

Ajax (Asynchronous javascript and XML)

동의과학대학교 컴퓨터정보과
김진숙

AJAX : 비동기 통신 기술

Asynchronous : 비동기적

Javascript : 자바스크립트

And

XML(eXensible Markup Language) :

- 전송 데이터 양식
- XML, JSON, TEXT, HTML 등도 사용 가능
- 현재는 주로 JSON 사용

참고문서 : <https://developer.mozilla.org/ko/docs/Web/API/XMLHttpRequest>

AJAX

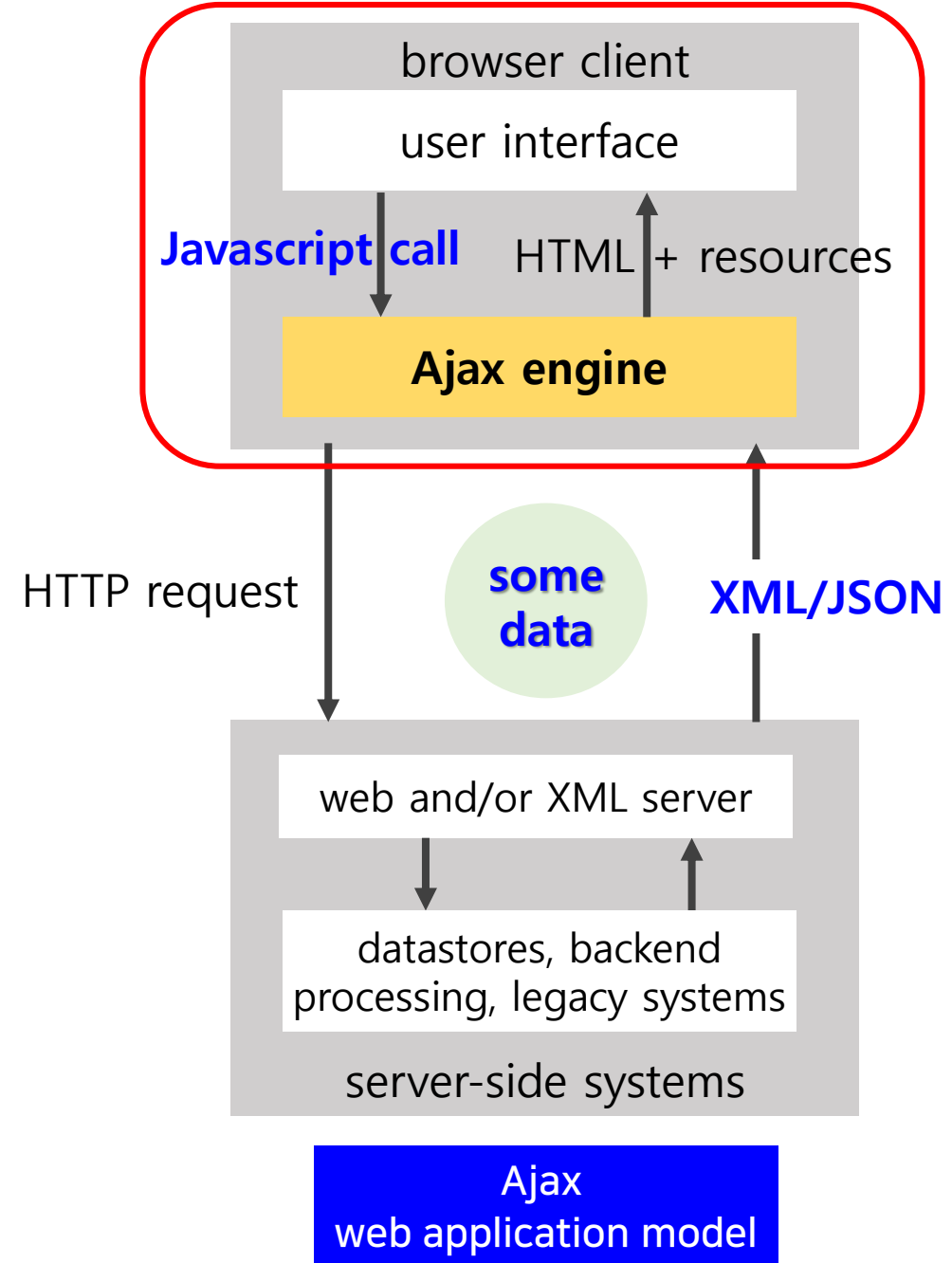
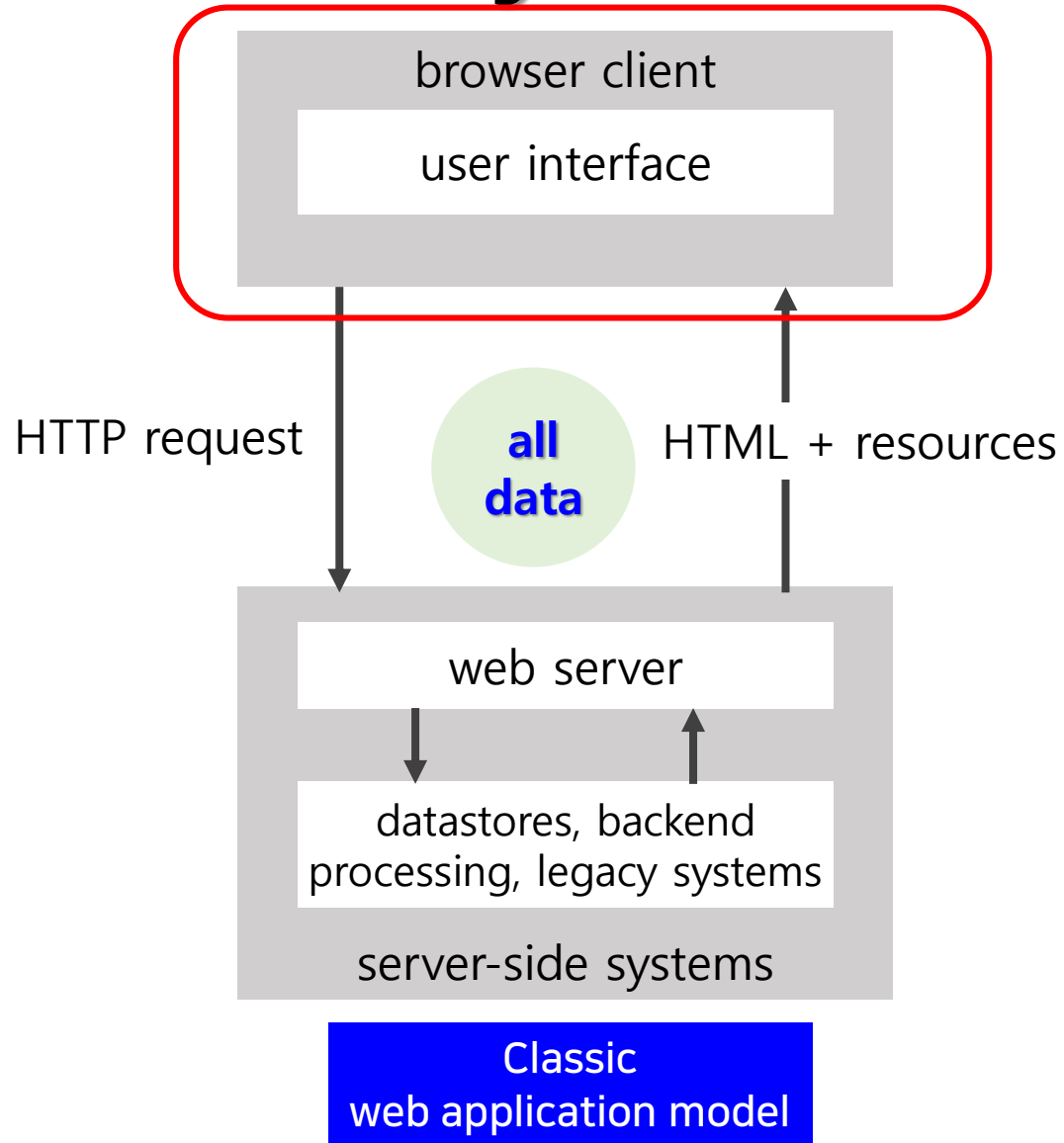
- Jess James Garrett 가 최초로 사용한 용어
 - "Ajax: A New Approach to Web Applications" 에서 최초 사용 – 2005년 2월
 - Ajax는 새로운 기술이 아니라 연계되는 기술들을 묶어서 사용하는 용어
 - <https://immagic.com/eLibrary/ARCHIVES/GENERAL/ADTVPATH/A050218G.pdf>



