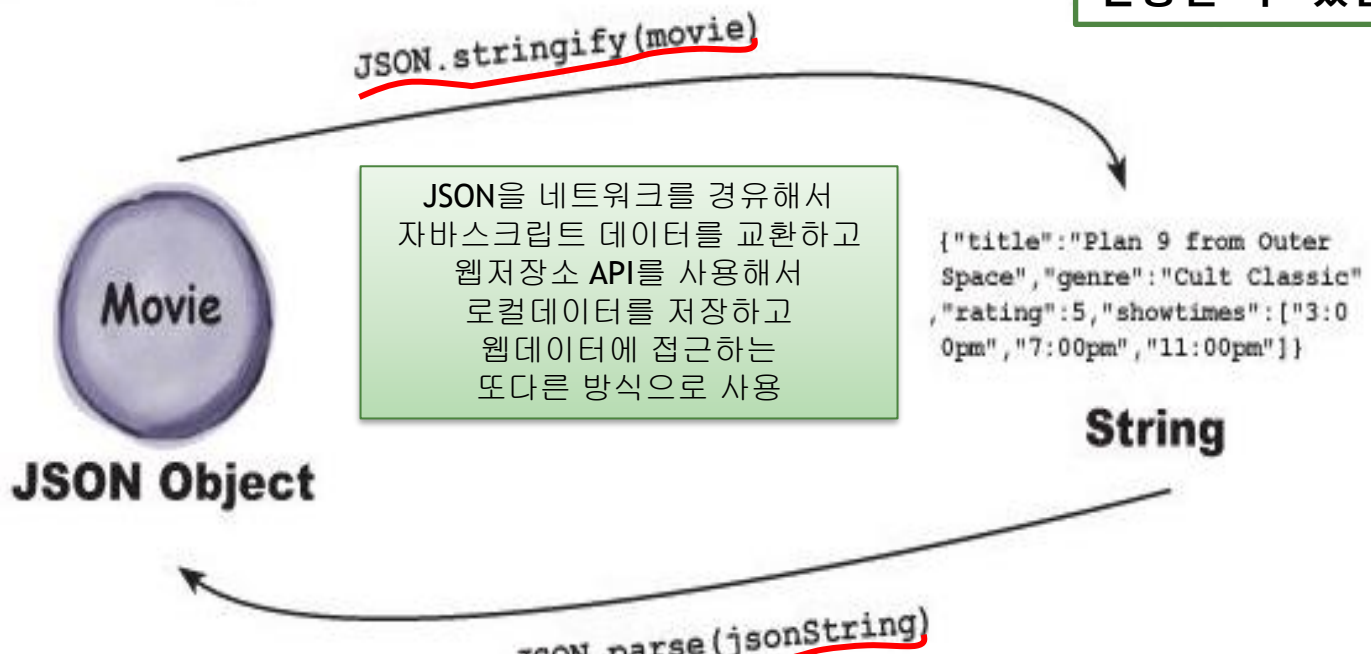
- 보안을 위해 소스 근원지가 같을 때만 허용한다는 Same Origin Policy (SOP) 정책 때문이다.
- 데이터를 호출하는 도메인과 데이터를 반환하는 도메인이 일치해야 한다는 것이다.
- 이러한 문제점을 해결하기 위해서 JSONP 형식으로 데이터를 주고 받아야 한다.

크로스도메인 이슈(CORS, Cross Origin Resources Sharing) https://brocess.tistory.com/168

JSON 메서드들

1. 교환하거나 저장하려는 자바스크립트 객체를 확보한 다음에 JSON.stringify 메서드를 호출해서 이 객체를 인자로 전달합니다.

2. 반환되는 것은 객체를 나타내는 문자열입니다.
이 문자열을 저장해서 함수로 전달하고 네트워크를 통해 다시 전송할 수 있습니다.



