HTML : 정적 웹페이지 표현

JavaScript

: 동적 웹페이지 표현

: JAVA와 별개

: 웹 브라우저에서 바로 실행 -> 현재 페이지 새로고침 불가

JSP (Java Server Pages)

: Back end에서 실행되는 JAVA기반 언어 (Java Servlet API) -> 현재 페이지 새로고침 가능

: HTML 문서에 코드 내재 (MVC의 View 역할)

Back end 실행 결과물은 HTML로 반환되어 Front end 에서 접근 가능하게 된다.

Back end 언어

: JSP, PHP, ASP, ASP.NET, Node.js

Servlet

: Java 코드에 HTML 내재 (MVC의 Controller 역할)

Servlet Container

: HTTP 요청을 받아 Servlet 실행 및 생명주기 관리

: Servlet과 웹 서버(ex : Tomcat) 통신 제공

JSX = Javascript + XML

DOM (Document Object Model)

: HTML / XML 문서 처리 API

ES5

: 2015년 이전의 Javascript 버전. 모든 브라우저에 완벽 대응 가능

이후 ES6, ES7 등 매년 신버전이 나오고 있음. (ES2015+)

But 지원안되는 브라우저 존재

* 트랜스파일러 이용하여 ES5 코드로 변환 작업 필요 (React의 경우 Babel 사용)

앵귤러 (Node.js + WebStrom + TypeScript)

: JS 프레임워크

: 쉽고 정확한 MVC 구현 가능

: 양방향 데이터 바인딩

: 타 프레임워크와 호환

Node.js

: REPL (Read, Eval, Print, Loop)

: Read – 메모리에 저장

: Eval – 입력 값 평가/실행

: Print – Eval로 반환된 값 출력

: Loop – REP 반복

Spring Framework

: JAVA 플랫폼을 위한 오픈소스 App Framework / 동적 웹 사이트 개발 Framework

: 크기 및 부하 경량

: 전자정부 표준 프레임워크 기반 기술

: 관점지향 프로그래밍 (AOP)

: MVC 모델 지원

: POJO(Plain Old Java Object) 방식 프레임워크

: IoC(Inversion of Control : 제어반전)

TypeScript

: Javascript 오픈소스 프로그래밍 언어

SPA (Single Page Application : 단일 페이지 어플리케이션)

: 최초 1회 페이지 전체 로딩 후 데이터만 변경하여 사용 가능한 웹 어플리케이션

* 브라우저 내부에서 동작하며 페이지 리로딩 불필요.

: 모바일 웹 만들기 용이 (웹과 모바일앱에 Back end 코드 재사용 가능)

: Backbone.js, Angular.js 등 사용

: MPA에 비해 보안에 취약

MPA (Multiple Page Application)  
: 브라우저는 매번 서버로부터 페이지 렌더링.

: AJAX를 이용하여 데이터 전달

: 검색 최적화

* 페이지 당 하나의 키워드 적용

: Front end / Back end 별도 개발 필요

AJAX (Asynchronous JavaScript and XML)

: 동적 웹페이지 개발 기법

: 웹페이지의 일부분만 갱신 가능

* 백그라운드 영역에서 서버와 통신하여 페이지에 결과 노출

: JSON, XML, HTML, TXT File 등 통신 가능

: 클라이언트가 서버로 요청하여 처리하는 동적 방식으로 서버가 클라이언트에 요청 불가.

* 클라이언트 풀링 가능 But 서버 푸시 불가능

: Prototype, script.aculo.us, dojo, jQuery 등의 프레임워크 존재

React

: UI 라이브러리

: 높은 재사용성

: 타 javascript 프레임워크 및 라이브러리와 호환 가능

: Virtual DOM 기능 지원

* React 컴포넌트가 return한 결과는 바로 브라우저에 렌더링 되지 않으며

Virtual DOM에서 기존 return 값과 비교하여 변경된 부분만 렌더링하여 적용한다.

* + 브라우저 처리 부담 감소 및 빠른 처리 속도

: 초기 구동시간은 느리다.

: MVC의 View 부분을 컴포넌트로 만들기 위한 라이브러리

: 단방향 Data Flow

: JSX 또는 Javascript로 구현 가능

MVC  
: Model + View + Controller

* Model 1 방식 : 사용자의 요청을 JSP가 전부 처리
* Model 2 방식 : Controller가 사용자의 요청을 받아 Model 처리 및 View 표출