Ajax의 필요성

- HTTP 프로토콜의 특징
 - 무상태성(**stateless**), 비연결성(**connectionless**)
 - 클라이언트쪽에서 Request를 보내고 Server쪽에서 Response를 받으면 연결이 끊어짐
- 화면의 내용을 갱신하기 위해서는 다시 **request**를 하고 **response**를 하면서 페이지 **전체를 갱신**
 - 페이지의 일부분만 갱신할 경우에도 페이지 전체를 다시 로드 해야 함
 - 이로 인해 자원과 시간낭비를 초래
- **Ajax는 html 페이지 전체가 아닌 일부분만 갱신 가능**
 - XMLHttpRequest객체를 통해 서버에 request하고 json이나 xml형태로 **필요한 데이터만 받아 갱신**함으로 자원과 시간을 아낄 수 있음

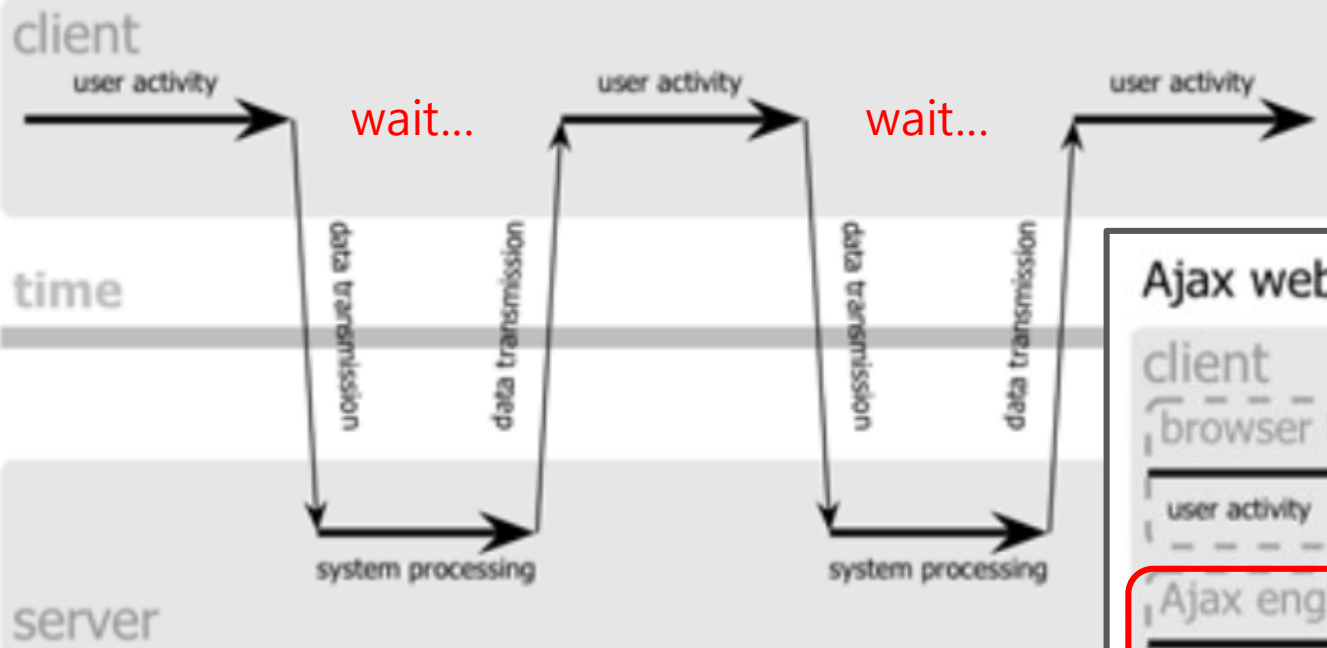
기존 vs. Ajax



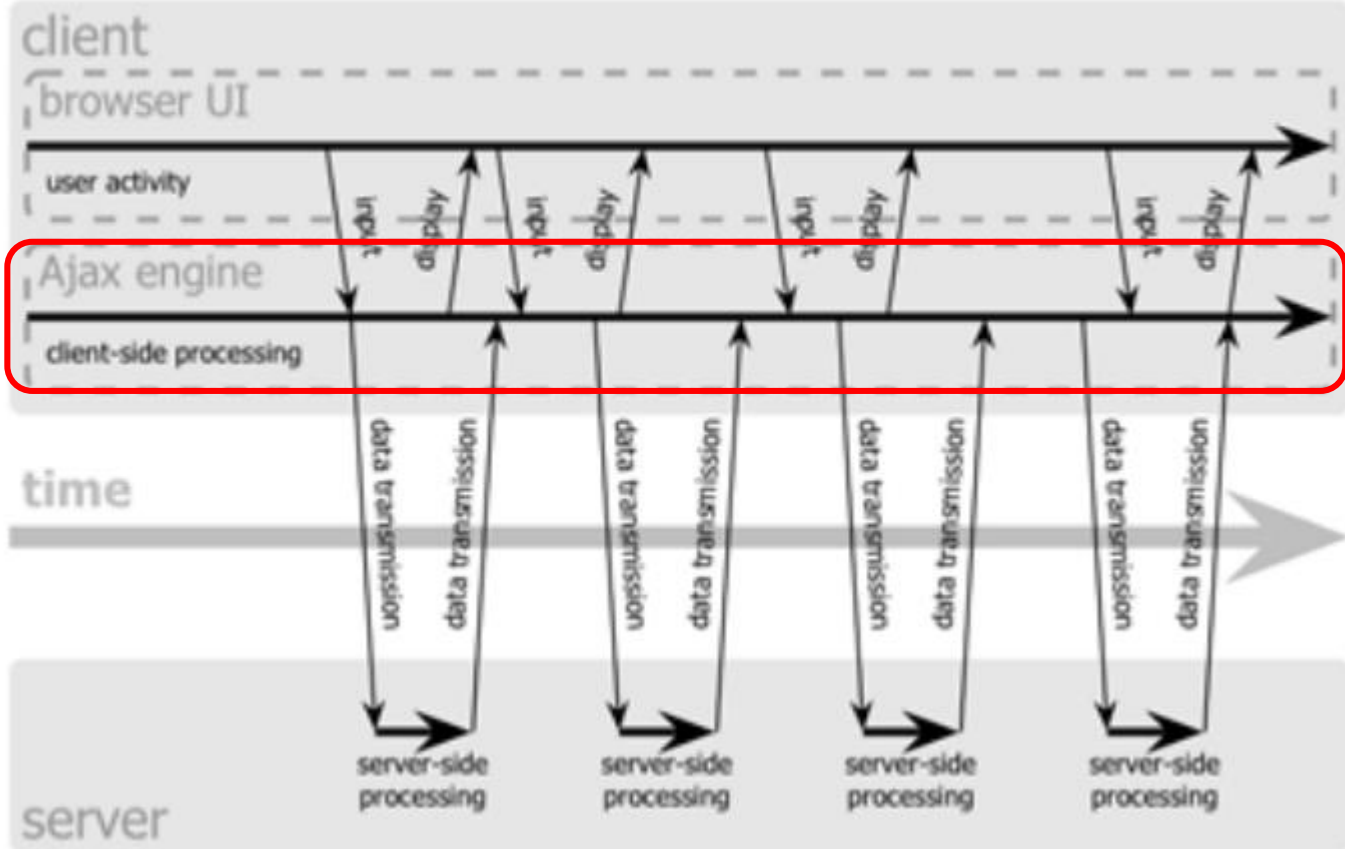
출처 : Ajax: A New Approach to Web Applications

<https://immagic.com/eLibrary/ARCHIVES/GENERAL/ADTVPATH/A050218G.pdf>

classic web application model (synchronous)



Ajax web application model (asynchronous)



