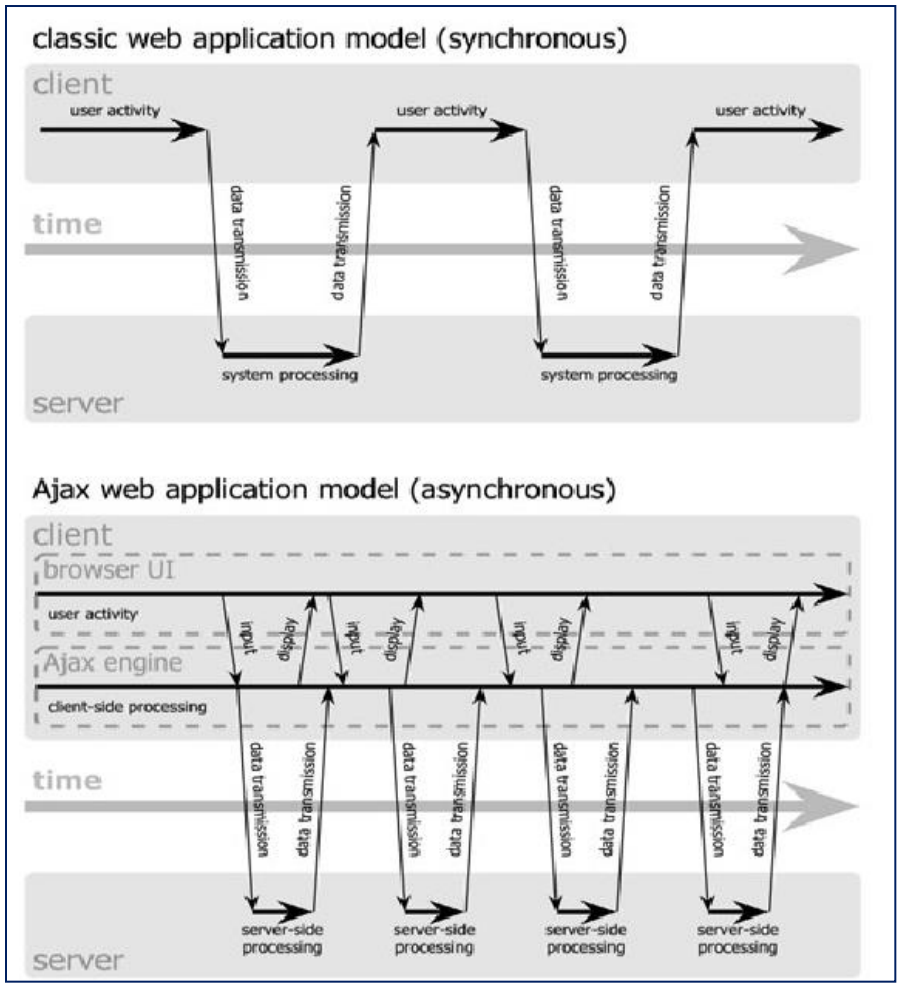
AJAX

AJAX 란?

- **AJAX(비동기 방식의 자바스크립트 XML)**
  - **Asynchronous JavaScript and XML**
- **AJAX Apps**
  - 구글맵, 야후맵, 네이버 검색 등..
- 직관적이고 자연스런 사용자 상호액션 방식
  - 기존 클릭이 필요하지 없음
- 화면의 일부분의 변경 :
  - 기존 - 서버 요청, 대기, 전체화면 새로고침
  - **AJAX** – 업데이터가 필요한 부분만 변경

