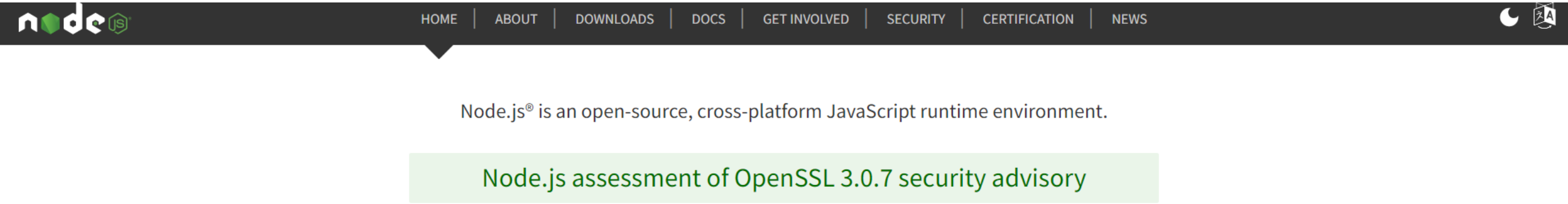
0. 개발 환경 설정

Node.js(+npm)

VS Code

0.1 Node.js는 무엇인가?



JavaScript runtime environment(자바스크립트 실행 환경)

말그대로 자바스크립트를 실행하기 위한 환경을 제공(가상머신)
→ 브라우저가 아닌 환경에서도 실행 가능 + 서버 개발 등 다양한 애플리케이션 개발 가능

0.2 npm(node package manager)

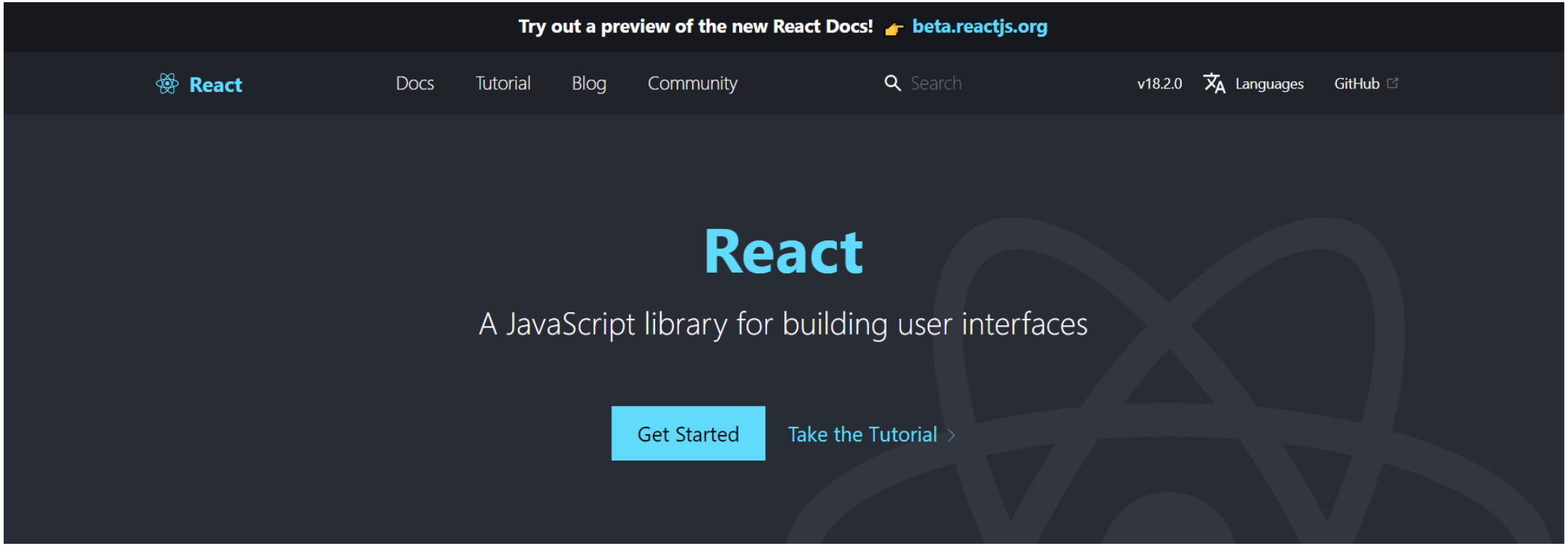
- Node.js 패키지 매니저
- 프로젝트에서 필요로 하는 다양한 외부 패키지들의 버전과 의존성을 관리
- 편하게 설치 및 삭제할 수 있게 도와주는 역할
- Node.js 설치 시 자동으로 설치됨

0.3 Node.js 설치하기

- Node.js 및 npm 버전 확인

1. 리액트 소개

1.1 리액트는 무엇인가?