Ajax의 장단점

Ajax의 장점

1. 웹 페이지의 **속도 향상**
2. 서버의 처리가 완료 될 때까지 기다리지 않고 처리 가능
3. 서버에서 Data만 전송하기 때문에 **전체적인 코딩 양 감소**
4. 기존 웹에서는 불가능했던 다양한 UI 구현 가능
 - 사진공유 사이트 Flickr의 경우 사진의 제목이나 태그를 페이지 **리로드 없이 수정**

Ajax 의 단점

1. 히스토리 관리 어려움
 - 보안에 좀 더 신경을 써야 함
2. 연속으로 데이터를 요청하면 서버 부하 증가
3. **XMLHttpRequest**를 통해 통신을 하는 경우 사용자에게 진행 정보 제공하지 않음
 - 아직 요청이 완료되지 않았는데 사용자가 페이지를 떠나거나 오작동할 우려가 발생 가능

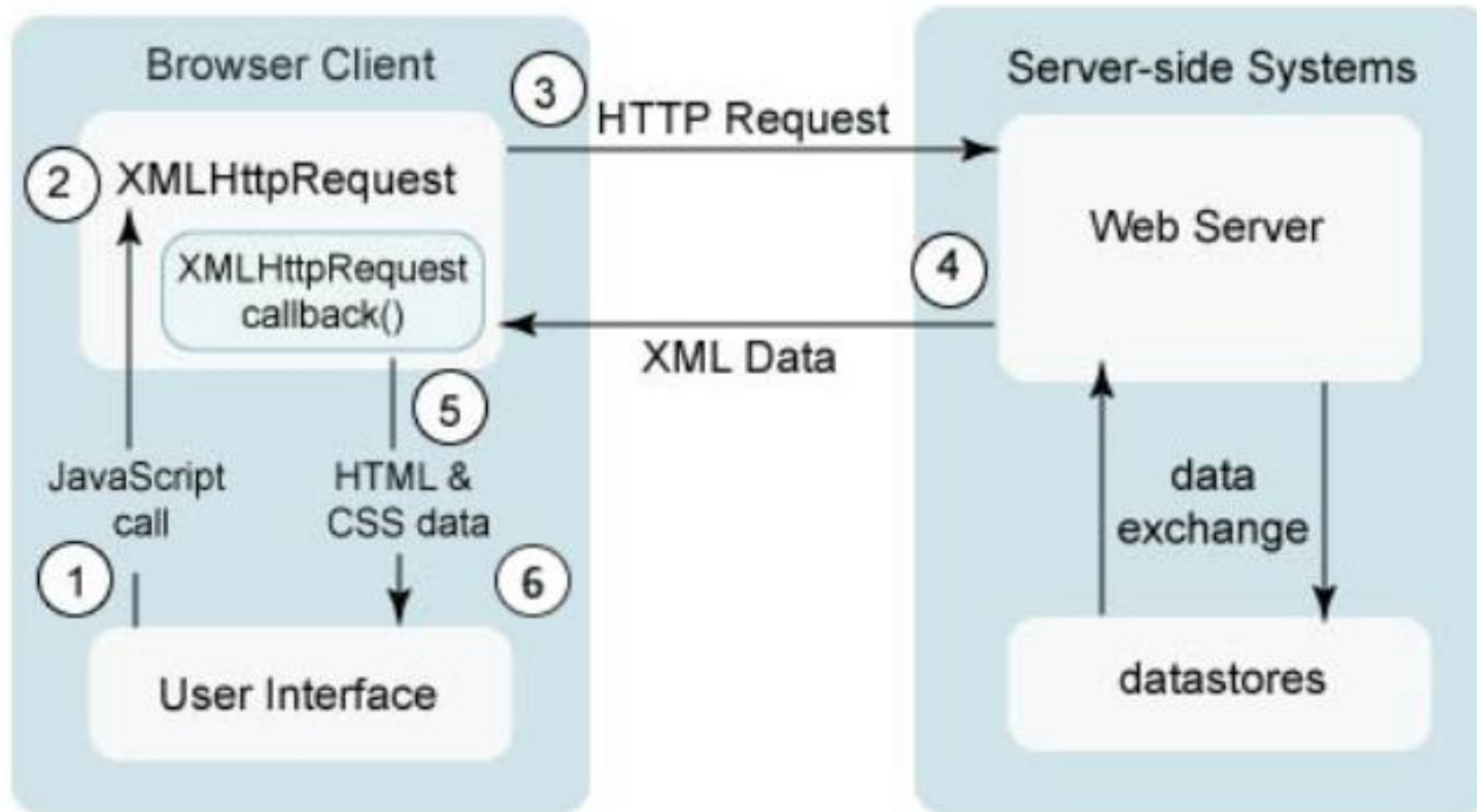
Ajax 용도와 사용 예시

- Ajax 용도
 - 페이지가 로드 된 후 웹 서버에서 데이터 읽기
 - 페이지를 다시로드하지 않고 웹 페이지 업데이트
 - 백그라운드에서 웹 서버로 데이터 보내기
- Ajax 사용 예시
 - 댓글
 - 좋아요
 - 검색어 자동 완성 등

XMLHttpRequest 객체

- Microsoft 사의 Internet Explorer5부터 ActiveX 컴포넌트 형식으로 XMLHttpRequest 제공
- 현재는 모든 브라우저에서 지원

Ajax 작동 방식



출처 : <https://webwox.wordpress.com/2011/08/30/asynchronous-javascript-and-xml-ajax-as-revolution/>