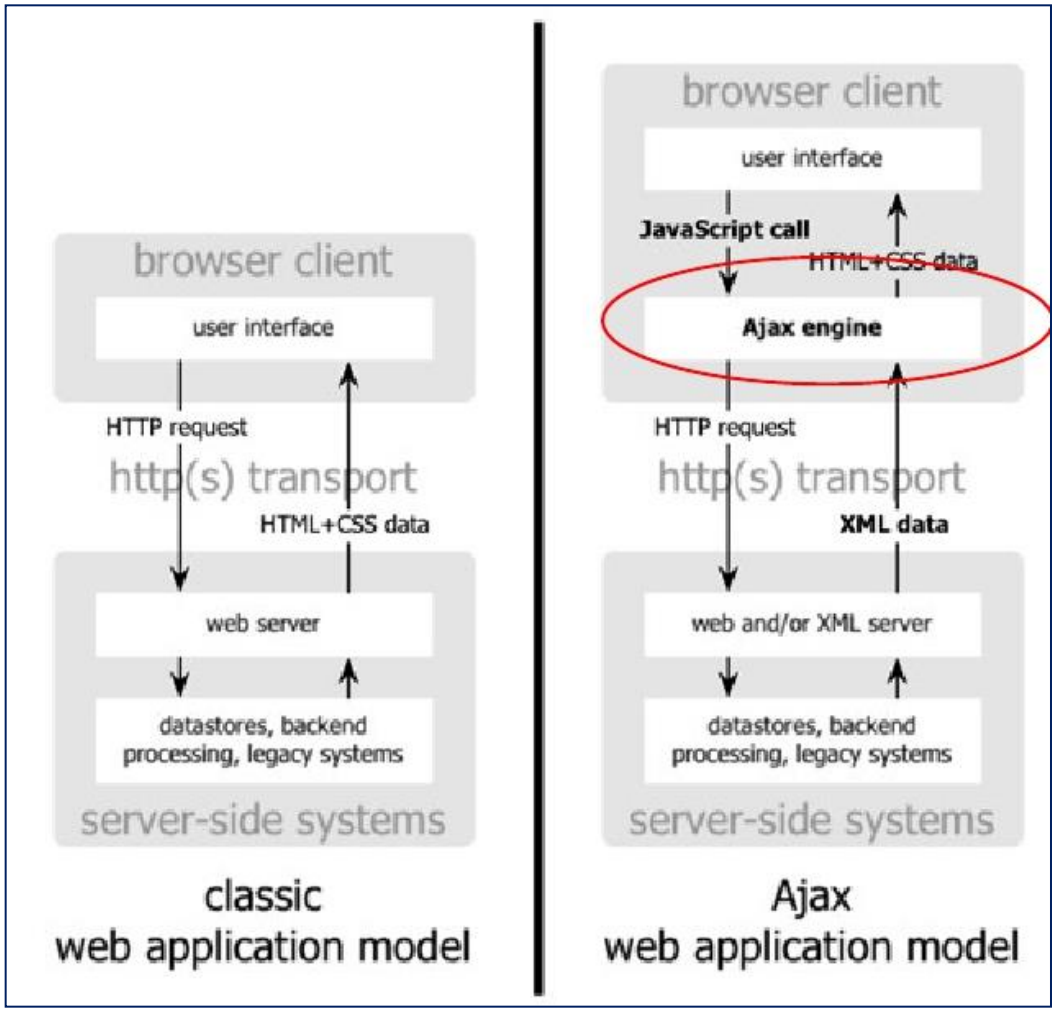
동작방식



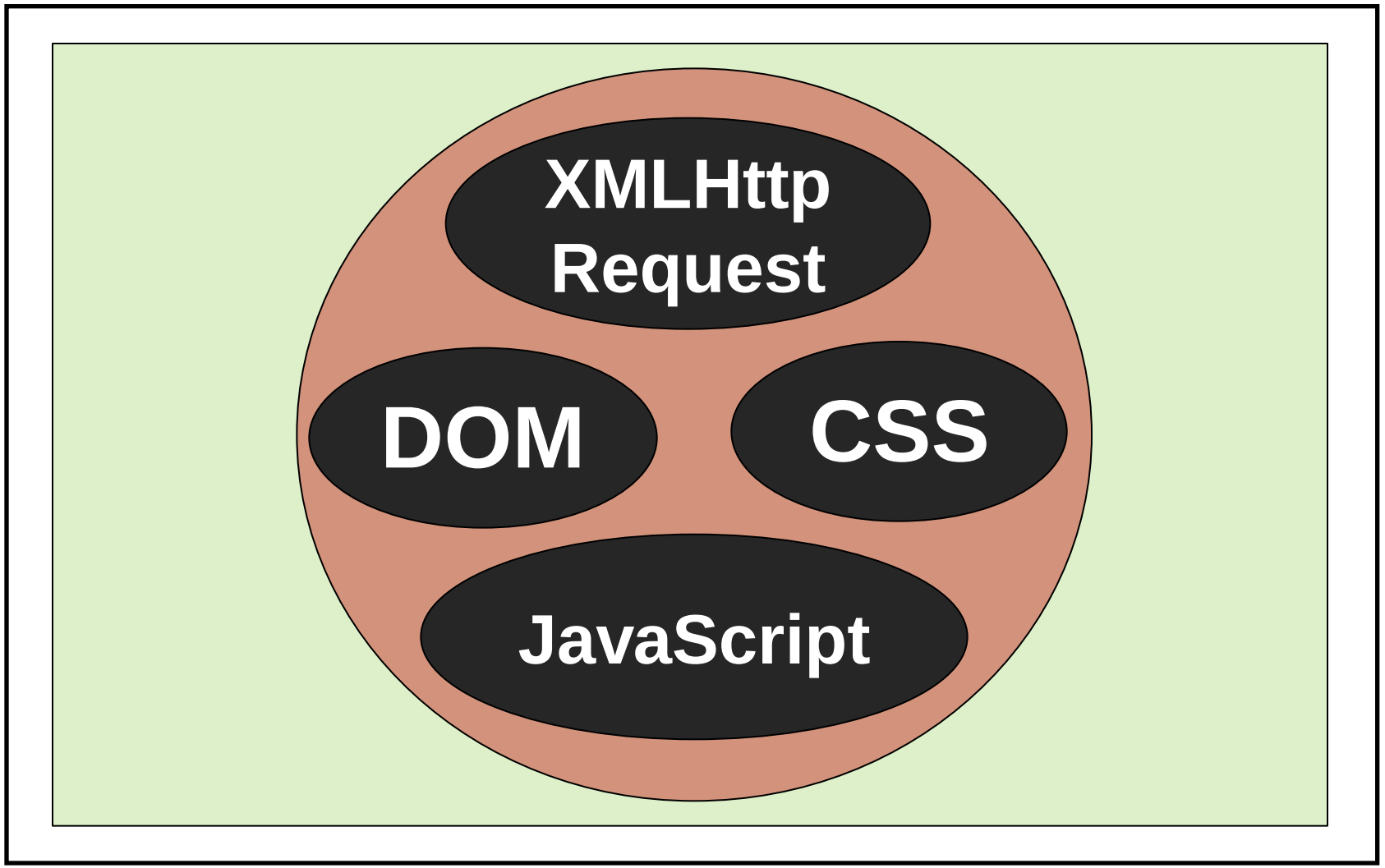
서버에 요청한 데이터가 도착할 때까지 클라이언트는 대기

서버에 요청한 데이터가 도착할 동안 클라이언트는 멈추지 않고 동작

동작방식


browser client
user interface
HTTP request
http(s) transport
HTML+CSS data
web server
datastores, backend processing, legacy systems
server-side systems
classic web application model
browser client
user interface
JavaScript call
HTML+CSS data
Ajax engine
HTTP request
http(s) transport
XML data
web and/or XML server
datastores, backend processing, legacy systems
server-side systems
Ajax web application model

