AJAX (Asynchronous JavaScript and XML)

AJAX란?

JavaScript와 XML을 이용한 비동기적 정보교환 기법이다.

전체 페이지를 새로 가져오지않고도 페이지 일부만을 변경할 수 있도록 서버에 데이터만을 별도로 요청하는 기법이다.

· 비동기성(Asynchronous): 서버 요청이 완료될 때까지 기다리지 않고 다른 작업을 계속 할 수 있다.

· 자바스크립트(JavaScript): AJAX는 주로 자바스크립트를 사용하여 서버와 통신한다.

· XML/JSON: 데이터는 보통 XML이나 JSON형식으로 주고받음. 요즘은 JSON이 더 많이 사용됨.

동작 과정

1. 자바스크립트 코드에서 `XMLHttpRequest` 객체를 생성.

2. `open` 메서드를 사용하여 서버에 요청을 설정.

3. `send` 메서드를 통해 서버에 요청을 보냄.

4. 서버에서 응답이 오면, `onreadystatechange` 이벤트 핸들러가 실행.

5. 응답데이터를 처리하고, 필요에 따라 웹 페이지 내용을 동적으로 업데이트.

AJAX를 사용하면 사용자 경험을 크게 향상시킬 수 있다. Ex) 검색창 자동완성, 무한스크롤, 실시간 채팅 등

Forward와 Redirect의 차이

클라이언트 요청을 처리한 후 다른 자원으로 이동시키기 위해 사용되는 방식

Forward

서버 내부에서 처리: 클라이언트의 요청을 서버 내부에서 다른 자원으로 전달

URL 변경 없음: 클라이언트는 URL이 변경되지 않으므로, 브라우저의 주소 표시줄에 표시된 URL이 그대로 유지됨.

속도 빠름: 서버 내부에서 이동하므로 네트워크 왕복이 없어서 더 빠름.

상태 유지: 원래 요청 객체와 응답 객체를 그대로 유지

Redirect

클라이언트 측에서 처리: 서버가 클라이언트에게 새로운 URL로 다시 요청을 보내도록 지시

URL 변경: 클라이언트의 브라우저 주소 표시줄에 새로운 URL이 표시

속도 느림: 클라이언트가 새로운 요청을 보내야 하므로 네트워크 왕복이 발생하여 속도가 느림.

상태 초기화: 새로운 요청이기 때문에 이전 요청 객체와 응답 객체는 유지되지 않음.

AJAX, Forward, Redirect 이 세방식의 차이를 간단하게 표로 보자면

방식	페이지 새로 고침	URL 변경
AJAX	X	X
Forward	O	X
Redirect	O	O

HTTP (HyperText Transfer Protocol)

클라이언트와 서버 사이에 이루어지는 요청/응답(request/response) 프로토콜.

쉽게 말하면 웹 브라우저가 서버와 통신하는 규칙.

HTTP 요청 (HTTP Request)

1. 요청 라인 (Request Line):

요청 메서드 (GET, POST, PUT, DELETE 등): 리소스에 대한 행동을 지정.

요청 URI (Uniform Resource Identifier): 접근하고자 하는 리소스의 경로를 지정.

HTTP 버전: 요청하는 클라이언트의 HTTP 프로토콜 버전을 명시.

속도 빠름: 서버 내부에서 이동하므로 네트워크 왕복이 없어서 더 빠름.

2. 헤더 (Headers)

요청의 메타데이터를 포함. Ex) 클라이언트 정보, 요청 본문의 타입, 쿠키 정보 등을 전송

3. 본문 (Body)

선택적으로 요청의 데이터를 포함할 수 있는 부분. 주로 POST 요청에서 사용되며, 데이터 전송에 필요한 정보를 담음.

HTTP 응답 (HTTP Response)

1. 상태 라인 (Status Line)

상태 코드 (Status Code): 요청의 처리 결과를 나타내는 숫자 코드. Ex) 404 Not Found

2. 헤더 (Headers)

응답의 메타데이터를 포함. Ex) 서버 정보, 응답 본문의 타입, 쿠키 설정 등을 포함.

3. 본문 (Body)

선택적으로 응답의 데이터를 포함하는 부분. 주로 HTML, JSON, XML 등의 형식으로 클라이언트에게 전송.

클라이언트(브라우저) -> HTTP요청(request) -> 웹서버 -> HTTP응답(response) -> 클라이언트

HTTPS (HyperText Transfer Protocol Secure)

HTTPS는 웹에서 데이터를 안전하게 주고받기 위해 HTTP에 SSL(Secure Sockets Layer) 또는 TLS(Transport Layer Security) 프로토콜을 적용한 것이다.

HTTPS는 웹 서버와 클라이언트 간의 통신을 암호화하여 데이터의 기밀성과 무결성을 보장함.

HTTPS 특징

1. 암호화: 모든 데이터가 암호화되어 전송 중에 도청을 방지.

2. 데이터 무결성: 전송된 데이터가 중간에서 변경되지 않았음을 보장.

3. 인증: 서버의 신원을 확인하여 피싱 공격과 같은 사기 행위를 방지.

SSL/TLS

웹 브라우저와 서버 간의 통신을 안전하게 암호화하는 프로토콜. SSL은 초기버전, TLS는 후속 버전

SSL/TLS 작동 원리

1. 핸드셰이크 단계: 클라이언트가 서버에 접속을 시도하면 SSL/TLS 핸드셰이크가 시작됨. 서버는 클라이언트에게 SSL/TLS 인증서를 보내고 클라이언트는 인증서를 확인하고 서버의 신원을 검증.

2. 세션 키 생성: 클라이언트와 서버는 대칭키 암호화를 성정하기 위해 세션키를 생성. 세션키는 공개키 암호화를 사용하여 안전하게 교환.

3. 데이터 암호화 전송: 핸드셰이크가 완료되면 클라이언트와 서버는 세션키를 사용하여 데이터를 암호화하고 전송.

SSL/TLS 인증서

웹 서버의 신원을 확인하고 클라이언트와 서버 간의 암호화된 통신을 설정하는데 사용, 인증 기관에 의해 발급됨.

주요 구성 요소

1. 도메인 이름: 인증서가 발급된 도메인의 이름..

2. 발급자: 인증서를 발급한 인증 기관의 이름.

3. 공개키: 서버의 공개키로, 클라이언트가 데이터를 암호화하는데 사용됨.

4. 유효 기간: 인증서의 유효 기간을 나타냄.

5. 서명: 인증 기관의 디지털 서명으로, 인증서의 무결성과 신뢰성을 보장함.

SSL/TLS 인증서의 종류

도메인 인증서 (Domain Validated, DV)

도메인 소유권만 검증됨. 발급이 빠르고 저렴함.

기관 인증서 (Organization Validated, OV)

도메인 소유권과 함께 조직의 신원도 검증됨. 신뢰도가 높음.

확장 인증서 (Extended Validation, EV):

도메인 소유권과 조직의 신원을 철저히 검증함. 브라우저 주소 표시줄에 녹색 자물쇠와 조직 이름이 표시됨.

간단하게 비유하자면

HTTPS는 클라이언트와 서버가 안전하게 대화할 수 있는 비밀통로(SSL/TLS 프로토콜)를 만드는 것이고 SSL/TLS인증서는 그 통로가 진짜 서버와 연결되어 있다는 것을 확인해주는 신분증과 같음.

따라서 HTTPS와 SSL/TLS 인증서는 인터넷 상에서 안전한 데이터 전송을 보장하고, 사용자가 신뢰할 수 있는 통신환경을 제공하는 중요한 역할을 한다.