Ajax 작동 방식

1. 웹 페이지 이벤트 발생 (페이지가 로드되고 버튼이 클릭 됨)
2. XMLHttpRequest 객체 생성
3. XMLHttpRequest는 웹 서버에 요청(request)
4. 서버가 웹 페이지로 응답(response)
5. JavaScript로 응답 읽어 냄
6. JavaScript에 의해 페이지 업데이트와 같은 적절한 조치가 수행

XMLHttpRequest 객체의 주요 메소드

메소드	설명
new XMLHttpRequest()	XMLHttpRequest 객체 생성
abort()	이미 요청을 전송한 경우, 그 요청을 중단
getAllResponseHeaders()	헤더 정보를 반환
getResponseHeader()	특정 헤더 정보 반환
open(<i>method, url, async, user, pwd</i>)	요청 정보 설정 <i>method</i> : 요청 타입(GET or POST) <i>url</i> : 파일 위치 <i>async</i> : true (asynchronous, default) or false (synchronous)
send() / send(request's body)	요청이 비동기식(기본값)인 경우 이 메서드는 요청이 전송되자마자 반환되고 결과는 이벤트를 사용하여 전달
	요청을 서버에게 전송
setRequestHeader(header, value)	HTML 양식처럼 데이터를 전송하려면 setRequestHeader()를 사용하여 HTTP 헤더를 추가 setRequestHeader(" Content-type ", "application/x-www-form-urlencoded");

XMLHttpRequest 객체의 주요 속성

속성	설명
onreadystatechange	readyState 속성이 변경될 때 호출되는 함수를 정의
readyState	XMLHttpRequest 의 상태를 값으로 가짐 0: 요청이 초기화되지 않음 1: 서버 연결/loading 2: 요청 받음/loaded 3: 요청 처리/interactive 4: 요청 완료/complete
responseText	요청에 대한 응답을 텍스트로 나타내는 string을 반환 요청이 실패했거나, 아직 전송하지않은 경우에는 null을 반환
responseXML	XML 로 응답 데이터 반환
status	요청의 상태 번호 반환 200: "OK" 403: "Forbidden" 404: "Not Found" For a complete list go to the Http Messages Reference
statusText	상태 텍스트 반환(e.g. "OK" or "Not Found")

HTTP : Content-Type

- Request 구조

Request line
Http header
<crLf>
Entity body

- Entity body에 들어가는 데이터 타입을 http heade에서 명시하는 속성이 Content-Type
 - text Type : text/css, text/javascript, text/html, text/plain 등
 - file Type : multipart/form-data
 - application Type
 - application/json : {key: value}의 형태로 전송
 - application/x-www-urlencoded : key=value&key=value의 형태로 전송(URL인코딩 방식)

요청(Request) - GET

- GET 요청

```
xhttp.open("GET", "demo_get.asp", true);  
xhttp.send();
```

```
xhttp.open("GET", "demo_get.asp?t=" + Math.random(), true);  
xhttp.send();
```

```
xhttp.open("GET", "demo_get2.asp?fname=Henry&lname=Ford", true);  
xhttp.send();
```

Example: GET

JS

```
const xhr = new XMLHttpRequest();  
xhr.open("GET", "/server", true);  
  
xhr.onload = () => {  
    // Request finished. Do processing here.  
};  
  
xhr.send(null);  
// xhr.send('string');  
// xhr.send(new Blob());  
// xhr.send(new Int8Array());  
// xhr.send(document);
```

요청(Request) - POST

- POST 요청
 - 서버에 대량의 데이터 전송 (POST에는 크기 제한이 없음).
 - 알 수 없는 문자를 포함 할 수 있는 사용자 입력을 보내는 POST는 GET보다 강력하고 안전