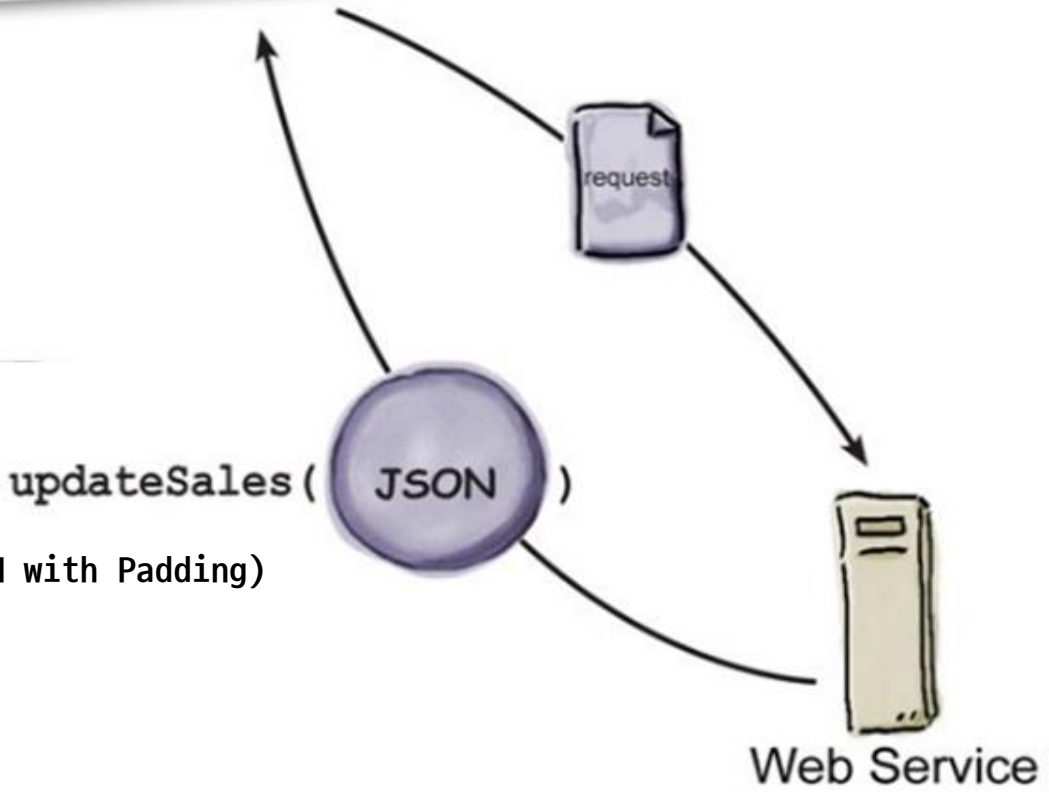
3. 이 문자열을 객체로 변환할 준비가 되면, 이를 JSON.parse 메서드로 전달합니다.

4. 결과로 반환되는 것은 원본 객체의 복사본입니다.

Callback 함수

```
<!doctype html>
<html lang="en">
  ...
<body>
  <h1>Mighty Gumball Sales</h1>
  <div id="sales">
  </div>
  <script src="http://gumball.wickedlysmart.com/"></script>
</body>
</html>
```

- callback 함수 - 자바스크립트의 매개변수에 JSON객체를 넣어준다.
- http://gumball.widkedlysmart.com/?callback=updateSales
- callback이란 URL 매개변수를 추가했는데 자바스크립트가 생성될 때 updateSales함수를 사용하라는 의미
- function updateSales(sales)이 실행



JSONP (JSON with Padding)