AJAX 구성요소



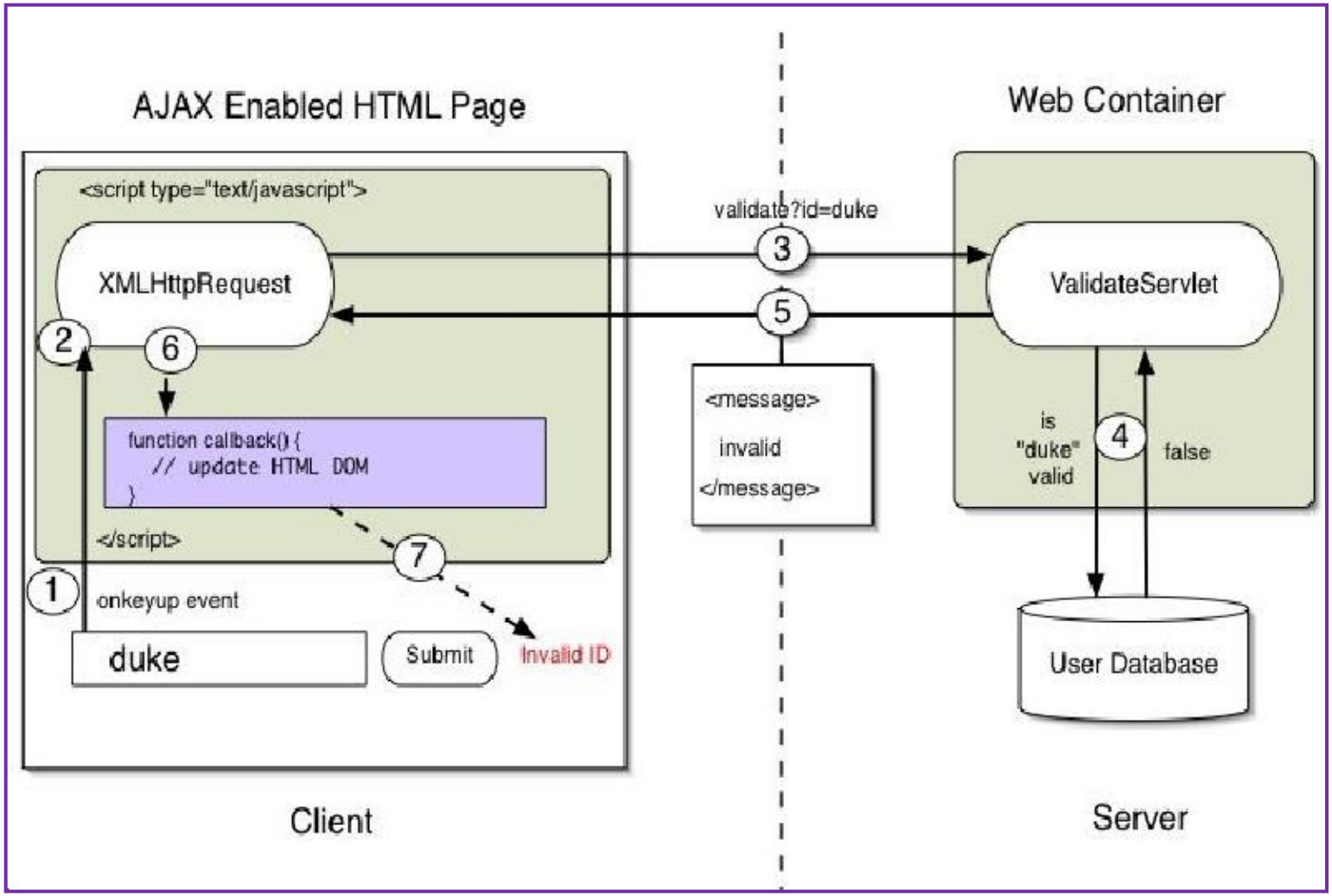
XMLHttpRequest

- 자바스크립트 객체
- 대부분의 브라우져에서 지원
  - Firefox, Opera, Safari, Mozilla
- 표준 HTTP방식(GET/POST) 으로 서버와 통신
- 서버와 통신시 비동기적으로 작업
  - 백그라운드에서 작업

Server Processing

- 기존 서버 작업 방식 사용
  - Servlet, JSP, ……
- 응답 문서 타입
  - 기존 html 외에 xml, json, 단순 텍스트 사용

AJAX Interaction 해석



AJAX 프로그래밍 순서

1. 클라이언트 이벤트 발생
2. XMLHttpRequest 객체 생성
3. XMLHttpRequest 객체 콜백함수 설정
4. XMLHttpRequest 객체를 통한 비동기화 요청
5. 서버 응답결과를 생성하여 클라이언트로 전송
6. XMLHttpRequest 객체는 서버 결과를 처리할 콜백함수 호출
7. 결과를 클라이언트 화면에 반영

1. 클라이언트 이벤트 발생

이벤트 발생

1. 사용자가 버튼을 클릭 시 자바스크립트 함수(requestMsg)를 호출한다.