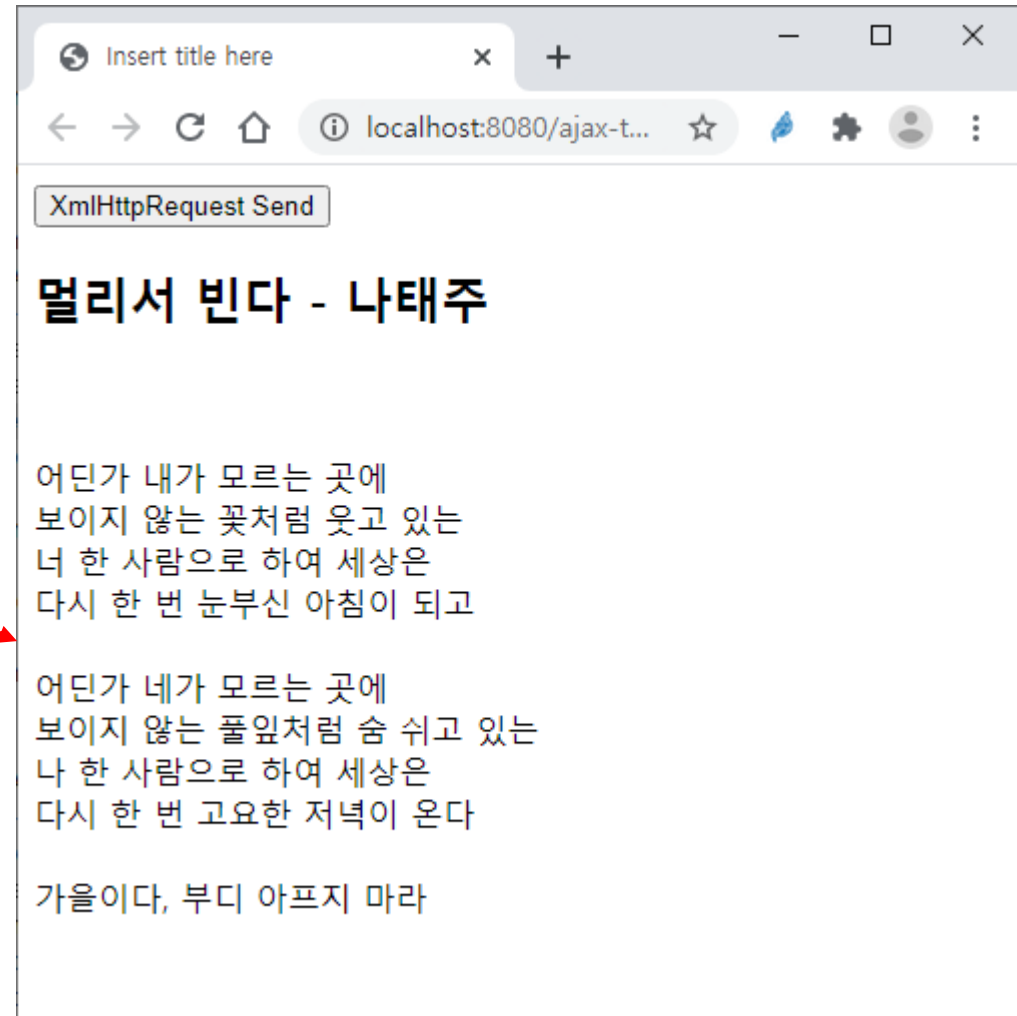
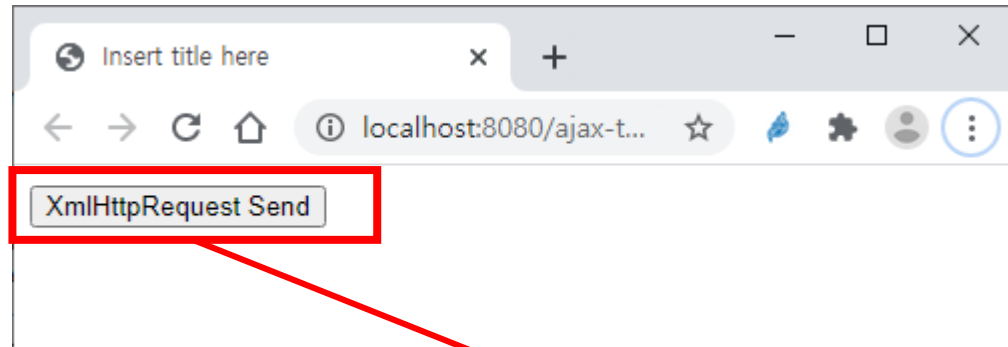
```
xhttp.open("POST", "demo_post2.asp", true);  
xhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");  
xhttp.send("fname=Henry&lname=Ford");
```

Example: POST

JS

```
const xhr = new XMLHttpRequest();  
xhr.open("POST", "/server", true);  
  
// Send the proper header information along with the request  
xhr.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");  
  
xhr.onreadystatechange = () => {  
  // Call a function when the state changes.  
  if (xhr.readyState === XMLHttpRequest.DONE && xhr.status === 200) {  
    // Request finished. Do processing here.  
  }  
};  
xhr.send("foo=bar&lorem=ipsum");  
// xhr.send(new Int8Array());  
// xhr.send(document);
```


실습1 - 텍스트 문서 요청



실습1 - 텍스트 문서 요청

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>ajax 실습 1</title>
6 <script>
7   function sendData(){
8     var xhr = new XMLHttpRequest(); //XMLHttpRequest 객체 생성
9
10    //onreadystatechange 속성은 readyState 속성이 변경 상태를 감지하여 호출될 함수를 정의
11    //readyState 상태값은
12    //0: 요청이 초기화되지 않음
13    //1: 서버 연결/loading)
14    //2: 요청 받음/loaded)
15    //3: 요청 처리/interactive)
16    //4: 요청 완료와 응답 준비/complete)
17    xhr.onreadystatechange = function(){
18
19      if(this.readyState == 4){ //4번의 상태일 때 처리하는 경우
20        //응답 데이터 responseText를 HTML 문서내 myDiv에 표시
21        document.getElementById('myDiv').innerHTML = this.responseText;
22      }
23    };
24    //요청정보 설정 open(method, url, sync, [user, pwd]);
25    xhr.open("POST", "myText.txt", true);
26    xhr.send();//요청을 서버에 전송
27  }
28 </script>
29 </head>
30 <body>
31
32   <input type="button" name="myname" onclick="sendData();" value="XMLHttpRequest Send">
33
34   <div id="myDiv">
35   </div>
36 </body>
37 </html>
```

참고 : JSON(JavaScript Object Notation)

- <http://www.json.org>
- 속성:값의 쌍으로 이루어진 경량(lightweight)의 **데이터 교환 형식**
- 비동기 브라우저/서버 통신 (AJAX)을 위해, 넓게는 XML(AJAX가 사용)을 대체하는 주요 데이터 포맷
- 특정 언어에 종속적이지 않음
- 대부분의 언어에서 JSON 포맷의 데이터를 처리할 수 있는 라이브러리 제공



javascript 언어 개발에 참여했던 미국의 컴퓨터 프로그래머인 **더글라스 크록포드**가 처음으로 JSON 포맷을 정의하고 보급