JSONP (JSON with Padding) 를 만나봅시다

- 오픈 소스들과 오픈 API들의 서비스가 많은 요즘에 사이트간 데이터교환은 빈번하게 발생한다. 이러한 문제를 해결하기 위해 JSONP 형식으로 데이터를 주고 받는다.
- JSONP-JSON with Padding(속을 꽉 채운 JSON) 속을 꽉 채운 모든 데이터는 요청으로 들어오기 전에 JSON 을 함수로 감싸고 있다
- <script> 태그를 사용해 JSON 객체를 회수 하는 하나의 방법으로 이때 callback 함수 이용
- 웹서비스할 때 JSOP 는 보안상의 커다란 문제 - 악의적인 코드가 함께 들어올 수 있으나 트위터, 구글, 페이스북의 경우는 일단 안심 그러나 항상 주의 필요!

```
function handleRefresh() {
    console.log("here");
    var url = "http://gumball.wickedlysmart.com" +
              "?callback=updateSales"
             +"&lastreporttime=" + lastReportTime;
```

```
function updateSales(sales) {
    var salesDiv = document.getElementById("sales");
```

```
[{"name":"seoul","time":1308774240601,"sales":10},
{"name":"busan","time":1308774240602,"sales":9},
{"name":"daegu","time":1308774240603,"sales":8},
{"name":"incheon","time":1308774240604,"sales":7},
{"name":"suwon","time":1308774240605,"sales":6}
```

프로그램 구성

HTML

```
<script src="mightygumball.js">
<div id="sales">
```

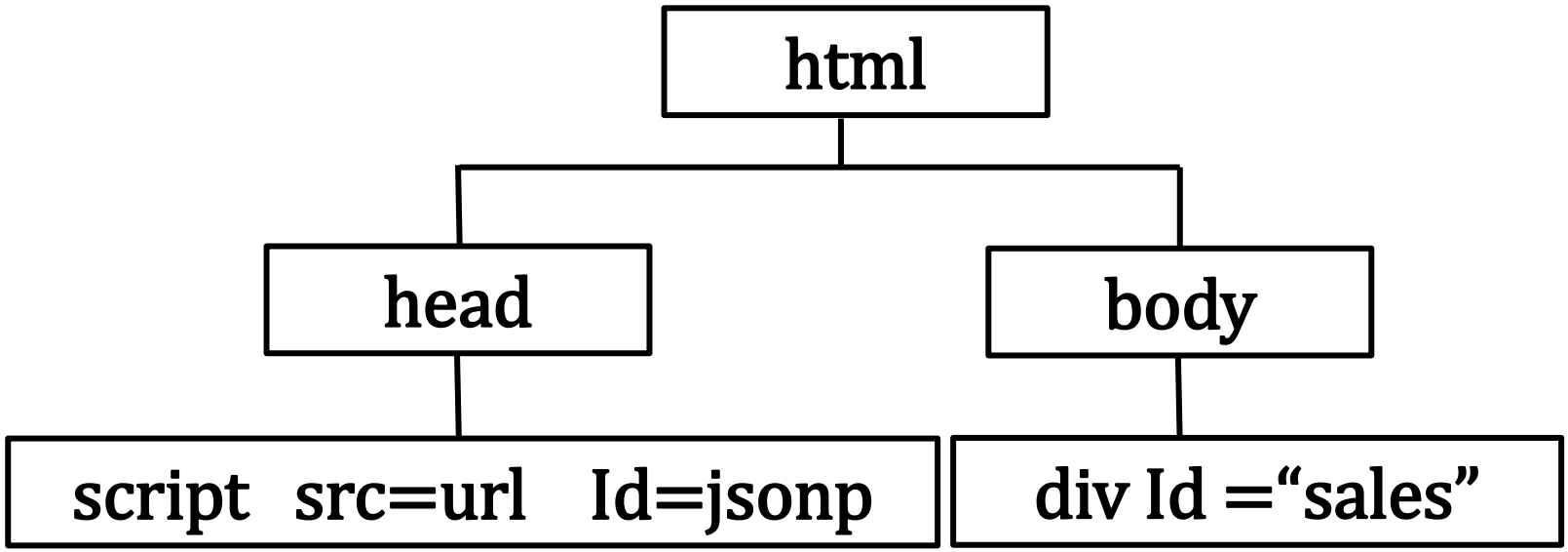
CSS

Body
div#sales
div.saleItem

JS

```
var lastReportTime = 0;
window.onload = init;
function init()
function handleRefresh()
function updateSales(sales)
```