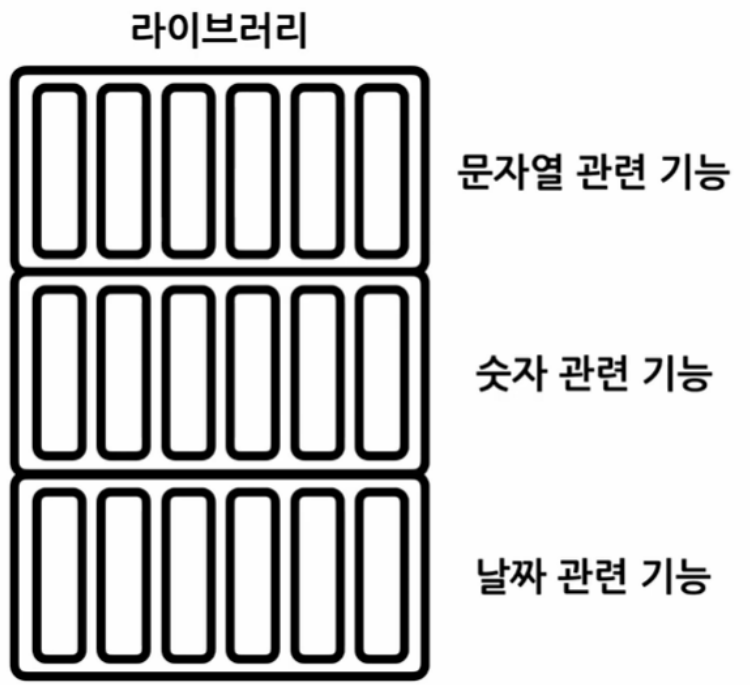


사용자 인터페이스를 만들기 위한 JavaScript 라이브러리

라이브러리? 도서관?

- 도서관에 가면 그림처럼 책장에 수많은 책이 정해진 기준에 따라서 잘 정리되어 있는 것을 볼 수 있음

- 특정 프로그래밍 언어에서 자주 사용되는 기능들을 모아서 정리해 놓은 모음집



사용자 인터페이스(User Interface, UI)

- 사용자와 프로그램이 서로 상호작용을 하기 위해 중간에서 입력과 출력을 제어해주는 것
- 웹 사이트를 예로 들면 버튼이랑 텍스트 입력창 등

리액트 =

사용자와 웹사이트의 상호작용을 돕는 인터페이스(UI)를
만들기 위한 자바스크립트 기능 모음집

쉽게 설명하면 화면을 만들기 위한 기능들을 모아 놓은 것

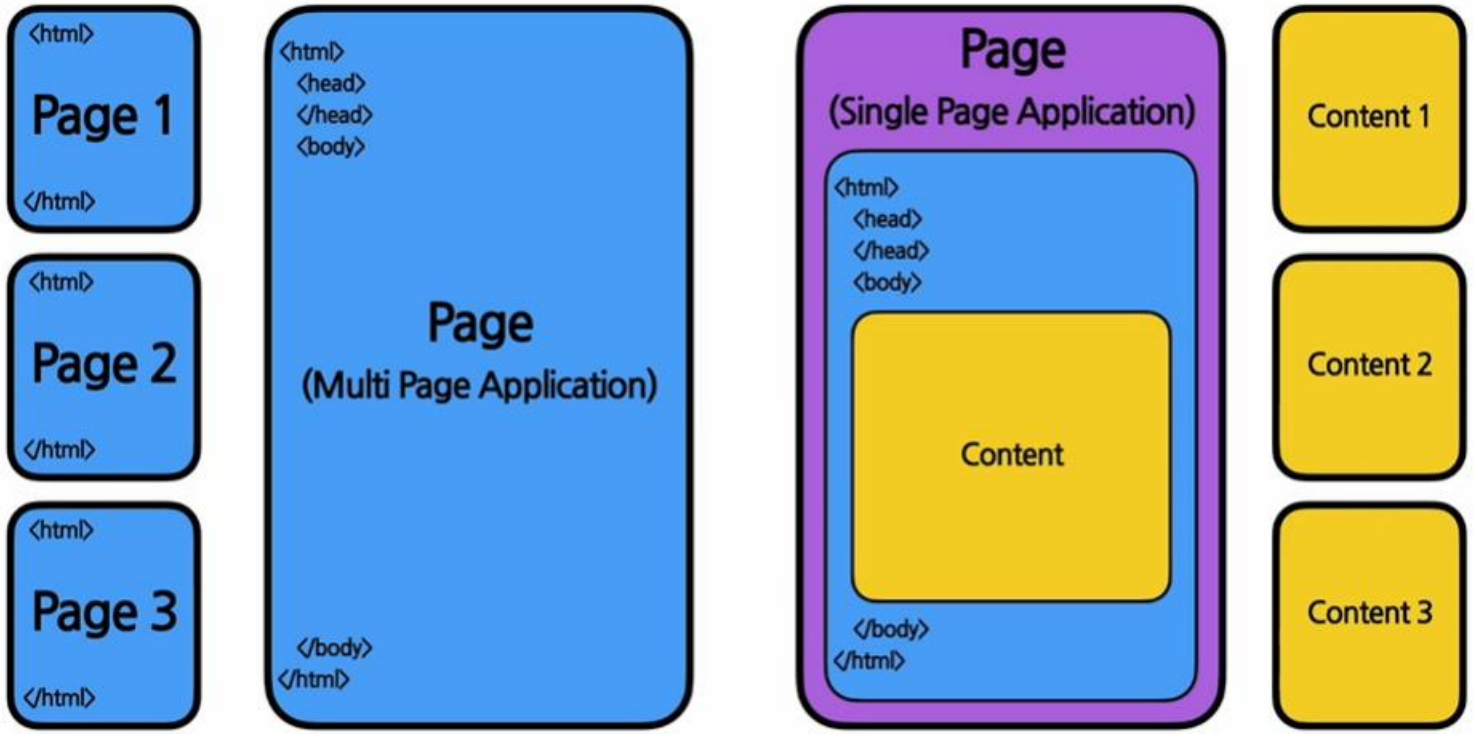
대표적인 자바스크립트 UI 라이브러리(그 외 Vue.js, Angular)