html

```
<input type="button" value="서버에 자료요청"
       onclick="requestMsg()">
```

2. XMLHttpRequest 객체 생성

```
Script
var httpRequest = null;
function getXMLHttpRequest() {
    if (window.ActiveXObject) {
        return new ActiveXObject("Microsoft.XMLHTTP");
    } else if (window.XMLHttpRequest) {
        return new XMLHttpRequest();
    } else {
        return null;
    }
}
function requestMsg() {
    httpRequest = getXMLHttpRequest();
    sendRequest("hello.jsp", null, responseMsg, "GET");
}
```

3. XMLHttpRequest 콜백함수 설정

```
Script


function sendRequest(url, params, callback, method) {
    httpRequest.onreadystatechange = callback;
    var httpUrl = url;
    httpRequest.open("GET", httpUrl, true);
    httpRequest.send(null);
}

function requestMsg() {
    httpRequest = getXMLHttpRequest();
    sendRequest("hello.jsp", null, responseMsg, "GET");
}
```

4. XMLHttpRequest 객체를 통한 비동기화 요청

```
Script


function sendRequest(url, params, callback, method) {
    httpRequest.onreadystatechange = callback;
    var httpUrl = url;
    httpRequest.open("GET", httpUrl, true);
    httpRequest.send(null);
}

function requestMsg() {
    httpRequest = getXMLHttpRequest();
    sendRequest("hello.jsp", null, responseMsg, "GET");
}
```

5. 서버 응답결과를 생성하여 클라이언트 전송

Client Request

```
function requestMsg() {
    httpRequest = getXMLHttpRequest();
    sendRequest("hello.jsp", null, responseMsg, "GET");
}
```

Server Response

hello.jsp

```
<%@ page language="java"
         contentType="text/html; charset=EUC-KR"
         pageEncoding="EUC-KR"%>
<h1>Hello AJAX World!!!</h1>
```

6. 서버 응답결과를 처리할 콜백함수 호출

Client Script

```
function responseMsg() {
  if (httpRequest.readyState == 4) {
    if (httpRequest.status == 200) {
      var msg_id = document.getElementById("msgView");
          msg_id.innerHTML += httpRequest.responseText;
    }
  }
}
```

7. 결과를 클라이언트 화면에 적용

Client Script

```
function responseMsg() {
  if (httpRequest.readyState == 4) {
    if (httpRequest.status == 200) {
      var msg_id = document.getElementById("msgView");
          msg_id.innerHTML += httpRequest.responseText;
    }
  }
}
```

XMLHttpRequest Methods

- open("HTTP method", "URL", syn/asyn)
  - ✓ 요청의 초기화 작업
  - ✓ GET / POST 지정
  - ✓ 서버 URL 지정
  - ✓ 동기 / 비동기 설정
- send(content)
  - ✓ GET 방식은 URL에 필요정보 추가 하기 때문에 null 적용
  - ✓ POST 방식에서 파라미터 설정 처리

XMLHttpRequest Properties

- onreadystatechange
  - ✓ 서버에서 응답이 도착했을 때 호출될 콜백함수 지정
  - ✓ 콜백함수는 상태(readyState)가 변경될 때 마다 호출
- readyState : 요청의 현재 상태
  - ✓ 0 : uninitialized (객체 생성 후 open 메서드 호출 전)
  - ✓ 1 : loading (open 메서드가 호출되고 send 호출 전)
  - ✓ 2 : loaded (send 메서드가 호출되었지만 서버응답 전)
  - ✓ 3 : interactive (데이터의 일부가 전송된 상태)
  - ✓ 4 : completed (모든 데이터 전송 완료)

XMLHttpRequest Properties

- status
  - ✓ 서버 처리 결과 상태 코드
  - ✓ 200 : OK (요청 성공)
  - ✓ 404 : Not Found (페이지를 못찾을 경우)
  - ✓ 500 : Server Error (서버에서 결과 생성 시 오류 발생)
- responseText
  - ✓ 서버의 응답결과를 문자열로 받기
- responseXML
  - ✓ 서버의 응답결과를 XML Document로 받기