DOM 구조



실행화면 디자인하기(mightygumball.css) 적용

salesDiv

```
var salesDiv = document.getElementById("sales");
        salesDiv  <=  Id=sales
```

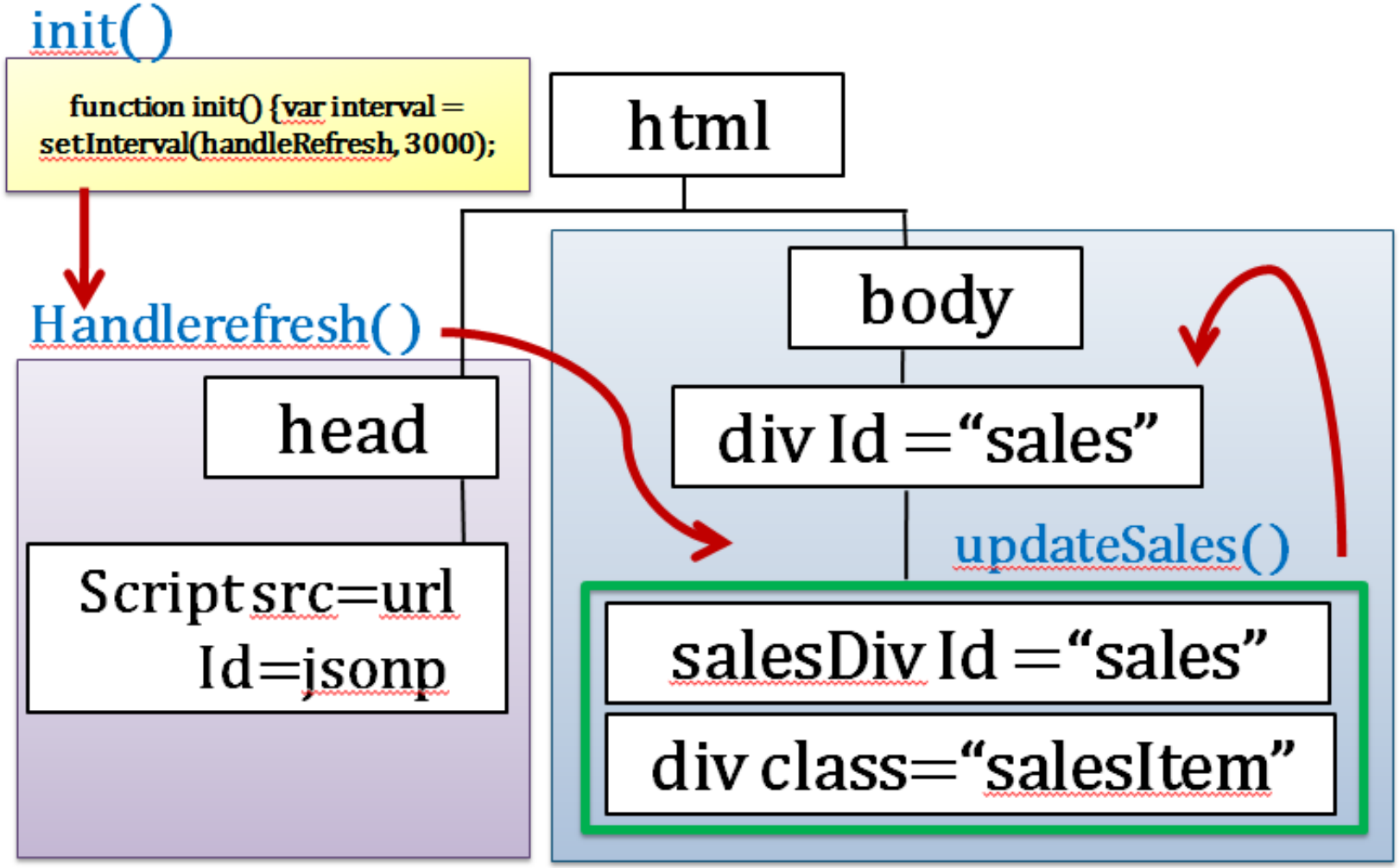
css 의 div#sales 적용

div

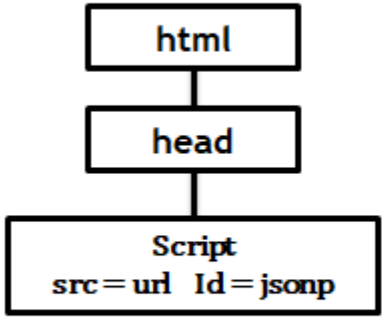
```
for
    var div = document.createElement("div");
    div.setAttribute("class", "saleItem");
    div.innerHTML = sale.name + "에서 볼을 " +  sale.sales + "개 판매하였습니다.";
                salesDiv.appendChild(div);
    }
```

css 의 div.saleItem 적용

함수 담당


init()
function init() {var interval =
setInterval(handleRefresh, 3000);
html
Handlerefresh()
head
Script src=url
Id=jsonp
body
div Id =“sales”
updateSales()
salesDiv Id =“sales”
div class=“salesItem”

callback 처리

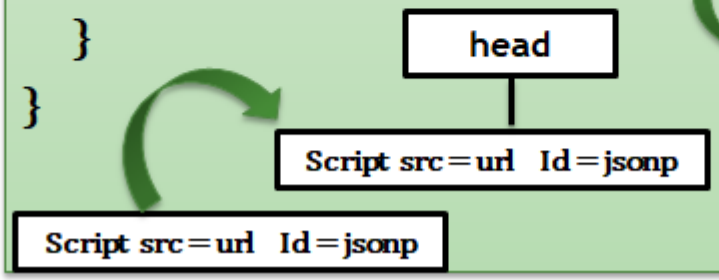


코드상에서 <head> 요소에 대한 참조를 가져와야 한다.

