# JSON, REST, AJAX 통합 정리

## 1. JSON 특징 요약

|  |  |
| --- | --- |
| 항목 | 설명 |
| 정의 | JSON은 JavaScript Object Notation의 약자로, 데이터를 저장하거나 전송할 때 사용하는 경량 데이터 교환 형식 |
| 특징 | 서버와 클라이언트 간 데이터 교환에 주로 사용, 자바스크립트 객체 표기법과 유사, 다양한 언어에서 사용 가능 |
| XML과의 비교 | XML은 태그로 감싸서 데이터를 표현하여 용량이 큼, JSON은 중괄호와 콤마로 간결하게 표현하여 가볍고 직관적 |
| 문법 | key와 value 쌍으로 이루어지며, key와 문자열은 반드시 쌍따옴표로 감싸야 함. 중첩 가능하며, 배열과 객체 구조를 사용할 수 있음 |
| 문제점 | 보안 문제로 인해 악성 스크립트가 실행될 위험이 있음. CORS 문제로 인해 동일 출처 정책이 적용됨 |
| 자바스크립트와의 호환성 | JSON.parse(): JSON 형식의 문자열을 자바스크립트 객체로 변환, JSON.stringify(): 자바스크립트 객체를 JSON 문자열로 변환 |

## 2. JSON Syntax Rules 및 데이터 타입

## JSON Syntax Rules

|  |  |
| --- | --- |
| 항목 | 설명 |
| 기본 규칙 | 이름과 값을 쌍으로 사용한다. |
| 자료 구분 | 자료는 콤마로 구분한다. |
| 객체 표기 | 객체는 {}로 감싼다. |
| 배열 표기 | 배열은 []로 감싼다. |
| 이름 표기 | 자바스크립트에서는 이름을 "로 감싸지 않지만, JSON에서는 이름을 "로 감싼다. |

## JSON과 JavaScript의 이름 사용 비교

|  |  |
| --- | --- |
| 종류 | 예시 |
| JSON | { "name":"John" } |
| JavaScript | { name:"John" } |

## JSON 데이터 타입

|  |  |
| --- | --- |
| 항목 | 설명 |
| 허용 데이터 타입 | string, number, object, array, boolean, null |
| 허용되지 않는 데이터 타입 | function, date, undefined |

## JSON 예시

|  |  |
| --- | --- |
| 종류 | 예시 |
| JSON Strings | { "name":"John" } |
| JSON Numbers | { "age":30 } |
| JSON Objects | { "employee":{ "name":"John", "age":30, "city":"New York" } } |
| JSON Arrays | { "employees":[ "John", "Anna", "Peter" ] } |
| JSON Booleans | { "sale":true } |
| JSON Null | { "middlename":null } |

## 3. JSON.stringify() / JSON.parse()

|  |  |
| --- | --- |
| stringify() | 자바스크립트의 값을 JSON 문자열로 변환합니다. |
| var obj = { name: "John", age: 30, city: "New York" }; var myJSON = JSON.stringify(obj); myJSON // "{"name":"John","age":30,"city":"New York"}" |

|  |  |
| --- | --- |
| parse() | JSON 문자열을 자바스크립트 객체로 변환하여 반환합니다. |
| var txt = '{"name":"John", "age":30, "city":"New York"}'; var obj = JSON.parse(txt); obj // {name: "John", age: 30, city: "New York"} |

## 4. REST 개념 및 특징

# REST 개념 및 특징 정리

## REST 정의 및 개념

REST(Representational State Transfer)는 자원을 이름으로 구분하여 해당 자원의 상태(정보)를 주고받는 모든 것을 의미합니다.

자원을 URI로 명시하고, HTTP Method(POST, GET, PUT, DELETE)를 통해 CRUD Operation을 적용하는 것을 의미합니다.

## REST 구성 요소

|  |  |
| --- | --- |
| 구성 요소 | 설명 |
| 자원(Resource) | URI를 통해 접근할 수 있는 모든 것 (예: /users, /groups) |
| 행위(Verb) | HTTP Method (GET, POST, PUT, DELETE) |
| 표현(Representation) | JSON, XML 등을 통해 자원의 상태를 전달 |

## HTTP Method와 CRUD 매핑

|  |  |
| --- | --- |
| HTTP Method | CRUD 연산 |
| GET | Read (조회) |
| POST | Create (생성) |
| PUT | Update (수정) |
| DELETE | Delete (삭제) |
| HEAD | Header 정보 조회 |

## REST API 특징

|  |  |
| --- | --- |
| 특징 | 설명 |
| Server-Client 구조 | 자원이 있는 쪽이 Server, 요청하는 쪽이 Client |
| Stateless | 각 요청은 독립적으로 처리되어야 함 |
| Cacheable | HTTP 캐싱을 활용하여 성능 향상 |
| Layered System | 클라이언트는 중간 서버를 인식하지 못함 |
| Uniform Interface | 일관성 있는 인터페이스 제공 |

## 5 REST API 설계 가이드

1. URI는 정보의 자원을 표현해야 한다.

2. 자원에 대한 행위는 HTTP Method(GET, POST, PUT, DELETE)로 표현한다.