SPA 를 쉽고 빠르게 만들 수 있도록 해주는 도구

SPA(Single Page Application)

- 하나의 페이지만 존재하는 웹 사이트(웹 애플리케이션)을 의미
- 규모가 큰 웹 사이트의 경우 수십~수백 개의 페이지가 존재 페이지마다 HTML 로 만드는 것은 굉장히 비효율적이고 관리하기도 힘듦
- 그래서 HTML 틀을 만들어 놓고 사용자가 특정 페이지를 요청할 때 그 안에 해당 페이지의 내용만 채우는 것이 바로 SPA
- 이러한 복잡한 웹 사이트를 SPA 방식으로 쉽고 빠르게 만들며 관리하기 위해 만들어진 것이 바로 리액트

MPA vs. SPA



- 각 페이지 별로 HTML 파일이 따로 존재, 페이지를 이동할 때 해당 페이지의 HTML 파일을 받아와서 표시(새로고침)
- 단 하나의 페이지만 존재, 페이지를 이동할 때 해당하는 콘텐츠를 가져와서 동적으로 채워 넣음

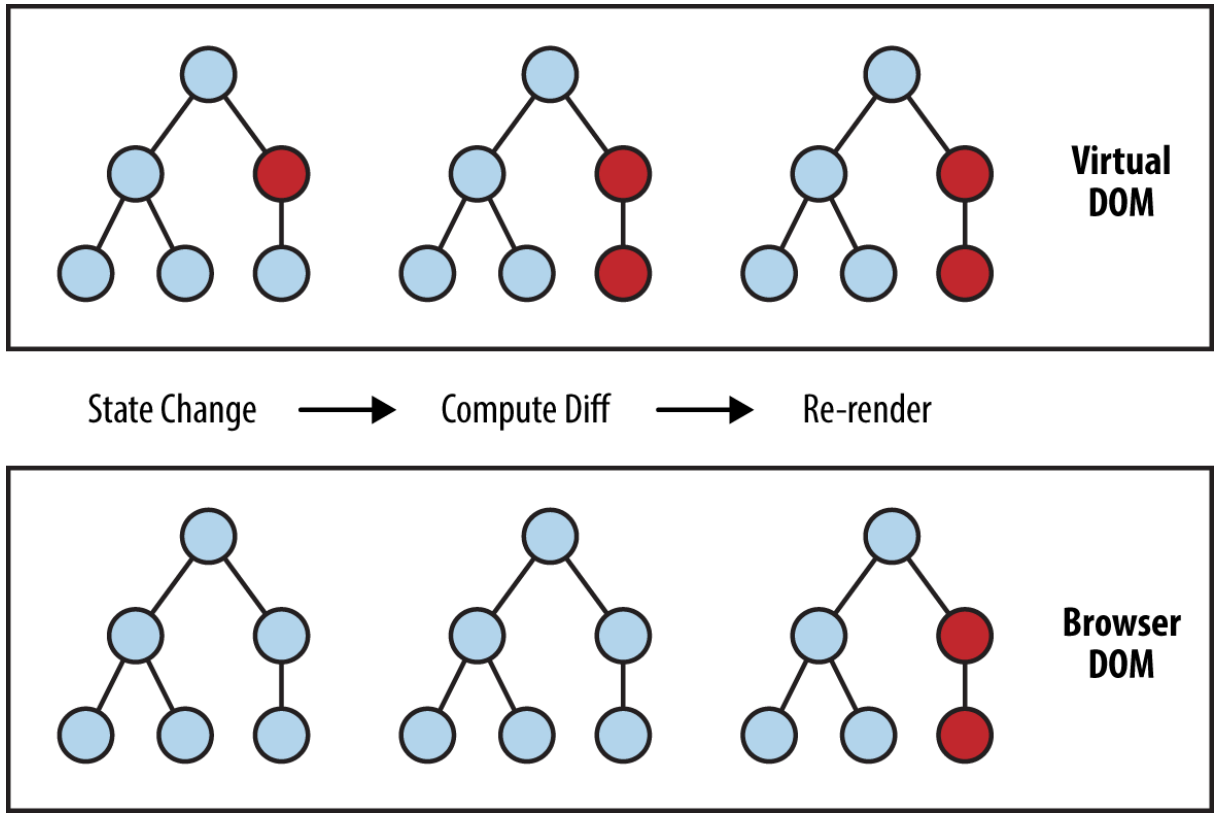
1.2 리액트의 장점

1) 빠른 업데이트 & 렌더링 속도

- 여기서 업데이트는 화면에 나타나는 내용이 바뀌는 것을 의미
- 이를 위해 내부적으로 Virtual DOM(가상 돔)을 사용
  - Real DOM(실제 돔)의 사본(가벼움, 빠름)
  - 웹 페이지와 Real DOM 사이에서 중간 매개체 역할
- 기존의 방식: 화면을 업데이트하려면 DOM을 직접 수정 → 성능에 영향(수정할 부분을 DOM에서 모두 찾고 여러 번 연산)
- 리액트 방식: DOM을 직접 수정하는 것이 아니라 가상 돔에 먼저 렌더링하고 이전 돔과 비교하여 업데이트 해야할 최소한의 부분만 찾아서 수정(한번의 연산)

1.2 리액트의 장점

- 해당 엘리먼트와 그 자식 엘리먼트를 이전의 엘리먼트와 비교하고 필요한 경우에만 DOM을 업데이트



1.2 리액트의 장점

2) 컴포넌트 구조

- 리액트에서 페이지는 컴포넌트로 구성
- 하나의 컴포넌트는 다른 여러 개의 컴포넌트의 조합으로 구성 될 수 있음