<script>요소가 있는지 체크하여 만약 존재하지 않는다면 새로운 <script>를 head에 추가합니다.

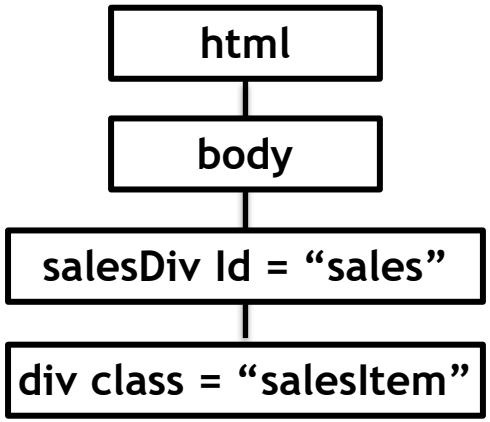
head <script>요소가 이미 존재한다면 이를 대체하기만 하면 됩니다. <head>요소에서 replaceChild 메서드를 사용해서 이전 것을 새로운 script로 대체합니다.

```
function handleRefresh() {
console.log("here");
var url = "http://gumball.wickedlysmart.com" +
"?callback=updateSales" + "&lastreporttime=" + lastReportTime
+ "&random=" + (new Date()).getTime();
var newScriptElement = document.createElement("script");
newScriptElement.setAttribute("src", url);
newScriptElement.setAttribute("id", "jsonp");
var oldScriptElement = document.getElementById("jsonp");
var head = document.getElementsByTagName("head")[0];
  if (oldScriptElement == null) {
   head.appendChild(newScriptElement);//첫번째연결
  }
  else{
  head.replaceChild(newScriptElement, oldScriptElement);
  }
}
```