```
[  
  { "title" : "XML Bible", "author" : "Gwyneth Paltrow", "price" : 40000 },  
  { "title" : "XML 클래스", "author" : "임순범", "price" : 19000 },  
  { "title" : "XML By Example", "author" : "홍길동", "price" : 25000 }  
]
```

참고 : JSON(JavaScript Object Notation)


- Javascript의 JSON 지원 API

- JSON.parse() : 파라미터로 전달된 문자열을 javascript 객체로 변환
- JSON.stringify() : 파라미터로 전달된 javascript 객체를 문자열로 변환

실습2 – JSP로 DB 데이터 목록 출력

• 출력 화면

XmlHttpRequest Send		
id		



XmlHttpRequest Send		
id	name	pwd
youngcheol	윤영철	3030
jimin	백지민	2929
sohee	성주희	99999
minwoo	김민우	1111
gildong	성주희	1111
chunhyang	성주희	2828
hongjin	박홍진	2727
minkyu	박민규	2626
jisung	노지성	2525
hyomin	김효민	2424
riwon	김리원	2323
kiwan	김기환	2222
yuna	한유나	2121

실습2 – JSP로 DB 데이터 목록 출력

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1">
6   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">
7   <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
8   <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script>
9   <title>ajax 실습 2</title>
10
11 </head>
12 <body>
13   <div class="container">
14     <form name="myForm" method="post">
15       <input type="button" name="myname" onclick="sendData()" value="XmlHttpRequest Send">
16     </form>
17     <br><br>
18     <table class="table" style="text-align:center; border:1px solid #dddddd">
19       <tr>
20         <th style="background-color:#fafafa; text-align:center">id</th>
21         <th style="background-color:#fafafa; text-align:center">name</th>
22         <th style="background-color:#fafafa; text-align:center">pwd</th>
23       </tr>
24       <tbody id="ajaxTable">
25       </tbody>
26     </table>
27   </div>
28 </body>
29 </html>
```

실습2 – JSP로 DB 데이터 목록 출력

- javascript

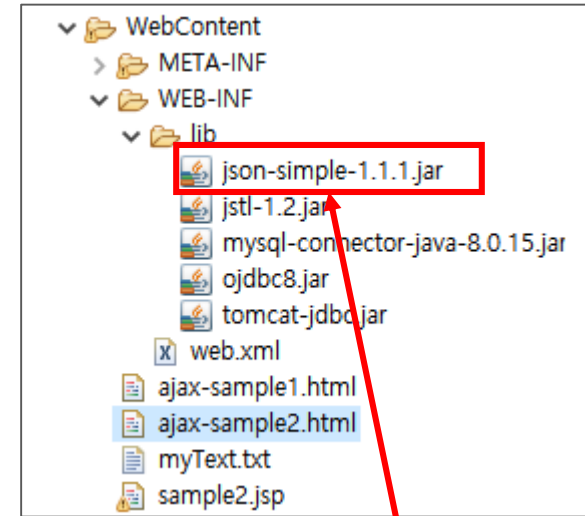
```
11 <script type="text/javascript">
12 var xhr = new XMLHttpRequest();
13
14 function sendData(){
15     var table = document.getElementById("ajaxTable");
16     table.innerHTML = ""; //기존 데이터를 지우기
17
18     xhr.onreadystatechange = function(){
19         //onreadystatechange : readyState 속성이 변경 상태를 감지함
20         //readyState 상태값은
21         //0: 요청이 초기화되지 않음      1: 서버 연결/loading) 2: 요청
22         //3: 요청 처리(interactive) 4: 요청 완료와 응답 준비(complete)
23
24         //status : 요청의 상태 번호 반환
25         //200: "OK" 403: "Forbidden" 404: "Not Found"
26         if(this.readyState == 4 && this.status == 200){
27             //서버에게 반환한 결과 값(json 포맷)을 받아 json 변수에 저장
28             var json = this.responseText;
29
30             // JSON 형식의 문자열을 자바스크립트 객체로 변환함.
31             var list = JSON.parse(json);
32
33             for(var i=0 in list){ //데이터 넣을 테이블 을 만들며 데이터 출력
34                 var row = table.insertRow(0);
35                 var cell1 = row.insertCell(0);
36                 var cell2 = row.insertCell(1);
37                 var cell3 = row.insertCell(2);
38
39                 cell1.innerHTML = list[i].id;
40                 cell2.innerHTML = list[i].name;
41                 cell3.innerHTML = list[i].pwd;
42             }
43         }
44     };
45
46     //요청정보 설정 open(method, url, sync, [user, pwd]);
47     xhr.open("POST", "list.do", true);
48     xhr.send();
49 }
50 </script>
```

실습2 – DB 데이터 목록 출력

• 사전 준비

- DB에 테이블 생성
- 관련 라이브러리 설치
- DBCP 관련 설정(META-INF/context.xml)

```
<!-- DBCP 설정 -->
<Resource name = "jdbc/jskim"
auth = "Container"
type="javax.sql.DataSource"
driverClassName = "org.mariadb.jdbc.Driver"
username="jinsook"
password="1111"
url="jdbc:mariadb://localhost:3306/jinsookdb"
maxWait = "5000"
/>
```



- <http://mvnrepository.com> 에서 json simple로 검색하여 라이브러리 다운로드
- WEB-INF/lib에 추가

실습2 – DB 데이터 목록 출력(Servlet)

```
1 package ditcs;  
2  
3 import java.io.IOException;  
4 import java.sql.Connection;  
5 import java.sql.PreparedStatement;  
6 import java.sql.ResultSet;  
7 import java.sql.SQLException;  
8  
9 import javax.naming.Context;  
10 import javax.naming.InitialContext;  
11 import javax.servlet.ServletException;  
12 import javax.servlet.annotation.WebServlet;  
13 import javax.servlet.http.HttpServlet;  
14 import javax.servlet.http.HttpServletRequest;  
15 import javax.servlet.http.HttpServletResponse;  
16 import javax.sql.DataSource;  
17  
18 import org.json.simple.JSONArray;  
19 import org.json.simple.JSONObject;
```