- 마치 레고 블록을 조립하여 하나의 모형을 만드는 것과 같음
- 컴포넌트를 조합하여 하나의 웹 사이트를 개발
- 예) 페이스북, 인스타그램, 에어비앤비, 넷플릭스, 트위치 등
- 장점: 컴포넌트의 재사용 가능!!

1.2 리액트의 장점

3) 재사용성(다시 사용이 가능한 성질)

- 개발 기간이 단축됨
  - 기존에 개발해 둔 모듈을 재사용
- 유지 보수가 용이함
  - 공통으로 사용하는 모듈에 문제가 생기면 해당 모듈만 수정
- 리액트 컴포넌트를 개발할 때 항상 쉽고 재사용 가능한 형태로 개발하는 것이 중요

1.2 리액트의 장점

4) 거대한 생태계와 커뮤니티

- 메타(페이스북)의 지원 = 지속적인 업데이트
- 활발한 지식 공유와 질의 응답
  - stack overflow에 질문 수와 watcher 수
- 다양한 라이브러리(for 리액트)
- 오픈소스 라이브러리 인기 지표
  - github 페이지 star수
  - npm trends 다운로드 수

1.2 리액트의 장점

5) 모바일 앱 개발 가능

- 리액트 네이티브
  - 리액트를 알면 쉽게 공부 가능
- 하나의 소스로 안드로이드 앱과 iOS 앱을 동시 출시
- 네이티브 앱 보다 성능, 속도 면에서 조금 부족

1.3 리액트의 단점

1) 러닝 커브(진입 장벽)

- 방대한 학습량
- 기존과 다른 철학
  - DOM를 직접 제어하던 방식 → 상태 변경
- 빠른 변화(업데이트)
  - 지속적인 공부 필요

1.3 리액트의 단점

2) 복잡한 상태 관리

- state라는 중요한 개념!!
- 리액트 컴포넌트의 상태를 의미
- Virtual DOM의 바뀐 부분만 찾아서 일괄 업데이트 할 때
  - 바뀐 부분 = state가 바뀐 컴포넌트
- 규모가 큰 앱일수록 높은 상태 관리 복잡도
  - 자체 Context API 사용
  - Redux, MobX, Recoil 등 외부 상태 관리 라이브러리 사용