handleRefresh()

salesDiv 처리




```
function updateSales(sales) {
var salesDiv = document.getElementById("sales");//body연결
 for (var i = 0; i < sales.length; i++) {
    var sale = sales[i];
    var div = document.createElement("div");
    div.setAttribute("class", "saleItem");
    div.innerHTML = sale.name + "에서 검볼을 "
    + sale.sales  +  "개 팔았습니다";
    //salesDiv.appendChild(div);//salesDiv에 div가 연결
      if (salesDiv.childElementCount == 0)
      {  salesDiv.appendChild(div);    }
       else
       {salesDiv.insertBefore(div, salesDiv.firstChild);
       }
 }//for문마지막
  if (sales.length > 0) {
     lastReportTime = sales[sales.length-1].time;
  }
}
```

updateSales(sales)

lastreport time

var url = "http://gumball.wickedlysmart.com" + "?callback=updateSales“
+ "&lastreporttime=" + lastReportTime

[{"name":"seoul","time":1308774240569,"sales":10},
{"name":"busan","time":1308774240579,"sales":9},
{"name":"daegu","time":1308774240589,"sales":8},
{"name":"incheon","time":1308774240599,"sales":7},
{"name":"suwon","time":1308774240600,"sales":6}

[{"name":"seoul","time":1308774240569,"sales":10},
{"name":"busan","time":1308774240579,"sales":9},
{"name":"daegu","time":1308774240589,"sales":8},
{"name":"incheon","time":1308774240599,"sales":7},
{"name":"suwon","time": 1308774240600,"sales":6}

[{"name":"seoul","time":1308774240601,"sales":10},
{"name":"busan","time":1308774240602,"sales":9},
{"name":"daegu","time":1308774240603,"sales":8},
{"name":"incheon","time":1308774240604,"sales":7},
{"name":"suwon","time":1308774240605,"sales":6}