JSON 라이브러리 import

```
21 @WebServlet("/list")  
22 public class LController extends HttpServlet{  
23     private static final long serialVersionUID = 1L;  
24  
25     protected void doHandle(HttpServletRequest request, HttpServletResponse  
26         Context initCtx=null;  
27         Context envCtx=null;  
28         DataSource ds=null;  
29         Connection con=null;  
30         PreparedStatement pstmt=null;  
31         ResultSet rs=null;  
32         JSONArray list=null;
```

객체 초기화

실습2 – DB 데이터 목록 출력

```
34     try {
35         request.setCharacterEncoding("utf-8");
36         //DB 연동
37         initCtx = new InitialContext();
38         envCtx = (Context) initCtx.lookup("java:comp/env");
39         ds = (DataSource) envCtx.lookup("jdbc/JSP");
40         //커넥션 얻고 SQL 실행
41         con = ds.getConnection();
42         String sql = "select * from login";
43         pstmt = con.prepareStatement(sql);
44         rs = pstmt.executeQuery();
45         //한글 처리
46         response.setContentType("text/html; charset=utf-8");
47
48         //JSON 배열 객체 생성
49         list = new JSONArray();
50
51         while (rs.next()){
52             JSONObject json = new JSONObject();
53             json.put("id", rs.getString("id"));
54             json.put("name", rs.getString("name"));
55             json.put("pwd", rs.getString("pwd"));
56             list.add(json); //JSON 객체로 배열을 만듦
57         }
58         //JSON 배열 객체 클라이언트에 반환
59         response.getWriter().print(list);
60     } catch (Exception e) {
```

이 전 예제들의 ArrayList와 dto 대신에 JSON 배열과 객체 사용

실습2 – DB 데이터 목록 출력

```
58         //JSON 배열 객체 클라이언트에 반환
59         response.getWriter().print(list);
60     } catch (Exception e) {
61         e.printStackTrace();
62     } finally {
63         if(rs!=null)
64             try {rs.close();} catch (SQLException e) {e.printStackTrace();}
65         if(pstmt!=null)
66             try {pstmt.close();} catch (SQLException e) {e.printStackTrace();}
67         if(con!=null)
68             try {con.close();} catch (SQLException e) {e.printStackTrace();}
69     }
70 }
71
72 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws Serv
73     doHandle(request, response);
74 }
75
76 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws Serv
77     doHandle(request, response);
78 }
79 }
```

Jquery로 Ajax 구현

- jquery로 ajax를 사용 장점
 - 크로스 브라우징 문제를 jquery가 알아서 해결(동일한 코드 사용)
 - 코드량이 적고 직관적인 코드 작성 가능
- Ajax 관련 자료 사이트
 - <https://api.jquery.com/category/ajax/>

크로스 브라우징(Cross Browsing) : 웹페이지의 상호 호환성

- 표준 웹기술을 채용하여 다른 기종이나 플랫폼에 따라 달리 구현되는 기술을 비슷하게 만들고 어느 한쪽에 최적화되어 치우치지 않도록 공통요소를 사용하여 웹페이지를 제작하는 기법

Jquery로 Ajax 구현

- jQuery.ajax([settings])

settings	Description
url	요청이 전송될 url
method	http 요청 방식 (default: 'GET')
type	method의 alias (default: 'GET')
data	서버로 전달될 데이터
datatype	서버로부터 반환될 데이터의 타입 default: Intelligent Guess (xml, json, jsonp, script, html)
async	요청 시 동기화 여부. 기본은 비동기(asynchronous) 요청 (default: true)
timeout	요청 제한 시간. 제한 시간 안에 요청이 완료되지 않으면 요청을 취소하거나 error 콜백을 호출.
jsonpCallback	JSONP 요청을 위한 콜백 함수 이름
success	요청 성공 이벤트 핸들러
error	요청 실패 이벤트 핸들러
complete	요청 완료 이벤트 핸들러

- **jQuery.ajax([settings])**

- **data**

- 서버로 데이터를 전송할 때 이 옵션을 사용한다.

- **datatype**

- 서버측에서 전송한 데이터를 어떤 형식의 데이터로 해석할 것인가를 지정한다. 값으로 올 수 있는 것은 xml, json, script, html이다. 형식을 지정하지 않으면 jQuery가 알아서 판단한다.

- **success**

- 성공했을 때 호출할 콜백을 지정한다.

- Function(PlainObject data, String textStatus, jqXHR jqXHR)

- **type**

- 데이터를 전송하는 방법을 지정한다. get, post를 사용할 수 있다.

실습3 - JQuery로 Ajax 구현

```
9  <title>ajax 실습 3 - jquery</title>
10
11 <script type="text/javascript">
12     function listup(){
13         $.ajax({
14             url: './list',
15             type: 'post',
16             async: true,
17             dataType: 'json',
18             success: function(data){ //json Parse를 이용해 자바 객체로 변환된 객체가 data가 됨
19                 var str = '';
20
21                 for(var i in data){
22                     str += '<tr><td>' + data[i].id + '</td>';
23                     str += '<td>' + data[i].name + '</td>';
24                     str += '<td>' + data[i].pwd + '</td></tr>';
25                 }
26                 $('#ajaxTable').html(str); //str의 내용을 id="ajaxTable"에 출력
27             }
28         });
29     }
30 </script>
```