3. URI 예시:

/users GET - 사용자 목록 조회  
/users/{id} GET - 특정 사용자 조회  
/users POST - 사용자 생성  
/users/{id} PUT - 사용자 수정  
/users/{id} DELETE - 사용자 삭제

## 5. Ajax 정리

Ajax는 JavaScript의 라이브러리 중 하나로, 비동기적으로 데이터를 서버와 교환하여 웹 페이지를 업데이트할 수 있는 기술입니다.

특징: 비동기 통신, XML 또는 JSON 데이터를 주고받음, XMLHttpRequest 객체를 이용하여 일부 데이터만 갱신

# 6 Ajax 정리

## AJAX란?

AJAX는 비동기적으로 데이터를 서버와 교환하여 웹 페이지를 업데이트할 수 있는 기술입니다. JavaScript 라이브러리 중 하나로, 클라이언트와 서버 간에 XML 또는 JSON 데이터를 주고받습니다.

## AJAX 기본 구조

AJAX 요청을 초기화하고 서버로 HTTP 요청을 보낸 후, 응답에 따라 콜백 함수를 실행하여 비동기적으로 데이터를 처리합니다.

기본 구조:

$.ajax({  
 type: "POST",  
 url: "some.php",  
 data: { name: "John", location: "Boston" }  
}).done(function(result) {  
 alert("Data Saved: " + result);  
}).fail(function(xhr, textStatus, errorThrown) {  
 alert("Error: " + errorThrown);  
});

## AJAX 주요 옵션 및 설명

|  |  |
| --- | --- |
| 옵션 | 설명 |
| type | 데이터 전송 타입 (GET, POST 등). POST는 데이터가 주소창에 보이지 않음. |
| url | 요청을 보낼 서버의 URL 경로. |
| data | 서버로 보낼 데이터를 설정 (문자열 또는 객체 형태). |
| success | 요청에 성공했을 때 실행되는 콜백 함수. |
| error | 요청에 실패했을 때 실행되는 콜백 함수. |

## AJAX 요청 성공 및 실패 처리

1. 성공 처리: .done(function(result) {...})

- 서버로부터의 응답이 성공했을 때 실행됩니다.

2. 실패 처리: .fail(function(xhr, textStatus, errorThrown) {...})

- 서버로부터의 응답이 실패했을 때 실행됩니다. xhr: XMLHttpRequest 객체, textStatus: 상태 정보, errorThrown: 예외 정보.

## AJAX 특징 요약

- 비동기적으로 데이터를 전송하여 페이지 리로드 없이 업데이트 가능

- 데이터를 서버와 교환하여 빠른 사용자 경험 제공

- 콜백 함수를 통해 성공 또는 실패를 처리하여 유연한 응답 제어

## Ajax란?

Ajax는 JavaScript의 라이브러리 중 하나이며, Asynchronous Javascript And Xml(비동기식 자바스크립트와 XML)의 약자입니다.

## 비동기란?

서버에 일을 요청하고 기다리지 않고 다른 일을 진행하는 것을 의미합니다. 내부에 콜백함수를 만들어 놓으면 서버에서 요청한 일을 콜백함수가 받아줍니다. 화면을 유지한 채(다른 페이지로 넘어가지 않고) 작업할 수 있습니다.

## Ajax 특징

- JavaScript를 사용한 비동기 통신

- 클라이언트와 서버 간에 XML 또는 JSON 데이터를 주고받는 기술

- 브라우저의 XMLHttpRequest 객체를 이용하여 전체 페이지를 다시 로드하지 않고도 일부 데이터만 갱신

- 데이터가 바뀌어도 페이지 전체를 다시 로드하지 않아 트래픽 감소

## Ajax 한마디로 정의

JavaScript를 사용한 비동기 통신 방식으로, 클라이언트와 서버 간에 XML 또는 JSON 데이터를 주고받는 기술입니다.

## 6. AJAX 사용법 및 예제

AJAX 요청 구조:

$.ajax({  
 type: "POST",  
 url: "some.php",  
 data: { name: "John", location: "Boston" }  
}).done(function(result) {  
 alert("Data Saved: " + result);  
}).fail(function(xhr, textStatus, errorThrown) {  
 alert("Error: " + errorThrown);  
});

AJAX 주요 옵션: type, url, data, success, error

AJAX 요청 성공 시: .done(function(result) {...})

AJAX 요청 실패 시: .fail(function(xhr, textStatus, errorThrown) {...})