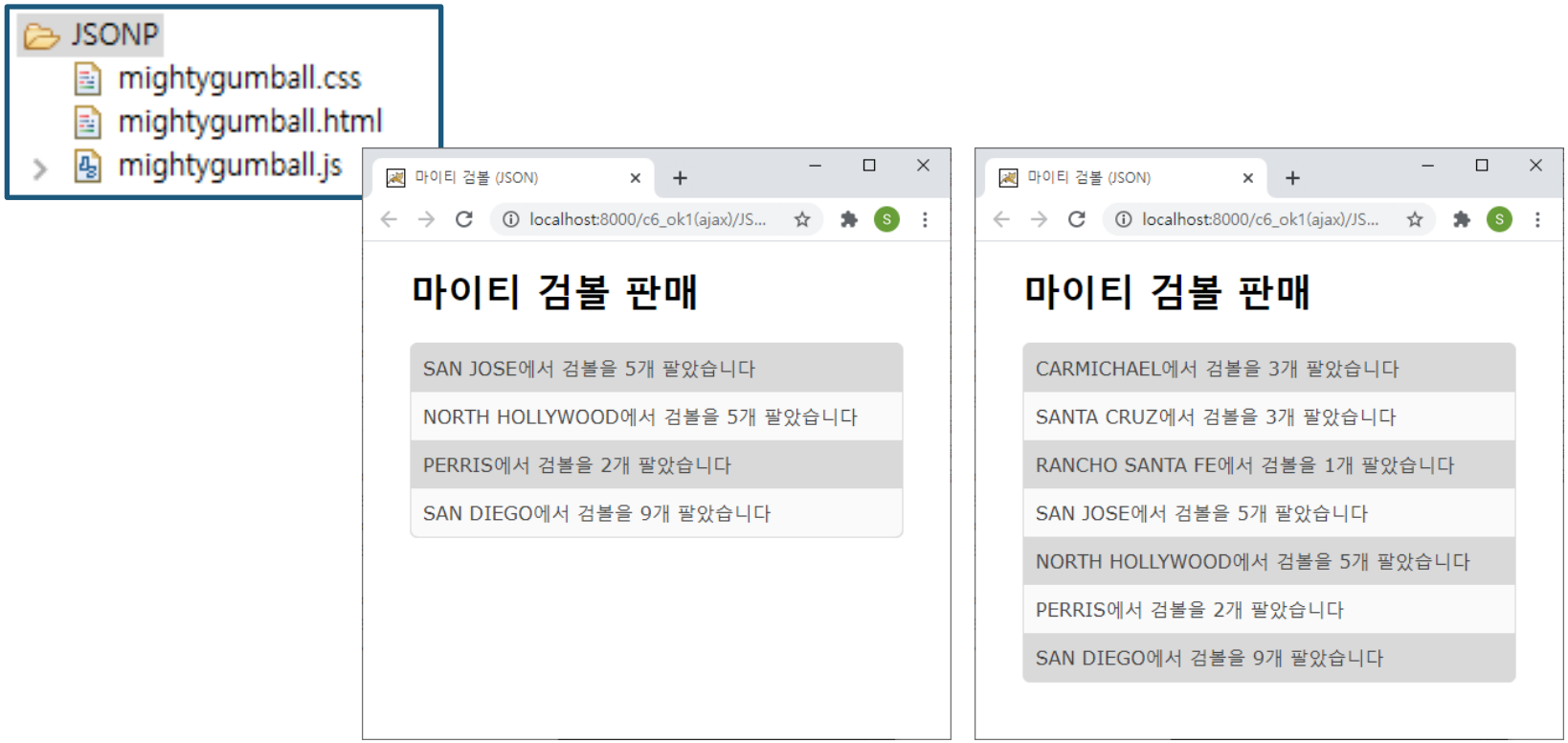
<실습7-1> /JSONP – 웹용 – mightgumball.js – 추가 1

```
1  var lastReportTime = 0;
2
3  window.onload = init;
4
5  function init() {
6      var interval = setInterval(handleRefresh, 3000);
7      handleRefresh();
8  }
9
10 function handleRefresh() {
11     console.log("here");
12     var url = "http://gumball.wickedlysmart.com" +
13                 "?callback=updateSales"
14                 +"&lastreporttime=" + lastReportTime;
15                 +   "&random=" + (new Date()).getTime();
16     var newScriptElement = document.createElement("script");
17     newScriptElement.setAttribute("src", url);
18     newScriptElement.setAttribute("id", "jsonp");
19     var oldScriptElement = document.getElementById("jsonp");
20     var head = document.getElementsByTagName("head")[0];
21     if (oldScriptElement == null) {
22         head.appendChild(newScriptElement);
23     }
24     else {
25         head.replaceChild(newScriptElement, oldScriptElement);
26     }
27 }
```

<실습7-2> /JONP – 웹용 – mightgumball.js – 추가 2

```
28
29 function updateSales(sales) {
30     var salesDiv = document.getElementById("sales");
31     for (var i = 0; i < sales.length; i++) {
32         var sale = sales[i];
33         var div = document.createElement("div");
34         div.setAttribute("class", "saleItem");
35         div.innerHTML = sale.name + "에서 검볼을 " + sale.sales + "개 팔았습니다";
36         //salesDiv.appendChild(div);
37         if (salesDiv.childElementCount == 0) {
38             salesDiv.appendChild(div);
39         }
40         else {
41             salesDiv.insertBefore(div, salesDiv.firstChild);
42         }
43     }
44
45     if (sales.length > 0) {
46         lastReportTime = sales[sales.length-1].time;
47     }
48 }
49
50
```

<실습7-3> /JONP – 웹용 – mightgumball.html



http://localhost:8000/js/c6_ok1(ajax)/JSONP/mightygumball.html

Q & A